Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

Improving the Data Quality of ConfSearch using LLMs

Semester Project

Samuel Bohl
bohls@ethz.ch

Distributed Computing Group
Computer Engineering and Networks Laboratory
ETH Zürich

Supervisors:
Andreas Plesner
Prof. Dr. Roger Wattenhofer

August 19, 2024

Acknowledgements

I thank Andreas Plesner for his supervision and guidance. I'm grateful to Prof. Dr. Roger Wattenhofer for approving this project. Acknowledgments to Dr. Michael Kuhn for initiating ConfSearch in 2007, and to Alex Thillen for his 2022 iteration.

Abstract

This project focuses on improving the data quality of ConfSearch, a tool for academic conference information retrieval. By integrating Large Language Models (LLMs), the system now extracts more comprehensive and accurate conference details from WikiCFP. Built on FastifyJS and PostgreSQL, the new backend enables parsing of complex event structures, including multiple tracks and workshops. The integration of CORE rankings provides additional context for conference quality. While achieving perfect accuracy for overlapping date fields with the previous version, the project acknowledges ongoing challenges in verifying LLM-generated content. Future work will address automating updates for non-WikiCFP conferences and enhancing search capabilities to fully utilize the enriched dataset.

Contents

Chapter 1

Introduction

1.1 Background and Problem Statement

ConfSearch is a tool designed to help researchers find and track academic conferences in computer science. It aims to provide up to date, comprehensive information about conferences, including their rankings, submission deadlines, and event details. The project has evolved through multiple iterations, with each version seeking to improve data quality and user experience.

1.1.1 Overview of ConfSearch and Its Purpose

ConfSearch serves as a centralized platform for academic conference information, allowing researchers to efficiently discover relevant venues for their work. It integrates data from various sources, including WikiCFP and CORE rankings, to offer a holistic view of conference quality and logistics. The tools primary purpose is to streamline the process of conference selection and submission planning for computer science researchers.

1.1.2 Data Quality Issues

Previous versions of ConfSearch faced challenges in extracting and structuring data from unstructured sources like WikiCFP. Key information about conference tracks, workshops, and specific deadlines was often embedded within event descriptions, making it difficult to parse and present accurately. These data quality issues limited the tool's effectiveness in providing comprehensive and reliable conference information.

1.2 Objectives

This projects aims to address these challenges through several key objectives. Primarily, I will leverage Large Language Models (LLMs) to enhance data ex-

traction and structuring capabilities, significantly improving the system's ability to parse complex, unstructured information. This approach is coupled with a focus on improving the accuracy and completeness of conference information, including detailed data on multiple tracks and workshops within each event.

Another crucial objective is the implementation of a robust backend architecture with automated data update mechanisms.

Lastly, a critical goal is to maintain data integrity while automating the process of keeping conference information current. This involves implementing safeguards against potential LLM hallucinations and ensuring that the automated update processes consistently provide accurate and reliable information to users.

Chapter 2

Related Work

2.1 ConfSearch 2022

I reviewed the previous iteration of ConfSearch, developed as a Bachelor's thesis by Alex Thillen in 2022 [1]. This version aimed to modernize the original ConfSearch tool from 2007. It introduced a more user-friendly interface with responsive design and updated the backend using Django. The architecture was modularized, separating frontend, backend, and data retrieval components. Search functionality was enhanced using weighted keywords, and the system integrated more up-to-date conference data. New features included bookmarking, visualization of important dates, and the ability to download conference dates as .ics files.

However, I found the documentation to be somewhat lacking, which made it challenging to build upon. This is why I decided to write the backend from scratch for this iteration of ConfSearch. Additionally, it could only partially utilize data from WikiCFP events, as crucial information about different tracks, workshops, special dates, and submission details is often embedded within the conference description, making it difficult to extract and structure programmatically.

2.2 WikiCFP

WikiCFP [2] is a community-driven platform that serves as a central hub for academic conference information. It provides comprehensive data about conferences, including their names, acronyms, submission deadlines, notification dates, and conference start and end dates. The platform boasts a sizeable and active user base, with over 100,000 monthly users contributing to and maintaining the data. This collaborative approach ensures that the information remains relatively current and reliable.

