

A Neural Model for Contextual Biasing Score Learning and Filtering

Wanting Huang

Computer Science Department

University of Iowa

Iowa City, USA

wanting-huang@uiowa.edu

Weiran Wang

Computer Science Department

University of Iowa

weiran-wang@uiowa.edu

Abstract—Contextual biasing improves automatic speech recognition (ASR) by integrating external knowledge, such as user-specific phrases or entities, during decoding. In this work, we use an attention-based biasing decoder to produce scores for candidate phrases based on acoustic information extracted by an ASR encoder, which can be used to filter out unlikely phrases and to calculate bonus for shallow-fusion biasing. We introduce a per-token discriminative objective that encourages higher scores for ground-truth phrases while suppressing distractors. Experiments on the Librispeech biasing benchmark show that our method effectively filters out majority of the candidate phrases, and significantly improves recognition accuracy under different biasing conditions when the scores are used in shallow fusion biasing. Our approach is modular and can be used with any ASR system, and the filtering mechanism can potentially boost performance of other biasing methods.

Index Terms—contextual biasing, shallow-fusion biasing, phrase filtering

I. INTRODUCTION

In recent years, Automatic Speech Recognition (ASR) has made significant strides, largely thanks to the rise of end-to-end neural (E2E) models such as Connectionist Temporal Classification (CTC, [1]), Recurrent Neural Network Transducer (RNN-T, [2]), and Attention-based Encoder–Decoder (AED, [3], [4]). These models have achieved impressive results on general benchmarks. However, they still face notable challenges when it comes to recognizing rare or domain-specific terms—such as proper nouns, technical jargon, or personalized content—that appear infrequently in the training data. This shortcoming becomes especially evident in real-world scenarios like virtual assistants or personalized transcription services, where users naturally expect the system to understand context-specific vocabulary.

To bridge this gap, contextual biasing has emerged as an effective strategy. It works by injecting a curated list of task- or user-specific phrases into the ASR decoding process, helping the model give preference to relevant terms. Biasing methods are broadly divided into two categories. The first includes inference-based biasing methods, which augment the scores of hypotheses containing specified biasing phrases during beam search. These methods typically operate without learnable parameters and do not necessitate a training regimen. The second category encompasses model-based biasing methods, characterized by their direct incorporation of contextual information into the ASR architecture during the training phase.

We provide a detailed review of both approaches in Section II. Both inference-based and model-based approaches have their drawbacks. Inference-based methods offer modularity and flexibility, easily integrating into any ASR system, but they struggle to differentiate the probability of various phrases. Conversely, model-based methods can lead to increased architectural complexity. A shared limitation is that both incur substantial computational costs when dealing with a vast number of potential phrases.

In this paper, we present a new approach: we train a biasing decoder on candidate biasing phrases to produce a score for each phrase which reflects the likelihood of the phrase appearing in audio, based on acoustic information extracted by an ASR encoder. Different from most model-based approaches that perform cross-attention to phrase embeddings, the biasing decoder works in the same way as an autoregressive attention decoder (and thus our method can be interpreted as an audio-based language model), except that it only produces the likelihood for candidate phrases instead of the full ASR transcript. Borrowing ideas from standard ASR, our training objective for the biasing decoder ensures it learns a discriminative per-token score. The learned score is used to filter out majority of unlikely phrases, and can also be used to calculate bonus scores in inference-based biasing. This design allows for plug-and-play integration with existing systems, without requiring any changes to the underlying ASR components.

We validate our method on the public Librispeech biasing benchmark [5]. Paired with shallow-fusion biasing [6], our method consistently achieves significant reductions in Word Error Rate (WER) with different number of phrases, and compares favorably with prior works. In the most challenging case of 2000 distractors, our method keeps less than 1% of the phrases during search, reduces the test-clean ASR WER from 2.7% to 2.1%, and test other from 6.3% to 5.0%, and achieves over 50% relative WER reduction on infrequent words.

II. RELATED WORK

Contextual biasing has emerged as a key research focus in automatic speech recognition (ASR), particularly for improving the recognition of rare or domain-specific terms. Broadly, contextual biasing approaches can be grouped into inference-based and model-based methods.

A. Inference-Based Biasing

Inference-based methods incorporate biasing contextual information primarily during the decoding process, often without modifying the ASR model itself. This approach offers modularity and flexibility. Predating the widespread adoption of E2E models, researchers focused on injecting biasing contexts to boost decoding scores for specific words or phrases [7], [8]. Typically, this involved constructing a compact search graph for the phrases and composing it with the normal ASR search graph, in the Weighted Finite State Transducer (WFST) framework [9]. Weights along the biasing graph edges would then add bonus scores to hypotheses matching these phrases. For on-device speech recognition with an E2E model, namely RNN-T, [10] adapted this approach by incorporating bonuses at the subword level. When the bonus scores are incorporated before beam pruning, this method is known as *shallow fusion* because it's similar to how external language model scores are used during inference.

