JSPG: Dynamic Dictionary Filtering via Joint Semantic-Pinyin-Glyph Retrieval for Chinese Contextual ASR

Anonymous ACL submission

Abstract

Contextual Automatic Speech Recognition (ASR) faces challenges with large-scale keyword dictionaries, as excessive irrelevant candidates introduce noise that degrades accuracy. To address this, dynamic filtering typically uses a base ASR model to generate preliminary hypotheses, followed by semantic text retrievers to fetch a concise subset of relevant keywords. However, this approach frequently fails in Chinese ASR. Base models often produce homophonic or near-homophonic errors that preserve the phonetic cues of the target keywords but severely distort their semantic meaning, rendering standard semantic retrievers ineffective. To resolve this, we propose a filtering framework that jointly integrates Semantic, Pinyin, and Glyph features (JSPG). Pinyin effectively retrieves targets based on phonetic similarity, while glyph provides complementary structural cues to filter out numerous irrelevant homophones inherent in Chinese. To bridge the gap between character-level pinyin/glyph metrics and sequence-level filtering, we introduce an extended Smith-Waterman algorithm that computes similarity scores between the N-best hypothesis sequences and keywords. Experiments on the Aishell-1 and RWCS-NER datasets demonstrate that JSPG significantly outperforms single-feature baselines. Furthermore, downstream contextual ASR models guided by JSPG achieve substantial improvements in keyword recognition accuracy.

1 Introduction

Automatic Speech Recognition (ASR) systems have achieved remarkable success in general transcription tasks (Radford et al., 2022; Bai et al., 2024; Omnilingual et al., 2025). However, accurately recognizing contextual keywords, such as technical terminologies and rare entities, remains a challenge. The keywords appear infrequently in general training corpora, which makes it difficult for models to transcribe them effectively (Peters et al., 2018; Sudo et al., 2024b).

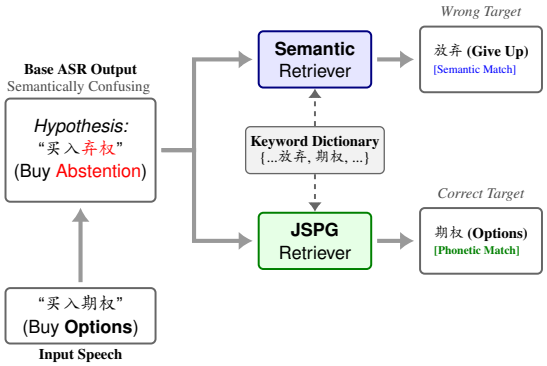


Figure 1: Illustration of the retrieval process. **Upper Path:** The semantic retriever is misled by the ASR error "弃权 (Abstention)" and retrieves "放弃 (Give Up)". **Lower Path:** Our JSPG retriever utilizes the joint semantic-pinyin-glyph features to correctly retrieve the target "期权 (Options)".

To address this, researchers have proposed the contextual ASR task (Pundak et al., 2018; Alon et al., 2019; Zhou et al., 2024b), which utilizes a predefined dictionary of potential keywords to guide the recognition process. Current approaches have achieved promising results when the keyword dictionary is small. However, experiments show that as the dictionary size increases, the presence of irrelevant keywords leads to a significant decline in transcription accuracy (Zhou et al., 2024b; Sudo et al., 2024a; Zhou and Li, 2025).

To alleviate the negative impact of large dictionaries, recent work has explored dynamically filtering the dictionary into a high-relevance subset of keywords for each speech input (Mathur et al., 2024; Li et al., 2024a; Chen et al., 2025; Lei et al., 2025). In a typical cascaded pipeline, a base ASR model first transcribes the speech input into preliminary N-best hypotheses. Then these methods usually either select the top-ranked hypothesis as the query or concatenate all hypotheses into a single

query sequence. Subsequently, the query and the keywords in the dictionary are fed into a pre-trained text embedding model, which converts them into embedding representations. Finally, the system calculates the cosine similarity scores between them to select the top-scoring keywords as the filtered subset (Xiao et al., 2025; Dimitrios et al., 2025). Since text embeddings are designed to encode the contextual meaning of the input, we refer to these methods as semantic retrievers.

However, relying solely on semantic features is insufficient for precise keyword filtering in Chinese contextual ASR. Since ASR is acoustic-based and Chinese has abundant homophones, ASR errors typically sound similar to the target keyword but carry entirely different meanings. Semantic retrievers fetch keywords that are semantically related to the erroneous transcription but completely irrelevant to the actual target. For instance, as illustrated in Figure 1, the target "期权" (Options) is misrecognized as "弃权" (Abstention) due to homophonic confusion. Consequently, the retriever incorrectly fetches the irrelevant keyword "放弃" (Give Up) based on its high semantic similarity with "弃权".

Since these errors preserve phonetic similarity to the target, we can leverage this cue for retrieval. In Chinese, pinyin directly encodes pronunciation, making it a natural metric for phonetic matching. However, many different Chinese characters share the exact same pronunciation. Therefore, relying solely on pinyin introduces substantial noise, as it retrieves many irrelevant characters alongside the correct keyword.

Fortunately, the glyph features of Chinese characters offer a complementary solution to this phonetic noise. Since most characters are phono-semantic compounds, ASR phonetic errors frequently share structural components with the target keywords (Tan et al., 2005). Crucially, while many different characters share the exact same pronunciation, every individual character has a unique glyph structure. This one-to-one property creates sharp discriminative signals, allowing the system to easily distinguish the target from unrelated homophones retrieved by pinyin.

In this work, we propose JSPG, a Joint Semantic-Pinyin-Glyph retrieval approach for Chinese contextual ASR. JSPG integrates two types of features: a global *semantic* score derived from pre-trained text embeddings to capture the broad context (Mathur et al., 2024; Lei et al., 2025), and a unified *phonetic-glyph* score that fuses character-level pinyin and glyph similarities (Qiao et al., 2025) to ensure precise local matching. Since pinyin and glyph metrics inherently operate at the character level, we introduce an extended Smith-Waterman algorithm (Smith et al., 1981) that extends them to measure the sequence-level similarity between a short keyword and a longer, noisy ASR hypothesis sequence.