In the context of this project, WikiCFP proved to be an invaluable resource. I utilized it as a primary source for scraping conference information to populate the database. By using WikiCFP, I was able to gather data on a large number

of conferences from a single source, rather than having to search and compile information from multiple individual conference websites.

2.3 CORE Ranking

I used the CORE Conference Ranking [3] to assess conference quality in my project. CORE ranks computer science conferences as A*, A, B, or C. I focused on populating the database with A* conferences, representing the top-tier venues in each field. This helps users identify the most prestigious conferences, though I acknowledge that paper quality should be judged independently of venue ranking. The CORE ranking is determined by considering several factors, including citation impact, the strength of researchers involved, and the quality of the reviewing process. A* conferences are characterized by their high visibility beyond their specific research focus and are often the venues of choice for top researchers in the field. A and B ranked conferences may have similar paper quality but typically have less broad visibility or impact.

In my implementation, I allowed for manual overrides of the CORE ranking. This feature enables researchers to adjust rankings based on their domain expertise or more recent assessments. This provides flexibility when the community's perception of a conference's quality differs from the official CORE ranking.

2.4 Large Language Models (LLMs)

As mentioned in the title, the goal of this project is to leverage Large Language Models (LLMs) to enhance the capabilities of ConfSearch. LLMs are advanced machine learning systems trained on vast amounts of text data, enabling them to understand and generate human-like text. I primarily used OpenAI's GPT [4] models, accessed through their API, to parse and structure conference information from unstructured text.

LLMs proved particularly useful in extracting key details about conferences, such as dates, tracks, and submission guidelines, from the often inconsistent and unstructured descriptions found on platforms like WikiCFP. This approach allowed me to overcome the limitations of previous iterations in handling embedded information. By using LLMs, I was able to more accurately populate the database with structured conference data, improving the overall quality and completeness of the information presented to users. With proper prompting, LLMs can produce highly structured output, which was crucial for my project. Initially, I experimented with Llama 3.1 8B [5], but ultimately switched to GPT-4o due to its superior performance, especially after OpenAI introduced a structured output feature. This feature eliminated the 5-6% invalid output rate I experienced with Llama 31. 8B.

2.4.1 Structured Output using Modelfusion

I initially experimented with ModelFusion's [6] strucutred output method, which uses a combination of prompt engineering and post-processing. ModelFusion allows developers to define a schema using Zod, a TypeScript-first schema validation library. The LLM is then prompted to generate output that adheres to this schema. After generation, ModelFusion attempts to parse the LLM's output according to the defined schema. If parsing fails, it can retry with adjusted prompts or fall back to error handling mechanisms.

This approach is more flexible as it can work with various LLM providers and models, not just those specifically trained for structured output. However, it may require multiple attempts to generate valid output, potentially increasing latency and token usage. While this method proved useful in my initial experiments, I ultimately opted for OpenAI's more deterministic approach for its guaranteed schema adherence and efficiency in processing conference information.

2.4.2 OpenAIs Structured output

OpenAI took a two-part approach [7] to improve reliability for model outputs that match JSON Schema. They trained their newest model gpt-4o-2024-08-06 to understand complicated schemas and produce matching outputs. Additionally, they implemented a deterministic, engineering-based approach called constrained decoding to achieve 100% reliability.

Constrained decoding limits the model's token choices to only those that would produce valid output according to the supplied schema. This is implemented dynamically, determining valid tokens after each generation step. The process involves converting the JSON Schema into a context-free grammar (CFG) [8] and preprocessing it for efficient access during sampling. During token generation, the inference engine determines which tokens are valid based on previously generated tokens and the grammar rules. This list of valid tokens is used to mask the next sampling step, effectively setting the probability of invalid tokens to zero. The preprocessed schema allows this to be done efficiently, with minimal latency overhead. This approach ensures that the model consistently produces structured output that conforms to the specified schema.

Chapter 3

System Architecture and Implementation

3.1 Backend Structure and API Design

The backend architecture Figure 3.1 is built around a FastifyJS [9] server interfacing with a PostgreSQL [10] database. At its core, an API Service handles incoming requests, routing them to appropriate handlers. A CRON [11] Service manages scheduled tasks for data updates, while a Scrape Engine is responsible for extracting and parsing event information from external sources. The database, utilizing Drizzle ORM [12] for operations, stores conference and event data. A Reverse Proxy acts as an intermediary, managing client requests to the backend services. The API follows REST [13] principles with endpoints for conferences, events, and search. It includes middleware for handling database connections and CORS [11].