Implementation In this work, we will be testing our method with the inference-based approach. We use the GPU-friendly implementation of shallow-fusion biasing proposed by [6], which potentially facilitates better parallelization. Before search starts, a partial match table is built to provide the index to backup to when encountering a token mismatch for each phrase. During search, we maintain for each hypothesis the partial matching lengths with each biasing phrase. These are essentially the “search state” of shallow fusion, and the transition between states follows the pre-built partial match table. In this approach, a per-token bonus is added to a token expansion during beam search, if this token extends the matching into a phrase in the biasing list, and when the token expansion stops matching into any phrase the accumulated bonus is canceled. In [6], the per-token bonus is a user parameter tuned on the development set. The goal of our work is to learn a neural model that provides a discriminative score.

B. Model-Based Biasing

Model-based methods embed a biasing component directly into the E2E ASR model’s architecture or training process. To accept biasing contexts as an additional input, the cross-attention mechanism [11] is often employed to condition the model’s outputs on these contexts. With many architecture variants, the summarized contextual information is then propagated to the rest of the ASR model to influence predictions [12]–[18].

More recently, using contextual adapters [17] for both encoder and predictor in the RNN-T model, [19] proposed a guided attention method to enforce the cross-attention weights to reflect the existence of a phrase at each audio frame and every output token, with additional cross-entropy or CTC loss. [20] used a bias decoder to predict, for each output token, the index of phrase it belongs to from a phrase list (and a “no-bias” phrase is used when the token does not belong to any given phrase). They extracted an embedding for each phrase in order to make phrase level prediction, and the bias decoder was trained with a cross-entropy loss. At inference time, they

would boost the score of a token if it belongs to a phrase predicted by the bias decoder. In other words, they performed ranking and filtering of phrases on the fly during decoding. [21] identified issues with context adapters and proposed a balanced learning objective to guide attention mechanisms more effectively, particularly for rare phrases.

For deeper integration of contextual information into the joint RNN-T/CTC model, [22] proposed to use biasing losses on intermediate representations from the audio encoder. The intermediate biasing losses are computed with an CTC decoder, and the target sequences are obtained from the ASR transcripts by keeping only the biasing phrase and replacing the rest with “no-bias” tokens. In [23], the authors explored methods for enhancing contextualization within an RNN-T model. Their approach involved introducing biasing lists at the intermediate encoder layers and employing text perturbations, specifically alternative spellings, to compel the model to utilize contextual information [24] used a multi-label synchronous output CTC loss to enhance the synchronization between the ASR output by CTC and the bias outputs by another CTC, and improved phrase-level contextual representation.

Different from most model-based approaches, [25] and [26] incorporated tree-based symbolic representation of biasing list into E2E model’s forward function. Intuitively, this approach is one step closer to shallow fusion.

With the advent of large language model (LLM)-based ASR systems [27]–[31], multiple works have tried to enhance their biasing capability. Besides cross-attention based biasing, there exists another two major approaches. One approach is text-based rescoring or hypothesis editing. [32] explored a generative error correction method using task-activating prompts, improving the error correction capabilities of ASR using zero-shot and few-shot learning. Similarly, [33] proposed a contextual spelling correction method that optimizes LLM prompts to reduce spelling errors, particularly for out-of-vocabulary words, by adjusting ASR outputs dynamically. Both methods leverage LLMs’ ability to process contextual information and improve the accuracy of the final transcription in post-processing. The other approach is prompt-based biasing, where specific prompts are injected into LLMs to steer responses toward more accurate transcriptions based on expected terms or prior knowledge. [34] employed a retrieval-based method using speech similarity to provide named entities from personal databases to LLMs, reducing the error rate of named entity recognition. In a similar vein, [35] used dynamic prompts combined with few-shot learning to bias the model toward recognizing rare named entities, effectively improving ASR accuracy in specialized contexts. To reduce the cost of LLM biasing, [36] proposed using vector quantization for the efficient retrieval of context, enhancing dynamic adaptation during recognition.

III. OUR METHOD

A. Training

Let \mathbf{X} denote the input acoustic feature sequence, which is the output of the ASR encoder in this work, and $\mathbf{Y} =$

(y_1, \dots, y_T) be the ASR label sequence. We are also given a set of candidate phrases $\{\mathbf{p}_1, \dots, \mathbf{p}_M\}$ sampled from the transcripts within the minibatch, as well as their corresponding labels $\{l_1, \dots, l_M\}$ where $l_i = 1$ if \mathbf{p}_i is a segment of Y and $l_i = 0$ otherwise. Our goal is to compute a score for each phrase \mathbf{p}_i conditioned on \mathbf{X} , which help distinguish the positive phrases (those with label 1) from the negative ones (those with label 0). We additionally introduce a special empty phrase \mathbf{p}_0 , which plays the role of “no-bias” in other works: if all sampled phrases are distractors, we assign a label $l_0 = 1$ to \mathbf{p}_0 , and otherwise assign $l_0 = 0$.

Biasing Decoder. We treat each phrase as a token sequence, i.e., $\mathbf{p}_i = \{p_{i1}, \dots, p_{iL_i}\}$ where L_i is the token length of p_i , and use an attention decoder to model the probability of a phrase p_i as:

$$P(\mathbf{p}_i | \mathbf{X}) = \prod_{t=1}^{L_i} P(p_{it} | \{<\text{sos}>, p_{i1}, \dots, p_{i(t-1)}\}, \mathbf{X})$$

where $<\text{sos}>$ denotes the start of phrase and $p_{iL_i} = <\text{eos}>$ denotes the end of phrase. Note that for the empty phrase \mathbf{p}_0 , we still make one prediction on $<\text{eos}>$. Essentially, the biasing decoder implements the same functionality as a normal ASR decoder, except that it models the (shorter) phrase sequences only.

