RECAST: Retrieval-Augmented Contextual ASR via Decoder-State Keyword Spotting

Anonymous ACL submission

Abstract

Contextual biasing in ASR systems is critical for recognizing rare, domain-specific terms but becomes impractical with large keyword dictionaries due to prompt size and latency constraints. We present RECAST—a lightweight retrieval-augmented approach that repurposes decoder states of a pretrained ASR model to retrieve relevant keywords without requiring audio exemplars. RECAST introduces a contrastively trained retriever that aligns decoder-state embeddings with textual keyword representations, enabling fast token-level retrieval over large dictionaries. Retrieved keywords are ranked and formatted into a prompt to guide a downstream speech language model. Trained solely on LibriSpeech and evaluated on out-of-domain benchmarks covering up to 4,000 keywords across diverse domains, RECAST consistently outperforms full-list prompt biasing and strong phonetic/text baselines. It achieves up to 54.3% relative reduction in entity WER and 41.3% overall WER improvement over the baseline, along with up to 2.5× higher recall in challenging settings. Furthermore, RECAST remains effective for diverse languages such as Hindi, demonstrating its scalability, language-agnostic design, and practicality for real-world contextual ASR.

1 Introduction

Contextual biasing in ASR via domain-specific keywords improves recognition of rare terms but fails to scale: large keyword inventories degrade transcription quality and exceed prompt-size constraints (Liu et al., 2020; Gourav et al., 2021; Sun et al., 2021; Mittal et al., 2023a). While recent advances in speech language models (Radford et al., 2023; Abouelenin et al., 2025; Saon et al., 2025) and LLM-based error correction (Li et al., 2024a; Ma et al., 2025) enable effective keyword prompting (Peng et al., 2023), domains like medicine or finance involve thousands of rare entities, making full-list inclusion infeasible. This raises a central challenge: how can we efficiently identify a small, relevant subset of keywords for any given utterance?

Unlike prior contextual biasing methods that struggle to scale with large keyword dictionaries—such as attention-based selection over the full list (Jain et al., 2020; Sun et al., 2023), or keyword spotting approaches that require audio exemplars and perform independent searches for each keyword (Navon et al., 2024; Li et al., 2024b)—we propose a unified and scalable solution: RECAST (***Re**trieval-Augmented **C**ontextual **A**SR via Decoder-**St**ate Keyword Spotting*). RECAST leverages the final decoder state of a pretrained encoder–decoder ASR model as a query to retrieve relevant keywords directly from a large text-only inventory, without modifying the ASR backbone or relying on audio examples.

Beyond the challenge of large keyword lists, real-world utterances are often long, with target entities appearing anywhere in the audio. This makes it crucial not only to retrieve relevant keywords but also to localize their aligned positions. To this end, RECAST introduces two lightweight modules: a unidirectional LSTM-based keyword encoder and a feed-forward decoder-state projector, trained with a contrastive loss to align subword-level decoder states with keyword token embeddings in a shared retrieval space. At inference time, each decoder state is treated as a fine-grained retrieval query into a precomputed keyword index. Retrieved token-level matches are aggregated into candidate keyword spans and ranked using a position-aware scoring function to form a top-$\mathcal{K}$ shortlist. This shortlist is then incorporated into the decoder prompt of a downstream speech language model, enabling fluent and grounded contextual biasing without modifying the ASR model.

We validate RECAST by training the retriever solely on LibriSpeech and evaluating it on diverse

out-of-domain benchmarks with up to 4,000 keywords across domains such as locations, names, and medical terms. Despite no exposure to these entities during training, RECAST consistently improves retrieval and ASR performance. It outperforms strong retrieval-style baselines—common in information retrieval but underexplored in ASR—which rely on ASR hypotheses and apply fuzzy matching (e.g., BM25) or phonetic algorithms like Soundex (Knuth, 1973) and Double Metaphone (Philips, 2000). In contrast, RECAST operates directly over decoder states and text keywords, enabling seamless generalization to languages such as Hindi, where phonetic methods tailored for English—like Soundex and Double Metaphone—are ineffective. It integrates with existing speech language models, offering low-latency inference and practical gains for real-world contextual ASR.

Our contributions are threefold: **(1)** a contrastively trained retriever that aligns ASR decoder states with keyword token embeddings for efficient large-vocabulary retrieval; **(2)** a token-level tracking and ranking algorithm that builds contextually relevant shortlists in real time; and **(3)** robust generalization across domains and languages, improving accuracy and latency without modifying the underlying ASR backbone.

2 Related Work and Background

Keyword Spotting (KWS). In keyword spotting the goal is to detect specific keywords within a speech signal. Early systems performed keyword spotting via large vocabulary continuous speech recognition with lattice-based search (Mamou et al., 2007) or HMM-based keyword-filler models (Rohlicek et al., 1989). Query-by-example methods using DTW (Zhang and Glass, 2009) eliminated the need for transcription but were sensitive to speaker and channel variability. The advent of deep learning introduced frame-level classifiers using DNNs and CNNs for small-vocabulary tasks (Arik et al., 2017; Tucker et al., 2016), while embedding-based methods—such as Siamese networks (Settle and Livescu, 2016) and audio-text dual encoders (Kamper et al., 2016)—enabled open-vocabulary detection for short audio but struggled with long audio. Recent works apply self-supervised speech models like wav2vec 2.0 (Baevski et al., 2020) and HuBERT (Hsu et al., 2021) to spoken language understanding tasks—classifying short utterances into discrete labels (Lugosch et al., 2019; Wang et al., 2021; Lim et al., 2023)—but they struggle to generalize to long-form audio and unseen keywords. Large-scale pretrained models like Whisper (Radford et al., 2023) have also been adapted for KWS. Notably, Li et al.(Li et al., 2023, 2024b) use CNNs on cross-attention similarity matrices, and AdaKWS(Navon et al., 2024) applies keyword-guided normalization in transformers for open-vocabulary spotting.

However, most methods still process keywords independently, leading to linear scaling in computation and latency with large dictionaries.

Contextual ASR. Contextual ASR integrates external information—such as dynamic vocabularies or keywords—into the decoding process to improve recognition. Traditional approaches combine an external language model (LM) via shallow (Ravi et al., 2020; Liu et al., 2020; Gourav et al., 2021) or deep fusion (Le et al., 2021b,a), but fixed interpolation weights can misbias rare terms and require LM retraining for new domains (Mittal et al., 2023a); moreover, they are ill-suited for keyword dictionaries that lack contextual structure. Attention-based methods allow ASR models to attend over keyword lists by embedding context tokens (Jain et al., 2020; Huber et al., 2021; Sun et al., 2023; Munkhdalai et al., 2023), but they struggle to scale or generalize to out-of-domain dictionaries. Inference-time methods inject keyword lists into beam search via class- or tree-based biasing (Williams et al., 2018; Huang et al., 2020; Sun et al., 2021), or use synthesized keyword exemplars for on-the-fly biasing (Mittal et al., 2023b), though these can lead to disfluent outputs. Prompt-based strategies, inspired by in-context learning, prepend keywords to speech LLM inputs (Peng et al., 2023; Chang et al., 2024; Yang et al., 2024), enabling biasing without model updates, but are limited by prompt length and do not scale to large dictionaries.

Unlike prior methods that require keyword-level audio exemplars or rely on ASR hypotheses for retrieval, our approach learns a contrastive alignment between ASR decoder states and text-only keyword embeddings. This enables scalable retrieval from large keyword inventories without modifying the ASR backbone or requiring audio templates.