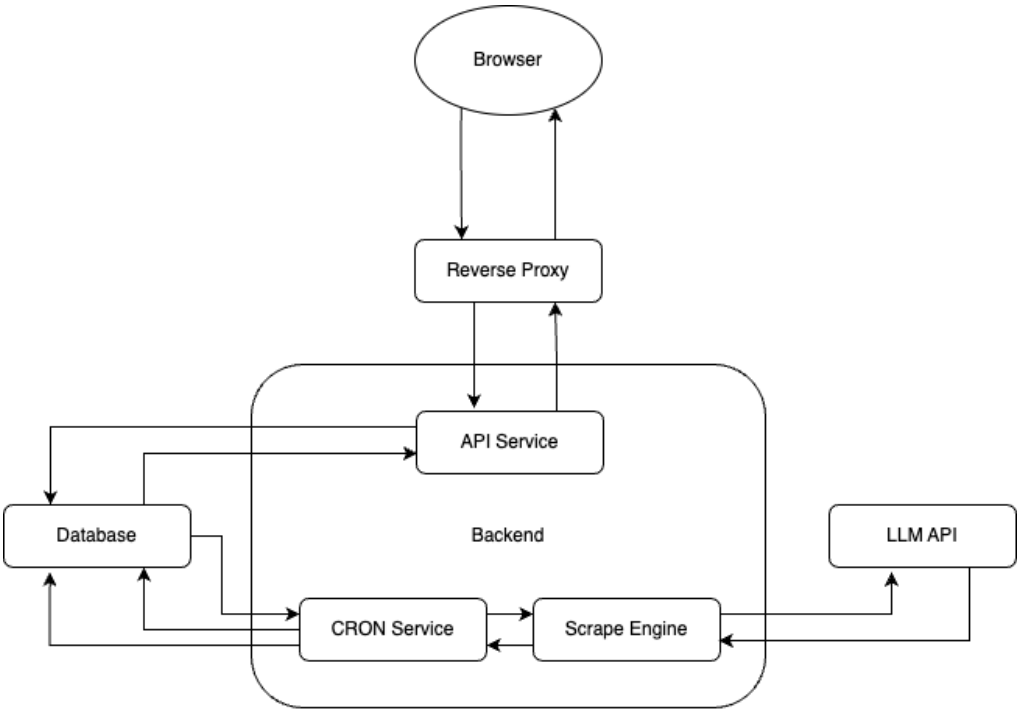

Figure 3.1: The ConfSearch backend architecture.

3.1.1 Data Model

The data model organizes information about conferences, their events, and related tracks or workshops. At the top level, we have conferences, which are broad series like "ICML" (International Conference on Machine Learning). Each conference can have several yearly events, such as "ICML 2024." Each event includes details like dates, submission deadlines, and relevant URLs.

Within these events, there are sometimes different tracks or workshops, which focus on specific topics or themes. For example, ICML 2024 might feature a workshop on "Deep Learning," with its own set of deadlines and details. This model helps manage and organize information about the overall conference series, individual events, and the specialized sessions within those events. Figure 3.2