Loss Functions. Our training loss combines the following two components.

- 1) **Phrase-level Log Loss:** We would like all positive phrases to have high probability under the biasing decoder. This is implemented by the log loss below computed over the positive phrases:

$$\mathcal{L}_{\log} = - \sum_{i=1}^M l_i \cdot \log P(\mathbf{p}_i | \mathbf{X}).$$

- 2) **Discriminative Loss:** To ensure that true biasing phrases are strongly preferred over distractors, we introduce a discriminative loss, similar to those used for ASR [37]–[39]. We first compute the phrases-level log-probabilities of both positive and negative phrases, and then divide them by the corresponding phrase lengths to obtain averaged per-token scores:

$$s_i = \log P(\mathbf{p}_i | \mathbf{X}) / L_i. \quad (1)$$

We then normalize the resulting scores in the space of $M + 1$ phrases (including the empty phrase \mathbf{p}_0), with a softmax. Finally, the discriminative loss is defined as:

$$\mathcal{L}_{\text{disc}} = - \sum_{i=0}^M l_i \cdot \log \frac{\exp(s_i)}{\sum_{j=0}^M \exp(s_j)}$$

which encourages the positive phrases to have high per-token scores relative to the negative phrases.

For one training utterance, the final biasing loss $\mathcal{L}_{\text{bias}}$ is a convex combination of the above two losses, weighted by a hyperparameter β :

$$\mathcal{L}_{\text{bias}}(X, \{\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_M\}) = (1 - \beta) \mathcal{L}_{\log} + \beta \mathcal{L}_{\text{disc}}. \quad (2)$$

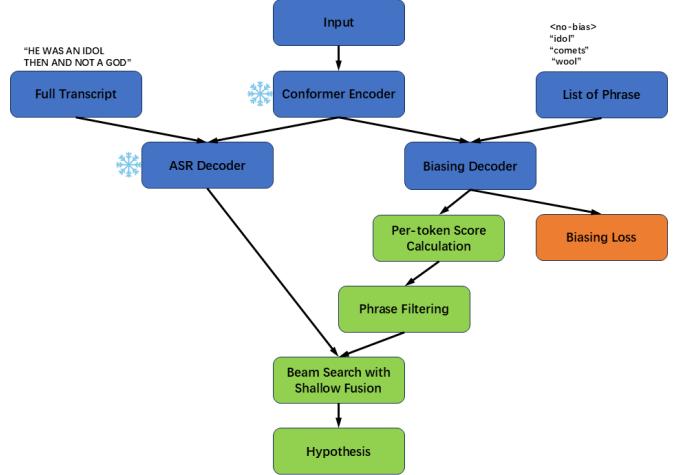


Fig. 1. The overall architecture of our method.

This loss is averaged over a minibatch of utterances for updating the biasing decoder.

B. Inference

At inference time, we perform beam search with search-based (shallow-fusion) biassing using the method of [6]. Note the cost of computing per-token bonus by [6] depends linearly on the number of phrases. To reduce this cost, we use our neural model to perform quick filtering of the large biassing list before search starts, and compute a per-token bonus used by KMP search on the remaining phrases.

We use the following strategy for filtering: for each phrases, we compute the per-token score as in (1), and compare it with the score of \mathbf{p}_0 (no-bias), we keep a phrase \mathbf{p}_i for beam search if its per-token score is high enough so that

$$tol + s_i - s_0 \geq 0 \quad (3)$$

for some user parameter $tol > 0$. In other words, a larger tol keeps more phrases in consideration for which our model has moderate confidence. Furthermore, we use

$$\text{bonus} = \max_i \{tol + s_i - s_0\} \quad (4)$$

as the per-token bonus for the input utterance. Note different utterances can have different per-token bonuses. In the initial exploration, we investigated the use of a different bonus $\{tol + s_i - s_0\}$ for each phrase (which boils down to removing \max_i in (4)) but obtained worse performance, so we stick to the scheme (4) in this paper.

As we will see in the experiments, the default value of $tol = 0$ already works very well, and a small positive value of tol can further boost the accuracy without introducing much more phrases.

An overall illustration of our method is given in Figure 1.

IV. EXPERIMENTS

A. Setup

We demonstrate our method with the Librispeech biassing setup, which is commonly used in prior work [5], [19], [20],

[22], [24]. The Librispeech dataset [40] consists of 960 hours of audio for training. The 5000 most common words in the training set accounts for 90% of all word occurrences, and the remaining 209.2K words in the training vocabulary are considered as *rare* words. At inference time, the ASR model is supplied with a biasing phrase list containing all rare words in the reference, as well as $N = \{100, 500, 1000, 2000\}$ randomly sampled distractors (consisting of also rare words). **Model training.** Our overall model is trained in two stages. We first train the hybrid Attention-CTC ASR model with multi-task objective [41] to capture general acoustic and linguistic knowledge. Then, we freeze the ASR model and update only the biasing decoder, which is trained on bias-augmented data (with 32 phrases per utterance) with the composite loss (2). We sample biasing phrases for each training utterance on the fly. We first randomly sample 3 phrases (with 1 to 3 consecutive words) from the transcript of each utterance, to create a pool of phrases for the minibatch. Then for each utterance, we randomly sample 1 ground truth phrase from the same utterance, and 31 potential distractors from other utterances. Given that we do not filter out frequent words when sampling phrases, it is possible that phrases from other utterances end up being ground truth phrases, and we verify the phrase labels by checking each phrase against the transcript. In summary, each utterance may have multiple positive phrases as a result of the sampling process, and it is future work to avoid frequent words for phrase sampling.

We follow the `librispeech/asr1/` recipe from Espnet [42] for ASR modeling. The conformer encoder has 12 layers and the transformer decoder has 6 layers, both with attention dimension 512 (8 attention heads) and feed-forward dimension 2048. We perform 2x time reduction at the encoder output, by averaging every two consecutive output frames, before CTC and attention decoder. The biasing decoder has the same architecture as the ASR attention decoder. The ASR model has a total of 116M parameters and the biasing decoder has another 30M parameters. The ASR model is trained for 100 epochs, while the biasing decoder is trained for 30 epochs. **Inference.** For decoding, we perform beam search with a beam size of 30. Before search starts, we apply the biasing decoder to filter out un-likely phrases and compute the per-token bonus as discussed in Sec III-B. Note this is a one-time cost and the model forward is done in batch mode (possibly with GPU) over the entire biasing list.

During search, The attention decoder leads the search by proposing for each hypothesis the top expansions. The log-probabilities from attention decoder are then combined with the CTC prefix scores to reduce the number of expansions to 30. Afterwards, the biasing bonuses for theses expansions are calculated and incorporated before pruning.

Evaluation. We measure the word error rate (**WER**), unbiased word error rate (**U-WER**), and biased word error rate (**B-WER**) similarly to previous work. WER is the overall word error rate measured on all words, U-WER measures the WER of words not in the biasing list, and B-WER measures the WER of words in the biasing list. Ideally, the contextual bi-

TABLE I
SENSITIVITY WITH RESPECT TO β . HERE $N = 1000$, AND $tol = 0$. THE AVERAGE NUMBER OF GROUND TRUTH PHRASES IS 2.1 FOR DEV-CLEAN AND 1.6 FOR DEV-OTHER.

| β | dev-clean | # phrases | dev-other | # phrases |
|---------|----------------------|-----------|----------------------|-----------|
| | WER(U-/B-WER) | | WER(U-/B-WER) | |
| ASR | 2.5 (1.6/9.8) | | 6.2 (4.8/19.1) | |
| 0.5 | 1.9 (1.6/4.5) | 3.0 | 5.3 (4.9/9.6) | 2.5 |
| 0.8 | 1.9 (1.6/4.6) | 2.7 | 5.3 (4.9/9.4) | 2.2 |
| 0.9 | 1.9 (1.6/4.1) | 3.2 | 5.2 (4.8/8.8) | 2.8 |
| 0.95 | 1.9 (1.6/4.2) | 3.4 | 5.3 (4.9/9.0) | 2.9 |

TABLE II
SENSITIVITY WITH RESPECT TO tol . HERE $N = 1000$ AND $\beta = 0.9$. THE AVERAGE NUMBER OF GROUND TRUTH PHRASES IS 2.1 FOR DEV-CLEAN AND 1.6 FOR DEV-OTHER.

| tol | dev-clean | # phrases | dev-other | # phrases |
|-------|----------------------|-----------|----------------------|-----------|
| | WER(U-/B-WER) | | WER(U-/B-WER) | |
| 0.0 | 1.9 (1.6/4.1) | 3.2 | 5.2 (4.8/8.8) | 2.8 |
| 1.0 | 1.8 (1.6/3.7) | 5.5 | 5.2 (4.8/7.8) | 5.4 |
| 2.0 | 1.9 (1.7/3.4) | 10.1 | 5.1 (4.9/7.1) | 10.7 |
| 3.0 | 1.9 (1.7/3.1) | 17.9 | 5.2 (5.1/6.7) | 20.2 |
| 4.0 | 2.0 (1.9/2.9) | 30.2 | 5.4 (5.3/6.4) | 35.1 |
| 5.0 | 2.1 (2.0/2.8) | 47.9 | 5.6 (5.6/5.9) | 56.2 |

asing model shall have a lower B-WER without increasing its U-WER significantly. Furthermore, it shall also have minimal B-WER degradation as the number of distractors N increases.

B. Sensitivity with respect to β

First, we perform sensitivity of the discriminative loss weight β used in the biasing training loss (2), for $N = 1000$ distractors and $tol = 0$. We provide the WER/U-WER/B-WER for models trained with a range of β values in Table I, as well as the number of active biasing phrases, i.e., those satisfying the condition (3).

Observe that for a wide range of β , the method works similarly well. The baseline ASR system achieves 9.8% and 19.1% B-WER for dev-clean and dev-other without biasing. With $\beta = 0.9$, we reduce the B-WERs to 4.1% and 8.8% respectively, without degrading the U-WERs. Furthermore, these improvements are achieved with very small numbers of active phrases: the average number of ground truth phrases is 2.1 for dev-clean and 1.6 for dev-other, and our method only keeps on average 3.2 phrases and 2.8 phrases respectively for beam search. Therefore, our method is accurate at removing large amounts of distractors.

