


 kliewerdaniel / agentsearch01








[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)





Agentic Search by Ingesting Files into ChromaDB and accompanying script to convert .json to .md for extracting chats from OpenAI backup .json


★ 0 stars    🍴 0 forks    👁 0 watching    🌿 Branches    🔄 Activity  
🏷 Tags


🌐 Public repository









  1 Branch     0 Tags     












 kliewerdaniel	2 months ago	 
 .gitignore	init	2 months ago
 README.md	Update README.md	2 months ago
 agent_search.py	init	2 months ago
 extract_chats.py	init	2 months ago
 requirements.txt	init	2 months ago

 README  

# Multi-Agent Research System with ChromaDB for Searching OpenAI `.json` Backups

This Python-based toolkit lets you transform your OpenAI Chat export ( `.json` ) into a searchable knowledge base via ChromaDB, enabling advanced, iterative, agentic workflows for generating detailed research reports with your preferred local LLM (e.g. Ollama).

## Why This Exists

OpenAI now allows you to request an export of all your chat data in `.json` format. While HTML versions are provided, they're hard to search programmatically. This tool was built to:

- Convert your `.json` archive into `.md` files (organized by date).
  - Ingest these `.md` files (and any other document set) into ChromaDB.
  - Interact with that data via a multi-agent system that surpasses simple vector-chunk recall by applying a research-style, iterative agent orchestration.
- 

## What It Does

---

### 1. Extract chats to Markdown

Converts OpenAI `.json` exports and optionally scraped Reddit archives into time-stamped `.md` files, making them easier to ingest and scan.

### 2. Chunk and embed data into ChromaDB

Processes all `.md` files (or your own docs), chunks them, and stores embeddings in a local ChromaDB instance.

### 3. Multi-agent research workflow

Six specialized agents collaborate iteratively to answer your query:

- **ContextBot**: defines background and concepts.
- **SynthAI**: integrates disparate info.
- **ValidatorPro**: verifies claims and evidence.
- **ExplorerX**: finds related threads and surprising connections.
- **ChronoAgent**: handles timelines and sequencing.
- **TechSpec**: dives into technical and implementation specifics.

### 4. Dynamic collaboration & fallback

Agents monitor each other's confidence scores. If one agent is weak, it requests help from others—forming a dynamic peer-assistance network to enhance overall accuracy.

### 5. Synthesis & report generation

A master synthesizer agent merges all insights into a coherent research report. Output includes:

- Confidence scores per agent and chunk
- Source citations
- Contributions by agent
- Discovered topical relationships
- Recommendations for further investigation

### 6. Local LLM output via Ollama

Final report is generated using your local Ollama model of choice—no cloud inference required.

---

## Requirements

- Python 3.8+
- ChromaDB
- Ollama (local LLM runtime; must be installed and model downloaded)
- Standard Python libraries (install via `pip install -r requirements.txt`)

## ⚙️ Installation & Usage

```
git clone https://github.com/kliewerdaniel/agentsearch01.git
cd agentsearch01
pip install -r requirements.txt
```



Step 1: Prepare your data

- Download your OpenAI Chat .json export.
- Run `extract_chats.py` to convert it into .md files, organized by date.
- Optionally include any additional .md documents (e.g. scraped Reddit threads).

Step 2: Process Documents and Search

```
# Process files and start research
python agent_search.py --process --search

# Multi-agent research on a complex question
python agent_search.py --question "What are the main factors contributing to climate
change and their historical timeline?"

# Interactive multi-agent session
python agent_search.py --search

# Single question with custom iterations
python agent_search.py --max-iterations 10 --question "Complex research question"
```

## 📊 Example Workflow

1. OpenAI .json → Markdown files via `extract_chats.py`
2. .md documents → ChromaDB via `python analyze.py --process`
3. User query → `agent_search.py` runs:
  - Agents: ContextBot → SynthAI → ValidatorPro → ExplorerX → ChronoAgent → TechSpec
  - Agents collaborate, pass messages, and fallback dynamically.



- Final synthesizer orchestrates complete report via Ollama.

## 💡 Future Ideas & Improvements

- Improved chunking strategies: hierarchical splitting, semantic-aware segmentation
- Modular agent framework: break out agents into clean modules for reuse and extension
- RSS scraper integration: auto-fetch news updates, summarize to .md, and periodically ingest into ChromaDB
- Alternate agent architectures: explore better coordination strategies
- Optional UI: e.g. simple Streamlit interface to ask questions and view sources and confidence scores

## 🔍 Feedback & Collaboration

I'd love to hear your thoughts on:

- Better chunking or embedding workflows
- Improvements to agent collaboration or orchestration logic
- Best practices or frameworks for multi-agent modular design
- Integration of additional data sources like RSS, bookmarks, etc.

This approach has already outperformed basic semantic recall—especially for creating transparent, trustworthy research reports. Let's explore how to make agentic research on personal data even better.

## ⚙️ Quick Summary

Step Description

🗨️ Extract .json → dated .md files

Ingest .md → ChromaDB

Research Multi-agent search for your query

## Report Synthesized output via Ollama

Let me know how it goes if you try it out—or if you've already tested it! 🚀

---

### Releases

No releases published

---

### Packages

No packages published

---

### Languages

---

● Python 100.0%