Background: Encoder-Decoder ASR. Encoder-decoder architectures, as employed in state-of-the-art ASR systems such as Whisper (Radford et al., 2023), process audio and generate transcriptions us-

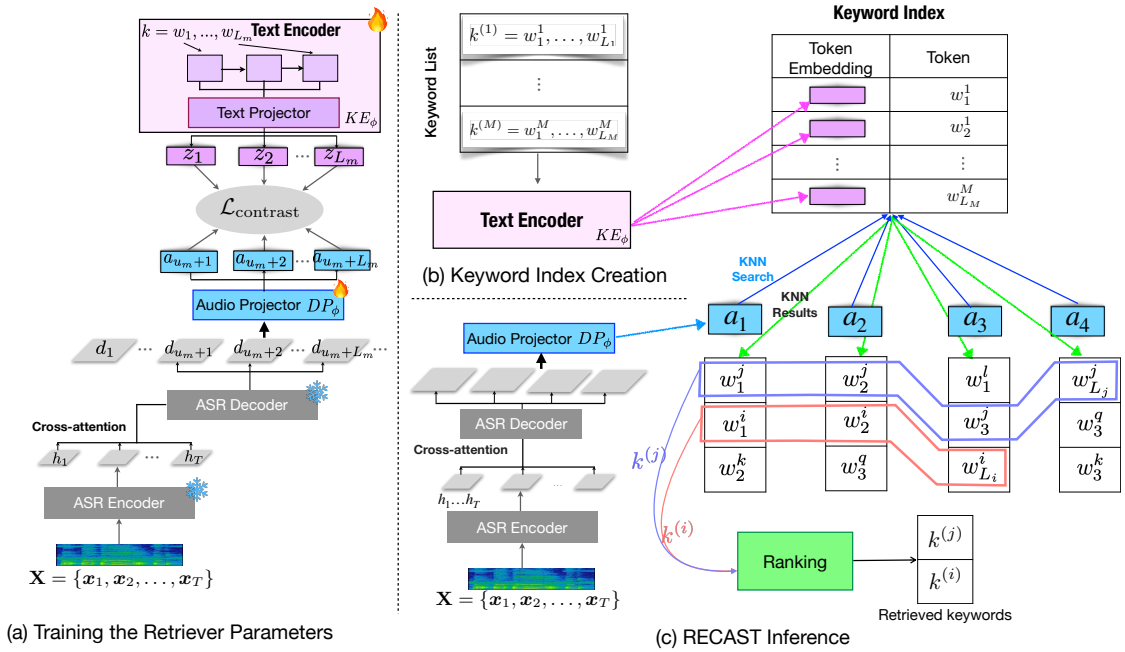

Figure 1: Overview of RECAST. (a) Training - We train a retrieval model with contrastive loss to align keyword-level audio and text representations in a shared embedding space. (b) Indexing - Encoded token-level keyword embeddings are stored in an index for efficient similarity search. (c) Inference - At each decoding step, token-level KNN matches are performed using the audio embedding, and continuous token spans forming keywords are extracted and ranked based on similarity.

ing a sequence-to-sequence framework with attention mechanisms. The encoder first transforms the input speech sequence $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$ into a sequence of latent representations $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_T$ using architectures such as RNNs (Hochreiter and Schmidhuber, 1997), Transformers (Vaswani et al., 2017), or Conformers (Gulati et al., 2020).

The decoder generates the output sequence autoregressively. At each step u, it attends to previous tokens $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{u-1}$ and performs cross-attention over encoder states $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_T$ to compute a context vector $\mathbf{e}_u$. This is combined with self-attention to update the decoder state $\mathbf{d}_u$, which is then used to compute the output distribution $P_\theta(y|\mathbf{d}_u, \boldsymbol{h})$ via a softmax layer. Inference typically uses beam search to find the most likely sequence.

3 RECAST

We formalize the retrieval problem as follows. Let $D = \{\tilde{\boldsymbol{y}}^1, \ldots, \tilde{\boldsymbol{y}}^N\}$ denote a large dictionary of keywords, where N is large. Given an audio utterance $\boldsymbol{x}$ to be transcribed, the goal is to efficiently retrieve a small subset of at most $\mathcal{K}$ keywords from D that are likely to appear somewhere in $\boldsymbol{x}$.

The key insight behind RECAST is that the decoder of a pretrained encoder–decoder ASR model uses cross-attention to focus on relevant segments of the input audio when generating each text token. These decoder states implicitly encode fine-grained audio–text alignment. RECAST leverages this property by introducing a lightweight extension to the decoder that repurposes these intermediate states to retrieve relevant keywords from D. The architecture and overall workflow of this retrieval-augmented framework are illustrated in Figure 1.

Overview of RECAST The two new trainable components we introduce are: a keyword-encoder KE_ϕ and a decoder-state projector DP_ϕ with parameters ϕ. The keyword-encoder KE_ϕ converts any keyword $\boldsymbol{k}$ to a sequence of vectors $S(\boldsymbol{k}) = \boldsymbol{z}_1, \ldots, \boldsymbol{z}_{l_k}$ where l_k denotes the number of tokens into which the ASR tokenizer would decompose $\boldsymbol{k}$. The decoder-state projector DP_ϕ converts at each decoder step t, the last hidden vector of the decoder $\boldsymbol{d}_t$ into an audio snippet embedding $\boldsymbol{a}_t$. More details about estimating ϕ are in Section 3.1. During domain-specific deployment, first given any arbitrary list of keywords D in that domain, we first input each keyword $\boldsymbol{k} \in D$ to KE_ϕ and get its corresponding vector sequence. This vector sequence is inserted into a vector search data struc-

ture $\mathcal{I}$. More details about constructing such an index appear in Section 3.2. Once the index is created, for each subsequent transcription task on an audio $\boldsymbol{x}$, we retrieve a short list of at most $\mathcal{K}$ keyword matches from D. This retrieval is performed auto-regressively where at each step t, the decoder state is used as search key to probe into the $\mathcal{I}$ for potential matches. The matches across multiple consecutive decoding steps are stitched together to obtain the short list $\mathcal{K}$. More details of this retrieval process appears in Section 3.3. Finally, the retrieved shortlist is used for contextual biasing the transcription of $\boldsymbol{x}$ into the text as described in Section 3.4.

3.1 Training the Retriever Parameters

To learn fine-grained associations between spoken content and its textual counterpart, we train a retrieval model that aligns keyword-level audio and text representations using a contrastive loss. Given a paired example $(\mathbf{x}, \mathbf{y})$, where $\mathbf{x} = x_1, \ldots, x_T$ denotes the input audio and $\mathbf{y} = y_1, \ldots, y_U$ the corresponding transcript tokens, the model is trained to bring matching audio-text keyword representations closer in a shared embedding space.

Keyword Extraction. From each transcript $\mathbf{y}$, we extract a set of salient keywords $\{\mathbf{k}^{(1)}, \ldots, \mathbf{k}^{(r)}\}$ using KeyBERT. Each keyword $\mathbf{k}^{(m)}$ may be a single word or a multi-word phrase. We tokenize each of these using the ASR tokenizer. Let the resulting sequence of subword tokens be $\texttt{tok}(\mathbf{k}^{(m)}) = [w_1^{(m)}, \ldots, w_{L_m}^{(m)}]$. Each keyword corresponds to a span $y_{u_m}, \ldots, y_{u_m+L_m}$ in the transcript where u_m denotes the start index of the m-th keyword within the tokenized transcript $\mathbf{y}$.

