YedidAI: A Hebrew RAG-based Assistant for Social Rights

Short Project Report

Abstract

We present YedidAI, a prototypical retrieval-augmented generation (RAG) system designed to assist Hebrew-speaking users in understanding and exercising social and legal rights in Israel. The pipeline integrates a dense semantic search index (SentenceTransformers) over curated rights pages with a conversational front-end driven by large language models (Anthropic / OpenAI). This short report summarizes the dataset, architecture, methods, qualitative results, and immediate future work and limitations. The project was motivated by the confusion and information gaps following the October 7th war in Israel: many evacuees, injured soldiers and civilians, families of hostages, and bereaved families were unsure of their entitlements or how to locate relevant guidance on the KOLZchut website. YedidAI aims to provide a conversational interface where users can describe their situation in free text, receive clarifying questions, and obtain personalized rights and links to authoritative pages.

1 Introduction

Accessing authoritative, personalized information about rights and entitlements can be challenging. This work was motivated by the confusion at the beginning of the October 7th war in Israel, when many people who deserved rights and assistance (evacuees, injured soldiers and civilians, families of hostages, bereaved families, and others) were either unaware of their entitlements or unable to find the relevant information on the KOLZchut website.

YedidAI addresses this gap by combining information retrieved from a structured corpus of Israeli rights pages with an empathetic, social-worker-style conversational agent. Our goal was to create a chatbot that allows users to tell their story in free language, asks clarifying follow-up questions to uncover missing or forgotten details, and returns the likely entitlements along with links to authoritative pages for next steps. The system is aimed at producing concise, actionable guidance in Hebrew, while surfacing relevant legal/administrative references.

2 Related Work

Prior work on RAG systems shows that coupling retrieval with generative models improves factuality and grounding. Systems focused on public advice and legal aid have demonstrated the value of domain-specific corpora and careful prompt/system prompt design. YedidAI follows these principles and emphasizes empathetic communication appropriate for social services.

3 Dataset

The project uses a curated JSON corpus (derived from public rights pages / XML dumps) covering domains such as disability benefits, pensions, parking permit for disabled, and employment rights. Documents include structured fields ("title", "eligibility_section", "full_text") and were preprocessed and optionally split into passages. A set of serialized embeddings is stored in `data/embedded_data.json` to speed up retrieval.

4 Methodology

4.1 Retrieval

We build document embeddings using the multilingual sentence-transformers model `paraphrase-multilingual-Mi`
A semantic search engine computes cosine similarity to find top-k passages for a user query.

4.2 Generation / Conversational Flow

A chat module controls the dialogue, driven by a system prompt designed to emulate a highly empathetic social worker (Hebrew). After gathering a user profile via a short dialog, the system triggers retrieval using the profile as a query. Retrieved passages are assembled into a prompt that is sent to an LLM (Anthropic Claude or OpenAI models via wrappers) with instructions to produce a structured, concise Hebrew answer describing eligibility, entitlements, and next steps.

4.3 Safety, Policy and Prompting

System prompts enforce brevity, empathy, and factual grounding; function-call style endpoints exist for conversation summaries. We adopt conservative behavior around sensitive personal data and avoid offering legal advice that would require professional judgment beyond informational guidance.

5 Implementation Details

The codebase includes:

- `src/rag.py` – semantic search engine (embedding, save/load, semantic search).
- `src/chatbot.py` – conversational loop, system prompts, message history management.
- `src/anthropic_wrapper.py`, `src/openai_tools.py` – LLM wrappers and system prompts in Hebrew.
- `src/app.py` – Gradio interface plus orchestration between chat and retrieval pipeline.

All components are implemented in Python and configured via environment variables (API keys and model names). Embeddings are saved and re-loaded to avoid repeated compute costs.

6 Experiments and Results

At this stage the evaluation is largely qualitative and exploratory:

- Retrieval: the sentence-transformer model returns semantically relevant passages for user-profile-like queries; top-5 often contains passages covering eligibility and next-steps for typical queries.
- Generation: the LLMs produce empathetic, actionable instructions in Hebrew when provided with retrieved context and a clear system prompt template. Output quality depends strongly on retrieval precision and prompt structure.
- Observed issues: hallucination risk when retrieval misses core evidence; inconsistent citation of sources; occasional overly long responses despite brevity constraints.

We recommend a formal evaluation plan consisting of: precision@k for retrieval, ROUGE/BLEU-style overlap for grounded answers, and human evaluation for helpfulness, clarity, and empathy on a balanced set of user scenarios.

7 Conclusions and Future Work

YedidAI demonstrates a practical RAG-based assistant for Hebrew rights-related queries with a clear engineering design and initial qualitative promise. Next steps:

1. Quantitative evaluation (retrieval and end-to-end answer quality) and creation of an evaluation dataset.
2. Add provenance tokens and explicit citations for all generated claims.
3. Implement a safety and escalation flow (when to suggest consulting a professional).
4. Add fine-grained indexing (passage-level, improved preprocessing) and caching to reduce latency.

Acknowledgements

This report documents work in the `YedidAI` repository. Thanks to contributors and to public sources of rights information used to compile the dataset.