Our method fails at $\beta = 1.0$ (with more than 100% WERs, not shown in the table), indicating that log loss is necessary to learn useful scores for biasing. From now on, we use the model trained with $\beta = 0.9$.

C. Sensitivity with respect to tol

We have seen in Table I that our bonus strategy (4) already works well with $tol = 0$. Next we investigate if it is possible to improvement the performance with $tol > 0$, at the cost of including more active phrases during search.

In Table II we provide the results for several values of tol . Observe that indeed a positive value of 1 or 2 leads

TABLE III
BIASING WITH MANUALLY TUNED BONUS. HERE $N = 1000$.

| Manual bonus | dev-clean | dev-other |
|--------------|----------------------|----------------------|
| 0.0 | 2.5 (1.6/9.8) | 6.2 (4.8/19.1) |
| 1.0 | 2.3 (1.6/8.0) | 5.8 (4.8/15.8) |
| 2.0 | 2.1 (1.6/6.3) | 5.5 (4.8/12.9) |
| 3.0 | 1.9 (1.6/5.1) | 5.2 (4.7/9.8) |
| 4.0 | 1.9 (1.6/4.1) | 5.1 (4.8/8.2) |
| 5.0 | 1.9 (1.7/3.3) | 5.0 (4.8/6.9) |
| 6.0 | 1.9 (1.8/2.7) | 5.3 (5.2/6.0) |
| 7.0 | 2.3 (2.3/2.5) | 5.6 (5.6/5.4) |

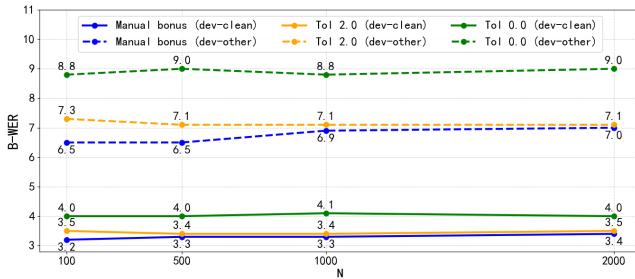


Fig. 2. B-WERs of our method on dev sets, across different values of N .

to improved B-WER without degrading U-WER, while the number of active phrases mildly increases from roughly 3 to about 5 and 10 respectively. Further increasing tol , however, tend to degrade the U-WER.

As a comparison, we exhaustively tune a constant per-token bias bonus, i.e., we replicates the method of [6], and the results are given in Table III. We note that a carefully tuned bonus can work well for search-based biasing: with a per-token bonus of 5, the WERs are matching the best of Table II. However, without the biasing decoder, we can not filter out any phrases and the number of active phrases stays at $N = 1000$ during search, which is 100 times more than that of our approach.

D. Sensitivity with respect to N

Next we investigate the model’s performance across a range of N , the number of distractors. Intuitively, as N increases, there is more chance for the model to be confused and we expect the WERs to deteriorate.

The results of our method with $tol = 0$ and $tol = 2$, as well as the manually tuned constant bonus 5, are shown in Figure 2. Since the fluctuation of WER is small, we plot only the U-WERs, which remains stable across different N .

In Figure 3, we plot the number of active (non-filtered) phrases in our method for varying N . Without biasing decoder, manual bonus cannot filter out phrases and the number of active phrases is N . Observe that the number active phrases grows slowly for both $tol = 0$ and $tol = 2$, indicating that the biasing decoder scores are consistently discriminative.

We plot the the WERs of our method, with both $tol = 0$ and $tol = 2$ in Figure 2. We observe that the WERs stays more or less constant across N . For example, with $tol = 2$, B-WER

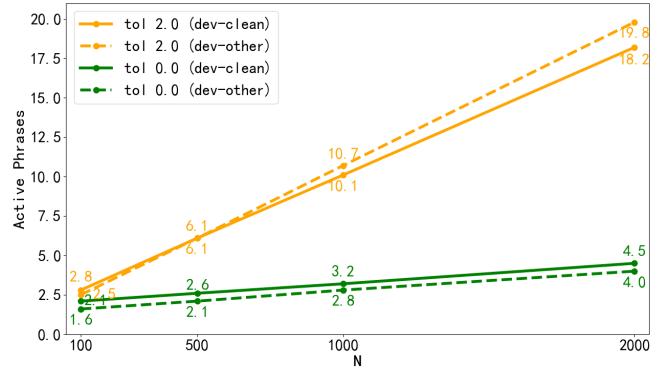


Fig. 3. Number of active phrases of our method on dev sets, across different values of N .

stays at roughly 3.4% for dev-clean, and 7.1% for dev-other. On the other hand, there is slight increase in U-WER, and this is because the model incorrectly bias towards negative phrases. In Figure 3, we plot the number of active phrases across N . Even with $N = 1000$ and $N = 2000$, our model keeps less than 1% of biasing list.

E. Final results

In Table IV, we provide the results of a few methods on the test sets. For our method, we observe that with $tol = 2$ our method performs as well as the best manually tuned bonus 5, but we only search over a minimal amount of active phrases so that the search phase is much more efficient. We consistently achieve over 20% relative improvement on WER, and over 50% relative improvement on B-WER.