Text Representation. We represent each keyword $\mathbf{k}^{(m)}$ by embedding its token sequence $[w_1^{(m)}, \ldots, w_{L_m}^{(m)}]$ through a unidirectional LSTM encoder and then linearly projecting to obtain the keyword embedding:

$$\boldsymbol{z}_1^{(m)}, \ldots, \boldsymbol{z}_{L_m}^{(m)} = \mathrm{KE}_\phi(w_1^{(m)}, \ldots, w_{L_m}^{(m)}) \quad (1)$$

Audio Representation. The corresponding audio $\mathbf{x}$ is processed through the encoder-decoder ASR model. At the decoding steps that produce tokens $w_1^{(m)}, \ldots, w_{L_m}^{(m)}$, we extract the decoder hidden states $\{\mathbf{d}_{\ell+u_m}^{(m)}\}_{\ell=1}^{L_m}$. Each decoder state $\mathbf{d}_{\ell+u_m}^{(m)} \in \mathbb{R}^{d_{\text{audio}}}$ is passed through a linear projection layer $\mathrm{DP}_\phi : \mathbb{R}^{d_{\text{audio}}} \rightarrow \mathbb{R}^{d_{\text{text}}}$ to obtain audio embeddings:

$$\mathbf{a}_{\ell+u_m}^{(m)} = \mathrm{DP}_\phi(\mathbf{d}_{\ell+u_m}^{(m)}) \quad (2)$$

Training Objective. To align audio and text representations, we employ a token-level contrastive loss where for each keyword token $w_\ell^{(m)}$, we increase the similarity between its contextual text embedding $\boldsymbol{z}_\ell^{(m)}$ and the audio embedding $\mathbf{a}_{\ell+u_m}^{(m)}$ at position $\ell + u_m$ by contrasting with two kinds of negatives: (1) In-batch negatives, i.e., mismatched keyword-audio pairs $(\boldsymbol{z}_\ell^{(m)}, \mathbf{a}_j^{(n)})$, and (2) Hard-negative keyword tokens $\mathcal{N}(w_\ell^{(m)})$ mined as described in Section 3.1.1 for greater contextual awareness of text embeddings. The overall training objective is:

$$\max_\phi \sum_{(\boldsymbol{x},\boldsymbol{y})} \sum_{\mathbf{k}^{(m)} \in \boldsymbol{y}} \sum_{\ell=1}^{L_m} \log \frac{\mathrm{S}(\boldsymbol{z}_\ell^{(m)}, \mathbf{a}_{\ell+u_m}^{(m)})}{\sum_{(j,n)} \mathrm{S}(\boldsymbol{z}_\ell^{(m)}, \mathbf{a}_j^{(n)}) + \sum_{\boldsymbol{z} \in \mathcal{N}(w_\ell^{(m)})} \mathrm{S}(\boldsymbol{z}, \mathbf{a}_{\ell+u_m}^{(m)})} \quad (3)$$

where $\mathrm{S}(\cdot,\cdot) = \exp(\mathrm{cosine}(\cdot,\cdot)/\tau)$ denotes exponentiation of cosine similarity scaled by a temperature hyper-parameter τ. This ensures that embeddings of corresponding audio and text tokens for each keyword are embedded close together, facilitating effective cross-modal retrieval.

3.1.1 Hard Negative Mining

To enhance the discriminative power of the learned embeddings and encourage sensitivity to left-context in keyword representations, we incorporate hard negatives during training. For each token $w_\ell^{(m)}$ within a keyword $\mathbf{k}^{(m)}$, we construct additional negative examples $\mathcal{N}(w_\ell^{(m)})$ that differ only in contextual prefix, thereby enforcing context-aware alignment. The first type of hard negative is constructed by identifying another keyword $\mathbf{k}^{(n)}$ that ends in the same token $w_\ell^{(m)}$ but is preceded by a different left context. We extract the contextual embedding $\mathbf{z}_\ell^{(n)}$ of this token (via Equation 1) and include it in $\mathcal{N}(w_\ell^{(m)})$ as a hard negative. This penalizes the model if it aligns representations of identical tokens that appear in different contextual settings. The second type of hard negative in $\mathcal{N}(w_\ell^{(m)})$ is obtained by stripping the left context of the keyword and recomputing the representation

of the token $w_\ell^{(m)}$ in isolation. This ensures that the model respects the full contextual information present during actual decoding.

3.2 Keyword Index Creation

Once the retrieval parameters are trained, RECAST enables contextual biasing for any set of keywords through a one-time index creation process. This process involves tokenizing each keyword using the ASR tokenizer, encoding the resulting subword tokens with the keyword encoder (KE_ϕ) to produce contextualized embeddings, and storing them in a list E along with a parallel list K that records the originating keyword and token position. A kNN index $\mathcal{I}$ is then constructed from E and K, enabling efficient similarity-based retrieval of token-level keyword embeddings during inference. The full procedure is provided in Appendix D.

3.3 RECAST Inference

At inference time, given an input audio x and index $\mathcal{I}$, RECAST interleaves greedy ASR decoding with token-level keyword retrieval to identify the $\mathcal{K}$ keywords appearing in x.

The inference algorithm begins by initializing the decoded transcript prefix $\hat{y}$ to the empty sequence and the decoder state d_0 to its designated initial value (line 3). Two collections are initialized: B, for active keyword hypotheses, and C, for completed keyword matches.

During each decoding step (lines 6–32), the model first performs greedy ASR decoding to extend the transcript by one token and updates the decoder state accordingly (lines 7–9). The updated state is then processed by the audio projector to obtain an embedding a_u, which is used to query the kNN index for its $\hat{\mathcal{T}}$ nearest neighbors and associated distances (lines 10–11). Each hypothesis in B is then updated: it verifies whether its next expected token index remains within the length of its corresponding keyword and whether its error count is below the threshold $E_{\max}$. If the token at position ℓ is not among the retrieved kNN neighbors, the hypothesis incurs an error; otherwise, the error count remains unchanged. In all cases, the current similarity score is appended to the hypothesis's score list before advancing its suffix index. We allow up to $E_{\max}$ such mismatches to accommodate discrepancies between keyword tokenization and ASR output (lines 14–26). Hypotheses that complete their keyword are added to C, while the remaining hypotheses persist to the next iteration.

Algorithm 1 RECAST Inference

1: **Input:** Encoder Context $\boldsymbol{h}$, Max Steps U, kNN Index $\mathcal{I}$, Audio Projector DP_ϕ, Neighbors $\hat{\mathcal{T}}$, Output Count $\mathcal{K}$
2: **Output:** transcript $\hat{y}$, top-$\mathcal{K}$ keywords
3: $\hat{y} \leftarrow \langle\rangle, d_0 \leftarrow \text{init}$ ▷ Initial decoder state
4: $B \leftarrow \emptyset$ ▷ Active hypotheses
5: $C \leftarrow \emptyset$ ▷ Completed hypotheses
6: **for** $u = 1$ **to** U **do**
7: $(p(\cdot),\, d_u) \leftarrow P_\theta(\hat{y} \mid d_{u-1}, \boldsymbol{h}, \hat{y})$ ▷ Decoding step
8: $y^* \leftarrow \arg\max p(\cdot)$
9: $\hat{y} \mathrel{+}= y^*$
10: $a_u \leftarrow \text{DP}_\phi(d_u)$
11: $(\text{dists}, \{(m_j, \ell_j)\}_{j=1}^{\hat{\mathcal{T}}}) \leftarrow \mathcal{I}.\text{search}(a_u, \hat{\mathcal{K}})$ ▷ KNN
12: $B_{\text{next}} \leftarrow \emptyset$
13: **for all** $(kw, m, \ell, err, dist_list) \in B$ **do**
14: **if** $err < E_{\max}$ **and** $\ell \leq |\boldsymbol{k}^{(m)}|$ **then**
15: **if** $(m, \ell) \notin \{(m_j, \ell_j)\}_{j=1}^{\hat{\mathcal{K}}}$ **then**
16: $err \mathrel{+}= 1$ ▷ Not in $\hat{\mathcal{K}}$, Count as Error
17: **end if**
18: $\text{sim} \leftarrow e_u \cdot \mathcal{I}.\text{get_embedding}(m, \ell)$
19: $dist_list.\text{append}(\text{sim})$
20: $\ell \mathrel{+}= 1$
21: **if** $\ell > |\boldsymbol{k}^{(m)}|$ **then** ▷ End of Keyword
22: $C.\text{append}((kw, dist_list))$
23: **else**
24: $B_{\text{next}}.\text{append}((kw, m, \ell, err, dist_list))$
25: **end if**
26: **end if**
27: **end for**
28: $B \leftarrow B_{\text{next}}$
29: **for** $j = 1$ **to** $\hat{\mathcal{K}}$ **do** ▷ Spawn new hypotheses
30: $B.\text{append}((\boldsymbol{k}^{(m_j)}, m_j, \ell_j + 1, \ell_j, [\text{dists}[j]]))$
31: **end for**
32: **end for**
33: $\text{top-}\mathcal{K} \leftarrow \text{DistanceRanker}(C, \mathcal{K})$
34: **return** $(\hat{y}, \text{top-}\mathcal{K})$