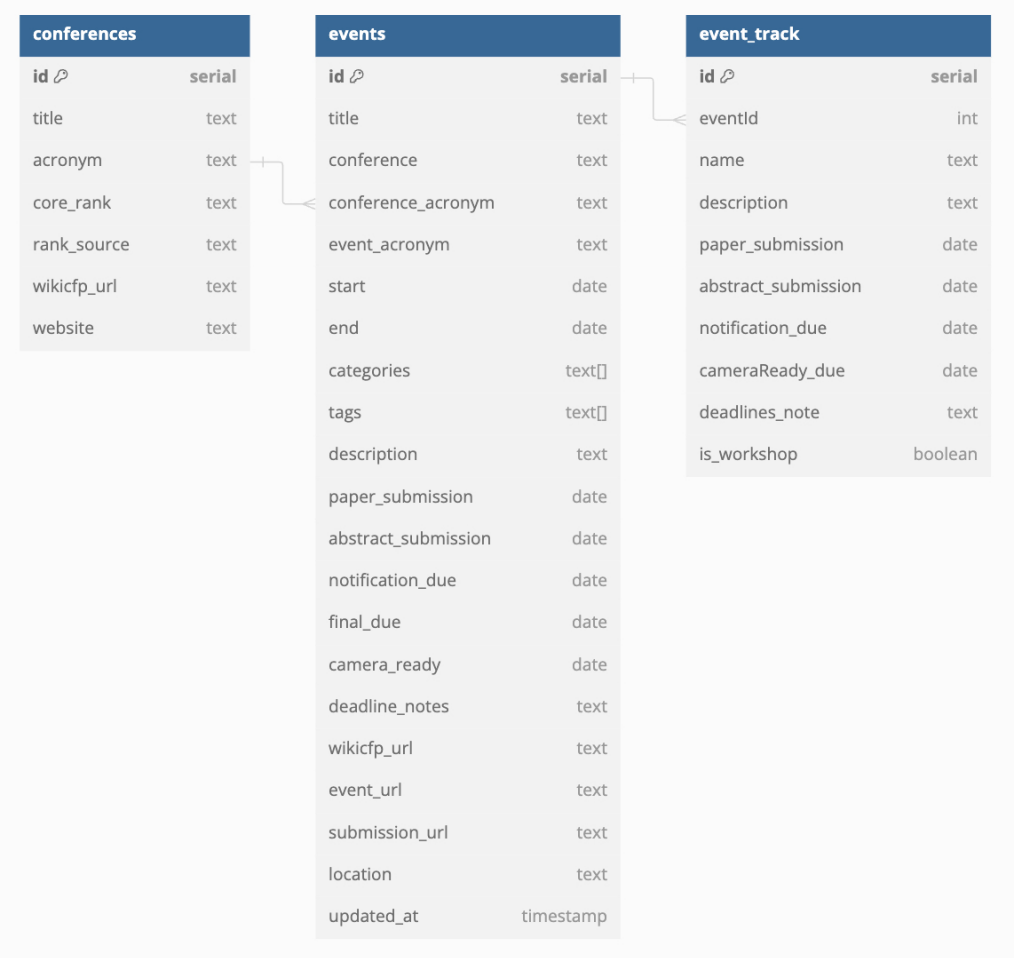

Figure 3.2: The ConfSearch database data model.

3.2 LLM Integration

3.2.1 Event Parsing with GPT Model

The system integrates OpenAI's GPT model to enhance event information parsing. The `openaiParseEvent` function, found in openai-event-parser.ts, serves as the bridge between raw context data and the GPT model. It sends contextual information to the model, which then generates structured event data based on a predefined schema. This approach allows for intelligent interpretation of unstructured or semi-structured event information. The parsed results undergo validation against a Zod schema, which ensures the data is in the correct format before I could insert or update the data in the database.

3.2.2 Schema Definition and Validation

Event data within the system is subject to strict typing and validation processes. The event-schema.ts file is central to this, defining comprehensive Zod schemas for both events and their associated tracks. These schemas play a crucial role in maintaining consistent data structure and type safety throughout the application. They include custom transformations applied to categories and dates, ensuring standardization across all event entries. This robust schema definition and validation process significantly reduces the likelihood of data inconsistencies and enhances the overall reliability of the system's event information.

3.3 Data Update Mechanisms

The system employs CRON jobs to manage data updates efficiently. These jobs, defined in cron.ts, are scheduled to regularly check for new conference events on WikiCFP. When new events are detected, the `localPrepareWikicfp` function scrapes the relevant content from the WikiCFP pages. This data is then processed using an AI model to extract and structure the information. Specialized functions like `upsertCoreRankings`, `upsertWikiCFPSeries`, and `updateEventFromLLMResponse` handle the database updates and ensure that the information is stored and refreshed accurately. This multi-step approach combines scheduled checks with advanced AI processing to keep the database current with the latest conference details.