We evaluate JSPG on the standard Aishell-1 (Bu et al., 2017; Chen et al., 2022) and the complex, real-world RWCS-NER datasets (Zhou et al., 2024a). Experimental results demonstrate that JSPG significantly outperforms baselines relying solely on individual semantic, pinyin, or glyph features. The ablation studies further reveal that each component contributes uniquely to the system, with their integration yielding the most robust retrieval performance. Furthermore, when contextual ASR models are guided by the keywords filtered by JSPG, they achieve substantial improvements in keyword recognition accuracy.

2 Related Work

Standard Contextual ASR approaches face significant scalability bottlenecks when handling massive dictionaries, as the influx of irrelevant candidates introduces noise that degrades recognition accuracy (Zhou et al., 2024b; Sudo et al., 2024a). To address this, recent research adopts dynamic dictionary filtering, which retrieves a concise, high-relevance keyword subset for each speech input before feeding it into the downstream contextual ASR model. Existing implementations of this mechanism typically fall into two primary categories: text-based cascaded retrieval and audio-based end-to-end retrieval.

2.1 Text-based Cascaded Retrieval

The first category, text-based cascaded retrieval, operates by first utilizing a lightweight base ASR model to transcribe audio into preliminary hypotheses or N-best lists. These hypotheses serve as queries to match against the keyword dictionary. Existing methods predominantly rely on dense text embeddings to perform this matching. For instance, Mathur et al. (2024) utilize BERT (Devlin et al., 2019), while Xiao et al. (2025) employ MPNet (Song et al., 2020) to encode both hypotheses and keywords, ranking candidates based on semantic similarity. The advantage of this paradigm lies in its training-free nature, allowing for the direct ap-

plication of powerful text embedding models to capture rich semantic knowledge.

While effective for capturing semantic similarity, these semantic-only retrievers are susceptible to phonetic recognition errors, which are prevalent in Chinese. Since base ASR models frequently produce homophonic or near-homophonic errors, the transcribed text often diverges significantly in meaning despite sounding similar to the target. Consequently, retrievers relying solely on semantic embeddings are misled by these textual errors, fetching keywords that are semantically related to the incorrect transcripts but irrelevant to the actual speech. Such irrelevant candidates fail to provide accurate context, rendering them ineffective for guiding the downstream contextual ASR model. This limitation underscores that relying solely on global semantics is insufficient, necessitating the incorporation of other features, such as phonetic and glyph features.

2.2 Audio-based End-to-End Retrieval

In text-based cascaded methods, errors in this preliminary transcription will propagate to the retrieval stage. It may lead to the failure of retrieving the correct keywords. To avoid the reliance on intermediate text, audio-based end-to-end retrieval directly utilizes the input speech to retrieve keywords. Specifically, these methods employ a speech encoder to generate embeddings for the audio and a text encoder to generate embeddings for the keywords. They are trained using a contrastive loss to maximize the similarity score between the input speech and the relevant ground-truth keywords (Li et al., 2024b; Gong et al., 2025). While this approach effectively bypasses error propagation, it typically necessitates extensive additional training on large-scale paired datasets to align the audio and text representations, making it data-hungry and computationally expensive.

In contrast, our proposed JSPG framework maintains the cascaded paradigm to directly leverage the sophisticated semantic capabilities of LLMs without additional training. Crucially, we mitigate the negative impact of transcription errors by integrating phonetic and glyph features.

2.3 Glyph and Pinyin Features in Chinese

Chinese characters are unique in that they integrate semantics, phonetics, and structures (Tan et al., 2005). Specifically, phonetic and structural attributes are commonly measured by pinyin and glyph features, respectively. Previous works in Chinese Spelling Check (CSC) and Named Entity Recognition (NER) have demonstrated that integrating these features is essential for resolving ambiguity.

Recently, Qiao et al. (2025) proposed an approach for quantifying the similarity between Chinese characters based on their phonetic and structural attributes. However, it is designed for *character-level* comparison and cannot be directly applied to our task. The core challenge lies in computing the similarity between the keyword and the hypothesis sequence. Typically, there is a significant length discrepancy between the short keyword and the longer hypothesis, and the hypothesis sequence usually contains recognition errors. To bridge this gap, our JSPG method adapts the character-level similarity scores into a modified Smith-Waterman algorithm (Smith et al., 1981). This allows us to leverage phonetic and glyph cues to identify the optimal matching part within the hypothesis and return a sequence-level similarity score, which is used to assist the retrieval process in JSPG.

3 Our JSPG Approach

Big Picture. Our goal is to determine a small number of keywords that are really related to the input speech from a keyword dictionary containing numerous keywords. Only the selected few keywords are provided to the subsequent contextual ASR models. Specifically, we follow the N-best hypothesis paradigm (Mathur et al., 2024; Dimitrios et al., 2025), which consists of two steps. First, we obtain N-best transcriptions using a non-contextual basic ASR model. Second, we obtain the K most related keywords from the dictionary using the N transcriptions. Previous works use implicit embedding-based similarity to represent relatedness, using LLMs to encode each transcription and each keyword into dense vectors. The main contribution of this work is incorporating explicit character similarity based on pronunciation and glyph. We follow the recent work on Chinese spelling check of (Qiao et al., 2025), and borrow the method for computing character similarity. To compute the relatedness between a transcription and a keyword, we extend the standard Smith-Waterman algorithm (Smith et al., 1981) to incorporate character similarity.