Concurrently, the algorithm spawns new hypotheses for each retrieved nearest neighbor (lines 29–31). Each new hypothesis encodes the keyword from the retrieved pointer, initializes its suffix index one position beyond the matched token index, seeds its error count to account for unmatched prefix tokens, and begins its similarity list with the retrieved distance. After completing U steps, all entries in C are passed to `DistanceRanker`, which selects the top-$\mathcal{K}$ keywords. The final output consists of the full transcript $\hat{y}$ alongside these ranked keywords (line 33).

Ranking Algorithm for RECAST. To identify relevant keywords for contextual ASR, we rank candidates by combining similarity, informativeness, and transcript coverage. Each keyword is first scored using average similarity between its token embeddings and decoder states (via cosine or dot product), then scaled by keyword length to favor

longer (Wu et al., 2016), more informative terms:

$$\text{scaled similarity} = \text{average similarity} \times |\text{tokens}|^{0.6}$$

To ensure broad coverage, we enforce positional diversity by selecting high-scoring keywords from distinct transcript regions. Remaining slots are filled by top-scoring keywords regardless of position. This strategy favors contextually aligned, content-rich keywords while maintaining positional spread. Ablations in §5.2.1 explore the impact of length scaling and diversity constraints.

3.4 Contextual ASR with RECAST

We use RECAST to retrieve relevant keywords and inject them into the decoder prompt of Whisper or other speech LLMs (e.g., Phi-4), following prior work (Li et al., 2024b; Shamsian et al., 2024). Decoder state embeddings are used to query a precomputed keyword index, and the top-$\mathcal{K}$ ranked keywords are formatted into a prompt. This retrieval-augmented prompting improves grounding and disambiguation without modifying the model, enhancing ASR performance in out-of-domain settings.

4 Experimental Setup

Models and Implementation Details. We use Whisper large-v2 (1.5B parameters) as the frozen ASR backbone. Our retrieval model consists of two lightweight modules: a single-layer LSTM keyword encoder KE_ϕ, which projects 1024-dimensional token embeddings into a 512-dimensional space, and a feedforward decoder-state projector DP_ϕ, which maps 384-dimensional decoder states into the same embedding space. Combined, these modules introduce only 6.5M additional parameters. Keyword embeddings are precomputed and indexed to enable efficient kNN-based retrieval during inference. Training and implementation details are provided in Appendix A.[1]

Metrics. We evaluate RECAST primarily using recall-based metrics to assess keyword retrieval quality from large candidate pools. **Recall@50** measures retrieval effectiveness, while **Keyword Recovery Rate (KRR)** captures the proportion of keywords missed by the baseline ASR but recovered through retrieval—highlighting gains in contextual recall. To assess downstream ASR impact, we also report **Word Error Rate (WER)** and **Entity-WER (E-WER)**, the latter computed over dictionary entity spans in test utterances

[1]Code will be released upon acceptance.

Retrieval Baselines. For these baselines, we use the 1-best ASR hypothesis and perform keyword retrieval via fuzzy text matching. While many such methods exist, we focus on scalable, index-based approaches supported by Elastic Search (and its Phonetic Analysis plugin)[2]. The five baselines are: (1) **Soundex.** Matches exact Soundex (Knuth, 1973) codes between transcript and keywords. (2) **Metaphone.** Same as above but uses Metaphone codes. (3) **Double Metaphone.** Matches on either primary or alternate codes from Double Metaphone (Philips, 2000). (4) **NYSIIS.** Uses NYSIIS (Moore, 1977) for phonetic normalization and exact matching. (5) **BM25.** Ranks keywords using Elastic Search's built-in BM25 based on term frequency and document relevance.

Contextual ASR Baseline. We compare against PRISM (Mittal et al., 2023b), a contextual ASR system that 1) synthesizes keywords via TTS, 2) indexes their audio & text embeddings in a kNN key–value store, and 3) biases the decoder by linearly interpolating kNN-derived probabilities during beam-search decoding.

Datasets. We evaluate RECAST on two benchmarks. First, we use the entity-rich PRISM dataset (Mittal et al., 2023b), which contains out-of-domain utterances with named entities across domains like people, locations, and medical terms. Second, we use the standard LibriSpeech benchmark (Panayotov et al., 2015), with keyword dictionaries curated from prior contextual ASR studies (Sun et al., 2021; Le et al., 2021a).

Contextual ASR Models. For contextual ASR evaluation, we consider two state-of-the-art models: Whisper large-v2 (Radford et al., 2023) and Phi-4 (Abouelenin et al., 2025), a recent speech-language model (SpeechLLM). For Whisper, we adopt the prompting strategy from Peng et al. (Peng et al., 2023), where retrieved keywords are added to the decoder prompt at inference time. For Phi-4, keywords are incorporated according to its designed transcription prompt interface (Appendix B). Additionally, we introduce a third baseline based on LLM error correction, where the GPT-4o-mini (Hurst et al., 2024) model is prompted with the initial ASR prediction along with the retrieved key-

[2]Elastic Search: https://www.elastic.co/elasticsearch; Phonetic Analysis plugin: https://www.elastic.co/docs/reference/elasticsearch/plugins/analysis-phonetic.

Table 1: Comparison of performance on the Entity-rich dataset. We report Recall@50 (with Keyword Recovery Rate, KKR, in parentheses), along with Word Error Rate (WER) and Entity-WER (E-WER), using Whisper large-v2 as the underlying ASR model.

Method	LOCATION (SMALL)		LOCATION (BIG)		DRUGS		NAMES	
	Recall ↑ (KKR ↑)	WER / E-WER ↓	Recall ↑ (KKR ↑)	WER / E-WER ↓	Recall ↑ (KKR ↑)	WER / E-WER ↓	Recall ↑ (KKR ↑)	WER / E-WER ↓
Whisper Baseline	53.7 (-)	19.6 / 37.7	54.2 (-)	19.7 / 39.0	16.1 (-)	16.5 / 74.8	51.2 (-)	9.0 / 13.6
Baselines								
Soundex	74.2 (206)	13.1 / 21.1	72.9 (530)	13.8 / 23.5	50.7 (988)	14.1 / 51.2	67.5 (1167)	12.1 / 12.4
Metaphone	73.2 (196)	13.4 / 21.7	73.0 (532)	13.8 / 23.6	50.8 (991)	13.7 / 52.4	70.1 (1238)	11.7 / 12.4
Double Metaphone	75.0 (214)	12.9 / 20.3	74.3 (569)	12.2 / 22.4	53.8 (1077)	14.2 / 49.3	66.0 (1130)	11.7 / 12.3
NYSIIS	70.7 (171)	13.6 / 22.5	69.5 (434)	14.9 / 25.4	41.3 (716)	14.1 / 58.1	67.0 (1007)	12.8 / 12.6
BM25	62.8 (92)	16.1 / 28.7	61.7 (214)	17.4 / 29.0	20.8 (127)	16.2 / 72.3	49.5 (3)	9.2 / 12.6
PRISM	-	13.8 / 23.1	-	17.4 / 29.0	-	14.4 / 51.9	-	11.1 / 12.7
RECAST								
$\hat{\mathcal{T}} = 100$	79.3 (266)	12.4 / 19.3	74.0 (609)	13.7 / 21.8	48.6 (994)	14.9 / 52.2	52.8 (681)	11.6 / 12.4
$\hat{\mathcal{T}} = 50$	79.6 (267)	12.4 / 19.2	75.0 (661)	13.7 / 21.8	48.1 (1043)	14.7 / 51.4	61.6 (907)	9.7 / 12.3
$\hat{\mathcal{T}} = 20$	**82.5 (292)**	**11.5 / 17.2**	**75.6 (671)**	**12.4 / 20.4**	**55.3 (1149)**	**12.3 / 49.6**	**66.0 (1087)**	**9.6 / 11.6**

words and tasked with correcting the transcription based on keyword grounding (Appendix C)—an approach aligned with prior work on LLM-based ASR correction (Li et al., 2024a; Ma et al., 2025).

