Efficient Answer Retrieval System (EARS): Combining Local DB Search and Web Search for Generative QA

Nikita Krayko[1] Ivan Sidorov[1,2] Fedor Laputin[1,2] Daria Galimzianova[1] Vasily Konovalov[3,4]

[1]MTS AI [2]HSE University [3]AIRI, Moscow, Russia
[4]Moscow Institute of Physics and Technology, Russia

Motivation and Objectives

Developing a virtual assistant for a commercial company is essential for enhancing customer experience. This paper presents **a factual QA skill** as part of the virtual assistant for Mobile TeleSystems (MTS). Our production-ready QA system integrates two methodologies: knowledge base search and an LLM-based solution, enhanced with search engine context. Key contributions include:

- Effective integration of knowledge base search with advanced LLM techniques.
- A language-agnostic pipeline for developing factual QA systems in any language.

Conclusions

EARS is a lightweight, production-ready voice-oriented LFQA system designed for real-time user interaction. Key results include:

- The ranker component improves local knowledge base retrieval by 23%.
- The LLM utilizes web search API context, resulting in a 92.8% increase in voice response Usefulness.
- Automatic validation significantly accelerates benchmarking by nearly three times, offering comparable quality to the LLM-as-Judge approach.

Pipeline

In this work, we introduce **EARS**, a production-ready factual question answering system. Answers can be derived either from a local knowledge base or generated by an LLM using context obtained from the web search API. There are 3 types of answers:

1. Answer by Local Base.
2. Answer by LLM.
3. Answer by Web Search if LLM refused (not available in audio channel).

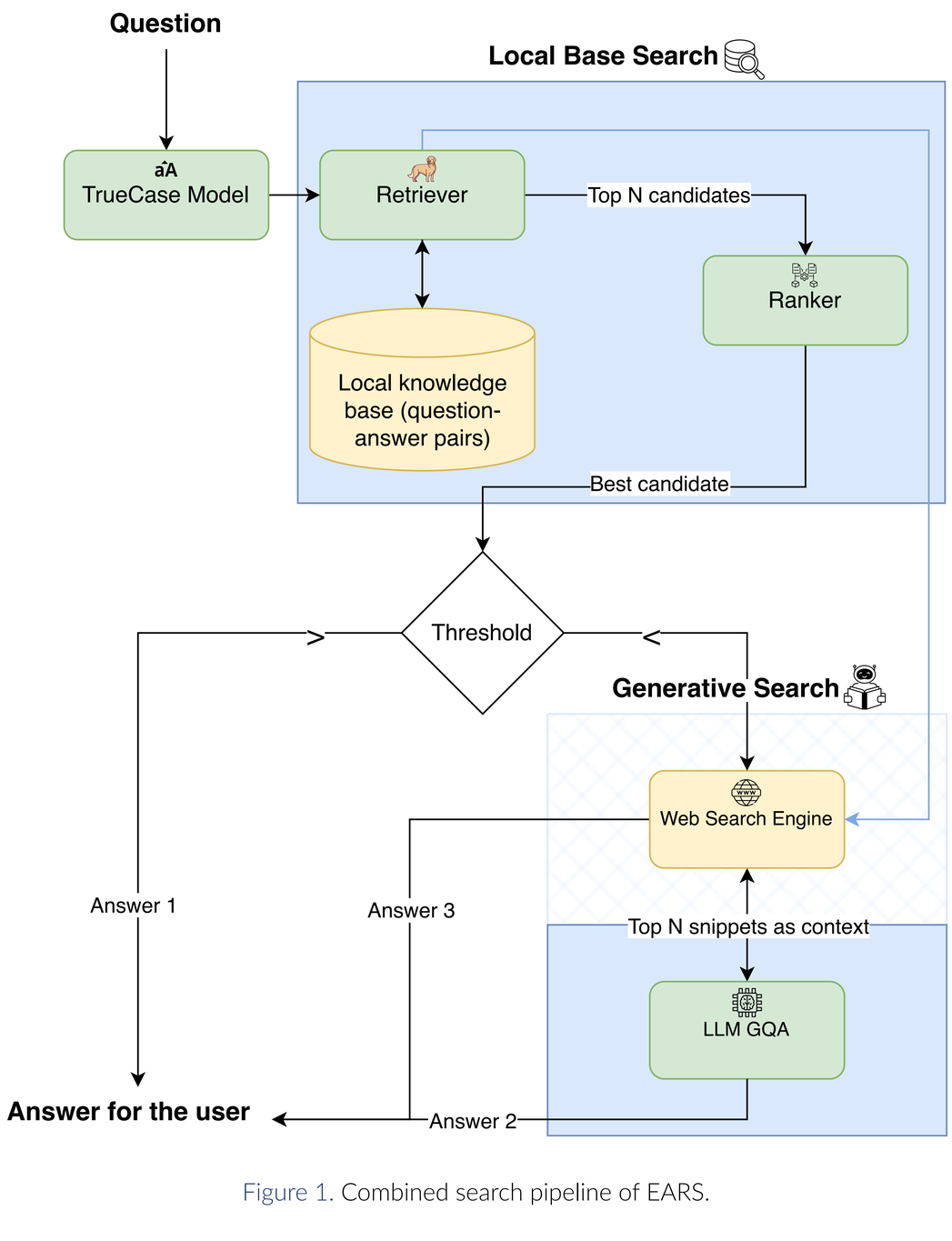

Figure 1. Combined search pipeline of EARS.

Components

Local base search. Designed to meet customer needs, it provides pre-prepared answers in specific domains, such as product advertising. We selected 300,000 popular Wikipedia articles from the last five years for our local database and developed a pipeline for auto-updating data from Wikipedia and Wikidata.

Retriever. This component includes an embedder and Approximate Nearest Neighbor (ANN) search, utilizing both symmetric (query-query) and asymmetric (query-passage) semantic searches. We encode triplets (query-title-passage) from the local database using the multilingual E5 large embedding model, with embeddings indexed via Faiss.

Ranker. A gradient boosting model trained with a tailored pairwise or listwise loss function identifies the most accurate answer among candidates retrieved by the Retriever.

Generative Search. To answer questions using external search engines, we follow two stages: (1) retrieve relevant context; (2) provide the LLM with the request and context. If the context is insufficient, the model outputs *"No information"*.

System performance

We created a domain-specific golden set of 1,600 factual questions and their human-crafted benchmark answers (ground-truth), representative of user queries in length and complexity.

The Usefulness metric evaluates QA system responses, defined as an answer that addresses the question. Scores range between 0, 0.5, or 1, averaged over the validation set for the final Usefulness score.

Model	Usefulness		*"No info"* proportion		Usefulness excl. *"No info"*	
	top-5	top-10	top-5	top-10	top-5	top-10
Mistral 7B	**59.58**	**65.43**	13.74	8.56	69.02	71.50
GPT-3.5	55.79	63.17	13.44	8.68	64.41	69.13
GPT-4o	50.09	50.00	**40.87**	**40.87**	**84.62**	**84.47**

Table 1. Usefulness on different LLMs evaluated on the golden set. Top-5 and top-10 indicate the number of search engine snippets passed to the model as context.

In Figure 2, the distribution of our Usefulness metric varies significantly based on the presence of the web search component. For the voice channel, the optimal service version combines Ranker and LLM, achieving approximately 55% Usefulness.

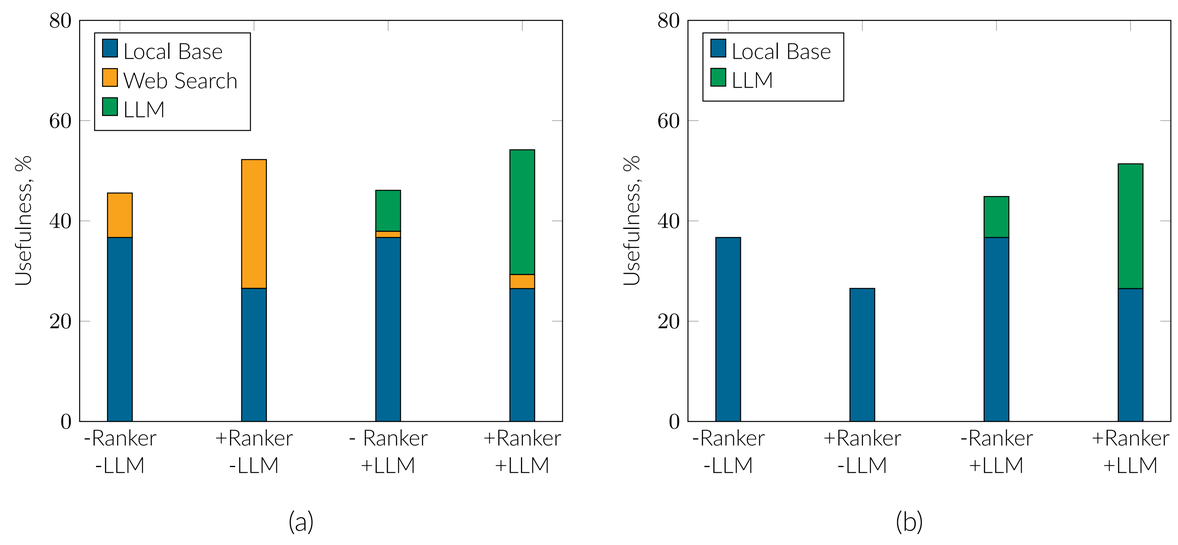

Figure 2. The Usefulness measured across the Ranker and LLM components. The histogram (a) includes Usefulness scores for all answers provided to the users. The histogram (b) excludes the web search component and only shows the scores for the answers our system was able to voice.

We pre-calculate deterministic metrics, similar to those in Figure 3, for pairs of generative responses and reference answers, then approximate these values with an ensemble model. This method enables us to create a resource-efficient ensemble for the Correctness task.

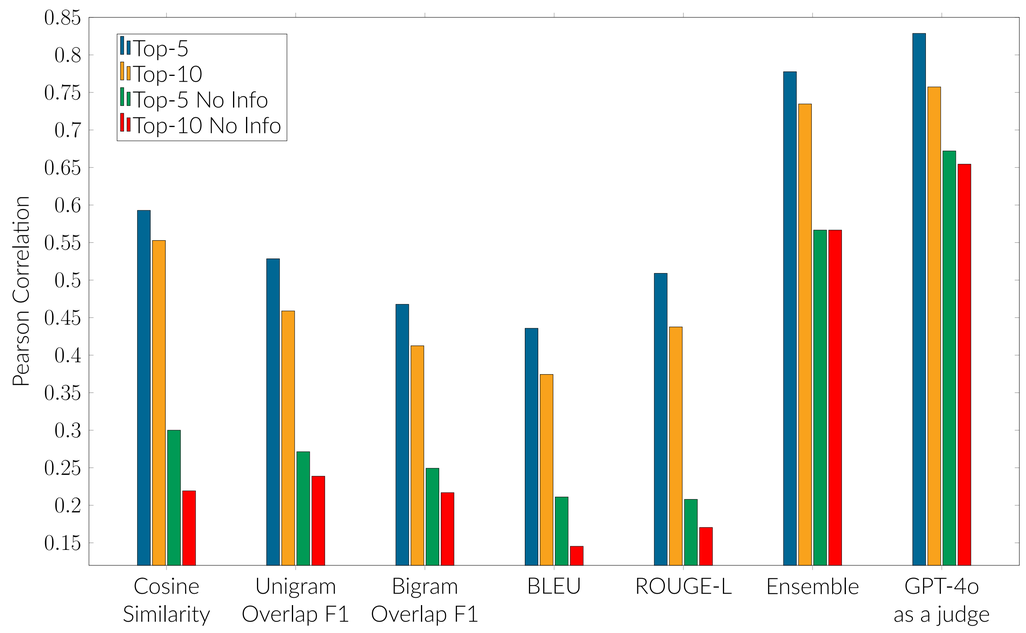

Figure 3. Metrics evaluated on the golden set and compared using Pearson's correlation with human evaluation (Usefulness) labels.

n.kraiko@mts.ai