

Product Gap Analyzer Overview

LLM-Based Product Feature Gap Analyzer

Overview: This Streamlit app lets product teams compare their product narrative against competitor descriptions and identify missing or weaker features. The UI accepts a JSON file describing one primary product plus any number of competitors (sample stored in `data/sample_products.json`). The backend formats these entries into a prompt (`scripts/prompts.py`) and, when an OpenAI key is provided, calls the OpenAI ChatCompletion API. Without a key it returns the prompt snippet as a placeholder, letting you inspect the generated instructions.

Flow: 1. `app/streamlit_app.py` handles file upload (or uses the bundled sample). 2. `scripts/analyzer.py` loads the JSON via `load_products`, identifies the main product vs. competitors, builds the prompt, and invokes `call_llm`. 3. `scripts/embeddings.py` contains an `Embedder` class powered by `sentence-transformers` (default `'all-MiniLM-L6-v2'`). It can be wired into the analyzer for semantic retrieval or feature clustering. 4. `scripts/prompts.py` centralizes the gap-analysis prompt template, ensuring consistent instructions across UI and CLI usage.

Tech Stack: - Python 3.9 - Streamlit for the web UI - sentence-transformers + PyTorch for embeddings - Optional vector stores: Chroma or FAISS (templates included) - OpenAI API (or any LLM) for generating structured JSON insights - pandas/numpy for potential data wrangling, pdfplumber for reading PDFs, tqdm for progress instrumentation

Usage Tips: - Run `streamlit run app/streamlit_app.py`. - Supply your own JSON or reuse `data/sample_products.json`. Each entry needs `'id'`, `'name'`, `'type'` ('product' or 'competitor'), and `'description'`. - Add an OpenAI API key in the text field to get real model output; otherwise you will see a prompt preview (`'LLM_OUTPUT_PLACEHOLDER'`). - Extend `Embedder` + vector DB utilities to ground responses in retrieved evidence or to deduplicate competitor claims.

Extensibility Ideas: - Swap in local Hugging Face models by customizing `call_llm`. - Persist embeddings and product snapshots into Chroma/FAISS for historical comparisons. - Add evaluation notebooks under `'notebooks/'` for experimentation or fine-tuning.