5 Experiments and Results

As shown in Table 1, among phonetic baselines, *Double Metaphone* achieves the best performance—improving Recall by up to 19.4% and reducing Entity-WER by 29.3% over *BM25* on the LOCATION (SMALL) benchmark. On the more error-prone DRUGS benchmark, it yields a 158.7% increase in Recall and a 32.1% drop in Entity-WER compared to BM25, demonstrating the effectiveness of phonetic matching in noisy ASR settings.

RECAST extends these improvements further. At its best configuration ($\hat{\mathcal{T}} = 20$), it achieves a 10.0% higher Recall and a 15.3% lower Entity-WER than *Double Metaphone* on LOCATION (SMALL). On the DRUGS dataset, it reduces WER by an additional 13.4%, even over the strongest phonetic baseline. On NAMES, it matches the top recall and achieves a 5.7% relative improvement in E-WER.

These gains, summarized in Table 1, highlight the strength of RECAST's contrastive retrieval approach in mitigating phonetic ambiguity and ASR noise. As shown in Appendix E, RECAST remains effective with smaller ASR models, balancing quality and efficiency. All WER gains are statistically significant under the MAPSSWE test ($p < 0.01$).

As shown in Table 2, LIBRISPEECH serves as our in-domain benchmark. RECAST with $\hat{\mathcal{T}} = 20$ consistently outperforms all baselines across both **test-clean** and **test-other**. On the more challenging **test-other** split, it achieves a 10.3% relative reduction in E-WER compared to the best phonetic baseline, while improving Recall by 3.4%. On **test-clean**, RECAST yields a 6.1% drop in E-WER and the highest recall overall.

Table 2: Comparison of performance on the in-domain LibriSpeech **test-clean** and **test-other** datasets with 1,000 distractors. We report Recall@50, Keyword Recovery Rate, along with Word Error Rate (WER) and Entity-WER (E-WER), using Whisper large-v2 as the underlying ASR model.

Method	test-clean			test-other		
	Recall ↑	KKR ↑	WER ↓ / E-WER ↓	Recall ↑	KKR ↑	WER ↓ / E-WER ↓
Whisper Baseline	89.2	-	4.2 / 10.2	79.5	-	6.7 / 10.6
Baselines						
Soundex	90.9	320	3.9 / 8.5	84.34	464	6.2 / 9.2
Metaphone	90.6	309	4.0 / 9.1	83.8	436	6.4 / 9.0
Double Metaphone	91.4	365	3.7 / 8.2	85.1	499	6.1 / 8.7
NYSISS	90.0	264	4.1 / 9.7	82.7	366	6.7 / 9.2
BM25	89.2	2	4.2/10.2	79.5	2	6.7 / 10.6
RECAST						
$\hat{\mathcal{T}} = 20$	**92.8**	**380**	**3.6 / 7.7**	**88.0**	**639**	**5.4 / 7.8**

Applicability to New Languages. Appendix F demonstrates the ability of RECAST to scale effectively to a new language—Hindi. Despite challenges such as high token fertility, RECAST achieves strong retrieval and contextual ASR performance. Unlike phonetic baselines such as *Soundex* and *Double Metaphone*, which are tailored to English, RECAST remains effective out of the box, reinforcing its language-agnostic applicability when paired with a suitable ASR model.

5.1 Using RECAST with LLMs

We evaluate the utility of RECAST beyond Whisper prompting by using its top-$\mathcal{K}$ keywords to guide two contextual ASR strategies: (1) prompting a speech LLM (Phi-4 Multimodal Instruct), and (2) text-based LLM error correction (Li et al., 2024a; Ma et al., 2025) using GPT-4o-mini (Hurst et al., 2024).

Table 3: Ablation of contextual ASR on the Location benchmarks: Speech LLMs (Phi-4, Whisper large-v2) and text-based ASR correction by GPT-4o-mini using keywords retrieved via RECAST. We report WER and E-WER for all methods, with RECAST varying by retrieval size ($\mathcal{K}$).

Method	LOCATION (SMALL)		LOCATION (BIG)	
	WER ↓	E-WER ↓	WER ↓	E-WER ↓
Baselines				
Whisper large-v2	19.6	37.7	19.7	39.0
Phi-4	21.9	42.5	18.9	41.3
GPT-4o-mini (w/ all keywords)	82.5	36.8	71.7	44.5
RECAST ($\hat{\mathcal{T}} = 20$)				
Phi-4 ($\mathcal{K} = 10$)	18.3	32.1	22.7	33.2
Phi-4 ($\mathcal{K} = 20$)	20.9	33.4	25.2	33.8
Phi-4 ($\mathcal{K} = 50$)	38.9	33.7	40.0	34.2
GPT-4o-mini ($\mathcal{K} = 50$)	16.3	26.6	18.1	32.5
Whisper (large-v2) ($\mathcal{K} = 50$)	**11.5**	**17.2**	**12.4**	**20.4**

Table 4: Ablation of ranking strategies on the Location benchmarks. We evaluate four rescoring methods for top-50 candidates—Avg/Scaled Distance with or without Positional Diversity—using Recall@50, Keyword Recovery Rate (KKR), WER, and E-WER.

Method	LOCATION (SMALL)			LOCATION (BIG)		
	Recall ↑	KKR ↑	WER ↓ / E-WER ↓	Recall ↑	KKR ↑	WER ↓ / E-WER ↓
Average Distance	78.9	264	12.3 / 18.6	71.6	601	13.8 / 23.3
Scaled Distance	81.2	288	11.7 / 17.2	72.9	630	13.6 / 22.7
P.D & Average Distance	79.9	280	11.8 / 18.1	72.6	625	13.8 / 22.8
P.D & Scaled Distance	**82.5**	**292**	**11.5 / 17.2**	**75.6**	**671**	**12.4 / 20.4**

As shown in Table 3, Phi-4 hallucinates with large keyword lists ($\mathcal{K} = 50$), increasing WER and E-WER, while smaller, focused subsets ($\mathcal{K} = 10$) improve performance—highlighting the need for precise keyword selection.

GPT-4o-mini shows similar behavior: prompting with all keywords leads to spurious insertions and degraded E-WER, while using RECAST-selected subsets yields consistent gains.

Overall, Whisper large-v2 with $\mathcal{K} = 50$ achieves the best performance, outperforming both LLM-based strategies.

5.2 Ablation Study

5.2.1 Analysis on Rankers

As shown in Table 4, using scaled distance normalization over plain average distance improves recall and reduces E-WER, with up to a 7.5% relative reduction. Incorporating positional diversity further enhances both metrics, yielding up to a 10.1% relative drop in E-WER over the base average distance method. These results highlight that combining distance scaling with positional diversity is crucial for selecting more relevant keyword candidates, leading to better retrieval quality and improved transcription performance.

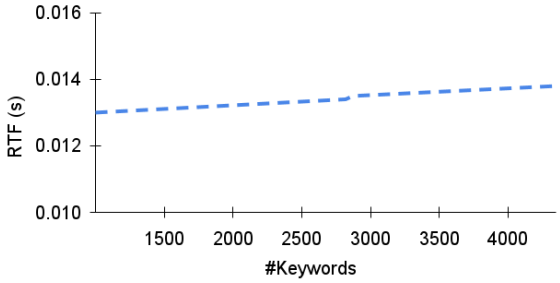

Figure 2: Comparison of Real-Time Factor (RTF) versus number of keywords for RECAST with $\hat{\mathcal{T}} = 20$.

5.2.2 Latency

Figure 2 shows that RECAST maintains consistently low Real-Time Factor (RTF) between 0.013 and 0.015 with $\hat{\mathcal{T}} = 20$, even as dictionary size scales from 1K to over 4K entities. This demonstrates the method's efficiency and suitability for real-time applications requiring accurate keyword retrieval with minimal latency. Notably, the near-constant RTF highlights the scalability of our token-level retrieval design, which avoids linear growth in inference time. These results underscore RECAST 's practicality for deployment in streaming ASR systems, where both responsiveness and retrieval quality are critical.

6 Conclusion

We introduced RECAST, a retrieval-augmented framework for contextual ASR that leverages decoder states of a pretrained encoder–decoder model to query large text-only keyword dictionaries without audio exemplars. At the core of RECAST is a contrastively trained retriever and a token-level span aggregation algorithm that constructs and ranks keyword hypotheses using contextual similarity, length-based scaling, and positional diversity. RECAST achieves state-of-the-art results on in-domain and out-of-domain benchmarks for both retrieval and contextual ASR, with substantial gains in recognition quality. It maintains low latency despite large vocabularies, owing to its lightweight design and efficient kNN retrieval. These results position RECAST as a scalable, accurate, and practical solution for keyword-guided speech recognition—even in languages where traditional phonetic baselines like *Soundex* and *Double Metaphone* are ineffective. While the current setup is language-specific, future work may explore multilingual extensions with a shared retriever.

Limitations

Our current implementation of RECAST is limited to encoder–decoder ASR models, where decoder states offer natural alignment for contrastive training. Extending this framework to CTC or RNN-T architectures would require estimating output alignments and identifying appropriate intermediate representations—a direction we leave for future work. Additionally, while our evaluation covers dictionaries of up to 4,000 keywords, real-world deployments may require scaling to tens of thousands of entities, for which suitable benchmarks are currently unavailable. We also note that performance in specialized domains such as medicine could further benefit from domain-specific finetuning of the base ASR model, which was not feasible due to data limitations.

Our current system performs keyword retrieval as a separate first pass, followed by contextual ASR in a second stage; an exciting direction for future work is to integrate retrieval and decoding more tightly—e.g., by guiding beam search with retrieved keywords in real time to avoid a two-stage pipeline.

While RECAST generalizes well across languages, its performance still depends on the quality of the underlying ASR backbone, which may underperform on certain dialects or low-resource languages.

In terms of broader implications, several risks merit consideration. If keyword dictionaries contain sensitive or personally identifiable information (PII), there is potential for unintended exposure in transcriptions. Furthermore, overly aggressive biasing—particularly with low-precision retrieval—can cause hallucinations or the insertion of incorrect entities. Security concerns also arise if malicious keyword dictionaries are introduced to manipulate transcription output.

References

Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, and 1 others. 2025. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras. *arXiv preprint arXiv:2503.01743*.

Sercan O Arik, Markus Kliegl, Rewon Child, Joel Hestness, Andrew Gibiansky, Chris Fougner, Ryan Prenger, and Adam Coates. 2017. Convolutional recurrent neural networks for small-footprint keyword spotting. *arXiv preprint arXiv:1703.05390*.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460.

Kai-Wei Chang, Haibin Wu, Yu-Kai Wang, Yuan-Kuei Wu, Hua Shen, Wei-Cheng Tseng, Iu-thing Kang, Shang-Wen Li, and Hung-yi Lee. 2024. Speechprompt: Prompting speech language models for speech processing tasks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Aditya Gourav, Linda Liu, Ankur Gandhe, Yile Gu, Guitang Lan, Xiangyang Huang, Shashank Kalmane, Gautam Tiwari, Denis Filimonov, Ariya Rastrow, and 1 others. 2021. Personalization strategies for end-to-end speech recognition systems. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7348–7352. IEEE.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and 1 others. 2020. Conformer: Convolution-augmented transformer for speech recognition. *arXiv preprint arXiv:2005.08100*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM transactions on audio, speech, and language processing*, 29:3451–3460.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Rongqing Huang, Ossama Abdel-Hamid, Xinwei Li, and Gunnar Evermann. 2020. Class lm and word mapping for contextual biasing in end-to-end asr. *arXiv preprint arXiv:2007.05609*.

Christian Huber, Juan Hussain, Sebastian Stüker, and Alexander Waibel. 2021. Instant one-shot word-learning for context-specific neural sequence-to-sequence speech recognition. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–7. IEEE.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Mahaveer Jain, Gil Keren, Jay Mahadeokar, Geoffrey Zweig, Florian Metze, and Yatharth Saraf. 2020. Contextual rnn-t for open domain asr. *arXiv preprint arXiv:2006.03411*.

Tahir Javed, Janki Atul Nawale, Eldho Ittan George, Sakshi Joshi, Kaushal Santosh Bhogale, Deovrat Mehendale, Ishvinder Virender Sethi, Aparna Ananthanarayanan, Hafsah Faquih, Pratiti Palit, and 1 others. 2024. Indicvoices: Towards building an inclusive multilingual speech dataset for indian languages. *arXiv preprint arXiv:2403.01926*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Herman Kamper, Weiran Wang, and Karen Livescu. 2016. Deep convolutional acoustic word embeddings using word-pair side information. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4950–4954. IEEE.

Donald E. Knuth. 1973. *The Art of Computer Programming, Volume 3: Sorting and Searching*. Addison-Wesley.

Duc Le, Mahaveer Jain, Gil Keren, Suyoun Kim, Yangyang Shi, Jay Mahadeokar, Julian Chan, Yuan Shangguan, Christian Fuegen, Ozlem Kalinli, and 1 others. 2021a. Contextualized streaming end-to-end speech recognition with trie-based deep biasing and shallow fusion. *arXiv preprint arXiv:2104.02194*.

Duc Le, Gil Keren, Julian Chan, Jay Mahadeokar, Christian Fuegen, and Michael L Seltzer. 2021b. Deep shallow fusion for rnn-t personalization. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 251–257. IEEE.

Sheng Li, Chen Chen, Chin Yuen Kwok, Chenhui Chu, Eng Siong Chng, and Hisashi Kawai. 2024a. Investigating asr error correction with large language model and multilingual 1-best hypotheses. In *Proc. Interspeech*, pages 1315–1319.

Yuang Li, Yinglu Li, Min Zhang, Chang Su, Jiawei Yu, Mengyao Piao, Xiaosong Qiao, Miaomiao Ma, Yanqing Zhao, and Hao Yang. 2024b. Cb-whisper: Contextual biasing whisper using open-vocabulary keyword-spotting. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2941–2946.

Yuang Li, Min Zhang, Chang Su, Yinglu Li, Xiaosong Qiao, Mengxin Ren, Miaomiao Ma, Daimeng Wei, Shimin Tao, and Hao Yang. 2023. A multitask training approach to enhance whisper with contextual biasing and open-vocabulary keyword spotting. *arXiv preprint arXiv:2309.09552*.

Hyungjun Lim, Younggwan Kim, Kiho Yeom, Eunjoo Seo, Hoodong Lee, Stanley Jungkyu Choi, and Honglak Lee. 2023. Lightweight feature encoder for wake-up word detection based on self-supervised speech representation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Da-Rong Liu, Chunxi Liu, Frank Zhang, Gabriel Synnaeve, Yatharth Saraf, and Geoffrey Zweig. 2020. Contextualizing asr lattice rescoring with hybrid pointer network language model. *arXiv preprint arXiv:2005.07394*.

Loren Lugosch, Mirco Ravanelli, Patrick Ignoto, Vikrant Singh Tomar, and Yoshua Bengio. 2019. Speech model pre-training for end-to-end spoken language understanding. *arXiv preprint arXiv:1904.03670*.

Rao Ma, Mengjie Qian, Mark Gales, and Kate Knill. 2025. Asr error correction using large language models. *IEEE Transactions on Audio, Speech and Language Processing*.

Jonathan Mamou, Bhuvana Ramabhadran, and Olivier Siohan. 2007. Vocabulary independent spoken term detection. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 615–622.

Ashish Mittal, Sunita Sarawagi, and Preethi Jyothi. 2023a. In-situ text-only adaptation of speech models with low-overhead speech imputations. In *The Eleventh International Conference on Learning Representations*.

Ashish Mittal, Sunita Sarawagi, Preethi Jyothi, George Saon, and Gakuto Kurata. 2023b. Speech-enriched memory for inference-time adaptation of asr models to word dictionaries. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14820–14835.

Gwendolyn B Moore. 1977. *Accessing individual records from personal data files using non-unique identifiers*, volume 13. US Department of Commerce, National Bureau of Standards.

Tsendsuren Munkhdalai, Zelin Wu, Golan Pundak, Khe Chai Sim, Jiayang Li, Pat Rondon, and Tara N Sainath. 2023. Nam+: Towards scalable end-to-end contextual biasing for adaptive asr. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 190–196. IEEE.

Aviv Navon, Aviv Shamsian, Neta Glazer, Gill Hetz, and Joseph Keshet. 2024. Open-vocabulary keyword-spotting with adaptive instance normalization. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11656–11660. IEEE.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE.

Puyuan Peng, Brian Yan, Shinji Watanabe, and David Harwath. 2023. Prompting the hidden talent of web-scale speech models for zero-shot task generalization. *arXiv preprint arXiv:2305.11095*.

Lawrence Philips. 2000. The double-metaphone search algorithm. *C/C++ Users Journal*, pages 38–43.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR.

Vijay Ravi, Yile Gu, Ankur Gandhe, Ariya Rastrow, Linda Liu, Denis Filimonov, Scott Novotney, and Ivan Bulyko. 2020. Improving accuracy of rare words for rnn-transducer through unigram shallow fusion. *arXiv preprint arXiv:2012.00133*.

Jan Robin Rohlicek, William Russell, Salim Roukos, and Herbert Gish. 1989. Continuous hidden markov modeling for speaker-independent word spotting. In *International Conference on Acoustics, Speech, and Signal Processing,*, pages 627–630. IEEE.

George Saon, Avihu Dekel, Alexander Brooks, Tohru Nagano, Abraham Daniels, Aharon Satt, Ashish Mittal, Brian Kingsbury, David Haws, Edmilson Morais, and 1 others. 2025. Granite-speech: open-source speech-aware llms with strong english asr capabilities. *arXiv preprint arXiv:2505.08699*.

Shane Settle and Karen Livescu. 2016. Discriminative acoustic word embeddings: Tecurrent neural network-based approaches. In *2016 IEEE spoken language technology workshop (SLT)*, pages 503–510. IEEE.

Aviv Shamsian, Aviv Navon, Neta Glazer, Gill Hetz, and Joseph Keshet. 2024. Keyword-guided adaptation of automatic speech recognition. *arXiv preprint arXiv:2406.02649*.

Guangzhi Sun, Chao Zhang, and Phil Woodland. 2023. Graph neural networks for contextual asr with the tree-constrained pointer generator. *arXiv preprint arXiv:2305.18824*.

Guangzhi Sun, Chao Zhang, and Philip C Woodland. 2021. Tree-constrained pointer generator for end-to-end contextual speech recognition. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 780–787. IEEE.

George Tucker, Minhua Wu, Ming Sun, Sankaran Panchapagesan, Gengshen Fu, and Shiv Vitaladevuni. 2016. Model compression applied to small-footprint keyword spotting.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Yingzhi Wang, Abdelmoumene Boumadane, and Abdelwahab Heba. 2021. A fine-tuned wav2vec 2.0/hubert benchmark for speech emotion recognition, speaker verification and spoken language understanding. *arXiv preprint arXiv:2111.02735*.

Ian Williams, Anjuli Kannan, Petar S Aleksic, David Rybach, and Tara N Sainath. 2018. Contextual speech recognition in end-to-end neural network systems using beam search. In *Interspeech*, pages 2227–2231.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, and 1 others. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Guanrou Yang, Ziyang Ma, Zhifu Gao, Shiliang Zhang, and Xie Chen. 2024. Ctc-assisted llm-based contextual asr. In *2024 IEEE Spoken Language Technology Workshop (SLT)*, pages 126–131. IEEE.

Yaodong Zhang and James R Glass. 2009. Unsupervised spoken keyword spotting via segmental dtw on gaussian posteriorgrams. In *2009 IEEE Workshop on Automatic Speech Recognition & Understanding*, pages 398–403. IEEE.

A Implementation Details

Training Setup. The retrieval model is trained on the LibriSpeech 960-hour corpus (Panayotov et al., 2015) using the contrastive loss described in Section 3.1. We use a batch size of 256 and a learning rate of $1\mathrm{e}{-4}$, training for 6 epochs with the AdamW optimizer. The temperature parameter used in the contrastive loss is set to $\tau = 0.07$. For each keyword span, up to 10 hard negatives are considered during training to improve contextual discrimination. The best checkpoint is selected based on validation accuracy.

Keyword Extraction. Bigram keywords are extracted from the capitalized LibriSpeech transcripts using the KeyBERT[3] model. These are then tokenized using Whisper's vocabulary and encoded using the keyword encoder KE_ϕ.

Retrieval Infrastructure. For efficient nearest-neighbor retrieval, keyword token embeddings are indexed using FAISS (Johnson et al., 2019). At inference time, decoder state embeddings are queried against this index to retrieve relevant token spans.

Inference Hyperparameters For all English experiments, the error threshold $E_{\max}$ was set to 2, and the number of nearest neighbors retrieved, $\hat{\mathcal{T}}$, was set to 20.

Hardware. All experiments are conducted on NVIDIA A100 GPUs with 40GB memory.

B Prompting Strategy for Phi-4 Multimodal Instruct Model

We employ different prompting strategies for baseline and contextual ASR using the Phi-4 Multimodal Instruct model (Abouelenin et al., 2025).

Baseline Prompt. For zero-context evaluation, we use a simple instruction-only prompt: `"Transcribe the audio to text."`

Contextual Prompt with Retrieved Keywords. To enable contextualization, we provide a list of relevant keywords retrieved by RECAST at inference time. The prompt is structured as:

> "Transcribe the audio to text. Transcribed text may contain the following words: <keyword_1>, <keyword_2>, ..., <keyword_N>."

[3] https://pypi.org/project/keybert/

Here, <keyword_i> denotes the i-th retrieved keyword. This formulation allows the model to bias transcription toward relevant entities without additional fine-tuning.

C Prompting Strategy for GPT-4o-mini Error Correction

For ASR error correction, we use GPT-4o-mini in a text-only setting, leveraging retrieved keywords from RECAST to provide contextual guidance. The prompt consists of a system instruction and a user input, combined into a single unified prompt presented to the model:

> System: You are given a set of keywords and an ASR prediction. Your task is to correct the ASR transcript using the keywords as contextual guidance. Only output the corrected transcript. Do not include any additional text.
>
> User: Keywords: <keyword_1>, <keyword_2>, ..., <keyword_N>
> ASR Prediction: <asr_output>

Here, <keyword_i> denotes the i-th retrieved keyword, and <asr_output> is the original ASR hypothesis. The model is expected to return only the corrected transcription without any additional explanation or formatting.

D Algorithm for Keyword Index Creation

Algorithm 2 Keyword Index Creation

Input: Keywords $\{\boldsymbol{k}^{(1)}, \ldots, \boldsymbol{k}^{(M)}\}$, `Tokenizer`, keyword-encoder KE_ϕ
Output: kNN index $\mathcal{I}$
$E \leftarrow []$, $K \leftarrow []$
for $m = 1$ **to** M **do**
  $\boldsymbol{w} \leftarrow \texttt{Tokenizer}(\boldsymbol{k}^{(m)})$
  $[\mathbf{z}_1^{(m)}, \ldots, \mathbf{z}_{|\boldsymbol{w}|}^{(m)}] \leftarrow \mathrm{KE}_\phi(w_1^{(m)}, \ldots, w_{|\boldsymbol{w}|}^{(m)})$ ▷ Eq. 1
  for $\ell = 1$ **to** $|\boldsymbol{w}|$ **do**
    E.append($\mathbf{z}_\ell^{(m)}$) ▷ Token Embedding
    K.append((m, ℓ)) ▷ Keyword & Token index
  end for
end for
$\mathcal{I} \leftarrow \texttt{FAISSIndex}(E, K)$
return $\mathcal{I}$

E Effect of ASR Model Size on RECAST

In Section 5, RECAST is trained on Whisper large-v2, a 1.5B parameter encoder–decoder ASR

Table 5: Performance comparison on the Entity-rich dataset. We report Recall@50 (with Keyword Recovery Rate, KKR, in parentheses), Word Error Rate (WER), and Entity-specific Word Error Rate (E-WER). RECAST is trained with different Whisper model variants, while all contextual ASR baselines use Whisper large-v2 as the underlying ASR model.

Method	LOCATION (SMALL)		LOCATION (BIG)		DRUGS		NAMES	
	Recall ↑ (KKR ↑)	WER / E-WER ↓	Recall ↑ (KKR ↑)	WER / E-WER ↓	Recall ↑ (KKR ↑)	WER / E-WER ↓	Recall ↑ (KKR ↑)	WER / E-WER ↓
Baseline								
Large-v2 (1.5B)	53.7 (-)	19.6 / 37.7	54.2 (-)	19.7 / 39.0	16.1 (-)	16.5 / 74.8	51.2 (-)	9.0 / 13.6
RECAST ($\hat{\mathcal{T}} = 20$)								
Tiny (39M)	67.8 (245)	14.4 / 24.7	58.0 (602)	14.7 / 24.7	24.8 (650)	17.3 / 63.2	49.1 (742)	10.6 / 12.9
Base (74M)	69.9 (262)	12.8 / 19.7	59.6 (612)	14.5 / 24.6	25.7 (661)	16.8 / 62.6	49.6 (756)	10.6 / 12.8
Small (244M)	74.2 (270)	12.6 / 18.0	68.5 (646)	13.4 / 22.2	29.6 (692)	16.5 / 60.9	52.3 (778)	10.5 / 12.7
Medium (769M)	75.6 (279)	12.1 / 17.8	69.3 (662)	13.2 / 21.8	32.2 (726)	16.3 / 59.6	54.1 (791)	10.3 / 12.6
Large-v2 (1.5B)	**82.5 (292)**	**11.5 / 17.2**	**75.6 (671)**	**12.4 / 20.4**	**55.3 (1149)**	**12.3 / 49.6**	**66.0 (1087)**	**9.6 / 11.6**

model. To assess sensitivity to ASR backbone size, we train RECAST on smaller Whisper variants: `medium.en` (769M), `small.en` (244M), `base.en` (74M), and `tiny.en` (39M), with the keyword retriever trained from scratch in each case. At inference, retrieved keywords are passed as prompts to Whisper large-v2 (Peng et al., 2023) to isolate retrieval quality from decoding performance.

As shown in Table 5, we find that smaller backbones yield competitive performance on LOCATION benchmarks (e.g., `Tiny` incurs only a 17.8% relative drop in Recall on LOCATION (SMALL) compared to `Large-v2`), but show larger degradations on entity-rich or ambiguity-prone datasets. On DRUGS, the E-WER increases by 27.5% for `Tiny` relative to `Large-v2`, while on NAMES, recall drops by 25.6%. These results indicate that while RECAST remains robust across model scales, larger ASR backbones offer significant benefits for complex retrieval settings. Nonetheless, smaller models remain a viable option in resource-constrained scenarios or domains with fewer rare entities.

F Evaluation on Hindi

To assess the effectiveness of RECAST on a linguistically diverse language, we conduct experiments on Hindi using the IndicVoices dataset (Javed et al., 2024). Retrieval and contextual ASR evaluations follow the same setup described in Section 4.

As the baseline Whisper large-v2 model performs poorly on IndicVoices (WER: 60.0), we first fine-tune it on the training split using LoRA (Hu et al., 2022), reducing the WER to 28.1. All subsequent RECAST training and contextual ASR experiments are conducted on this LoRA-adapted model. This setup enables fair comparison and demonstrates the applicability of RECAST in languages with challenging tokenization characteristics. Notably, since Hindi is underrepresented in the training data of Whisper's tokenizer, its tokenization quality is significantly worse than for English. The token fertility—the average number of tokens per word—is approximately 2 for English but rises to around 6 for Hindi, resulting in sequences that are roughly 3× longer. This further underscores the importance of evaluating retrieval methods in token-heavy settings.

Table 6: Performance on the Hindi contextual biasing benchmark constructed from the IndicVoices dataset. We report Recall@50 (with Keyword Recovery Rate, KKR, in parentheses), along with Word Error Rate (WER) and Entity-specific WER (E-WER) under two matching tolerance settings: $E_{\max} = 2$ and $E_{\max} = 3$. All models are evaluated using the LoRA-adapted Whisper large-v2 model.

Method	$E_{max} = 2$			$E_{max} = 3$		
	Recall ↑	KKR ↑	WER ↓ / E-WER ↓	Recall ↑	KKR ↑	WER ↓ / E-WER ↓
Baseline	43.6	-	28.1 / 43.6	43.6	-	28.1 / 43.6
RECAST ($\hat{\mathcal{T}} = 20$)	52.6	192	27.2 / 40.4	71.5	396	26.3 / 38.2
RECAST ($\hat{\mathcal{T}} = 50$)	66.1	322	26.4 / 38.5	75.6	479	25.9 / 37.1
RECAST ($\hat{\mathcal{T}} = 100$)	**73.0**	**420**	**26.2 / 38.1**	**77.7**	**531**	**25.2 / 35.1**

Since no standard contextual biasing benchmark exists for Hindi, we construct one by extracting keywords from all test utterances and compiling them into a dictionary of 800 entries, which serves as the biasing list for retrieval and contextual ASR evaluation in this experiment.

Moreover, most phonetic baselines—such as Double Metaphone (Philips, 2000) and Soundex (Knuth, 1973)—are not applicable to Hindi, as they were primarily designed for English and lack multilingual phonetic support. In contrast, RECAST is inherently applicable across languages with sufficient ASR capabilities, making it particularly well-suited for multilingual scenarios.

Table 6 shows that RECAST yields substantial relative improvements over the baseline across all

settings. At $E_{\max} = 2$, recall improves by **67.4%** and E-WER is reduced by **12.7%**. With $E_{\max} = 3$, the relative gains are even higher, with Recall improving by **78.2%** and E-WER dropping by **19.5%**. These trends highlight that, for languages with high token fertility, larger $\hat{\mathcal{T}}$ values are especially beneficial due to longer and more fragmented entity spans. Additionally, increasing $E_{\max}$ enables more tolerant entity matching, which is important for compensating for tokenization-induced mismatches.

G Acknowledgment of AI Assistance

We used GPT-4o for spell checking and text editing assistance only. All technical content and experimental results were developed and written by the authors.