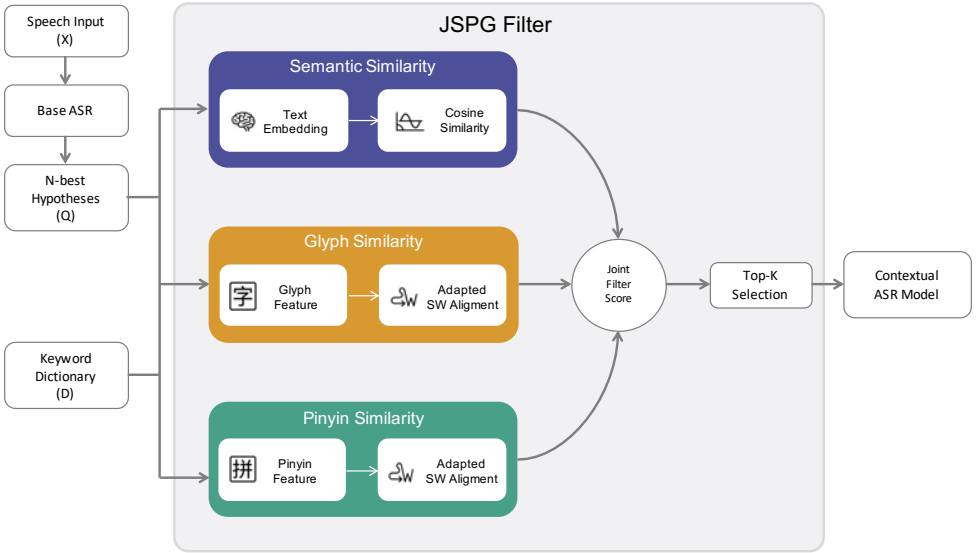


Figure 2: Overview of the proposed JSPG filtering framework. Given an input utterance, a base ASR model generates N-best hypotheses, which are compared to each candidate keyword in the dictionary via semantic, pinyin, and glyph similarity. The scores are combined to rank and retrieve the most relevant keywords for downstream contextual ASR.

3.1 Problem Formulation

Given an input speech x and a large dictionary $D = \{w_1, w_2, \ldots, w_M\}$, the goal of our JSPG is to filter D into a concise, high-relevance subset of a predefined size K ($K \ll M$). Following previous practices (Xiao et al., 2025), the process begins with a base ASR model that transcribes the speech x into a preliminary list of N-best hypotheses $Q = \{q_1, q_2, \ldots, q_N\}$, where each q_j is a character sequence representing a transcription. JSPG computes a joint score $F(Q, w)$ between w and Q. The top-K keywords are selected as the subset. Specifically, $F(Q, w)$ is computed by integrating a semantic score $F^s(Q, w)$ and a phonetic-glyph score $F^{pg}(Q, w)$. The phonetic-glyph score fuses pinyin-based score $F^p(Q, w)$ and glyph-based score $F^g(Q, w)$. The final score combines semantic and phonetic-glyph scores:

$$F^{pg}(Q, w_i) = \alpha \cdot F^p(Q, w_i) + (1-\alpha) \cdot F^g(Q, w_i). \quad (1)$$

$$F(Q, w_i) = \beta \cdot F^s(Q, w_i) + (1-\beta) \cdot F^{pg}(Q, w_i). \quad (2)$$

3.2 Semantic-based Sequence Similarity

To calculate the semantic similarity between the ASR hypotheses and the candidate keywords, we leverage dense text embeddings. We begin by concatenating the N-best hypotheses into a single sequence Q', following the approach in (Dimitrios et al., 2025). Next, we employ the Qwen3-Embedding model (Zhang et al., 2025) to encode both Q' and the candidate keywords w_i. To maximize the model's retrieval capability, we follow its recommended settings and add a task-specific instruction before Q': "*Given a list of candidate transcriptions predicted by a speech recognition model as a query, retrieve keywords relevant to the query. The candidate transcriptions are:* q_1, q_2, $\ldots$, q_N.". The semantic score $F^s(Q, w_i)$ is then derived using cosine similarity:

$$F^s(Q, w_i) = \frac{\text{Emb}(Q') \cdot \text{Emb}(w_i)}{\|\text{Emb}(Q')\| \, \|\text{Emb}(w_i)\|} \quad (3)$$

While text embeddings effectively capture the global context, ASR errors in Q' often change the actual meaning of the transcription, causing pure semantic retrieval to fail. This limitation necessitates the incorporation of pinyin and glyph features.

3.3 Phonetic-Glyph Character Similarity

ASR prediction errors typically disrupt the semantic context but preserve the phonetic cues of the target keyword. To retrieve the target keywords using this cue, we compute character-level similarity using two complementary features: pinyin and glyph. First, pinyin directly encodes pronunciation in Chinese, so it can be used to measure phonetic similarity and recall the target keywords. However, in Chinese, a single pronunciation often corresponds to multiple distinct characters. Therefore, using pinyin to retrieve keywords inevitably

fetches many irrelevant candidates. Second, the glyph feature serves as another effective metric to retrieve the target keywords. Because most Chinese characters are phono-semantic compounds, characters with similar pronunciations frequently share similar structural components. Furthermore, unlike the one-to-many mapping of pinyin, every character has a unique glyph structure. The one-to-one property enables the glyph feature to effectively exclude irrelevant keywords that sound similar to the target.

Pinyin Similarity. Following Qiao et al. (2025), for any two characters c_1 and c_2, we extract their complete pinyin sequences, denoted as $\mathsf{py}(c_1)$ and $\mathsf{py}(c_2)$. The pinyin similarity score $sim_p(c_1, c_2)$ is then calculated using the normalized Levenshtein Distance (LD):

$$sim_p(c_1, c_2) = 1 - \frac{\mathsf{LD}(\mathsf{py}(c_1), \mathsf{py}(c_2))}{|\mathsf{py}(c_1)| + |\mathsf{py}(c_2)|}, \quad (4)$$

where $|\cdot|$ represents the length of the pinyin string.

Glyph Similarity. To measure the glyph similarity between characters, we adopt the character-level metric proposed by Qiao et al. (2025). This metric comprehensively evaluates glyph similarity across four distinct dimensions. For any two characters c_1 and c_2, the glyph similarity score $sim_g(c_1, c_2)$ is defined as the average of four sub-metrics:

1. **Four-corner Code** (sim_g^1): Measures the similarity between numerical codes that represent the shapes of the character's four corners.

2. **Structure-aware Code** (sim_g^2): Encodes radical decomposition and spatial layout (e.g., left-right vs. top-bottom structures).

3. **Stroke Sequence Edit Distance** (sim_g^3): Computes the normalized LD between standard stroke-order sequences to capture fine-grained writing differences.

4. **Stroke Sequence LCS** (sim_g^4): Computes the normalized Longest Common Subsequence (LCS) of stroke sequences to identify shared internal writing patterns.

The final character-level glyph similarity is the average of these four components:

$$sim_g(c_1, c_2) = \frac{1}{4} \sum_{k=1}^{4} sim_g^k(c_1, c_2). \quad (5)$$

Currently, both sim_p and sim_g operate strictly at the character level. However, a key challenge is extending them to quantify the sequence-level relevance between a short keyword and a longer, noisy ASR hypothesis sequence. To bridge this gap, we introduce an extended Smith-Waterman algorithm to extend these character-level metrics into sequence-level scores, yielding $F^p(Q, w_i)$ and $F^g(Q, w_i)$.

3.4 Sequence-Level Similarity via an Extended Smith-Waterman Algorithm

The similarities defined above operate at the single-character level. In other words, given a pair of characters, $sim_g(.)$ (or $sim_p(.)$) can give a similarity score between them.

However, our retrieval module needs to assess how likely it is that a short keyword sequence $\boldsymbol{w} = c_1c_2...c_s$ is related to a longer sequence $\boldsymbol{q} = x_1x_2...x_n$. Specifically, $\boldsymbol{w}$ is considered related to $\boldsymbol{q}$ if $\boldsymbol{w}$ itself, or a modified version of it, is a substring of $\boldsymbol{q}$. To achieve this, we employ an extended Smith-Waterman algorithm (Smith et al., 1981), a dynamic programming approach based on edit distance.

Edit Distance. As a preliminary, the edit distance algorithm uses dynamic programming to efficiently compute the distance between two sequences, such as $a_1a_2...a_s$ and $b_1b_2...b_t$. We use $\mathsf{ED}(\cdot)$ to denote the edit distance. Specifically, $\mathsf{ED}(a_1a_2...a_s, b_1b_2...b_t)$ represents the total weight of the operations required to transform $a_1a_2...a_s$ into $b_1b_2...b_t$. There are four standard operations: match (no change), deletion, insertion, and substitution. Their corresponding weights are typically set to 0, 1.0, 1.0, and 1.0, respectively.

Standard Smith-Waterman. The standard Smith-Waterman algorithm is built upon $\mathsf{ED}(.)$ and a more complex dynamic programming algorithm. Given $\boldsymbol{w} = c_1...c_s$ and $\boldsymbol{q} = x_1...x_n$, the algorithm finds the optimal indices u, v ($1 \le u \le v \le s$) for $\boldsymbol{w}$ and i, j ($1 \le i \le j \le n$) for $\boldsymbol{q}$, which yield the lowest edit distance between the substrings $c_u...c_v$ and $x_i...x_j$.

$$\mathsf{SW}(\boldsymbol{w}, \boldsymbol{q}) = \min_{u,v,i,j} \mathsf{ED}(c_u...c_v, x_i...x_j)$$

Extended $\mathsf{SW}(.)$. To accommodate our situation, we make a two-fold extension. First, we further consider character similarity for the substitution operation. Taking sim_g as an example, we change

the weight of substituting c_1 with c_2 from a fixed value of 1.0 to:

$$1 - sim_g(c_1, c_2)$$

The more similar c_1 is to c_2, the lower the weight. When $c_1 = c_2$, which corresponds to the not-change operation, the weight is zero.

Second, we prohibit deletion operations at the beginning and end positions of $\boldsymbol{w}$ (i.e., for c_1 and c_s). This constraint mathematically enforces $u = 1$ and $v = s$, forcing the entirety of $\boldsymbol{w}$ to be aligned against a substring of $\boldsymbol{q}$.

Then, the relatedness likelihood is computed as:

$$\mathsf{RL}(\boldsymbol{w}, \boldsymbol{q}) = \frac{|\boldsymbol{w}| - \mathsf{SW}_{\text{ext}}(\boldsymbol{w}, \boldsymbol{q})}{|\boldsymbol{w}|}$$

Considering that the base ASR model typically generates an N-best list of hypotheses $Q = \{q_1, q_2, ..., q_N\}$ rather than a single transcription, we compute the relatedness likelihood $\mathsf{RL}(w, q_j)$ for each individual hypothesis $q_j \in Q$. We take the maximum likelihood score across all N hypotheses. Therefore, the final sequence-level similarity score between the keyword w and the N-best hypotheses Q is formulated as:

$$F^*(Q, w) = \max_{q_j \in Q} \mathsf{RL}(w, q_j)$$

where $*$ represents either p or g, depending on whether pinyin or glyph features are used when calculating the relatedness likelihood RL.

4 Experiments

4.1 Experimental Setup

Datasets. To comprehensively evaluate the proposed approach across varying domains and speaking styles, we conduct experiments on two datasets. Aishell-1 (Bu et al., 2017; Chen et al., 2022) serves as our standard benchmark, containing 150 hours of Mandarin speech recorded in a clean environment. To validate robustness in complex, real-world scenarios, we utilize the RWCS-NER dataset (Zhou et al., 2024a), which comprises two distinct domains: open-domain daily conversation (DC) and intelligent cockpit instructions (ICI). It features spontaneous speech, background noise, and a high density of rare entities, making it a challenging task for contextual ASR.

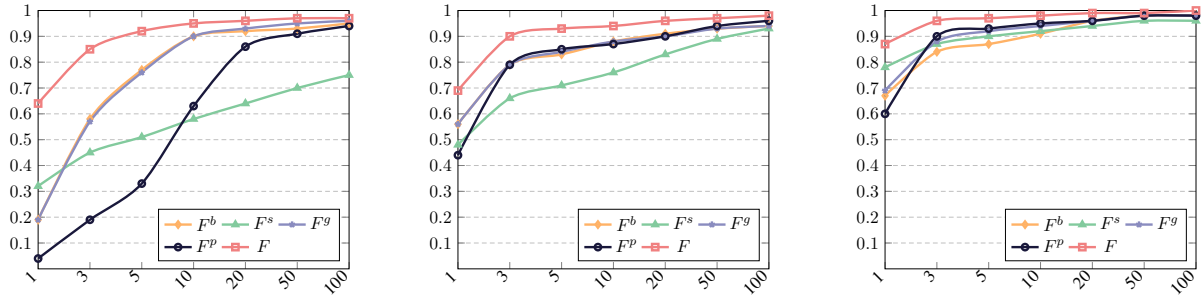
Pipeline Settings. We evaluate our approach using a three-stage pipeline. First, we employ Whisper (Radford et al., 2022), fine-tuned on the Aishell-1 training set, as the base ASR model to generate the initial N-best hypotheses Q. Second, for the retrieval stage, we compare our JSPG against an exact character-matching baseline (F^b) that uses the same Smith-Waterman framework, as well as single-feature retrievers relying solely on semantic (F^s, powered by Qwen3-Embedding-0.6B (Zhang et al., 2025)), glyph (F^g), or pinyin (F^p) scores. Finally, to verify downstream effectiveness, we integrate the retrieved keywords into two distinct contextual ASR paradigms: end-to-end models (CopyNE (Zhou et al., 2024b) and a multi-grained model (Zhou and Li, 2025)) and a two-stage LLM-refinement paradigm utilizing GPT-OSS-120B (OpenAI, 2025) to correct the N-best hypotheses based on the retrieved keywords. Detailed descriptions of these downstream models are provided in Appendix A.2.

Evaluation Metrics We evaluate our system from two perspectives: the quality of the retrieved keywords and the final ASR recognition accuracy. To assess retrieval quality, we adopt Recall@K (R@K) as our core metric (Chen et al., 2025). It measures the proportion of ground-truth keywords captured within the top-K retrieved list. In our experiments, we report R@K for $K \in \{1, 3, 5, 10, 20, 50, 100\}$.

To assess downstream recognition accuracy and verify whether the retrieved keywords effectively guide the ASR model, we report four metrics. We measure the overall Character Error Rate (**CER**) on the full text, and the Unbiased-CER (**U-CER**) on non-keyword text to ensure general transcription quality is preserved. For keyword-specific performance, we report Keyword-CER (**K-CER**) on the keyword segments and Keyword Recall (**Recall**), which represents the percentage of ground-truth keywords correctly transcribed.

4.2 Results of Retrieval

Overall Trend. Figure 3 illustrates the recall performance as K varies. A consistent trend is observed across all datasets: the recall of all retrieval methods increases monotonically with K and eventually saturates. This aligns with the theoretical expectation that a larger candidate pool covers more ground-truth targets. Crucially, our proposed JSPG algorithm (F) consistently achieves the highest re-



(a) Aishell-1 (b) DC (c) ICI

Figure 3: The performance of retrieval methods on Aishell, DC, and ICI datasets. The x-axis indicates the K value in top-K, and the y-axis shows the corresponding Recall@K.

call among all methods across all K settings. This empirical evidence validates that fusing semantic, glyph, and pinyin features provides a more robust retrieval signal than any single modality or the character-matching baseline.

Comparison between Different Retrieval Approaches. The performance gap between methods is most pronounced at lower K values (e.g., $K = 1, 3$), which reflect the precision of the retriever. In this setting, the joint algorithm F demonstrates a decisive advantage, significantly outperforming other methods. Notably, on the relatively clean Aishell-1 and DC datasets, the character-matching baseline F^b performs comparably to the glyph-based method F^g. This suggests that many keywords in these scenarios can be recalled via surface-level or visual similarity. Conversely, while the pinyin-based retriever F^p underperforms at $K = 1$ (especially on Aishell, dropping to 0.04), its recall rate grows rapidly as K increases, eventually matching or surpassing F^b and F^g (e.g., reaching 0.96 on DC at $K = 100$). This indicates that pinyin features are highly effective at capturing phonetic confusion and homophonic errors, provided a slightly larger candidate pool is allowed to accommodate phonetic ambiguity.

Ablation on Phonetic-Glyph Feature Weights. We conduct a two-stage ablation study to investigate the complementarity among semantic, pinyin, and glyph features, as summarized in Table 2.

Stage 1: Pinyin vs. Glyph (α) Varying the pinyin weight α reveals a strong symbiotic relationship. Interestingly, R@1 remains highly stable ($\sim$18.7%) across all settings. This stability demonstrates that top-1 precision is heavily anchored by the glyph feature, whose one-to-one structural mapping successfully filters out exact homophone confusions. However, for a larger candidate pool, R@100 rises substantially to peak at 99.13% when $\alpha = 0.7$. This proves that the pinyin feature is indispensable for capturing a broader range of structurally distinct but phonetically similar ASR errors. We set $\alpha = 0.7$ to optimally balance broad phonetic tolerance and precise structural filtering.

Stage 2: Semantic vs. Phonetic-Glyph (β). Fixing $\alpha = 0.7$, we evaluate the semantic weight β (Table 2) to demonstrate the necessity of feature fusion. As shown in Figure 3, the recall curve for pure semantic retrieval (F^s) flattens out early. Because homophonic ASR errors severely distort sentence meaning, pure semantics assigns extremely low scores to the correct targets, pushing them completely out of the top-100 pool. Table 2 confirms this limitation: over-relying on semantics ($\beta = 0.9$) causes the overall recall to drop to 89.11%. In contrast, fusing semantic and phonetic-glyph features effectively overcomes this bottleneck. At the optimal weight ($\beta = 0.4$), the system achieves R@1 = 64.52% while sustaining a high overall recall (R@100 = 97.03%). This confirms that the semantic and phonetic-glyph features are highly complementary, and their fusion leads to significantly better retrieval performance.

4.3 Results of Contextual ASR

Having validated the retrieval performance, we now evaluate the impact of the proposed JSPG retriever on the downstream contextual ASR task. We integrate JSPG into the pipeline and test it with three different recognition models: CopyNE (Deep Biasing) (Zhou et al., 2024b), Multi-grained (LLM-Fusion) (Zhou and Li, 2025), and LLM-Refine (Two-stage correction); more details are provided in Appendix A.2. Table 1 details the performance of the retrieval-augmented systems, with the zero-

	CopyNE				Multi-grained				LLM-Refine			
Top-K	CER↓	K-CER↓	U-CER↓	R↑	CER↓	K-CER↓	U-CER↓	R↑	CER↓	K-CER↓	U-CER↓	R↑
Aishell-1 *(Whisper Base | CER: 5.20 | K-CER: 10.40 | U-CER: 4.70 | R: 80.60)*												
1	4.80	5.60	**4.72**	88.19	3.97	4.90	**3.87**	89.43	4.45	7.00	**4.18**	85.77
10	**4.70**	**3.90**	4.78	**93.51**	**3.84**	**2.70**	3.96	**95.53**	**4.16**	**3.30**	4.25	**93.23**
50	4.74	4.20	4.80	92.80	3.86	3.10	3.94	94.42	4.19	3.90	4.22	91.55
100	4.78	4.35	4.82	92.53	3.90	3.35	3.96	94.06	4.27	4.40	4.26	90.80
DC *(Whisper Base | CER: 12.80 | K-CER: 22.90 | U-CER: 11.70 | R: 71.10)*												
1	12.25	18.90	**11.65**	76.62	11.40	14.76	11.09	82.05	11.51	14.50	11.24	81.56
10	**12.10**	**15.55**	11.79	**81.33**	**10.94**	**11.90**	**10.85**	**86.02**	**11.13**	**12.00**	**11.05**	**86.17**
50	12.18	15.90	11.84	80.80	11.04	12.30	10.92	85.25	11.32	12.70	11.19	85.09
100	12.26	16.10	11.91	80.44	11.22	12.90	11.07	84.80	11.36	13.10	11.20	84.98
ICI *(Whisper Base | CER: 11.50 | K-CER: 30.70 | U-CER: 6.90 | R: 40.80)*												
1	9.30	19.40	**6.95**	64.91	7.90	12.70	**6.78**	76.68	6.96	8.40	6.63	87.22
10	**8.96**	**17.30**	7.02	**69.12**	**7.71**	**11.40**	6.85	**79.19**	**5.78**	**4.85**	5.99	**93.20**
50	9.06	17.70	7.05	68.47	7.83	12.15	6.82	77.56	5.91	5.95	**5.90**	90.58
100	9.13	17.95	7.08	68.14	8.03	12.95	6.88	76.10	5.94	6.05	5.92	90.58

Table 1: Results of different contextual ASR models with JSPG retrieval. The performance of base ASR is provided in the dataset headers for direct comparison.

Ablation Phase	Weight	R@1	R@100
Stage 1: α	0.1	**18.72**	96.50
	0.7	18.68	**99.13**
	0.9	18.68	98.82
Stage 2: β	0.1	62.31	96.94
	0.4	**64.52**	**97.03**
	0.9	43.16	89.11

Table 2: Ablation on feature weights. It reveals their complementary roles: glyph heavily anchors top-1 precision, pinyin expands the top-100 recall ceiling, and semantics acts as a vital tie-breaker upon a solid phonetic-glyph foundation.

shot performance of the fine-tuned Whisper baseline provided directly in the dataset headers for direct comparison.

Improvements over Baseline Experimental results demonstrate that our framework significantly outperforms the Whisper baseline across all datasets. On the standard Aishell-1, the best-performing configuration reduces the K-CER from 10.40% to 2.70% and improves R from 80.60% to 95.53% (Multi-grained with Top-K=10). The gains are even more pronounced in the real-world datasets. On the noisy, task-oriented ICI dataset, K-CER drops dramatically from 30.70% to 4.85%, and R surges from 40.80% to 93.20% (LLM-Refine with Top-K=10). These results confirm that incorporating retrieved context via JSPG effectively mitigates the keyword recognition errors.

Performance of Different Recognition Models The effectiveness of the recognition models varies by domain. The Multi-grained model performs best on Aishell-1 dataset. However, for the highly complex ICI dataset, the LLM-Refine method (utilizing GPT-OSS-120B) achieves the lowest K-CER. This indicates that while specialized end-to-end models are efficient for general tasks, large-scale LLMs possess stronger capabilities for keyword correction in difficult scenarios. Importantly, the U-CER remains stable across all experiments, ensuring that our retrieval-based biasing does not negatively impact the recognition of general non-keyword text.

5 Conclusion

In this paper, we introduced JSPG, a novel filtering framework designed to address the challenge of utilizing large-scale contextual dictionaries in ASR. By synergizing semantic, glyph, and pinyin information, JSPG effectively bridges the gap between noisy hypotheses and target keywords, ensuring robust recall even in the presence of severe recognition errors. We believe this work offers a practical solution for deploying contextual ASR in open-domain scenarios and paves the way for future research in speech processing.

Limitations

While JSPG demonstrates promising results, we acknowledge two limitations in this work.

First, our evaluation is currently limited to the Chinese language. The specific implementation of our method relies on Chinese pinyin and character structures. However, we believe that the core principle of leveraging phonetic and structural features is universal. With appropriate modifications, we expect our approach can be adapted to other languages that face similar homophonic challenges, such as Japanese or Korean.

Second, this work mainly focuses on the retrieval stage. We aim to find the most relevant keywords to narrow down the search space. We have not optimized how the downstream contextual ASR model utilizes these keywords. Since the retrieved candidates are often phonetically or visually similar to the target, guiding the model to strictly distinguish the correct keyword from these confusing candidates is a critical step. We plan to explore better ways to utilize the retrieved context in future work.

References

Uri Alon, Golan Pundak, and Tara N Sainath. 2019. Contextual speech recognition with difficult negative training examples. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6440–6444. IEEE.

Ye Bai, Jingping Chen, Jitong Chen, Wei Chen, Zhuo Chen, Chuang Ding, Linhao Dong, Qianqian Dong, Yujiao Du, Kepan Gao, and 1 others. 2024. Seed-asr: Understanding diverse speech and contexts with llm-based speech recognition. *arXiv preprint arXiv:2407.04675*.

Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, and Hao Zheng. 2017. Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline. In *2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*, pages 1–5.

Boli Chen, Guangwei Xu, Xiaobin Wang, Pengjun Xie, Meishan Zhang, and Fei Huang. 2022. Aishell-ner: Named entity recognition from chinese speech. In *2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8352–8356.

Yifu Chen, Shengpeng Ji, Haoxiao Wang, Ziqing Wang, Siyu Chen, Jinzheng He, Jin Xu, and Zhou Zhao. 2025. WavRAG: Audio-integrated retrieval augmented generation for spoken dialogue models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12505–12523, Vienna, Austria. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Siskos Dimitrios, Stavros Papadopoulos, Pablo Peso Parada, Jisi Zhang, Karthikeyan Saravanan, and Anastasios Drosou. 2025. Retrieval augmented generation based context discovery for ASR. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 14247–14254, Suzhou, China. Association for Computational Linguistics.

Xun Gong, Anqi Lv, Zhiming Wang, Huijia Zhu, and Yanmin Qian. 2025. Br-asr: Efficient and scalable bias retrieval framework for contextual biasing asr in speech llm. *arXiv preprint arXiv:2505.19179*.

Zhihong Lei, Xingyu Na, Mingbin Xu, Ernest Pusateri, Christophe Van Gysel, Yuanyuan Zhang, Shiyi Han, and Zhen Huang. 2025. Contextualization of asr with llm using phonetic retrieval-based augmentation. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Shaojun Li, Hengchao Shang, Daimeng Wei, Jiaxin Guo, Zongyao Li, Xianghui He, Min Zhang, and Hao Yang. 2024a. La-rag: Enhancing llm-based asr accuracy with retrieval-augmented generation. *arXiv preprint arXiv:2409.08597*.

Shaojun Li, Hengchao Shang, Daimeng Wei, Jiaxin Guo, Zongyao Li, Xianghui He, Min Zhang, and Hao Yang. 2024b. La-rag: Enhancing llm-based asr accuracy with retrieval-augmented generation. *arXiv preprint arXiv:2409.08597*.

Puneet Mathur, Zhe Liu, Ke Li, Yingyi Ma, Gil Karen, Zeeshan Ahmed, Dinesh Manocha, and Xuedong Zhang. 2024. DOC-RAG: ASR language model personalization with domain-distributed co-occurrence retrieval augmentation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 5132–5139, Torino, Italia. ELRA and ICCL.

ASR Omnilingual, Gil Keren, Artyom Kozhevnikov, Yen Meng, Christophe Ropers, Matthew Setzler, Skyler Wang, Ife Adebara, Michael Auli, Can Balioglu, and 1 others. 2025. Omnilingual asr: Open-source multilingual speech recognition for 1600+ languages. *arXiv preprint arXiv:2511.09690*.

OpenAI. 2025. gpt-oss-120b and gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Golan Pundak, Tara N Sainath, Rohit Prabhavalkar, Anjuli Kannan, and Ding Zhao. 2018. Deep context: end-to-end contextual speech recognition. In *2018 IEEE spoken language technology workshop (SLT)*, pages 418–425. IEEE.

Ziheng Qiao, Houquan Zhou, Yumeng Liu, Zhenghua Li, Min Zhang, Bo Zhang, Chen Li, Ji Zhang, and Fei Huang. 2025. DISC: Plug-and-play decoding intervention with similarity of characters for Chinese spelling check. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 28312–28324, Vienna, Austria. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision. *arXiv preprint arXiv:2212.04356*.

Temple F Smith, Michael S Waterman, and 1 others. 1981. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *Advances in neural information processing systems*, 33:16857–16867.

Yui Sudo, Yosuke Fukumoto, Muhammad Shakeel, Yifan Peng, and Shinji Watanabe. 2024a. Contextualized automatic speech recognition with dynamic vocabulary. In *2024 IEEE Spoken Language Technology Workshop (SLT)*, pages 78–85. IEEE.

Yui Sudo, Muhammad Shakeel, Yosuke Fukumoto, Yifan Peng, and Shinji Watanabe. 2024b. Contextualized automatic speech recognition with attention-based bias phrase boosted beam search. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 10896–10900. IEEE.

Li Hai Tan, Angela R Laird, Karl Li, and Peter T Fox. 2005. Neuroanatomical correlates of phonological processing of chinese characters and alphabetic words: A meta-analysis. *Human brain mapping*, 25(1):83–91.

Cihan Xiao, Zejiang Hou, Daniel Garcia-Romero, and Kyu J Han. 2025. Contextual asr with retrieval augmented large language model. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, and 1 others. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Shilin Zhou and Zhenghua Li. 2025. Improving contextual asr via multi-grained fusion with large language models. *arXiv preprint arXiv:2507.12252*.

Shilin Zhou, Zhenghua Li, Chen Gong, Lei Zhang, Yu Hong, and Min Zhang. 2024a. Chinese spoken named entity recognition in real-world scenarios: Dataset and approaches. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1872–1884, Bangkok, Thailand. Association for Computational Linguistics.

Shilin Zhou, Zhenghua Li, Yu Hong, Min Zhang, Zhefeng Wang, and Baoxing Huai. 2024b. CopyNE: Better contextual ASR by copying named entities. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2675–2686, Bangkok, Thailand. Association for Computational Linguistics.

A Appendix

A.1 Algorithmic Walkthrough of Extended Smith-Waterman

To demonstrate the process of our extended SW algorithm, we present a step-by-step walkthrough here. This example computes the sequence-level similarity score between the keyword $w =$ "语音识别" (speech recognition, $|w| = 4$) and the ASR hypothesis "关于雨音的识别" (about the recognition of rain sound). Here the target character "语" is misrecognized as the homophone "雨" and an extra character "的" is inserted.

Input: Substitution Cost Matrix Table 3 presents the substitution costs between the hypothesis characters and keyword characters. As defined in Section 3.4, each cost is computed as $1 - sim_p(c_i, c_j)$, where sim_p is the pinyin similarity defined in Section 3.3. A cost of 0 indicates identical pronunciation. Notably, the homophonic pair ("雨", "语") shares identical pinyin (yu3), resulting in a substitution cost of 0.

Process: Cost Accumulation Matrix (D) Using the substitution costs from Table 3 and a gap cost of 1, we compute the cost matrix D, where $D_{i,j}$ represents the minimum alignment cost of the first j keyword characters against hypothesis characters ending at position i. The initialization is $D_{i,0} = 0$ (free start in the hypothesis) and $D_{0,j} = \infty$ for $j \geq 1$.

cost	语$_1$(yu3)	音$_2$(yin1)	识$_3$(shi2)	别$_4$(bie2)
关$_1$(guan1)	0.50	0.33	0.56	0.56
于$_2$(yu2)	0.17	0.43	0.43	0.43
雨$_3$(yu3)	**0.00**	0.43	0.57	0.57
音$_4$(yin1)	0.43	**0.00**	0.50	0.38
的$_5$(de)	0.60	0.67	0.67	0.50
识$_6$(shi2)	0.57	0.50	**0.00**	0.38
别$_7$(bie2)	0.57	0.38	0.38	**0.00**

Table 3: Substitution costs ($1 - sim_p$) derived from pinyin similarity. A cost of 0 indicates identical pronunciation.

D	$\emptyset_0$	语$_1$	音$_2$	识$_3$	别$_4$
$\emptyset_0$	0	∞	∞	∞	∞
关$_1$	0	0.50	1.50	2.50	∞
于$_2$	0	0.17	0.93	1.93	2.93
雨$_3$	0	**0.00**	0.60	1.50	2.50
音$_4$	0	0.43	**0.00**	1.00	1.88
的$_5$	0	0.60	**1.00**	0.67	1.50
识$_6$	0	0.57	1.10	**1.00**	1.04
别$_7$	0	0.57	0.95	1.48	**1.00**

Table 4: The cost matrix D. Lower values indicate better alignment. The arrows visualize the optimal alignment path.

For general positions, the recurrence is $D_{i,j} = \min(D_{i-1,j-1} + \text{cost}_{i,j},\ D_{i-1,j} + 1,\ D_{i,j-1} + 1)$. As described in Section 3.4, for the first and last keyword positions ($j = 1$ and $j = |w|$), the deletion move $D_{i,j-1} + 1$ is excluded to ensure the entire keyword is matched. The resulting matrix is shown in Table 4. We highlight the calculation for the following critical nodes.

For $D_{3,1}$: Aligning 雨$_3$ with 语$_1$. Since $j = 1$, deletion of the first keyword character is prohibited. With a substitution cost of 0 (identical pinyin), the diagonal move yields the minimum:

$$\begin{aligned} D_{3,1} &= \min(D_{2,0} + \text{cost}_{3,1},\ D_{2,1} + 1) \\ &= \min(0 + 0,\ 0.17 + 1) = \mathbf{0}. \end{aligned}$$

For $D_{4,2}$: Aligning 音$_4$ with 音$_2$. An exact match (cost $= 0$) continues the diagonal path:

$$\begin{aligned} D_{4,2} &= \min(D_{3,1} + \text{cost}_{4,2},\ D_{3,2} + 1,\ D_{4,1} + 1) \\ &= \min(0 + 0,\ 0.60 + 1,\ 0.43 + 1) = \mathbf{0}. \end{aligned}$$

For $D_{5,2}$: The insertion character 的$_5$ has a high substitution cost with 音$_2$ (cost $= 0.67$). The algorithm selects the vertical move $D_{4,2} + 1 = 1$, which inserts 的 as a gap. This effectively extends the alignment of "雨音" ↔ "语音" to handle the extra character, rather than breaking the existing match:

$$\begin{aligned} D_{5,2} &= \min(D_{4,1} + \text{cost}_{5,2},\ D_{4,2} + 1,\ D_{5,1} + 1) \\ &= \min(1.10,\ 1.00,\ 1.60) = \mathbf{1.00}. \end{aligned}$$

For $D_{6,3}$ and $D_{7,4}$: After bypassing the insertion, exact matches (cost $= 0$) resume. Since $j = 4$ at the last step, deletion is again prohibited:

$$\begin{aligned} D_{6,3} &= \min(D_{5,2} + \text{cost}_{6,3},\ D_{5,3} + 1,\ D_{6,2} + 1) \\ &= \min(1.00,\ 1.67,\ 2.10) = \mathbf{1.00}, \end{aligned}$$

$$\begin{aligned} D_{7,4} &= \min(D_{6,3} + \text{cost}_{7,4},\ D_{6,4} + 1) \\ &= \min(1.00,\ 2.04) = \mathbf{1.00}. \end{aligned}$$

Consequently, the keyword "语音识别" is successfully matched to the hypothesis part "雨音的识别", with a total alignment cost of $\mathsf{SW}_{\text{ext}} = D_{7,4} = 1.0$. The relatedness likelihood is then computed as $\mathsf{RL} = (|w| - \mathsf{SW}_{\text{ext}})/|w| = (4-1)/4 = 0.75$.

A.2 Details of Downstream Contextual ASR Models

To verify that JSPG is generally useful for downstream keyword utilization, we evaluate three contextual ASR models that represent different ways of incorporating retrieved keywords. All models receive the same JSPG Top-K keyword list for each utterance, so the comparison focuses on whether the retrieved keywords can improve recognition under different contextual ASR paradigms.

CopyNE. CopyNE (Zhou et al., 2024b) is an end-to-end contextual ASR model designed to preserve the integrity of multi-token keywords. Instead of treating a keyword as a sequence of independently generated tokens, CopyNE introduces a phrase-level copy mechanism. It encodes the external keyword list with a keyword encoder, computes a copy probability for each keyword at every decoding step, and uses the resulting dictionary representation to assist token prediction. During inference, the decoder can either generate a normal vocabulary token or copy an entire keyword from the provided list. In our experiments, the JSPG Top-K keywords are used as the contextual keyword

list for CopyNE, so this setting evaluates whether JSPG can supply a compact dictionary that enables reliable phrase-level copying.

Multi-grained. The multi-grained model (Zhou and Li, 2025) represents an LLM-enhanced contextual ASR paradigm. It combines speech information from the ASR model and contextual knowledge from an LLM at two granularities. At the token level, the model fuses token-level scores from the ASR model and the LLM, where the LLM receives the keyword list as textual prompt context. At the phrase level, it further estimates the probability of selecting a complete keyword phrase by combining ASR hidden states, LLM hidden states, and keyword representations. The final decoder jointly uses token-level generation and phrase-level keyword selection, aiming to benefit from both fine-grained token prediction and complete keyword copying. In our experiments, the JSPG Top-K keywords are provided as the contextual keyword list, allowing us to test whether JSPG improves recognition in a model that explicitly fuses ASR and LLM knowledge at multiple granularities.

LLM-Refine. LLM-Refine represents a two-stage correction paradigm. Different from the above end-to-end models, it does not re-decode the input speech. Instead, the base ASR model first generates N-best hypotheses, and GPT-OSS-120B (OpenAI, 2025) then refines the final transcription using both the N-best hypotheses and the JSPG Top-K keywords as textual evidence. The refinement prompt is composed of three parts: the ASR-predicted N-best hypotheses, a list of keywords that may appear in the speech, and an instruction asking the LLM to generate the final ASR transcription based on both sources. Concretely, the prompt follows the template: "The following are the ASR-predicted hypotheses: {N-best hypotheses}. The following keywords may appear in the speech: {retrieved keywords}. Please correct the recognition errors using these keywords and generate the final ASR transcription." This setting evaluates whether the retrieved keywords are also useful for post-recognition correction with a large language model.