We also conduct an experiment where we use biasing decoder solely for filtering (with $tol = 2.0$) but use manually tuned bonus 5 for search; this illustrates the potential of using our biasing decoder for filtering while using another method for biasing. The results are shown in the last row of Table IV. We achieve the best accuracy at $N = 2000$, indicating that filtering not only improves the computational efficiency but also prevents over-biasing on distractors.

To put our results into context, we compare our results with those of a few recent works [5], [19], [20], [22], [24]. Some of these prior works have similar baseline WERs (without biasing) to our Attention+CTC system, yet our biasing results are much stronger than theirs. We also include the results from [23], who achieved better WERs for the Librispeech biasing setup, with a stronger RNN-T recipe. Observe that, while their model-based biasing has an advantage for $N = 100$ and $N = 500$, the performance of shallow fusion remains strong, especially for $N = 1000$. It is interesting future work to combine our filtering strategy with state-of-the-art model-based biasing methods.

V. CONCLUSIONS

We have proposed a neural model for ASR contextual biasing. Our model provides discriminative scores that can be used to effectively filter out unlikely phrases from a given

TABLE IV
BIASING WERS OF DIFFERENT MODELS ON LIBRISPEECH TEST SETS. U-WER/B-WER ARE GIVEN IN BRACKETS. FOR OUR METHOD, THE BEST WERS FOR EACH N ARE SHOWN IN BOLDFACE.

| Method | test-clean | | | | test-other | | | |
|---|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
| | $N = 100$ | $N = 500$ | $N = 1000$ | $N = 2000$ | $N = 100$ | $N = 500$ | $N = 1000$ | $N = 2000$ |
| [5], RNN-T | | 3.7 (2.4/14.1) | | | | 9.6 (7.2/30.6) | | |
| DB-RNNNT s3 | 2.8 (2.2/7.4) | 2.9 (2.3/8.1) | 3.0 (2.3/8.5) | 3.0 (2.3/8.9) | 8.1 (7.0/17.7) | 8.3 (7.1/19.1) | 8.5 (7.1/20.5) | 8.8 (7.3/21.8) |
| [20], Attention | | 5.1 (3.9/14.1) | | | | 8.8 (6.6/27.9) | | |
| Biasing with BPB | 2.8 (2.3/6.0) | 3.2 (2.7/7.0) | 3.5 (3.0/7.7) | | 5.6 (4.9/12.0) | 6.3 (5.5/13.5) | 7.3 (6.4/15.8) | |
| [19], Transducer | | 2.8 (1.9/10.2) | | | | 6.6 (4.9/22.0) | | |
| Biasing with CA + GA-CE | 2.2 (1.8/5.1) | | 2.4 (1.9/6.4) | | 5.4 (4.7/12.2) | | 6.0 (5.0/15.3) | |
| [22], Transducer + CTC | | 2.9 (1.6/13.0) | | | | 7.2 (4.8/28.1) | | |
| Intermediate + Joint decoding | 2.3 (1.5/8.6) | 2.8 (1.6/11.2) | 2.8 (1.7/12.4) | | 6.4 (4.7/21.2) | 7.1 (4.9/26.3) | 7.3 (5.0/27.9) | |
| [24], Attention | | 3.4 (2.0/14.4) | | | | 8.8 (6.1/32.5) | | |
| Biasing | 2.3 (1.8/6.5) | 2.5 (1.9/7.4) | 2.7 (2.0/8.2) | | 6.8 (5.8/15.6) | 7.5 (6.2/18.2) | 7.7 (6.2/20.3) | |
| [23], Transducer | | 2.2 (1.3/9.7) | | | | 5.2 (3.3/21.8) | | |
| Shallow Fusion Biasing | 1.5 (1.2/4.0) | 1.6 (1.3/4.2) | 1.6 (1.3/4.3) | | 4.0 (3.3/10.5) | 4.1 (3.3/11.1) | 4.3 (3.5/11.2) | |
| Neural biasing, Intermediate layers + text perturbation | 1.5 (1.1/4.7) | 1.7 (1.2/5.8) | 1.9 (1.3/6.6) | | 3.7 (3.1/9.2) | 4.0 (3.2/11.4) | 4.4 (3.4/13.5) | |
| Ours, Attention + CTC (without biasing) | | 2.7 (1.7/11.1) | | | | 6.3 (4.3/23.3) | | |
| Manual bonus (5.0) (no filtering) | 1.9 (1.6/4.4) | 2.0 (1.7/4.6) | 2.0 (1.7/4.6) | 2.2 (1.8/4.9) | 4.6 (4.1/9.2) | 4.7 (4.2/9.4) | 4.8 (4.2/9.6) | 4.9 (4.4/9.9) |
| Biasing decoder (tol 0.0) | 2.0 (1.6/5.2) | 2.1 (1.7/5.2) | 2.1 (1.7/5.2) | 2.1 (1.7/5.3) | 4.9 (4.2/11.4) | 4.9 (4.2/11.4) | 5.0 (4.2/11.6) | 5.0 (4.3/11.5) |
| Biasing decoder (tol 2.0) | 2.0 (1.6/4.8) | 2.0 (1.7/4.6) | 2.0 (1.7/4.6) | 2.1 (1.8/4.6) | 4.7 (4.1/9.6) | 4.7 (4.2/9.4) | 4.8 (4.3/9.3) | 5.0 (4.5/9.5) |
| Biasing decoder filtering (tol 2.0) + manual bonus (5.0) | 1.9 (1.6/4.5) | 2.0 (1.6/4.6) | 2.0 (1.7/4.6) | 2.1 (1.7/4.9) | 4.6 (4.1/9.3) | 4.7 (4.1/9.6) | 4.8 (4.2/9.7) | 4.8 (4.2/9.7) |

biasing list, resulting in significant computational savings. When the scores are used to compute bonus for shallow-fusion biasing, they match the best manually tuned per-token bonus. We achieve about 50% relative improvement in B-WER on the Librispeech biasing setup, while keep only 1% of original phrases during search. There are a few future directions. First, when constructing biasing phrases on the fly for training, we could remove frequent words to better simulate the scenario at testing. Second, we shall test our model’s filtering capability with other systems that perform model-based biasing than shallow-fusion biasing, and with LLM-based ASR systems. Third, our approach uses an ASR decoder to provide audio-conditioned biasing scores, and it would be interesting to extend it to the streaming setup.

REFERENCES

- [1] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks,” in *International Conference on Machine learning*, 2006.
- [2] A. Graves, “Sequence transduction with recurrent neural networks,” in *ICML Workshop on Representation Learning*, 2012.
- [3] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attention-based models for speech recognition,” *CoRR*, 2015.
- [4] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, “Listen, attend and spell: A neural network for large vocabulary conversational speech recognition,” in *ICASSP*, 2016.
- [5] D. Le, M. Jain, G. Keren, S. Kim, Y. Shi, J. Mahadeokar, J. Chan, Y. Shangguan, C. Fuegen, O. Kalinli, Y. Saraf, and M. L. Seltzer, “Contextualized streaming end-to-end speech recognition with trie-based deep biasing and shallow fusion,” in *Interspeech*, 2021.
- [6] W. Wang, Z. Wu, D. Caseiro, T. Munkhdalai, K. C. Sim, P. Rondon, G. Pundak, G. Song, R. Prabhavalkar, Z. Meng, D. Zhao, T. Sainath, Y. He, and P. M. Mengibar, “Contextual biasing with the Knuth-Morris-Pratt matching algorithm,” in *Interspeech*, 2024.
- [7] P. Aleksic, M. Ghodsi, A. Michaely, C. Allauzen, K. Hall, B. Roark, D. Rybach, and P. Moreno, “Bringing contextual information to google speech recognition,” in *Interspeech*, 2015.
- [8] K. Hall, E. Cho, C. Allauzen, F. Beaufays, N. Coccaro, K. Nakajima, M. Riley, B. Roark, D. Rybach, and L. Zhang, “Composition-based on-the-fly rescoring for salient n-gram biasing,” in *Interspeech*, 2015.
- [9] M. Mohri, F. Pereira, and M. Riley, “Weighted finite-state transducers in speech recognition,” *Computer Speech and Language*, 2002.
- [10] D. Zhao, T. N. Sainath, D. Rybach, P. Rondon, D. Bhatia, B. Li, and R. Pang, “Shallow-Fusion End-to-End Contextual Biasing,” in *Interspeech*, 2019.
- [11] 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*. 2017.
- [12] G. Pundak, T. N. Sainath, R. Prabhavalkar, A. Kannan, and D. Zhao, “Deep context: end-to-end contextual speech recognition,” in *IEEE spoken language technology workshop (SLT)*, 2018.
- [13] M. Jain, G. Keren, J. Mahadeokar, and Y. Saraf, “Contextual rnn-t for open domain asr,” in *Interspeech*, 2020. [Online]. Available: <https://api.semanticscholar.org/CorpusID:219401479>
- [14] T. Munkhdalai, K. C. Sim, A. Chandorkar, F. Gao, M. Chua, T. Strohman, and F. Beaufays, “Fast contextual adaptation with neural associative memory for on-device personalized speech recognition,” *arXiv preprint arXiv:2110.02220*, 2021.
- [15] F.-J. Chang, J. Liu, M. Radfar, A. Mouchtaris, M. Omologo, A. Rastrow, and S. Kunzmann, “Context-aware transformer transducer for speech recognition,” in *Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2021.
- [16] M. Han, L. Dong, Z. Liang, M. Cai, S. Zhou, Z. Ma, and B. Xu, “Improving end-to-end contextual speech recognition with fine-grained contextual knowledge selection,” in *ICASSP*, 2022.
- [17] K. M. Sathyendra, T. Muniyappa, F.-J. Chang, J. Liu, J. Su, G. P. Strimel, A. Mouchtaris, and S. Kunzmann, “Contextual adapters for personalized speech recognition in neural transducers,” in *ICASSP*, 2022.
- [18] Z. Meng, Z. Wu, R. Prabhavalkar, C. Peyser, W. Wang, N. Chen, T. N. Sainath, and B. Ramabhadran, “Text injection for neural contextual biasing,” *Interspeech*, 2024.
- [19] J. Tang, K. Kim1, S. Shon, F. Wu, P. Sridhar, and S. Watanabe, “Improving ASR contextual biasing with guided attention,” in *ICASSP*, 2024.
- [20] Y. Sudo, M. Shakeel, Y. Fukumoto, Y. Peng, and S. Watanabe., “Contextualized automatic speech recognition with attention-based bias phrase boosted beam search,” in *ICASSP*, 2024.
- [21] Y.-C. Wang, L.-T. Pai, B.-C. Yan, H.-W. Wang, C.-H. Lin, and B. Chen, “An effective context-balanced adaptation approach for long-tailed speech recognition,” in *SLT*, 2024.
- [22] M. Shakeel, Y. Sudo, Y. Peng, and S. Watanabe., “Contextualized end-to-end automatic speech recognition with intermediate biasing loss,” in *Interspeech*, 2024.
- [23] R. Huang, M. Yarmohammadi, S. Khudanpur, and D. Povey, “Improving neural biasing for contextual speech recognition by early context injection and text perturbation,” in *Interspeech*, 2024.
- [24] M. Fang, T. Wei, K. Guo, Z. Zhuang, Y. Shi, N. Cheng, S. Wang, and J. Xiao, “Improving contextual asr with enhanced phrase-level representation based on mctc loss,” in *ICASSP*, 2025.
- [25] D. Le, G. Keren, J. Chan, J. Mahadeokar, C. Fuegen, and M. L. Seltzer, “Deep shallow fusion for rnn-t personalization,” in *SLT*, 2021.
- [26] G. Sun, C. Zhang, and P. C. Woodland, “Tree-constrained pointer generator for end-to-end contextual speech recognition,” in *ASRU*, 2021.
- [27] E. Lakomkin, C. Wu, Y. Fathullah, O. Kalinli, M. L. Seltzer, and C. Fuegen, “End-to-end speech recognition contextualization with large language models,” 2023. [Online]. Available: <https://arxiv.org/abs/2309.10917>
- [28] M. Wang, W. Han, I. Shafran, Z. Wu, C.-C. Chiu, Y. Cao, N. Chen, Y. Zhang, H. Soltau, P. K. Rubenstein, L. Zilka, D. Yu, G. Pundak, N. Siddhartha, J. Schalkwyk, and Y. Wu, “Slm: Bridge the thin gap between speech and text foundation models,” in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2023.
- [29] Z. Chen, H. Huang, A. Andrusenko, O. Hrinchuk, K. C. Puvvada, J. Li, S. Ghosh, J. Balam, and B. Ginsburg, “Salm: Speech-augmented language model with in-context learning for speech recognition and translation,” in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024.
- [30] Z. Kong, A. Goel, R. Badlani, W. Ping, R. Valle, and B. Catanzaro, “Audio flamingo: A novel audio language model with few-shot learning and dialogue abilities,” in *International Conference on Machine Learning*. PMLR, 2024.
- [31] T. Hori, M. Kocour, A. Haider, E. McDermott, and X. Zhuang, “Delayed fusion: Integrating large language models into first-pass decoding in end-to-end speech recognition,” 2025.
- [32] C.-H. H. Yang, Y. Gu, Y.-C. Liu, S. Ghosh, I. Bulyko, and A. Stolcke, “Generative speech recognition error correction with large language models and task-activating prompting,” in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, Dec. 2023, p. 1–8. [Online]. Available: <http://dx.doi.org/10.1109/ASRU57964.2023.10389673>
- [33] G. Song, Z. Wu, G. Pundak, A. Chandorkar, K. Joshi, X. Velez, D. Caseiro, B. Haynor, W. Wang, N. Siddhartha, P. Rondon, and K. C. Sim, “Contextual spelling correction with large language models,” in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2023, pp. 1–8.
- [34] Z. Lei, X. Na, M. Xu, E. Pusateri, C. V. Gysel, Y. Zhang, S. Han, and Z. Huang, “Contextualization of asr with llm using phonetic retrieval-based augmentation,” 2024. [Online]. Available: <https://arxiv.org/abs/2409.15353>
- [35] C. Sun, Z. Ahmed, Y. Ma, Z. Liu, L. Kabela, Y. Pang, and O. Kalinli, “Contextual biasing of named-entities with large language models,” 2023. [Online]. Available: <https://arxiv.org/abs/2309.00723>
- [36] A. Flemotomas, R. Hsiao, P. Swietojanski, T. Hori, D. Can, and X. Zhuang, “Optimizing contextual speech recognition using vector quantization for efficient retrieval,” in *SLT*, 2024.
- [37] D. Povey and P. Woodland, “Minimum phone error and I-smoothing for improved discriminative training,” in *ICASSP*, 2002.
- [38] K. Vesely, A. Ghoshal, L. Burget, and D. Povey, “Sequence-discriminative training of deep neural networks.” in *Interspeech*, 2013.
- [39] R. Prabhavalkar, T. N. Sainath, Y. Wu, P. Nguyen, Z. Chen, C.-C. Chiu, and A. Kannan, “Minimum word error rate training for attention-based sequence-to-sequence models,” in *ICASSP*, 2018.
- [40] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, “Librispeech: An ASR corpus based on public domain audio books,” in *ICASSP*, 2015.
- [41] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, “Hybrid CTC/Attention architecture for end-to-end speech recognition,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1240–1253, 2017.
- [42] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, “ESPNet: End-to-end speech processing toolkit,” in *Interspeech*, 2018.