Chapter 4

Results, Discussion, and Conclusion

4.1 Data Quality Improvements

The latest iteration of ConfSearch significantly enhances data extraction capabilities. Previously, the system could only extract start and end dates, deadline dates, and notification dates. Now, it captures additional crucial information such as separate paper and abstract submission deadlines, camera-ready dates, and supplementary instructions like deadline timezones. The system now also accommodates data for different conference tracks, a feature that was not possible in earlier versions. When available, submission URLs are now included in the extracted data.

4.1.1 Comparative Analysis

Verifying the accuracy of the new data extraction process posed challenges due to the absence of comparable old data for many new fields. However, manual spot checks of the final version of our scrape engine showed perfect accuracy. To assess the accuracy of the LLM-parsed dates that were common to both old and new systems, a script (`accuracy.js`) in the experiments folder was developed. This script compared the data from the old system with the new, yielding 100% accuracy for the overlapping date fields.

4.2 Challenges and Limitations

Verifying the extracted data and preventing hallucinations in LLM responses remain significant challenges in this projects. LLMs are prone to generating false information, which makes maintaining data integrity a constant concern.

To address this, I implemented a strategy of explicitly instructing the model to return null values when the data is unavailable, which notably improved the

quality of the extracted information. While the experiments in the previous section showed high accuracy for overlapping date fields with the previous system, there is not guarantee that this will hold true for future parsed events.

4.3 Future Work and Recommendations

There are several areas for future development. Firstly, conferences not originating from WikiCFP are not automatically updated in the current system. I propose using LLMs to predict new URLs for upcoming events and verify their availability.

Secondly, the current search functionality, while incorporating the new categories and tags, is still limited to a basic full-text search. This is a step back from ConfSearch 2022's more sophisticated ranking system. I implemented the search using PostgreSQL's text search capabilities, which includes the new fields (e.g. tags and categories) in the search scope but doesn't provide any ranking or weighting of results. While this approach allows users to find events based on all available data, it doesn't offer the nuanced, prioritized results that the previous iteration of Confseach has. Enhancing this area to better leverage our expanded data structure and implement a more sophisticated ranking algorithm is an area for improvement.

Lastly, I suggest implementing a user authentication system to restrict deletion rights to trusted users, which would help prevent malicious actions and improve the overall security and reliability of our system.

Bibliography

[1] A. Thillen, "Confsearch 2022," Zürich, Switzerland, August 2022, distributed Computing Group, Computer Engineering and Networks Laboratory.

[2] WikiCFP. Accessed: 2024-08-19. [Online]. Available: http://www.wikicfp.com

[3] CORE Raking. Accessed: 2024-08-19. [Online]. Available: https://portal.core.edu.au/conf-ranks/

[4] OpenAI, "Gpt-4 technical report," https://arxiv.org/abs/2303.08774, 2023, accessed on [Insert Date].

[5] Meta AI, "Introducing llama 3.1: Our most capable models to date," July 2024, retrieved 2024-08-19. [Online]. Available: https://ai.meta.com/blog/meta-llama-3-1/

[6] ModelFusion Contributors, "Modelfusion," https://github.com/lgrammel/modelfusion, 2023, accessed: 2024-03-15.

[7] OpenAI. Introducing structured outputs in the api. Accessed: 2024-08-19. [Online]. Available: https://openai.com/index/introducing-structured-outputs-in-the-api/

[8] N. Chomsky, *Three models for the description of language.* IEEE, 1956, vol. 2, no. 3.

[9] Fastify Contributors, "Fastify - fast and low overhead web framework, for node.js," 2017. [Online]. Available: https://www.fastify.io/

[10] PostgreSQL Global Development Group, *PostgreSQL: Documentation*, 1996, version 16.0. [Online]. Available: https://www.postgresql.org/docs/

[11] Paul Vixie, *Cron: A time-based job scheduler in Unix-like operating systems*, 1987, version 3. [Online]. Available: https://pubs.opengroup.org/onlinepubs/9699919799/utilities/crontab.html

[12] Drizzle Team, "Drizzle orm - type-safe sql queries for typescript," 2024. [Online]. Available: https://orm.drizzle.team/

[13] R. T. Fielding, "Architectural styles and the design of network-based software architectures," *Doctoral dissertation, University of California, Irvine*, 2000. [Online]. Available: https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm