HEISIR: Hierarchical Expansion of Inverted Semantic Indexing for Training-free Retrieval of Conversational Data using LLMs

Sangyeop Kim[†1,2], **Hangyeul Lee**[2], **Yohan Lee**[1]
[1] Coxwave
[2] Seoul National University
sangyeop.kim@coxwave.com, mikelee@snu.ac.kr, yohan.lee@coxwave.com

Abstract

The growth of conversational AI services has increased demand for effective information retrieval from dialogue data. However, existing methods often face challenges in capturing semantic intent or require extensive labeling and fine-tuning. This paper introduces HEISIR (Hierarchical Expansion of Inverted Semantic Indexing for Retrieval), a novel framework that enhances semantic understanding in conversational data retrieval through optimized data ingestion, eliminating the need for resource-intensive labeling or model adaptation. HEISIR implements a two-step process: (1) Hierarchical Triplets Formulation and (2) Adjunct Augmentation, creating semantic indices consisting of Subject-Verb-Object-Adjunct (SVOA) quadruplets. This structured representation effectively captures the underlying semantic information from dialogue content. HEISIR achieves high retrieval performance while maintaining low latency during the actual retrieval process. Our experimental results demonstrate that HEISIR outperforms fine-tuned models across various embedding types and language models. Beyond improving retrieval capabilities, HEISIR also offers opportunities for intent and topic analysis in conversational data, providing a versatile solution for dialogue systems.

1 Introduction

Conversational AI is being deployed across diverse industries, including personalized services (Kocaballi et al., 2019), code generation (Li et al., 2022), educational tutoring (Mousavinasab et al., 2021) and even in social welfare services (Jo et al., 2023). This rapid expansion has resulted in vast amounts of dialogue data, rich with insights into user needs and behavioral patterns. To harness this wealth of information, Information Retrieval approach tailored to the unique characteristics of conversational data is necessary. We define this area as *Conversational Data Retrieval* (CDR), which aims to enable efficient access to and extraction of relevant information from large-scale conversational data.

CDR is highly important for both end-users and conversational AI service providers. End-users frequently need to retrieve specific information from past conversations, such as previous agreements or discussion outcomes, using natural language queries. Service providers can utilize CDR to analyze user interactions and enhance their AI-powered services. For example, CDR can enhance Retrieval Augmented Generation (RAG) systems by providing relevant dialogue examples (Lewis et al., 2020; Wang et al., 2024), and support service improvement by identifying patterns in user behavior and recurring topics (Motger et al., 2022; Owoicho et al., 2022).

However, conventional sparse and dense retrieval methods exhibit limited performance when applied to conversational data. Unlike traditional document data, conversational data has distinct characteristics that make traditional retrieval approaches insufficient. First, each utterance is assigned to a specific speaker, meaning that the same message can be expressed in entirely different ways depending on who is speaking. Additionally, a speaker's intention can be multifaceted; a single utterance may have multiple, often complex, intents in it. Furthermore, the meaning of utterances depends heavily on the context of the dialogue and the relationship between participants, often without explicit contextual markers. These unique characteristics pose significant challenges that existing context-aware retrieval models, trained primarily on document-based QA tasks (Yang et al., 2015; Rajpurkar et al., 2016; Nguyen et al., 2016), struggle to address.

Apart from the limited effectiveness of existing retrieval methods on conversational data, the structure of the retrieval system itself poses a significant challenge to practical implementation. A typical re-

[†]Corresponding author.

Findings of the Association for Computational Linguistics:
NAACL 2025, pages 7143–7159
April 29 - May 4, 2025 ©2025 Association for Computational Linguistics

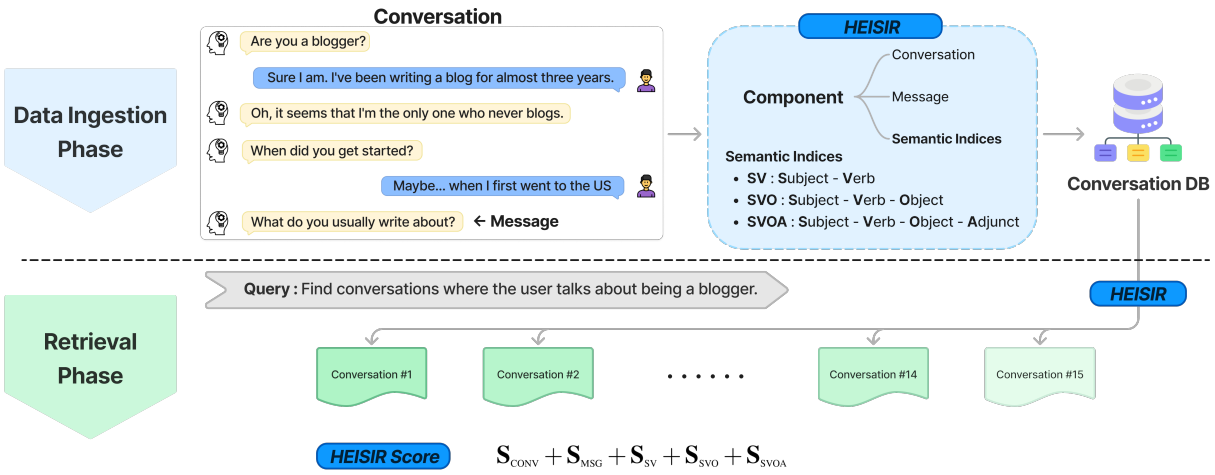

Figure 1: Architecture of HEISIR framework: Data Ingestion Phase

trieval system consists of two main phases (Wang et al., 2021): the data ingestion phase, where incoming data is processed and indexed offline, and the retrieval phase, where relevant information is searched based on user queries. Recent research on retrieval system leverages the powerful context understanding capability of LLMs through techniques such as re-ranking (Zhang et al., 2023), query rewriting (Yu et al., 2020), and using larger retrieval models (Ma et al., 2023; Peng et al., 2023). However, these approaches, especially when incorporating LLMs, introduce significant latency trade-off in the retrieval phase, making them impractical for real-time services.

Building on these insights, we propose a novel framework that extracts and processes the inverted semantic indices in the data ingestion phase, unlike traditional approaches that process indices in the retrieval phase. **HEISIR** (**H**ierarchical **E**xpansion of **I**nverted **S**emantic **I**ndexing for **R**etrieval) implements structured semantic indices that capture the inherent syntactic hierarchy within sentences. Overall scheme of our framework is detailed in Figure 1. HEISIR constructs search indices based on the inherent syntax present in all natural language sentences, eliminating the need for extensive labeling and training. In data ingestion phase, HEISIR extracts semantic indices and stores them as *inverted indices*. In retrieval phase, HEISIR computes score to retrieve the most relevant conversations. Our approach not only enhances retrieval performance but also offers practical advantages, as it eliminates latency during the retrieval phase. The key contributions of this research are:

1. **Improved retrieval performance** significantly enhances retrieval capabilities with only negligible increase of latency by optimizing data ingestion.
2. **Practical real-world applicability** enables deployment of effective dialogue retrieval systems in production environments lacking labeled training data.
3. **Practical-scale Robustness** consistently enhances performance when integrated with any combination of language model scales.
4. **Versatility beyond retrieval** provides highly interpretable atomic semantic units, enabling intuitive intent and topic analysis.

2 Linguistic Preliminaries

To design semantic indices for CDR, it is necessary to identify two fundamental components in a message: the speaker and the intent. These components correspond to the subject and the verb of a sentence. To capture complex intents while maintaining the structural integrity, HEISIR follows the syntactic comprehension process of humans.

Numerous studies support the view that humans comprehend sentences incrementally. Specifically, sentences are processed by first constructing the simplest syntactic structure and then integrating semantic adjuncts to achieve full understanding (Kamide et al., 2003; Altmann and Mirković, 2009; Fossum and Levy, 2012). **Phrase Structure Grammar (PSG)** (Chomsky, 2002; Gazdar, 1985) is one of the most widely used models to explain this top-down syntactic processing. PSG breaks down natural language sentences into **constituents**—syntactically significant

units—organized in a hierarchical binary tree structure. The overall scheme of how PSG parses messages for HEISIR is detailed in Figure 2.

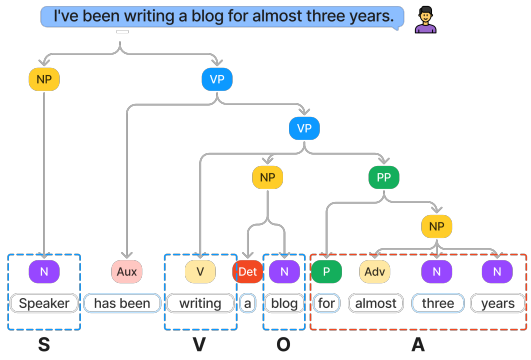

Figure 2: Phrase Structure Grammar and Constituents

Another key concept central to syntactic processing is **verb valency**. Verb valency refers to the number of arguments that a verb can take in a sentence, and it describes the relationship between the verb and other elements in the clause, such as the subject, direct object, and indirect object. Verbs can be categorized based on their valency as **avalent** (taking no arguments), **monovalent** (taking one argument), **divalent** (taking two arguments), or **trivalent** (taking three arguments). Verb valency plays a critical role in determining the constituent hierarchy in sentences, as different valency types lead to different phrase structures.

In this study, we develop a semantic indexing framework according to hierarchical syntactic processing schemes that represent the three most common types of verb valency: monovalent, divalent, and trivalent. We parse the constituents from sentences as follows:

- **Subject**: The entity performing the action or the one that the sentence is about.
- **Verb**: The action or state described in the sentence.
- **Object**: The entity directly affected by the verb's action.
- **Adjunct**: An optional or necessary element that provides additional information about the action, such as time, place, or manner.

HEISIR borrows these concepts from syntax, but uses them in a slightly different context. For instance, HEISIR fixes the **Subject** to the speaker of utterance, to perform information retrieval in conversational data. Therefore, avalent verbs are disregarded due to the existence of an explicit subject. This makes index tuples include at least two constituents: the **Subject** and the **Verb**. Furthermore, we use the word **Adjunct** in a slightly broader context than is conventional; For divalent verbs that take two objects, it is syntactically correct to allocate both direct and indirect objects under the **Object** constituent. Instead, we include indirect object into the **Adjunct** category for structural integrity and better search performance.

3 Hierarchical Expansion of Inverted Semantic Indexing for Retrieval

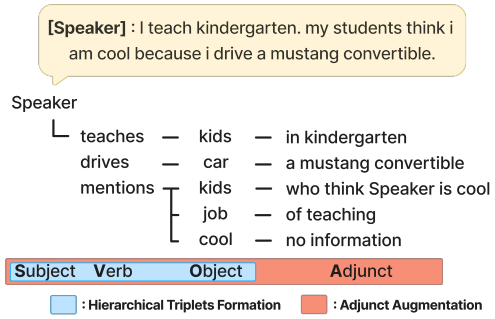

Figure 3: 2-Step Expansion Process of HEISIR

HEISIR is a novel conversational data retrieval framework that incorporates incremental syntactic processing with inverted indexing. The framework invests resources in the data ingestion phase, allowing for optimized performance during the retrieval phase without sacrificing latency.

Figure 3 outlines two key steps of the HEISIR framework: (1) **Hierarchical Triplets Formulation**, identifying the fundamental syntactic constituents – subject, verb, and object – forming the core structures of the sentence; (2) **Adjunct Expansion**, expanding SVO triplets to SVOA quadruplets, where A represents an Adjunct.

While a single-step approach for SVOA quadruplet extraction may seem straightforward, it has critical limitations in accuracy and control. For example, it generates redundant variations like "teaches kids at kindergarten" and "teaches children in kindergarten" that carry the same meaning, introducing unnecessary index noise.

In contrast, HEISIR's two-step approach first extracts core SVO triplets and then carefully adds adjuncts, effectively preventing redundancy while

maintaining high-quality indices. Furthermore, the experimental results supporting this analysis are presented in Section 6.2. The detailed prompts used in our approach can be found in Appendix A.

3.1 Data Ingestion Phase

Step 1: Hierarchical Triplets Formulation In this step, HEISIR breaks down sentences into one or more **SVO triplets**. These triplets capture the core syntactic hierarchy within the dialogue content, addressing two key limitations of traditional embedding methods (Mikolov, 2013; Pennington et al., 2014). First, HEISIR reduces ambiguity in the embeddings of complex sentences. HEISIR decomposes messages into multiple SVO triplets, significantly reducing sentence complexity. Additionally, HEISIR strictly excludes pronouns to ensure semantic completeness in each index. Second, our approach minimizes the impact of non-semantic elements on performance. As discussed in section 2, HEISIR fixes the subject to the speaker of the message to enhance retrieval performance. During this syntactic transformation, avalent verbs are transformed into their mono-, di-, or trivalent equivalents, and all tenses, auxiliaries, and syntactic markers are removed. This process refines HEISIR-generated triplets to capture only semantically significant elements of the message.

Step 2: Adjunct Augmentation SVO triplet indices extracted in step 1 identify hierarchical structures in the message. To enhance specificity, we introduce a *Adjunct Augmentation* process. We categorize four distinct patterns of adjuncts:

- **Detailed Content and Theme of Discussion** specifies the target of communication or conversational themes using prepositions.
- **Reason or Causation** explains the underlying causes or reasons for actions or situations, addressing the "why" of a scenario.
- **Condition or Accompanying Circumstance** describes the context or conditions under which actions occur or situations exist.
- **No information** indicates cases where adding detailed information is impossible or not meaningful for the given sentence structure.

Following this step, each message in a conversation is encoded into one or more SVOA quadruplets. These SVOA quadruplets break down complex messages into comprehensible semantic units. The indices are then stored in an inverted index structure, which facilitates efficient retrieval in the subsequent phase.

3.2 Retrieval Phase: Scoring

After deriving SVOA quadruplets from a conversation, we devise a method to collectively evaluate embeddings of five conversational components — conversation, message, SV, SVO, SVOA — in a single score metric.

To formalize our scoring method, we represent queries and conversations with embeddings E_q and E_{conv} respectively, while other conversational components $C = \{\text{message, SV, SVO, SVOA}\}$ are encoded as E_c. The relevance between components is computed using a vector similarity function $f(\cdot, \cdot)$. The scoring process is defined as follows:

$$S_{\text{conv}} = f(E_q, E_{\text{conv}}) \tag{1}$$

$$S_c = \max_{E_{c_m}} f(E_q, E_{c_m}) \text{ for } c \in C \quad \text{where } E_{c_m} \in \mathbb{E}_c(m), m \in \text{conv} \tag{2}$$

$$S_{\text{HEISIR}} = S_{\text{conv}} + \sum_{c \in C} S_c \tag{3}$$

Equation (1) measures the semantic similarity between the query and the entire conversation content to capture conversation-level relevance. Component scores are computed as shown in (2), where we evaluate each type separately by finding the highest similarity between the query and component instances within the conversation. For each conversational component $c \in C$, $\mathbb{E}_{c_m}$ denotes a set of embeddings of component c in the message m, and $\max_{E_{c_m}}$ picks best from multiple <component, message> pairs. Finally, the overall HEISIR score in (3) is determined by aggregating the conversation-level similarity with individual component scores to provide a comprehensive measure of relevance.

Potential alternatives of maximization function in (2) are summation and averaging. However, we employ maximization instead of these alternatives since summation and averaging introduce noise from less relevant messages, whereas focusing on the most salient component is intuitively more effective.

4 Experiment

4.1 Dataset

We select five dialogue datasets addressing key topics in conversational data analysis: profanity detec-

tion, user satisfaction evaluation, and personal information protection. We extract queries from five datasets and map them to corresponding conversation sessions. The statistics of the derived dataset is detailed in Table 1.

- **BAD** (Xu et al., 2021) focuses on improving safety of conversational agents.
- **DICES** (Aroyo et al., 2024) evaluates safety of AI responses in diverse contexts.
- **Daily Dialog** (Li et al., 2017) provides labeled multi-turn dialogues reflecting user sentiments.
- **PILD** (Xu et al., 2020) detects and protects sensitive personal information.
- **USS** (Sun et al., 2021) simulates user satisfaction in task-oriented dialogue systems.

Dataset	Conv.	Query	Utterances per Conv.	Mapped Conv. per Query
Train	4,096	3,000	12.3	15.5
Validation		590		14.9
Test	4,035	3,501	12.7	15.4

Table 1: Statistics of Dataset

4.2 Baseline

For the evaluation of Information Retrieval, we compare the following baseline models:

- **DPR** (Karpukhin et al., 2020): Learns dense embeddings for queries and passages using a dual-encoder architecture for efficient retrieval.
- **SPLADE-v3** (Lassance et al., 2024): Employs sparse lexical representations for expansion-based retrieval, enhancing earlier SPLADE versions (Formal et al., 2021).
- **LLM2Vec** (BehnamGhader et al., 2024): Creates dense vector representations of text using large language models to facilitate retrieval tasks.

For training-free models, we utilize the following in our comparison:

- **CoT Expansion** (Jagerman et al., 2023): Expands queries using CoT prompting and uses both original and expanded queries for retrieval.
- **HyDE** (Gao et al., 2022a): Generates hypothetical relevant documents using an LLM to enhance retrieval performance.
- **LameR** (Shen et al., 2023a): Produces hypothetical documents using BM25 initial results to improve retrieval effectiveness.

4.3 Experimental Setup

For context consideration, we set a window of $k = 2$ previous messages. We set LLMs temperature to 0.0, and use cosine similarity for embedding comparisons. All implementation details, including the versions of the embedding models and LLMs used, can be found in Appendix D.

Embedding for HEISIR For this study, We select embedding models based on their performance in the MTEB benchmark (Muennighoff et al., 2022). We explore Encoder-only models with extended context capabilities, such as GTE (Li et al., 2023) and Nomic (Nussbaum et al., 2024). Another category of embedding models is decoder-only models based on LLMs. Specifically, we utilize LLM2Vec (BehnamGhader et al., 2024) with its LLama-3 (AI@Meta, 2024) based version. Additionally, we use NV-Embed (Lee et al., 2024), which currently achieves the highest average MTEB benchmark score. For comprehensive evaluation, We employ latest OpenAI-small and large embedding models (OpenAI, 2024).

LLMs for HEISIR The study uses a range of LLM models for different computational environments. For low-resource settings, we assess Gemma (2B) (Team et al., 2024), Phi-3 (mini, 3.8B) (Abdin et al., 2024), LLama-3 (8B) (AI@Meta, 2024), and Qwen-2 (7B) (Qwen, 2024), which are suitable for environments with limited hardware. We also test API-based models including GPT-3.5-turbo (OpenAI, 2023) and Claude 3 Haiku (Anthropic, 2024).

5 Result

5.1 Baseline Result

Type	Model	acc@1	ndcg@5	ndcg@10	ndcg@20
Baseline	DPR	0.0337	0.0274	0.0257	0.0283
	SPLADE-v3	0.2048	0.1696	0.1614	0.1743
	LLM2Vec	0.2825	0.2225	0.2092	0.2220
Fine-tuned	DPR	0.1708	0.1420	0.1342	0.1444
	SPLADE-v3	0.2474	0.2015	0.1908	0.2034
	LLM2Vec	0.3533	0.2881	0.2745	0.2912
HEISIR	GPT-3.5-turbo + OpenAI-large	**0.4085**	**0.3260**	**0.3056**	**0.3198**

Table 2: Performance metrics of baseline, fine-tuned, and best models

As shown in Table 2, models that typically excel in general document retrieval struggle to achieve

	Model	GTE (Encoder-only)				LLM2Vec (Decoder-only)				OpenAI-large (API)			
		acc@1	ndcg@5	ndcg@10	ndcg@20	acc@1	ndcg@5	ndcg@10	ndcg@20	acc@1	ndcg@5	ndcg@10	ndcg@20
Base	Pre-trained	0.3156	0.2550	0.2394	0.2520	0.2825	0.2225	0.2092	0.2220	0.3425	0.2826	0.2670	0.2839
	Fine-tuned	0.3708	0.2994	0.2823	**0.2984**	0.3533	0.2881	0.2745	0.2912	-	-	-	-
Mini	Gemma	0.3096	0.2562	0.2415	0.2541	0.3136	0.2597	0.2428	0.2543	0.3422	0.2733	0.2574	0.2705
	Phi-3	0.3582	0.2934	0.2757	0.2901	0.3865	0.3138	0.2936	0.3055	0.3950	0.3192	0.2992	0.3136
Small	LLama-3	**0.3728**	**0.3005**	**0.2829**	0.2958	0.3902	0.3155	0.2944	0.3073	0.4053	**0.3263**	**0.3072**	**0.3207**
	Qwen-2	0.3613	0.2952	0.2786	0.2921	0.3879	0.3123	0.2912	0.3035	0.4045	0.3224	0.3026	0.3166
API	GPT-3.5-turbo	0.3633	0.2943	0.2776	0.2930	0.3916	0.3140	0.2934	0.3068	**0.4085**	0.3260	0.3056	0.3198
	Haiku	0.3716	0.2978	0.2809	0.2943	**0.3927**	**0.3161**	**0.2958**	**0.3086**	0.4056	0.3245	0.3028	0.3185

Table 3: Performance comparison of different models across Encoder-only (GTE), Decoder-only (LLM2Vec), and API (OpenAI-large) approaches: Underlined values outperform the Fine-tuned model, bold values indicate the best performing LLM model combination for each embedding model.

high performance in conversational data retrieval. The LLM2Vec model (BehnamGhader et al., 2024), which is based on a decoder-only architecture specialized for dialogue generation, outperforms other models, indicating that embedding-based retrieval methods are more effective for semantic search in the context of conversational data. Fine-tuning alone does not sufficiently address this issue, suggesting the need for retrieval methods specifically tailored to conversational data.

5.2 Main Result

In this section, we only discuss the best-performing combinations of LLMs and embedding models. Results for all other combinations also show the effectiveness of HEISIR, with details available in Appendix B.

Table 3 demonstrates the effectiveness of HEISIR method in conversational data retrieval across various model architectures and embedding types. Our model, in combination with various LLMs, outperforms all pre-trained encoder-only and decoder-only baselines and API models, except for when paired with Gemma. Notably, many of these models even surpass the fine-tuned baselines, highlighting the key strength of our method: achieving high performance without a need for resource-intensive data labeling and training. These results provide insights into optimal combinations for various constraints. OpenAI-large with GPT-3.5-turbo achieve the highest overall performance. For environments prohibiting external APIs, LLama-3 + LLM2Vec perform best, though other combinations show comparable results, offering flexibility. In resource-constrained settings, LLama-3 + GTE provides an efficient balance of performance and resource usage.

Model	acc@1	ndcg@5	ndcg@10	ndcg@20	Time (s)
BM25	0.1465	0.1233	0.1200	0.1299	0.0045
CoT Expansion	0.1225	0.0963	0.0894	0.0926	2.0059
HyDE	0.2702	0.2139	0.2026	0.2128	2.6192
LameR	0.2245	0.1853	0.1752	0.1866	3.5465
LLM2Vec	0.2825	0.2225	0.2092	0.2220	0.2091
$\text{HEISIR}_{\text{LLM2Vec}}$	0.3902	0.3155	0.2944	0.3073	0.2778
OpenAI-large	0.3425	0.2826	0.2670	0.2839	0.5966
$\text{HEISIR}_{\text{OpenAI-large}}$	0.4085	0.3260	0.3056	0.3198	0.7334

Table 4: Performance Comparison of Different Models

5.3 Practical Retrieval Efficiency

Table 4 demonstrates that HEISIR outperforms various training-free methods in both speed and accuracy. Methods that rely on LLMs for query expansion during the search phase, such as CoT Expansion (Jagerman et al., 2023), LameR (Shen et al., 2023b), and HyDE (Gao et al., 2022b), require significantly longer search times. In contrast, HEISIR adds minimal time during retrieval phase (0.0687 seconds to LLM2Vec, 0.1367 seconds to OpenAI-large) by requiring only simple score computation. This efficiency is achieved by shifting the computational load to the data ingestion phase, which takes 3.86 seconds with GPT-3.5-turbo and 1.22 seconds with LLama-3. These ingestion times are reasonable considering they can be significantly reduced through parallelization techniques. Additionally, HEISIR is more cost-effective. While other methods incur costs per search, HEISIR only requires a one-time data processing step during the data ingestion phase. As search frequency increases, HEISIR remains economically efficient, whereas costs for other methods continue to rise. LLM inference for indexing incurs an additional cost of approximately $0.00894 per conversation, based on the latest API pricing.

An interesting trend was observed in the experimental results: LameR, which is designed as an

improvement over HyDE, shows inferior performance than HyDE. LameR integrates BM25 for higher retrieval performance. This unexpected result suggests that while BM25 may be well-suited for traditional document retrieval, it appears to hinder performance in conversational data retrieval.

6 Analysis

6.1 Interacting with Each Component

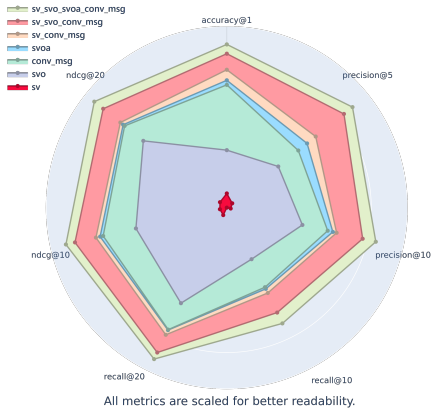

Figure 4: Marginal Performance of Components

Figure 4 shows the average performance across all experimental settings. Each contour represents a combination of the conversational components: Conversation, Message, SV, SVO, SVOA. Notably, SVOA index embeddings outperforms existing retrieval methods that rely on conversation and message embeddings. The best performance is achieved when embeddings of all conversational components are combined, demonstrating the effectiveness of HEISIR.

6.2 2-step index construction

The key strength of HEISIR lies in its 2-step SVOA quadruplet extraction process. This approach offers both qualitative and quantitative advantages over single-step extraction methods. First, the 2-step approach extracts SVOA quadruplets with greater precision. The **Hierarchical Triplet Formulation** step establishes the syntactic hierarchy within the message, while the **Adjunct Augmentation** step collects detailed information, simulating the incremental sentence comprehension process of humans. This method consistently produces higher-quality SVOA quadruplets compared to single-step extraction. Second, quantitative evaluations show that our 2-step approach consistently outperforms the single-step method (Table 5).

Model		Metric			
LLM	**embedding**	**acc@1**	**ndcg@5**	**ndcg@10**	**ndcg@20**
Phi-3	GTE	0.3530 (-1.45%)	0.2883 (-1.74%)	0.2724 (-1.20%)	0.2856 (-1.55%)
	LLM2Vec	0.3405 **(-11.90%)**	0.2724 **(-13.19%)**	0.2559 **(-12.84%)**	0.2692 **(-11.88%)**
	OpenAI-large	0.3879 (-1.80%)	0.3149 (-1.35%)	0.2970 (-0.74%)	0.3137 (+0.03%)
LLama-3	GTE	0.3513 **(-5.77%)**	0.2872 (-4.43%)	0.2713 (-4.10%)	0.2843 (-3.89%)
	LLM2Vec	0.3362 **(-13.84%)**	0.2710 **(-14.11%)**	0.2544 **(-13.59%)**	0.2693 **(-12.37%)**
	OpenAI-large	0.3865 (-4.64%)	0.3106 (-4.81%)	0.2957 (-3.74%)	0.3123 (-2.62%)
GPT-3.5-turbo	GTE	0.3510 (-3.39%)	0.2844 (-3.36%)	0.2682 (-3.39%)	0.2823 (-3.65%)
	LLM2Vec	0.3359 **(-14.22%)**	0.2696 **(-14.14%)**	0.2533 **(-13.67%)**	0.2673 **(-12.88%)**
	OpenAI-large	0.3876 **(-5.12%)**	0.3126 (-4.11%)	0.2954 (-3.34%)	0.3111 (-2.72%)

Table 5: Single-step Performance (% Indicates Performance Change Compared to 2-step)

6.3 Exploring the effects of score ensembles

While our results confirm that combining the extracted semantic indices enhances retrieval performance, this improvement can be claimed as a simple *ensemble effect*. In this subsection, we compare the ensemble effect with HEISIR. To measure the ensemble effect, we combine scores from all embedding models for the conversation and message components, excluding the significantly underperforming NV-Embed model (Lee et al., 2024) to avoid underestimating the ensemble effect. For comparison, we apply the HEISIR method to the Phi-3 model (Abdin et al., 2024), which is the smallest and least performant among our HEISIR variants. As shown in Table 6, HEISIR outperforms the ensemble effect, even in its least optimal setting. This demonstrates that HEISIR adds value beyond simple score aggregation.

combination	acc@1	ndcg@5	ndcg@10	ndcg@20
ensemble effect	0.3445	0.2784	0.2651	0.2802
(- NV-Embed)	0.3522	0.286	0.2715	0.2887
$\text{HEISIR}_{\text{Phi-3,GTE}}$	0.3582	0.2934	0.2757	0.2901
$\text{HEISIR}_{\text{Phi-3,LLM2Vec}}$	0.3865	0.3138	0.2936	0.3055
$\text{HEISIR}_{\text{Phi-3,OpenAI-large}}$	**0.3950**	**0.3192**	**0.2992**	**0.3136**

Table 6: Comparison of Ensemble Effects

6.4 Potential to Hybrid Search

In section 6.1, we observed that progressively adding conversational components improved retrieval performance. Following this approach, we

experimented with the addition of keywords to determine whether they enhance performance. Previous research on hybrid search methods report promising performance improvements by incorporating keyword to semantic search (Kuzi et al., 2020; Shen et al., 2023b). We explore addition of BM25 score into various HEISIR settings, with results shown in Table 7.

The performance results reveal interesting patterns in the integration of BM25 (Robertson et al., 1995) with various models. When combined with weaker models like Gemma and NV-Embed, BM25 provided slight improvements. However, its integration with stronger models ($\text{HEISIR}_{\text{others}}$) led to performance declines. This trend aligns with our previous observations in Section 5.3. These findings suggest that BM25's simplistic relevance estimation may struggle to capture the complex semantic relationships present in conversational contexts.

Model	acc@1	acc@5	ndcg@10	ndcg@20
Only BM25	0.1465	0.3610	0.1200	0.1299
$\text{HEISIR}_{\text{Gemma, All-embeddings}}$	0.3048	0.5919	0.2315	0.2430
+ BM25 score	**0.3105**	**0.5939**	**0.2346**	**0.2465**
$\text{HEISIR}_{\text{All-LLMs,NV-Embed}}$	0.2806	0.5542	0.2107	0.2196
+ BM25 score	**0.2912**	**0.5688**	**0.2189**	**0.2279**
$\text{HEISIR}_{\text{others}}$	**0.3822**	**0.6849**	**0.2886**	**0.3020**
+ BM25 score	0.3767	0.6791	0.2858	0.2997

Table 7: Metric Changes with and without BM25

6.5 Optimization of HEISIR through weighted sum

HEISIR currently calculates the final score by equally summing all component scores. However, components may contribute differently to sentence meaning and overall performance, suggesting potential benefits from a weighted sum approach. Our experiments using random search demonstrate clear potential for performance improvement through weight optimization, as shown in Table 8. These weights can be further refined for specific domains, potentially yielding even better results.

6.6 Applications of Semantic Indexing

Utilizing HEISIR for retrieval offers two key advantages beyond performance. First, it enhances interpretability: the highest-scoring SVOA indices provide structured, detailed insights into retrieval results. Second, HEISIR enables easy and effective result modification. Traditional retrieval systems often struggle to exclude unwanted results across semantically similar queries, but HEISIR overcomes this by allowing the removal of specific semantic indices responsible for undesired results,

Model	Baseline		Weighted	
	acc@1	ndcg@20	acc@1	ndcg@20
OpenAI-large + GPT-3.5-turbo	0.4085	0.3198	**0.4179**	**0.3202**
LLama-3 + LLM2Vec	0.3902	0.3073	**0.4050**	**0.3097**

Table 8: Comparison of model performance with baseline and weighted sum approaches

Furthermore, the SV, SVO, and SVOA indices provide a guidance to user intent and conversation topics, aiding dialogue state tracking. Analysis of SI embeddings from OpenAI-large reveals clear clustering patterns that reproduce most intent labels in the USS dataset and encompass intents from other datasets (Table 9). This demonstrates the potential of HEISIR for automatic intent classification without fine-tuning. The potential for topic analysis using SVOA is further explored in Appendix C.

Intent Label	Representative Samples
Rejection	Declines, denies, refuses, rejects
Suggestion & Planning	Suggests, wants, plans, advises, offers
Expression & Description & Explanation	Expresses, describes, implies, explains
Request	Requests, asks for, needs, seeks
Preference	Likes, loves, enjoys, dislikes
Acknowledgment & Gratitude	Acknowledges, greets, thanks, appreciates
Inquiry & Curiosity	Inquires about, wants to know, wonders
Mention & Specification	States, specifies, confirms, clarifies
Question	Asks, questions
Mention	Mentions

Table 9: Intent Clusters and Representative Samples

7 Related Work

Semantic Inverted Indexing for Retrieval Traditional retrieval methods streamline text searching with inverted indices (Zobel et al., 1998; Gormley and Tong, 2015; Dey et al., 2024), while knowledge graph embeddings (KGEs) focus on revealing the innate structure of questions and answers (Wang et al., 2017; Bordes et al., 2014). However, term-based approaches fail to capture context in multi-turn conversations, and KGEs struggle with representing implicit relationships (Ge et al., 2024). There are very few researches that utilized SVO constituents as semantic indices, but were limited to simplistic SVO structure losing all semantic adjuncts (Burek et al., 2007; Gao and Wang, 2009). Our research bridges these gaps with hierarchically expanded semantic indices, offering conversation-centric and detailed semantic representations than existing approaches.

Semantic Search Semantic search models have shown capabilities in handling the semantic complexity in natural language queries. Encoder-based models, such as SBERT (Reimers and Gurevych, 2019), DPR (Karpukhin et al., 2020) and SimCSE (Gao et al., 2021), have shown high performance upon their introduction but lacked scalability in general. Current research efforts, including SGPT (Muennighoff, 2022), RepLLaMA (Ma et al., 2023), leverage LLMs to address the limitation of encoder-based approaches. However, LLM-based models typically require substantial computational resources for training and deployment. In contrast, HEISIR can achieve high performance without the need for labeling and training.

8 Conclusion

In this paper, we introduced HEISIR, a novel framework for conversational data retrieval that reflects human sentence comprehension by capturing the syntactic hierarchy in natural language. HEISIR employs a 2-step extraction process to significantly improve the precision of semantic indices. Furthermore, by building semantic understanding from natural language syntax, HEISIR alleviates the need for extensive labeling and training.

Our experiments demonstrate that HEISIR consistently enhances retrieval performance across a variety of settings. Additionally, the inherent interpretability of the method offers significant advantages, making intent and topic analysis in dialogue datasets more accessible, thus providing a versatile solution for conversational AI systems. As chat-based services continue to grow rapidly, our findings have the potential to greatly enhance conversational data retrieval for both end-users and service providers.

Limitations

Multilingual: HEISIR extracts quadruplets from messages based on the syntax of English, which is an isolating language. In isolating languages, word order plays a crucial role in semantic comprehension since each word typically contains only one morpheme. This suggests that HEISIR may encounter challenges when extracting SVOA quadruplets from agglutinative and fusional languages, where word order does not directly reflect the syntactic hierarchy.

Expanding HEISIR to Document Data: Extending HEISIR from conversations to general documents presents several challenges. Conversational data have clear speakers as subjects in each utterance. However, documents often use passive voice or describe third-party actions, creating complex subject relationships that are difficult to parse. Furthermore, while conversations typically contain clear actions, documents tend to focus more on describing states and concepts, which makes SVOA analysis more challenging. This complexity necessitates advanced prompting and preprocessing techniques to accurately capture the subject and context. Additionally, it becomes important to account for avalent verbs in document retrieval, which are not considered in conversational data.

Weight Optimization: HEISIR currently employs simple summation to compute similarity scores. However, performance improvements were observed when assigning different weights to each conversational component. While uniform weight aggregation already outperforms current state-of-the-art retrieval models, further exploration is needed to optimize these weights for maximum efficiency.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

AI@Meta. 2024. Llama 3 model card.

Gerry TM Altmann and Jelena Mirković. 2009. Incrementality and prediction in human sentence processing. *Cognitive science*, 33(4):583–609.

Anthropic. 2024. Claude 3 haiku: Our fastest model yet. Accessed: 2024-07-06.

Lora Aroyo, Alex Taylor, Mark Diaz, Christopher Homan, Alicia Parrish, Gregory Serapio-García, Vinodkumar Prabhakaran, and Ding Wang. 2024. Dices dataset: Diversity in conversational ai evaluation for safety. *Advances in Neural Information Processing Systems*, 36.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. Llm2vec: Large language models are secretly powerful text encoders. *arXiv preprint arXiv:2404.05961*.

Antoine Bordes, Jason Weston, and Nicolas Usunier. 2014. Open question answering with weakly supervised embedding models. In *Machine Learning and*

Knowledge Discovery in Databases: European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings, Part I 14, pages 165–180. Springer.

Gaston Burek, Christian Pietsch, and Anne De Roeck. 2007. Svo triple based latent semantic analysis for recognising textual entailment.

Noam Chomsky. 2002. *Syntactic structures*. Mouton de Gruyter.

Snehasis Dey, Bhimasen Moharana, Utpal Chandra De, Tapaswini Samant, Trupti Mayee Behera, and Shobhan Banerjee. 2024. Search engine for qna using distributed inverted index system. In *2024 3rd International Conference for Innovation in Technology (INOCON)*, pages 1–4. IEEE.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. Splade: Sparse lexical and expansion model for first stage ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2288–2292.

Victoria Fossum and Roger Levy. 2012. Sequential vs. hierarchical syntactic models of human incremental sentence processing. In *Proceedings of the 3rd workshop on cognitive modeling and computational linguistics (CMCL 2012)*, pages 61–69.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2022a. Precise zero-shot dense retrieval without relevance labels. *arXiv preprint arXiv:2212.10496*.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2022b. Precise zero-shot dense retrieval without relevance labels. *arXiv preprint arXiv:2212.10496*.

Ming Gao and Ji-Cheng Wang. 2009. Semantic search based on svo constructions in chinese. In *2009 International Conference on Web Information Systems and Mining*, pages 213–216. IEEE.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Gerald Gazdar. 1985. *Generalized phrase structure grammar*. Harvard University Press.

Xiou Ge, Yun Cheng Wang, Bin Wang, C-C Jay Kuo, et al. 2024. Knowledge graph embedding: An overview. *APSIPA Transactions on Signal and Information Processing*, 13(1).

Clinton Gormley and Zachary Tong. 2015. *Elasticsearch: the definitive guide: a distributed real-time search and analytics engine*. " O'Reilly Media, Inc.".

Rolf Jagerman, Honglei Zhuang, Zhen Qin, Xuanhui Wang, and Michael Bendersky. 2023. Query expansion by prompting large language models. *arXiv preprint arXiv:2305.03653*.

Eunkyung Jo, Daniel A Epstein, Hyunhoon Jung, and Young-Ho Kim. 2023. Understanding the benefits and challenges of deploying conversational ai leveraging large language models for public health intervention. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–16.

Yuki Kamide, Gerry TM Altmann, and Sarah L Haywood. 2003. The time-course of prediction in incremental sentence processing: Evidence from anticipatory eye movements. *Journal of Memory and language*, 49(1):133–156.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Ahmet Baki Kocaballi, Shlomo Berkovsky, Juan C Quiroz, Liliana Laranjo, Huong Ly Tong, Dana Rezazadegan, Agustina Briatore, and Enrico Coiera. 2019. The personalization of conversational agents in health care: systematic review. *Journal of medical Internet research*, 21(11):e15360.

Saar Kuzi, Mingyang Zhang, Cheng Li, Michael Bendersky, and Marc Najork. 2020. Leveraging semantic and lexical matching to improve the recall of document retrieval systems: A hybrid approach. *arXiv preprint arXiv:2010.01195*.

Carlos Lassance, Hervé Déjean, Thibault Formal, and Stéphane Clinchant. 2024. Splade-v3: New baselines for splade. *arXiv preprint arXiv:2403.06789*.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2024. Nv-embed: Improved techniques for training llms as generalist embedding models. *arXiv preprint arXiv:2405.17428*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957*.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. 2022. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2023. Fine-tuning llama for multi-stage text retrieval. *Preprint*, arXiv:2310.08319.

Tomas Mikolov. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Quim Motger, Xavier Franch, and Jordi Marco. 2022. Software-based dialogue systems: survey, taxonomy, and challenges. *ACM Computing Surveys*, 55(5):1–42.

Elham Mousavinasab, Nahid Zarifsanaiey, Sharareh R. Niakan Kalhori, Mahnaz Rakhshan, Leila Keikha, and Marjan Ghazi Saeedi. 2021. Intelligent tutoring systems: a systematic review of characteristics, applications, and evaluation methods. *Interactive Learning Environments*, 29(1):142–163.

Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human-generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Zach Nussbaum, John X. Morris, Brandon Duderstadt, and Andriy Mulyar. 2024. Nomic embed: Training a reproducible long context text embedder. *Preprint*, arXiv:2402.01613.

OpenAI. 2023. Gpt-3.5: Generative pre-trained transformer. Accessed: 2024-07-06.

OpenAI. 2024. New embedding models and api updates. https://openai.com/blog/new-embedding-models-and-api-updates/. Accessed: 2024-07-06.

Paul Owoicho, Jeff Dalton, Mohammad Aliannejadi, Leif Azzopardi, Johanne R Trippas, and Svitlana Vakulenko. 2022. Trec cast 2022: Going beyond user ask and system retrieve with initiative and response generation. In *TREC*.

Zhiyuan Peng, Xuyang Wu, and Yi Fang. 2023. Soft prompt tuning for augmenting dense retrieval with large language models. *arXiv preprint arXiv:2307.08303*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Qwen. 2024. Qwen2 technical report. Accessed: 2024-07-06.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Tao Shen, Guodong Long, Xiubo Geng, Chongyang Tao, Tianyi Zhou, and Daxin Jiang. 2023a. Large language models are strong zero-shot retriever. *arXiv preprint arXiv:2304.14233*.

Tao Shen, Guodong Long, Xiubo Geng, Chongyang Tao, Tianyi Zhou, and Daxin Jiang. 2023b. Large language models are strong zero-shot retriever. *arXiv preprint arXiv:2304.14233*.

Weiwei Sun, Shuo Zhang, Krisztian Balog, Zhaochun Ren, Pengjie Ren, Zhumin Chen, and Maarten de Rijke. 2021. Simulating user satisfaction for the evaluation of task-oriented dialogue systems. In *Proceedings of the 44rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21. ACM.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Jianguo Wang, Xiaomeng Yi, Rentong Guo, Hai Jin, Peng Xu, Shengjun Li, Xiangyu Wang, Xiangzhou Guo, Chengming Li, Xiaohai Xu, et al. 2021. Milvus: A purpose-built vector data management system. In *Proceedings of the 2021 International Conference on Management of Data*, pages 2614–2627.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE transactions on knowledge and data engineering*, 29(12):2724–2743.

Xiaohua Wang, Zhenghua Wang, Xuan Gao, Feiran Zhang, Yixin Wu, Zhibo Xu, Tianyuan Shi, Zhengyuan Wang, Shizheng Li, Qi Qian, et al. 2024. Searching for best practices in retrieval-augmented generation. *arXiv preprint arXiv:2407.01219*.

Jing Xu, Da Ju, Margaret Li, Y-Lan Boureau, Jason Weston, and Emily Dinan. 2021. Bot-adversarial dialogue for safe conversational agents. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2950–2968.

Qiongkai Xu, Lizhen Qu, Zeyu Gao, and Gholamreza Haffari. 2020. Personal information leakage detection in conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6567–6580, Online. Association for Computational Linguistics.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2013–2018.

Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-shot generative conversational query rewriting. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 1933–1936.

Longhui Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, Meishan Zhang, and Min Zhang. 2023. Rankinggpt: Empowering large language models in text ranking with progressive enhancement. *arXiv preprint arXiv:2311.16720*.

Justin Zobel, Alistair Moffat, and Kotagiri Ramamohanarao. 1998. Inverted files versus signature files for text indexing. *ACM Transactions on Database Systems (TODS)*, 23(4):453–490.

A Prompts for HEISIR

We detail the prompts used for our method. All prompts use {{$variable}} as placeholders for external variables. While the prompts shown here are designed for GPT-3.5-turbo, we adapted them for other LLM models by adding predefined tokens according to each model's specific template. Each prompt consists of four rows in the prompt table: common LLM parameters, system prompt, 5-shot examples generated using GPT-4o (omitted in this appendix for brevity), and user message with input placeholders.

A.1 Prompt for Hierarchical Triplets Formulation

Temperature: 0.0
max_tokens: 1024

System:

You must extract all [$information triplet$] in the message, given the context and role, subject to the following conditions:

- [$information triplet$] should include anything you need to understand the message and the role, such as the role's emotions, the topic of conversation, the intent of the message and etc.
- You need to extract an [$information triplet$] for a new message according to the following instructions.

Structural Constraints of [$information triplet$]:

- The structure of [$information triplet$] : [$subject$] [$common verb$] [$target content$]
- [$subject$] : Must be the {{$role}}.
- [$common verb$]:
  - Must Use a singular present tense verb (e.g., "says", "asks", "wants to", "inquires about", "looks into") to describe the {{$role}}'s action or intention.
  - If you need to use the negative form, write it as not in front of the common verb. However, use negative forms only when they are essential to understanding the sentence.
- [$target content$]:
  - Must be a noun phrase of 3 characters or less
  - If [$target content$] contains content too specific to be generalized, such as a person's name, webpage address, or code, generalize it by using general expressions such as person, friend, url, website, code and etc.
  - Should contain only one content; if multiple noun phrases or contents are needed, separate them into individual [$information triplet$] items.

How to Construct an Information Triplet:

1. You receive the [$conversation context$] along with the [$message$] you need to analyze as input.
2. Review the [$conversation context$] to understand the nuance and content of the [$message$]. However, the [$conversation context$] should only be used to understand the message and should not be used to extract the [$information triplet$].
3. Extract all the [$information triplet$] that can be obtained from the message in the form of a JSON.
4. The JSON format is as follows:

```
{
  "information_triplet":
  [
    {"{{$role}} [$common verb$]": [$target content$]},
    {"{{$role}} [$common verb$]": [$target content$]},
    {"{{$role}} [$common verb$]": [$target content$]},
    ...
  ]
}
```

Few-shot examples

User:

[$conversation context$]
{{$context}}

[$message$]
{{$role}}: {{$message}}

Extract as much [$information triplet$] as possible from the message while maintaining all of the above conditions. We recommend extracting between 5 and 20 pieces of [$information triplet$], depending on the length of the sentence. The key of the information_triplet you create should start with {{$role}}.

Answer:

Table 10: Prompt and parameters for Hierarachical Triplets Expansion

A.2 Adjunct Augmentation

Temperature: 0.0
max_tokens: 1024

System:

You will be provided with $conversation context$, $message$ and $information list$. $message$ is a real conversation message from {{$role}} that you need to analyze. $information list$ is information extracted from $message$, consisting of $subject$, $verb$, and $target content$.
You need to elaborate on the $detail$ of $target content$ according to the following instructions.

- Prepositions must be actively used to describe the $detail$ of the target content.
- If the value placed in $detail$ is vague, such as a pronoun, please use a specific noun to make the meaning clear.
- Here are three recommended strategies for elaborating on the $detail$:
  1. **Detailed Content and theme of Discussion**
     This category involves prepositions that help specify the subject of communication, the object of emotions or actions, and the theme of a conversation. For example
     - user expresses anger: to assistant.
     - user asks questions: about climate change.
     - assistant maintains a professional attitude: towards the user's queries.
     - assistant offers advice: regarding data protection.
     - user expresses confusion: over the assistant's instructions.
     - user seeks clarification: with respect to the subscription plans.
     - assistant invests effort: in improving the user interface.
  2. **Reasons or Causations**
     This set of phrases explains the cause or reason behind an action or situation. It answers the "why" of a scenario. For example
     - assistant pauses the service: because of maintenance needs.
     - user misses the deadline: due to a technical glitch.
     - user returns the product: owing to a manufacturing defect.
     - assistant improves response time: thanks to the user's constructive feedback.
  3. **Conditions or accompanying Circumstances**
     These prepositions are used to describe the conditions under which something happens or the context that accompanies an action. For example
     - user solves problems: with patience.
     - assistant writes articles: for users.
     - user reads the manual: over the weekend.
     - user leaves feedback: on the website.
     - user places the report: under the book.
  4. **In cases where no specific details can be included**
     If a sentence structure consisting of {{$role}} $verb$ $target content$ $detail$ is impossible or adding detailed information is not meaningful, set "no information" as the value for $detail$. For example
     - user mentions Christmas: no information.
- The answer should be written in JSON format as follows:

```
{
  "detailed_information":
  [
    {"{{$role}} [$verb$] [$target content$]": "[$detail$]"},
    {"{{$role}} [$verb$] [$target content$]": "[$detail$]"},
    ...
  ]
}
```

Few-shot examples

User:

[$conversation context$]
{{$context}}

[$message$]
{{$role}}: {{$message}}

[$information list$]
{{$info$_{list}$}}

Choose the best of the three strategies above and write a 2-3 word answer that clarifies the content of the sentence. Do not include the content of the sentence in your answer, but start with the preposition. The entire contents of [$Information list$] should be used as the key for detailed$_{information without any changes at all}$.

Answer:

Table 11: Prompt and parameters for Detailed Description Augmentation

B All results

Table [12-17] displays performance for key component combinations across LLM and Embedding models. 'p' and 'r' represent precision and recall, respectively. The highest value per metric is in bold, with the second-highest underlined.

Table 18 shows the baseline performance of all embedding models when using only conversation and messages.

C Intent Topic Analysis

In the main text, we observed intent clustering using SV. SVOA or SVO can reveal specific conversation topics. Table C shows the results of clustering into 15 groups. Examining representative topics for each cluster demonstrates that we can recover most major conversation themes from our dataset, along with related topics.

This analysis uses a basic K-means algorithm, showcasing the potential of Semantic indices for intent and topic analysis. While the current approach is fundamental, it effectively captures and categorizes conversational content. We anticipate that more advanced methods could yield even more robust and nuanced models, opening up various possibilities for future research in natural language processing and conversational AI.

D Reproducing

For all experiments, we utilized a single NVIDIA H100 80GB HBM3 GPU. Our study employed various models from Hugging Face and API. The embedding models consisted of Alibaba-NLP/gte-large-en-v1.5 (GTE), nomic-ai/nomic-embed-text-v1 (Nomic), nvidia/NV-Embed-v1 (NV-Embed), and McGill-NLP/LLM2Vec-Meta-Llama-3-8B-Instruct-mntp-supervised (LLM2Vec). Language models included google/gemma-2b-it (Gemma), microsoft/Phi-3-mini-128k-instruct (Phi-3), meta-llama/Meta-Llama-3-8B-Instruct (LLama-3), Qwen/Qwen2-7B-Instruct (Qwen-2), GPT-3.5-turbo-0125, and claude-3-haiku-20240307. In the fine-tuning process, train/validation datasets were created like test data, using only train data and excluding DICES due to data unavailability. Fine-tuning ran for 10 epochs. For LLM2Vec, we used LoRA (r=16, alpha=32) with a 1e-4 learning rate; other models used 1e-5. All code and data are accessible in the supplementary materials.

E Impact of Context

We select GPT-3.5-turbo as the LLM as it is the most accessible and widely model used in industry. Including previous context for semantic indices construction shows minor improvement in retrieval performance, as shown in Table 20.

However, in terms of interpretability, a thorough analysis of SVOA indices reveals that SVOA quadruplets extracted without previous context fail

embedding	combination	acc@1	acc@5	p@5	p@10	r@5	r@10	ndcg@10	ndcg@20	mrr@10	mrr@20	map@10	map@20
GTE	sv	0.0229	0.0717	0.0159	0.0133	0.0058	0.0096	0.0158	0.0168	0.0441	0.0481	0.0059	0.0050
	sv _svo	0.1311	0.3353	0.1020	0.0873	0.0440	0.0724	0.1045	0.1136	0.2189	0.2277	0.0518	0.0494
	sv _svo_svoa	0.1899	0.4467	0.1492	0.1271	0.0709	0.1127	0.1552	0.1688	0.2983	0.3062	0.0831	0.0820
	svoa_conv_msg	**0.3253**	**0.6252**	**0.2424**	**0.1990**	**0.1179**	**0.1805**	**0.2504**	**0.2643**	**0.4514**	**0.4577**	**0.1486**	**0.1450**
	svo_svoa_conv_msg	0.3131	0.6181	0.2400	0.1969	0.1150	0.1776	0.2465	0.2594	0.4425	0.4489	0.1458	0.1414
	sv _svo_svoa_conv_msg	0.3096	0.6124	0.2350	0.1929	0.1120	0.1729	0.2415	0.2541	0.4374	0.4437	0.1423	0.1378
Nomic	sv	0.0157	0.0520	0.0115	0.0106	0.0041	0.0077	0.0120	0.0132	0.0325	0.0363	0.0044	0.0037
	sv _svo	0.1245	0.3056	0.0972	0.0827	0.0416	0.0676	0.0984	0.1065	0.2030	0.2114	0.0497	0.0473
	sv _svo_svoa	0.1799	0.4264	0.1459	0.1232	0.0674	0.1083	0.1497	0.1594	0.2874	0.2952	0.0806	0.0777
	svoa_conv_msg	**0.2959**	**0.5910**	0.2234	**0.1806**	0.1088	**0.1650**	**0.2295**	**0.2421**	**0.4197**	**0.4265**	**0.1354**	**0.1319**
	svo_svoa_conv_msg	0.2939	0.5855	**0.2246**	0.1803	**0.1089**	0.1628	0.2282	0.2400	0.4179	0.4245	0.1344	0.1299
	sv _svo_svoa_conv_msg	0.2919	0.5793	0.2203	0.1781	0.1064	0.1606	0.2244	0.2344	0.4117	0.4182	0.1317	0.1263
NV-Embed	sv	0.0183	0.0620	0.0137	0.0119	0.0049	0.0083	0.0137	0.0157	0.0376	0.0425	0.0050	0.0043
	sv _svo	0.1282	0.3196	0.0979	0.0824	0.0418	0.0672	0.0992	0.1052	0.2114	0.2192	0.0495	0.0464
	sv _svo_svoa	0.1879	0.4247	0.1403	0.1170	0.0652	0.1016	0.1442	0.1544	0.2878	0.2956	0.0769	0.0744
	svoa_conv_msg	0.2339	0.4664	0.1616	0.1297	0.0754	0.1143	0.1656	0.1723	0.3326	0.3406	0.0922	0.0868
	svo_svoa_conv_msg	0.2482	**0.4976**	**0.1754**	**0.1419**	**0.0819**	**0.1237**	**0.1793**	0.1870	0.3536	0.3607	**0.1008**	0.0955
	sv _svo_svoa_conv_msg	**0.2505**	0.4970	0.1745	0.1416	0.0808	0.1236	0.1791	**0.1871**	**0.3556**	**0.3629**	0.1004	**0.0953**
LLM2Vec	sv	0.0240	0.0686	0.0155	0.0143	0.0058	0.0104	0.0166	0.0184	0.0447	0.0496	0.0062	0.0054
	sv _svo	0.1394	0.3585	0.1136	0.0969	0.0497	0.0803	0.1151	0.1247	0.2326	0.2411	0.0585	0.0562
	sv _svo_svoa	0.1974	0.4622	0.1578	0.1317	0.0746	0.1171	0.1611	0.1739	0.3078	0.3153	0.0877	0.0863
	svoa_conv_msg	0.3082	0.6130	0.2338	0.1909	0.1134	0.1740	0.2410	0.2538	0.4361	0.4422	0.1419	0.1381
	svo_svoa_conv_msg	**0.3165**	0.6195	0.2378	**0.1930**	0.1150	**0.1743**	**0.2439**	**0.2563**	**0.4421**	**0.4485**	**0.1443**	**0.1397**
	sv _svo_svoa_conv_msg	0.3136	**0.6198**	**0.2387**	0.1923	**0.1151**	0.1737	0.2428	0.2543	0.4410	0.4474	0.1436	0.1387
OpenAI-small	sv	0.0171	0.0628	0.0138	0.0123	0.0050	0.0091	0.0139	0.0158	0.0376	0.0423	0.0049	0.0043
	sv _svo	0.1280	0.3376	0.1068	0.0909	0.0474	0.0757	0.1079	0.1178	0.2174	0.2263	0.0549	0.0529
	sv _svo_svoa	0.1962	0.4562	0.1546	0.1308	0.0728	0.1171	0.1599	0.1711	0.3059	0.3134	0.0867	0.0845
	svoa_conv_msg	**0.3253**	**0.6190**	**0.2398**	**0.1976**	**0.1173**	**0.1813**	**0.2503**	**0.2656**	**0.4508**	**0.4573**	**0.1495**	**0.1469**
	svo_svoa_conv_msg	0.3242	0.6178	0.2392	0.1957	0.1166	0.1779	0.2477	0.2614	0.4485	0.4549	0.1476	0.1442
	sv _svo_svoa_conv_msg	0.3208	0.6084	0.2355	0.1925	0.1150	0.1752	0.2440	0.2574	0.4448	0.4511	0.1451	0.1416
OpenAI-large	sv	0.0163	0.0603	0.0135	0.0119	0.0054	0.0089	0.0136	0.0152	0.0360	0.0405	0.0049	0.0042
	sv _svo	0.1325	0.3399	0.1062	0.0918	0.0472	0.0764	0.1091	0.1183	0.2218	0.2304	0.0554	0.0532
	sv _svo_svoa	0.1962	0.4582	0.1545	0.1313	0.0734	0.1176	0.1604	0.1727	0.3078	0.3157	0.0864	0.0848
	svoa_conv_msg	**0.3508**	**0.6590**	**0.2587**	**0.2130**	**0.1271**	**0.1941**	**0.2694**	**0.2851**	**0.4798**	**0.4856**	**0.1627**	**0.1600**
	svo_svoa_conv_msg	0.3473	0.6387	0.2527	0.2071	0.1226	0.1875	0.2620	0.2756	0.4714	0.4773	0.1575	0.1536
	sv _svo_svoa_conv_msg	0.3422	0.6347	0.2483	0.2031	0.1202	0.1836	0.2574	0.2705	0.4656	0.4714	0.1541	0.1502

Table 12: Results for Different Embeddings and **Gemma**

embedding	combination	acc@1	acc@5	p@5	p@10	r@5	r@10	ndcg@10	ndcg@20	mrr@10	mrr@20	map@10	map@20
GTE	sv	0.0374	0.1163	0.0280	0.0241	0.0113	0.0189	0.0286	0.0318	0.0728	0.0785	0.0116	0.0106
	sv_svo	0.2396	0.5330	0.1864	0.1564	0.0879	0.1390	0.1920	0.2053	0.3642	0.3709	0.1065	0.1047
	sv_svo_svoa	0.2816	0.5698	0.2107	0.1748	0.1015	0.1577	0.2187	0.2336	0.4033	0.4099	0.1267	0.1248
	svoa_conv_msg	0.3476	0.6587	0.2638	0.2158	0.1270	0.1949	0.2716	0.2856	0.4805	0.4861	0.1643	0.1603
	svo_svoa_conv_msg	**0.3610**	0.6621	0.2660	**0.2191**	0.1280	**0.1972**	**0.2760**	**0.2905**	0.4877	0.4933	**0.1680**	**0.1644**
	sv_svo_svoa_conv_msg	0.3582	**0.6695**	**0.2677**	0.2184	**0.1286**	0.1966	0.2757	0.2901	**0.4886**	**0.4947**	0.1677	0.1641
Nomic	sv	0.0386	0.1060	0.0262	0.0214	0.0105	0.0170	0.0265	0.0281	0.0693	0.0738	0.0111	0.0099
	sv_svo	0.2508	0.5079	0.1847	0.1544	0.0849	0.1343	0.1900	0.1995	0.3603	0.3673	0.1076	0.1034
	sv_svo_svoa	0.2913	0.5641	0.2163	0.1773	0.1020	0.1563	0.2216	0.2335	0.4077	0.4148	0.1306	0.1265
	svoa_conv_msg	0.3416	0.6264	0.2464	0.2022	0.1199	0.1841	0.2566	0.2701	0.4637	0.4700	0.1541	0.1505
	svo_svoa_conv_msg	0.3573	0.6484	0.2600	0.2121	0.1247	**0.1919**	0.2688	0.2817	0.4815	0.4880	0.1635	0.1590
	sv_svo_svoa_conv_msg	**0.3602**	**0.6590**	**0.2635**	**0.2138**	**0.1257**	0.1916	**0.2707**	**0.2839**	**0.4858**	**0.4920**	**0.1647**	**0.1603**
NV-Embed	sv	0.0371	0.1200	0.0297	0.0245	0.0115	0.0187	0.0287	0.0316	0.0736	0.0794	0.0116	0.0104
	sv_svo	0.2616	0.5330	0.1907	0.1572	0.0878	0.1359	0.1949	0.2043	0.3750	0.3819	0.1097	0.1052
	sv_svo_svoa	**0.2933**	0.5601	0.2089	0.1720	0.0996	**0.1534**	0.2168	**0.2277**	**0.4064**	**0.4134**	0.1259	**0.1219**
	svoa_conv_msg	0.2465	0.4990	0.1776	0.1447	0.0827	0.1269	0.1818	0.1885	0.3520	0.3589	0.1027	0.0973
	svo_svoa_conv_msg	0.2773	0.5541	0.2069	0.1702	0.0967	0.1497	0.2124	0.2210	0.3950	0.4011	0.1233	0.1182
	sv_svo_svoa_conv_msg	0.2893	**0.5630**	**0.2129**	**0.1739**	**0.0997**	0.1528	**0.2180**	0.2266	0.4053	0.4117	**0.1275**	0.1217
LLM2Vec	sv	0.0537	0.1557	0.0385	0.0330	0.0149	0.0251	0.0390	0.0421	0.0986	0.1051	0.0162	0.0143
	sv_svo	0.2862	0.5781	0.2155	0.1781	0.1009	0.1567	0.2203	0.2332	0.4101	0.4172	0.1272	0.1237
	sv_svo_svoa	0.3193	0.6073	0.2378	0.1930	0.1136	0.1725	0.2430	0.2568	0.4405	0.4470	0.1451	0.1421
	svoa_conv_msg	0.3579	0.6552	0.2610	0.2124	0.1263	0.1938	0.2699	0.2833	0.4838	0.4897	0.1627	0.1584
	svo_svoa_conv_msg	0.3790	0.6904	0.2817	0.2273	0.1351	0.2044	0.2880	0.2998	0.5081	0.5136	0.1769	0.1712
	sv_svo_svoa_conv_msg	**0.3865**	**0.6967**	**0.2860**	**0.2321**	**0.1368**	**0.2078**	**0.2936**	**0.3055**	**0.5166**	**0.5219**	**0.1807**	**0.1749**
OpenAI-small	sv	0.0497	0.1400	0.0356	0.0294	0.0141	0.0225	0.0353	0.0378	0.0896	0.0955	0.0147	0.0131
	sv_svo	0.2625	0.5410	0.1964	0.1636	0.0918	0.1453	0.2035	0.2163	0.3828	0.3904	0.1161	0.1130
	sv_svo_svoa	0.3022	0.5844	0.2233	0.1841	0.1066	0.1650	0.2314	0.2451	0.4239	0.4310	0.1362	0.1333
	svoa_conv_msg	0.3653	0.6550	0.2627	0.2167	0.1285	0.1986	0.2752	0.2915	0.4893	0.4956	0.1677	0.1654
	svo_svoa_conv_msg	0.3807	0.6724	0.2736	0.2246	0.1333	0.2032	0.2849	0.2989	0.5054	0.5112	0.1749	0.1713
	sv_svo_svoa_conv_msg	**0.3893**	**0.6821**	**0.2785**	**0.2283**	**0.1347**	**0.2060**	**0.2898**	**0.3034**	**0.5147**	**0.5201**	**0.1783**	**0.1740**
OpenAI-large	sv	0.0457	0.1328	0.0324	0.0284	0.0127	0.0215	0.0334	0.0362	0.0844	0.0907	0.0137	0.0121
	sv_svo	0.2594	0.5456	0.1969	0.1656	0.0909	0.1461	0.2036	0.2175	0.3824	0.3892	0.1157	0.1132
	sv_svo_svoa	0.3005	0.5907	0.2210	0.1834	0.1067	0.1647	0.2296	0.2451	0.4243	0.4312	0.1338	0.1325
	svoa_conv_msg	**0.3965**	0.6918	0.2831	0.2319	0.1394	0.2121	0.2950	0.3113	0.5223	0.5276	0.1812	0.1785
	svo_svoa_conv_msg	0.3899	0.7021	0.2864	0.2331	0.1396	0.2109	0.2959	0.3117	0.5230	0.5284	0.1823	0.1792
	sv_svo_svoa_conv_msg	0.3950	**0.7092**	**0.2898**	**0.2357**	**0.1405**	**0.2133**	**0.2992**	**0.3136**	**0.5285**	**0.5337**	**0.1844**	**0.1803**

Table 13: Results for Different Embeddings and **Phi-3**

embedding	combination	acc@1	acc@5	p@5	p@10	r@5	r@10	ndcg@10	ndcg@20	mrr@10	mrr@20	map@10	map@20
GTE	sv	0.0286	0.0985	0.0233	0.0201	0.0086	0.0146	0.0231	0.0245	0.0588	0.0635	0.0091	0.0078
	sv_svo	0.1965	0.4693	0.1576	0.1370	0.0731	0.1198	0.1644	0.1761	0.3131	0.3213	0.0885	0.0860
	sv_svo_svoa	0.2953	0.5995	0.2267	0.1896	0.1079	0.1695	0.2351	0.2478	0.4260	0.4321	0.1372	0.1339
	svoa_conv_msg	0.3582	0.6655	0.2699	0.2215	0.1300	0.2004	0.2783	0.2914	0.4886	0.4943	0.1691	0.1642
	svo_svoa_conv_msg	**0.3767**	**0.6735**	**0.2741**	**0.2244**	**0.1321**	**0.2021**	**0.2844**	**0.2975**	**0.5026**	**0.5086**	**0.1740**	**0.1690**
	sv_svo_svoa_conv_msg	0.3728	0.6730	0.2739	0.2240	0.1313	0.2012	0.2829	0.2958	0.5006	0.5066	0.1725	0.1675
Nomic	sv	0.0263	0.0857	0.0201	0.0177	0.0076	0.0131	0.0203	0.0219	0.0523	0.0564	0.0079	0.0069
	sv_svo	0.2125	0.4579	0.1587	0.1323	0.0728	0.1147	0.1623	0.1718	0.3177	0.3254	0.0890	0.0850
	sv_svo_svoa	0.3193	0.6013	0.2317	0.1900	0.1097	0.1674	0.2382	0.2503	0.4403	0.4466	0.1410	0.1371
	svoa_conv_msg	0.3510	0.6387	0.2567	0.2093	0.1243	0.1897	0.2648	0.2776	0.4730	0.4793	0.1603	0.1558
	svo_svoa_conv_msg	**0.3699**	0.6527	0.2679	0.2165	0.1286	0.1938	0.2749	0.2887	0.4902	0.4968	0.1684	0.1638
	sv_svo_svoa_conv_msg	0.3693	**0.6630**	**0.2699**	**0.2189**	**0.1289**	**0.1949**	**0.2770**	**0.2900**	**0.4929**	**0.4991**	**0.1695**	**0.1646**
NV-Embed	sv	0.0280	0.0954	0.0228	0.0217	0.0078	0.0149	0.0240	0.0255	0.0594	0.0643	0.0093	0.0079
	sv_svo	0.2025	0.4576	0.1578	0.1308	0.0711	0.1119	0.1609	0.1701	0.3139	0.3219	0.0884	0.0844
	sv_svo_svoa	**0.3153**	**0.5930**	**0.2257**	**0.1843**	**0.1060**	**0.1624**	**0.2327**	**0.2425**	**0.4325**	**0.4393**	**0.1370**	**0.1319**
	svoa_conv_msg	0.2539	0.5113	0.1856	0.1479	0.0871	0.1298	0.1880	0.1961	0.3651	0.3725	0.1075	0.1020
	svo_svoa_conv_msg	0.2819	0.5567	0.2097	0.1705	0.0980	0.1494	0.2141	0.2236	0.3989	0.4061	0.1254	0.1196
	sv_svo_svoa_conv_msg	0.2845	0.5644	0.2145	0.1732	0.1001	0.1516	0.2175	0.2264	0.4047	0.4117	0.1274	0.1211
LLM2Vec	sv	0.0371	0.1291	0.0309	0.0257	0.0115	0.0190	0.0299	0.0309	0.0764	0.0816	0.0119	0.0100
	sv_svo	0.2274	0.5159	0.1826	0.1536	0.0841	0.1344	0.1868	0.1992	0.3509	0.3586	0.1046	0.1013
	sv_svo_svoa	0.3433	0.6387	0.2532	0.2047	0.1192	0.1815	0.2586	0.2716	0.4700	0.4765	0.1559	0.1516
	svoa_conv_msg	0.3673	0.6735	0.2694	0.2178	0.1295	0.1966	0.2765	0.2897	0.4954	0.5012	0.1670	0.1625
	svo_svoa_conv_msg	0.3885	0.6924	0.2836	0.2307	0.1350	0.2070	0.2921	0.3055	0.5169	0.5223	0.1790	0.1740
	sv_svo_svoa_conv_msg	**0.3902**	**0.6978**	**0.2880**	**0.2326**	**0.1365**	**0.2084**	**0.2944**	**0.3073**	**0.5188**	**0.5244**	**0.1808**	**0.1754**
OpenAI-small	sv	0.0366	0.1183	0.0275	0.0247	0.0105	0.0179	0.0283	0.0307	0.0724	0.0780	0.0110	0.0098
	sv_svo	0.2179	0.4856	0.1686	0.1401	0.0784	0.1226	0.1723	0.1852	0.3290	0.3375	0.0960	0.0937
	sv_svo_svoa	0.3156	0.6061	0.2359	0.1943	0.1125	0.1737	0.2427	0.2559	0.4394	0.4460	0.1439	0.1409
	svoa_conv_msg	0.3802	0.6655	0.2703	0.2235	0.1310	0.2038	0.2827	0.2972	0.5012	0.5073	0.1727	0.1687
	svo_svoa_conv_msg	0.3867	0.6849	0.2820	0.2286	0.1351	0.2068	0.2905	0.3056	0.5118	0.5176	0.1796	0.1758
	sv_svo_svoa_conv_msg	**0.3905**	**0.6889**	**0.2849**	**0.2296**	**0.1364**	**0.2070**	**0.2919**	**0.3067**	**0.5166**	**0.5219**	**0.1801**	**0.1764**
OpenAI-large	sv	0.0314	0.1125	0.0265	0.0237	0.0100	0.0169	0.0268	0.0286	0.0673	0.0728	0.0104	0.0089
	sv_svo	0.2174	0.4896	0.1691	0.1459	0.0792	0.1289	0.1774	0.1922	0.3340	0.3425	0.0981	0.0972
	sv_svo_svoa	0.3253	0.6244	0.2408	0.2005	0.1156	0.1788	0.2504	0.2640	0.4535	0.4595	0.1490	0.1459
	svoa_conv_msg	0.3933	0.7072	0.2922	0.2381	0.1424	0.2174	0.3016	0.3165	0.5247	0.5299	0.1863	0.1828
	svo_svoa_conv_msg	0.4050	**0.7149**	**0.2982**	0.2418	**0.1437**	0.2189	0.3068	**0.3208**	**0.5351**	**0.5400**	0.1903	**0.1862**
	sv_svo_svoa_conv_msg	**0.4053**	0.7135	0.2971	**0.2431**	0.1427	**0.2194**	**0.3072**	0.3207	0.5339	0.5388	**0.1904**	0.1861

Table 14: Results for Different Embeddings and **LLama-3**

embedding	combination	acc@1	acc@5	p@5	p@10	r@5	r@10	ndcg@10	ndcg@20	mrr@10	mrr@20	map@10	map@20
GTE	sv	0.0366	0.1077	0.0267	0.0235	0.0102	0.0172	0.0272	0.0294	0.0683	0.0739	0.0110	0.0097
	sv_svo	0.2139	0.4790	0.1676	0.1433	0.0774	0.1248	0.1728	0.1864	0.3271	0.3352	0.0940	0.0920
	sv_svo_svoa	0.2842	0.5753	0.2159	0.1793	0.1038	0.1617	0.2231	0.2376	0.4054	0.4122	0.1292	0.1273
	svoa_conv_msg	0.3636	0.6638	0.2644	0.2168	0.1276	0.1969	0.2753	0.2887	0.4900	0.4960	0.1669	0.1623
	svo_svoa_conv_msg	**0.3693**	**0.6661**	**0.2692**	**0.2211**	0.1294	**0.1999**	**0.2801**	**0.2932**	**0.4962**	**0.5022**	**0.1712**	**0.1659**
	sv_svo_svoa_conv_msg	0.3613	**0.6661**	0.2688	0.2208	**0.1296**	0.1997	0.2786	0.2921	0.4916	0.4980	0.1696	0.1649
Nomic	sv	0.0331	0.1005	0.0251	0.0206	0.0099	0.0155	0.0247	0.0260	0.0635	0.0681	0.0102	0.0088
	sv_svo	0.2034	0.4682	0.1640	0.1385	0.0745	0.1193	0.1673	0.1796	0.3153	0.3232	0.0918	0.0895
	sv_svo_svoa	0.2885	0.5813	0.2228	0.1818	0.1050	0.1608	0.2263	0.2376	0.4119	0.4180	0.1325	0.1286
	svoa_conv_msg	0.3393	0.6361	0.2503	0.2055	0.1218	0.1882	0.2599	0.2725	0.4654	0.4709	0.1562	0.1523
	svo_svoa_conv_msg	0.3562	0.6558	0.2622	0.2122	0.1270	0.1917	0.2692	0.2819	0.4809	0.4872	0.1643	0.1594
	sv_svo_svoa_conv_msg	**0.3619**	**0.6587**	**0.2645**	**0.2147**	**0.1276**	**0.1927**	**0.2720**	**0.2845**	**0.4859**	**0.4920**	**0.1662**	**0.1610**
NV-Embed	sv	0.0346	0.1171	0.0290	0.0252	0.0104	0.0180	0.0284	0.0298	0.0696	0.0747	0.0113	0.0097
	sv_svo	0.2131	0.4784	0.1682	0.1420	0.0758	0.1216	0.1717	0.1828	0.3283	0.3361	0.0934	0.0900
	sv_svo_svoa	**0.2862**	**0.5821**	**0.2183**	**0.1787**	**0.1035**	**0.1583**	**0.2232**	**0.2333**	**0.4099**	**0.4158**	**0.1300**	**0.1251**
	svoa_conv_msg	0.2514	0.4996	0.1785	0.1450	0.0829	0.1273	0.1836	0.1908	0.3572	0.3644	0.1045	0.0988
	svo_svoa_conv_msg	0.2768	0.5501	0.2075	0.1685	0.0969	0.1479	0.2112	0.2199	0.3929	0.3999	0.1233	0.1175
	sv_svo_svoa_conv_msg	0.2825	0.5610	0.2125	0.1715	0.0992	0.1500	0.2154	0.2239	0.4008	0.4077	0.1263	0.1201
LLM2Vec	sv	0.0446	0.1451	0.0363	0.0311	0.0139	0.0226	0.0358	0.0383	0.0879	0.0940	0.0145	0.0127
	sv_svo	0.2371	0.5304	0.1907	0.1598	0.0880	0.1390	0.1936	0.2086	0.3581	0.3663	0.1081	0.1061
	sv_svo_svoa	0.3142	0.6170	0.2371	0.1976	0.1131	0.1769	0.2459	0.2587	0.4427	0.4489	0.1452	0.1419
	svoa_conv_msg	0.3599	0.6530	0.2620	0.2139	0.1268	0.1946	0.2709	0.2843	0.4858	0.4916	0.1631	0.1586
	svo_svoa_conv_msg	0.3799	0.6815	0.2791	0.2266	0.1347	0.2042	0.2866	0.2992	0.5093	0.5145	0.1746	0.1693
	sv_svo_svoa_conv_msg	**0.3879**	**0.6864**	**0.2847**	**0.2299**	**0.1365**	**0.2065**	**0.2912**	**0.3035**	**0.5152**	**0.5203**	**0.1786**	**0.1728**
OpenAI-small	sv	0.0380	0.1291	0.0320	0.0281	0.0121	0.0209	0.0322	0.0344	0.0787	0.0842	0.0130	0.0113
	sv_svo	0.2231	0.5021	0.1774	0.1510	0.0832	0.1326	0.1841	0.1976	0.3426	0.3507	0.1023	0.1001
	sv_svo_svoa	0.2942	0.5938	0.2283	0.1879	0.1093	0.1686	0.2345	0.2476	0.4220	0.4285	0.1378	0.1348
	svoa_conv_msg	0.3705	0.6618	0.2674	0.2194	0.1302	0.2006	0.2783	0.2931	0.4931	0.4995	0.1700	0.1663
	svo_svoa_conv_msg	0.3807	0.6778	0.2779	0.2253	0.1348	0.2050	0.2861	0.3001	0.5056	0.5112	0.1760	0.1718
	sv_svo_svoa_conv_msg	**0.3816**	**0.6824**	**0.2800**	**0.2283**	**0.1358**	**0.2069**	**0.2890**	**0.3032**	**0.5082**	**0.5136**	**0.1779**	**0.1738**
OpenAI-large	sv	0.0428	0.1322	0.0333	0.0293	0.0119	0.0207	0.0338	0.0356	0.0846	0.0904	0.0137	0.0116
	sv_svo	0.2291	0.5124	0.1815	0.1539	0.0844	0.1356	0.1871	0.2004	0.3488	0.3558	0.1036	0.1017
	sv_svo_svoa	0.3088	0.5950	0.2274	0.1889	0.1089	0.1702	0.2368	0.2510	0.4304	0.4371	0.1392	0.1367
	svoa_conv_msg	0.3873	0.6935	0.2852	0.2340	0.1407	0.2139	0.2966	0.3124	0.5186	0.5242	0.1828	0.1798
	svo_svoa_conv_msg	0.3965	0.7004	0.2922	0.2366	**0.1422**	0.2145	0.3005	0.3154	0.5255	0.5312	0.1863	0.1822
	sv_svo_svoa_conv_msg	**0.4045**	**0.7021**	**0.2926**	**0.2380**	**0.1422**	**0.2157**	**0.3026**	**0.3166**	**0.5305**	**0.5358**	**0.1874**	**0.1832**

Table 15: Results for Different Embeddings and **Qwen-2**

embedding	combination	acc@1	acc@5	p@5	p@10	r@5	r@10	ndcg@10	ndcg@20	mrr@10	mrr@20	map@10	map@20
GTE	sv	0.0448	0.1345	0.0351	0.0292	0.0132	0.0214	0.0342	0.0363	0.0843	0.0903	0.0146	0.0126
	sv_svo	0.2211	0.5024	0.1758	0.1482	0.0815	0.1291	0.1796	0.1931	0.3392	0.3475	0.0984	0.0962
	sv_svo_svoa	0.2913	0.5910	0.2214	0.1826	0.1057	0.1642	0.2279	0.2424	0.4183	0.4252	0.1329	0.1302
	svoa_conv_msg	0.3588	0.6495	0.2633	0.2160	0.1281	0.1967	0.2735	0.2882	0.4837	0.4899	0.1664	0.1623
	svo_svoa_conv_msg	**0.3676**	0.6621	**0.2696**	**0.2207**	**0.1307**	**0.1992**	**0.2790**	**0.2939**	**0.4925**	**0.4982**	**0.1705**	**0.1665**
	si_svo_svoa_conv_msg	0.3633	**0.6641**	0.2677	0.2199	0.1296	0.1980	0.2776	0.2930	0.4906	0.4964	0.1693	0.1657
Nomic	sv	0.0446	0.1188	0.0326	0.0267	0.0125	0.0198	0.0320	0.0341	0.0774	0.0832	0.0142	0.0123
	sv_svo	0.2336	0.4959	0.1782	0.1491	0.0826	0.1294	0.1825	0.1934	0.3454	0.3533	0.1025	0.0989
	sv_svo_svoa	0.3156	0.5921	0.2268	0.1869	0.1075	0.1662	0.2347	0.2463	0.4340	0.4406	0.1391	0.1349
	svoa_conv_msg	0.3482	0.6421	0.2515	0.2060	0.1222	0.1886	0.2618	0.2753	0.4720	0.4784	0.1578	0.1539
	svo_svoa_conv_msg	0.3622	0.6558	0.2651	0.2170	0.1284	0.1963	0.2751	0.2871	0.4895	0.4953	0.1682	0.1631
	si_svo_svoa_conv_msg	**0.3696**	**0.6750**	**0.2719**	**0.2195**	**0.1307**	**0.1973**	**0.2784**	**0.2899**	**0.4975**	**0.5029**	**0.1702**	**0.1649**
NV-Embed	sv	0.0448	0.1308	0.0345	0.0298	0.0128	0.0213	0.0344	0.0362	0.0834	0.0891	0.0146	0.0126
	sv_svo	0.2322	0.4944	0.1798	0.1508	0.0811	0.1278	0.1835	0.1929	0.3458	0.3538	0.1023	0.0976
	sv_svo_svoa	**0.3019**	**0.5898**	**0.2235**	**0.1805**	**0.1057**	**0.1604**	**0.2283**	**0.2405**	**0.4244**	**0.4309**	**0.1341**	**0.1301**
	svoa_conv_msg	0.2471	0.4996	0.1790	0.1452	0.0834	0.1272	0.1831	0.1897	0.3551	0.3625	0.1038	0.0981
	svo_svoa_conv_msg	0.2825	0.5470	0.2056	0.1696	0.0951	0.1480	0.2121	0.2212	0.3965	0.4033	0.1233	0.1177
	si_svo_svoa_conv_msg	0.2896	0.5590	0.2095	0.1720	0.0971	0.1496	0.2153	0.2255	0.4035	0.4104	0.1251	0.1198
LLM2Vec	sv	0.0634	0.1680	0.0448	0.0381	0.0174	0.0284	0.0454	0.0484	0.1102	0.1174	0.0200	0.0175
	sv_svo	0.2585	0.5490	0.2009	0.1694	0.0926	0.1464	0.2061	0.2192	0.3811	0.3891	0.1171	0.1137
	sv_svo_svoa	0.3259	0.6304	0.2464	0.2024	0.1178	0.1810	0.2531	0.2667	0.4530	0.4589	0.1517	0.1485
	svoa_conv_msg	0.3628	0.6590	0.2610	0.2137	0.1261	0.1940	0.2709	0.2850	0.4871	0.4932	0.1633	0.1592
	svo_svoa_conv_msg	0.3813	0.6867	0.2800	0.2277	0.1344	0.2054	0.2884	0.3018	0.5101	0.5155	0.1767	0.1718
	si_svo_svoa_conv_msg	**0.3916**	**0.6915**	**0.2861**	**0.2319**	**0.1366**	**0.2081**	**0.2934**	**0.3068**	**0.5184**	**0.5234**	**0.1801**	**0.1749**
OpenAI-small	sv	0.0626	0.1520	0.0404	0.0342	0.0158	0.0250	0.0413	0.0443	0.1024	0.1089	0.0183	0.0164
	sv_svo	0.2445	0.5310	0.1951	0.1616	0.0898	0.1410	0.1973	0.2097	0.3633	0.3711	0.1115	0.1084
	sv_svo_svoa	0.3236	0.6081	0.2329	0.1921	0.1111	0.1723	0.2419	0.2558	0.4425	0.4489	0.1432	0.1405
	svoa_conv_msg	0.3656	0.6664	0.2695	0.2194	0.1314	0.2006	0.2785	0.2937	0.4928	0.4987	0.1707	0.1674
	svo_svoa_conv_msg	0.3830	0.6781	0.2804	0.2286	0.1355	0.2086	0.2897	0.3035	0.5088	0.5148	0.1792	0.1744
	si_svo_svoa_conv_msg	**0.3873**	**0.6861**	**0.2844**	**0.2306**	**0.1371**	**0.2092**	**0.2928**	**0.3065**	**0.5144**	**0.5199**	**0.1812**	**0.1766**
OpenAI-large	sv	0.0531	0.1531	0.0402	0.0334	0.0151	0.0247	0.0394	0.0423	0.0962	0.1028	0.0170	0.0149
	sv_svo	0.2539	0.5293	0.1934	0.1612	0.0897	0.1401	0.1982	0.2112	0.3720	0.3796	0.1118	0.1089
	sv_svo_svoa	0.3202	0.6121	0.2374	0.1953	0.1144	0.1759	0.2455	0.2599	0.4457	0.4518	0.1456	0.1431
	svoa_conv_msg	0.4013	0.6992	0.2871	0.2360	0.1414	0.2167	0.3004	0.3153	0.5286	0.5343	0.1857	0.1819
	svo_svoa_conv_msg	0.4047	0.7112	0.2940	0.2381	0.1437	0.2168	0.3034	0.3180	0.5343	0.5395	0.1882	0.1844
	si_svo_svoa_conv_msg	**0.4085**	**0.7124**	**0.2955**	**0.2402**	**0.1439**	**0.2178**	**0.3056**	**0.3198**	**0.5362**	**0.5415**	**0.1898**	**0.1856**

Table 16: Results for Different Embeddings and **GPT-3.5-turbo**

embedding	combination	acc@1	acc@5	p@5	p@10	r@5	r@10	ndcg@10	ndcg@20	mrr@10	mrr@20	map@10	map@20
GTE	sv	0.0411	0.1297	0.0316	0.0279	0.0120	0.0205	0.0320	0.0340	0.0799	0.0854	0.0129	0.0113
	sv_svo	0.2228	0.5013	0.1797	0.1498	0.0829	0.1307	0.1826	0.1947	0.3431	0.3507	0.1012	0.0983
	sv_svo_svoa	0.2916	0.6010	0.2268	0.1876	0.1078	0.1673	0.2327	0.2469	0.4242	0.4302	0.1351	0.1324
	svoa_conv_msg	0.3628	0.6724	0.2668	0.2172	0.1282	0.1967	0.2748	0.2891	0.4918	0.4977	0.1658	0.1615
	sit_svoa_conv_msg	0.3665	0.6795	**0.2734**	0.2219	**0.1308**	0.1997	0.2796	**0.2944**	0.4969	0.5027	0.1702	**0.1659**
	sv_svo_svoa_conv_msg	**0.3716**	**0.6801**	0.2715	**0.2234**	0.1301	**0.2007**	**0.2809**	0.2943	**0.4995**	**0.5054**	**0.1707**	**0.1659**
Nomic	sv	0.0394	0.1163	0.0291	0.0242	0.0107	0.0180	0.0285	0.0320	0.0726	0.0788	0.0117	0.0105
	sv_svo	0.2359	0.4973	0.1787	0.1492	0.0814	0.1287	0.1828	0.1927	0.3470	0.3546	0.1028	0.0986
	sv_svo_svoa	0.3096	0.6147	0.2348	0.1897	0.1105	0.1669	0.2382	0.2486	0.4370	0.4429	0.1410	0.1362
	svoa_conv_msg	0.3365	0.6398	0.2514	0.2044	0.1220	0.1861	0.2587	0.2735	0.4647	0.4707	0.1553	0.1521
	sit_svoa_conv_msg	0.3559	0.6561	0.2666	0.2175	0.1281	0.1948	0.2735	0.2866	0.4829	0.4886	0.1670	0.1620
	sv_svo_svoa_conv_msg	**0.3602**	**0.6590**	**0.2695**	**0.2193**	**0.1294**	**0.1955**	**0.2757**	**0.2882**	**0.4867**	**0.4928**	**0.1687**	**0.1630**
NV-Embed	sv	0.0408	0.1282	0.0318	0.0291	0.0115	0.0204	0.0327	0.0343	0.0805	0.0858	0.0131	0.0113
	sv_svo	0.2276	0.5056	0.1826	0.1494	0.0838	0.1285	0.1831	0.1923	0.3460	0.3537	0.1024	0.0976
	sv_svo_svoa	**0.3079**	**0.5930**	**0.2248**	**0.1842**	**0.1084**	**0.1644**	**0.2320**	**0.2421**	**0.4306**	**0.4367**	**0.1357**	**0.1303**
	svoa_conv_msg	0.2539	0.5110	0.1821	0.1460	0.0853	0.1290	0.1853	0.1921	0.3621	0.3692	0.1049	0.0990
	sit_svoa_conv_msg	0.2851	0.5701	0.2127	0.1701	0.0994	0.1494	0.2138	0.2245	0.4030	0.4103	0.1239	0.1189
	sv_svo_svoa_conv_msg	0.2873	0.5810	0.2179	0.1751	0.1016	0.1531	0.2191	0.2279	0.4094	0.4162	0.1273	0.1212
LLM2Vec	sv	0.0548	0.1602	0.0399	0.0365	0.0149	0.0262	0.0418	0.0455	0.1015	0.1087	0.0172	0.0153
	sv_svo	0.2716	0.5658	0.2125	0.1743	0.0973	0.1506	0.2143	0.2261	0.3950	0.4021	0.1232	0.1192
	sv_svo_svoa	0.3390	0.6412	0.2516	0.2077	0.1195	0.1842	0.2597	0.2719	0.4662	0.4721	0.1556	0.1510
	svoa_conv_msg	0.3556	0.6684	0.2662	0.2143	0.1283	0.1945	0.2722	0.2848	0.4867	0.4928	0.1641	0.1587
	sit_svoa_conv_msg	0.3887	0.6801	0.2808	0.2302	0.1345	0.2063	0.2913	0.3043	0.5164	0.5224	0.1784	0.1727
	sv_svo_svoa_conv_msg	**0.3927**	**0.6944**	**0.2875**	**0.2340**	**0.1369**	**0.2087**	**0.2958**	**0.3086**	**0.5228**	**0.5282**	**0.1815**	**0.1758**
OpenAI-small	sv	0.0506	0.1514	0.0390	0.0331	0.0144	0.0239	0.0387	0.0422	0.0952	0.1019	0.0163	0.0145
	sv_svo	0.2582	0.5207	0.1926	0.1612	0.0891	0.1401	0.1991	0.2108	0.3734	0.3805	0.1132	0.1095
	sv_svo_svoa	0.3213	0.6233	0.2411	0.1997	0.1140	0.1777	0.2493	0.2633	0.4486	0.4549	0.1486	0.1449
	svoa_conv_msg	0.3730	0.6784	0.2744	0.2207	0.1329	0.2015	0.2809	0.2954	0.5017	0.5075	0.1709	0.1674
	sit_svoa_conv_msg	0.3856	0.6901	0.2842	0.2300	0.1363	0.2079	0.2916	0.3052	0.5146	0.5202	0.1795	0.1752
	sv_svo_svoa_conv_msg	**0.3910**	**0.6978**	**0.2888**	**0.2328**	**0.1384**	**0.2092**	**0.2952**	**0.3085**	**0.5198**	**0.5251**	**0.1824**	**0.1776**
OpenAI-large	sv	0.0494	0.1422	0.0354	0.0316	0.0133	0.0229	0.0366	0.0396	0.0914	0.0982	0.0151	0.0132
	sv_svo	0.2565	0.5367	0.1942	0.1613	0.0907	0.1408	0.1991	0.2103	0.3748	0.3823	0.1128	0.1089
	sv_svo_svoa	0.3276	0.6210	0.2433	0.2003	0.1164	0.1793	0.2509	0.2644	0.4543	0.4603	0.1486	0.1453
	svoa_conv_msg	0.3882	0.7072	0.2908	0.2355	0.1419	0.2157	0.2983	0.3132	0.5226	0.5277	0.1834	0.1794
	sit_svoa_conv_msg	0.3956	0.7121	0.2949	**0.2389**	0.1428	**0.2160**	0.3019	0.3171	0.5280	0.5330	0.1863	0.1827
	sv_svo_svoa_conv_msg	**0.4056**	**0.7129**	**0.2955**	0.2385	**0.1434**	0.2155	**0.3028**	**0.3185**	**0.5345**	**0.5395**	**0.1870**	**0.1835**

Table 17: Results for Different Embeddings and **Haiku**

embedding	combination	acc@1	acc@5	p@5	p@10	r@5	r@10	ndcg@10	ndcg@20	mrr@10	mrr@20	map@10	map@20
GTE	msg	0.2656	0.5278	0.1950	0.1601	0.0944	0.1462	0.2025	0.2151	0.3772	0.3843	0.1169	0.1140
	conv	0.2336	0.5016	0.1629	0.1295	0.0767	0.1158	0.1667	0.1729	0.3482	0.3558	0.0879	0.0814
	conv_msg	0.3156	0.6055	0.2323	0.1889	0.1118	0.1718	0.2394	0.2520	0.4380	0.4447	0.1404	0.1355
Nomic	msg	0.2294	0.4773	0.1678	0.1391	0.0835	0.1292	0.1766	0.1892	0.3358	0.3438	0.0997	0.0980
	conv	0.2131	0.4730	0.1529	0.1241	0.0732	0.1122	0.1582	0.1684	0.3255	0.3345	0.0832	0.0790
	conv_msg	0.2708	0.5487	0.1991	0.1631	0.0982	0.1522	0.2077	0.2221	0.3897	0.3972	0.1185	0.1165
NV-Embed	msg	0.1962	0.4170	0.1368	0.1097	0.0612	0.0924	0.1385	0.1391	0.2904	0.2968	0.0738	0.0663
	conv	0.0808	0.1839	0.0459	0.0356	0.0213	0.0309	0.0480	0.0510	0.1249	0.1317	0.0221	0.0202
	conv_msg	0.1571	0.3382	0.1049	0.0830	0.0474	0.0721	0.1074	0.1111	0.2346	0.2421	0.0563	0.0515
LLM2Vec	msg	0.2619	0.5467	0.1991	0.1613	0.0963	0.1455	0.2032	0.2154	0.3820	0.3899	0.1160	0.1123
	conv	0.1537	0.3179	0.0933	0.0735	0.0459	0.0690	0.0989	0.1089	0.2259	0.2359	0.0499	0.0479
	conv_msg	0.2825	0.5496	0.2008	0.1633	0.0985	0.1503	0.2092	0.2220	0.3988	0.4067	0.1189	0.1152
OpenAI-small	msg	0.2599	0.5256	0.1945	0.1601	0.0972	0.1498	0.2031	0.2197	0.3724	0.3799	0.1181	0.1183
	conv	0.2705	0.5470	0.1869	0.1493	0.0902	0.1359	0.1929	0.2027	0.3863	0.3942	0.1060	0.1005
	conv_msg	0.3116	0.6027	0.2280	0.1887	0.1126	0.1741	0.2390	0.2543	0.4350	0.4422	0.1408	0.1382
OpenAI-large	msg	0.2893	0.5733	0.2146	0.1797	0.1075	0.1677	0.2272	0.2444	0.4123	0.4194	0.1331	0.1332
	conv	0.3073	0.5901	0.2129	0.1717	0.1025	0.1551	0.2203	0.2313	0.4263	0.4343	0.1248	0.1185
	conv_msg	0.3425	0.6592	0.2591	0.2103	0.1286	0.1954	0.2670	0.2839	0.4759	0.4821	0.1601	0.1578

Table 18: Baseline for conversation data retrieval performance of each embedding model

Cluster	Representative Samples
Restaurants & Food	Asks restaurant, mentions food, asks food, asks music at restaurant
Movies	Likes movie, asks movie, mentions movie, acknowledges movie
Well-being & Health	Asks question about well-being, inquires well-being, asks opinion
Reservations	Specifies number of people, confirms booking, asks booking
Family & Pets	Mentions family, mentions work, mentions kids, mentions dog
Sports & Hobbies	Asks sports, asks interest in sports, mentions football, plays games
Pricing & Admission	Asks fee for entrance, asks method of payment, mentions price
Conversation Closure	Declines help, declines offer, says goodbye, ends conversation
Miscellaneous	Has kids, greets morning, feels tired, watches TV
Location Inquiries	Asks location, asks hotel, mentions hotel, asks duration of stay
Interest in Others	Asks you, greets person with question, asks activity, asks today
Greetings	Greets person, expresses gratitude, acknowledges response
Music & Pets	Likes music, mentions hobbies, has dog, enjoys music
Recommendations	Thinks better, asks for recommendations, suggests change of topic
Asking Opinions	Expresses opinion, states opinion, expresses frustration

Table 19: Intent Clusters and Representative Samples from Data

Model	Context	Metric			
		acc@1	**ndcg@5**	**ndcg@10**	**ndcg@20**
HEISIR $_{\text{GPT-3.5-turbo, All-embeddings}}$	x	0.3569	0.2869	0.2691	0.2813
	o	**0.3683**	**0.2959**	**0.2772**	**0.2902**

Table 20: Metric Changes with and without context

to capture relational dependencies between words. Since HEISIR avoids the use of pronouns to ensure semantic completeness, this lack of context makes it difficult to extract accurate information. Consequently, extracting SVOA quadruplets from context-less messages not only reduces the precision of the semantic indices but also results in fewer indices being generated. For future applications beyond retrieval tasks, incorporating previous context will be necessary.

F Dataset License and Disclaimer

The datasets used in this study are subject to the following licenses: BAD (Xu et al., 2021) and DICES (Aroyo et al., 2024) (CC BY 4.0), DailyDialog (Li et al., 2017) (CC BY-NC-SA 4.0), PILD (Xu et al., 2020) (MIT license), USS (Sun et al., 2021) (individual licenses), ReDial (CC-BY-4.0), MWOZ (MIT License), and SGD (CC-BY-SA-4.0). We strictly adhered to these licenses and confirm that no commercial use was made of any dataset in this research. While BAD and DICES datasets contain some inherently harmful content, we used these datasets solely as search targets and emphasize that our paper does not include any harmful content in its presented material.