LLM-Powered Argument Extractor: A Gemini-Based Web App for Academic Papers

1. Overview & Problem Statement

Modern research often requires reading lengthy documents to extract core arguments, theses, and supporting evidence.This process is repetitive and consumes significant time for students, researchers, and analysts.

To address this challenge, we've developed a **Gemini-powered web application** that allows users to:

- Upload research documents (PDFs),
- Extract structured arguments using an LLM,
- Preview the output in markdown, and
- Save or revisit previous extractions.

This app functions as a smart **academic assistant with memory**, optimized for clarity, convenience, and performance.

2. System Architecture & Technology Stack

Frontend – React.js

The interface is built with **React functional components and hooks**.

Key UI features include:

- **File Input** with custom instruction prompt
- **Markdown Preview** using `react-markdown` for rich formatting
- **Clean Left/Right Layout**: file list on the left, output preview on the right

Backend – Node.js + Express.js

The backend handles file processing, argument extraction, and storage through defined REST APIs:

Route	Functionality
`/extract`	Uploads file and triggers argument extraction via Gemini
`/save-doc`	Sends extracted content to Google Docs
`/regenerate/:id`	Reprocesses previously uploaded files
`/arguments`	Returns list of all processed files
`/arguments/:id`	Fetches arguments for a specific file

3. LLM Integration – Google Gemini API

The core argument extraction is powered by **Google's Gemini API**, accessed via the **GenAI SDK**. Features include:

- Accepts **base64-encoded PDFs**
- **Accepts Natural Language modifiers**, such as:
  - "Humanize this"
  - "Summarize and extract key findings"
- Extracts content into a **structured argument format**:
  - Main thesis
  - Claims

    - Evidence
    - Reasoning
    - Source sections (if available)

4. Cloud & Storage Integration

Google Docs API

After extraction, users can save the structured content to a **new Google Docs file**. This includes:

- Auto-creation of a doc with arguments inserted
- Link returned for viewing/editing
- Preserves formatting and structure

MongoDB via Mongoose

Every interaction is persisted using MongoDB, storing:

- Original filename
- Extracted arguments
- Timestamps
- Google Docs link (if saved)

This allows file history, reloading past extractions, and **regenerating arguments** on demand.

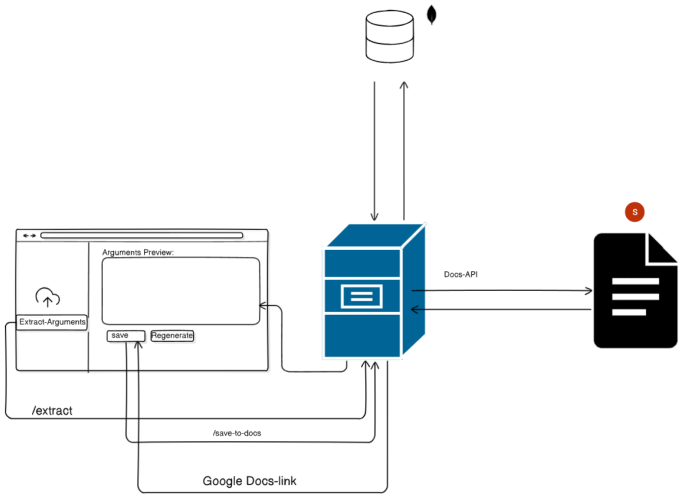
5. Key Features Summary

Feature	Description
File Upload	Upload any PDF; system reads & extracts arguments
Custom Prompt	Add instructions like “summarize” or “simplify” for custom output
Gemini Extraction	Uses GenAI SDK to analyze and return logical argument structures
Markdown Preview	Results shown in a formatted scrollable preview
Regenerate Arguments	Re-extract from any previously uploaded document with one click
Save to Google Docs	One-click export to a Docs file for editing, saving, or sharing
File History Panel	View, reopen, and manage previously uploaded files
Smart UX	Disables buttons contextually, clears input after tasks, resets file UI

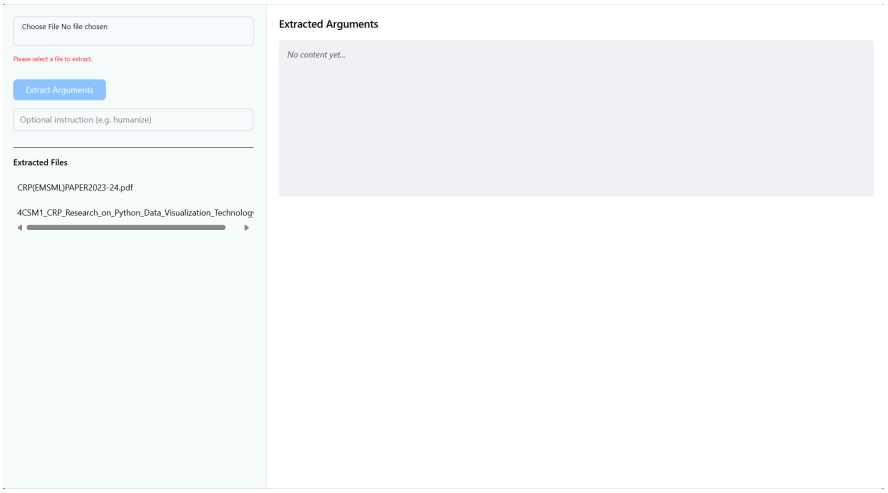
6. Conclusion

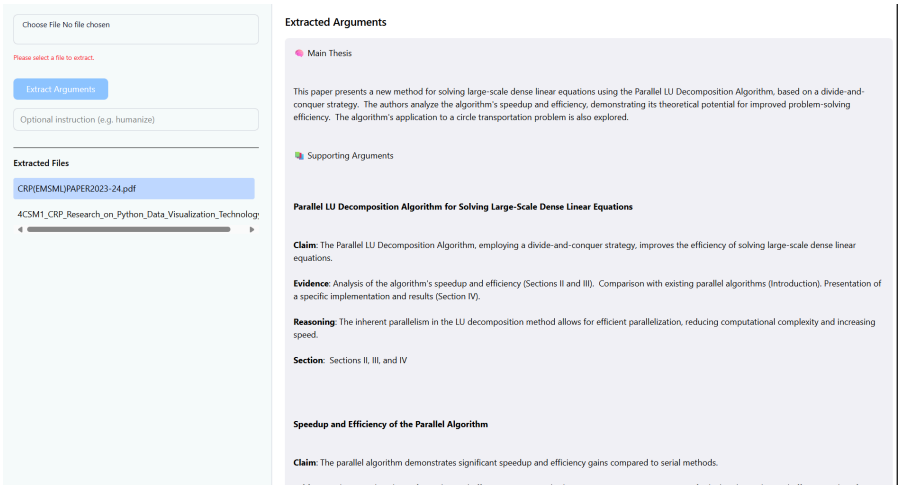
This web app combines the power of **Gemini LLM**, **React-based UX**, and **Google Docs APIs** to provide a fully automated academic assistant. Users can upload any research paper, extract meaningful arguments, preview and edit results, and maintain a persistent history for future use. This assistant streamlines scholarly work by enabling faster comprehension, clearer summaries, and seamless knowledge retention.

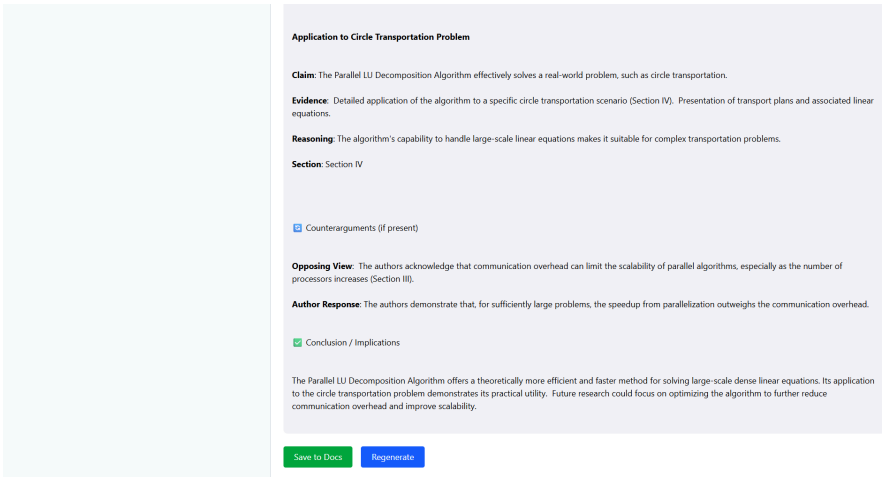
7.Workflow

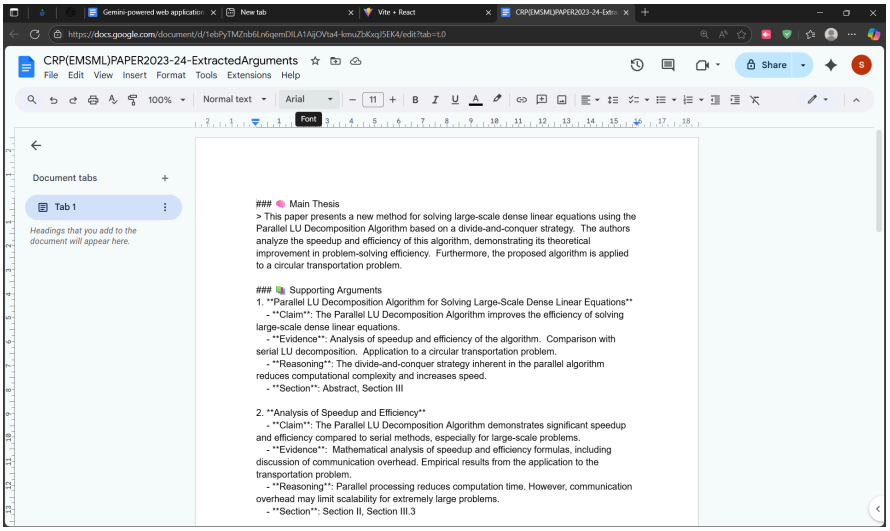


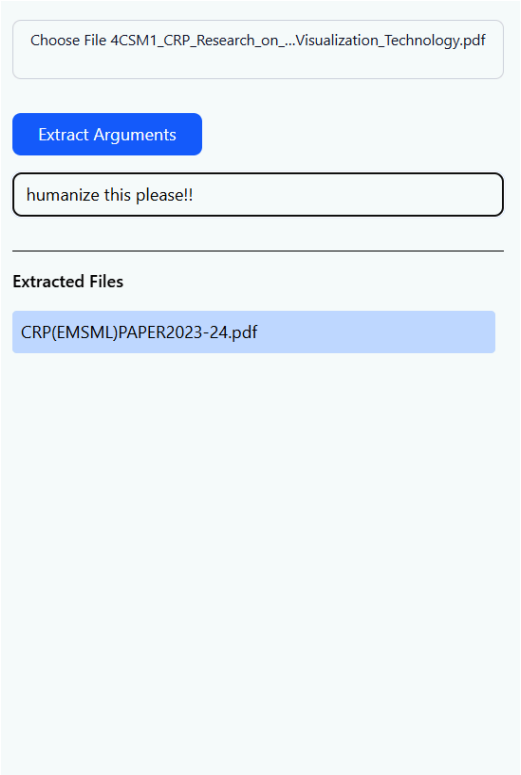
8.Images


Choose File No file chosen
Please select a file to extract.
Extract Arguments
Optional instruction (e.g. humanize)
Extracted Files
CRP(EMSML)PAPER2023-24.pdf
4CSM1_CRP_Research_on_Python_Data_Visualization_Technolog
Extracted Arguments
No content yet...


Choose File No file chosen
Please select a file to extract.
Extract Arguments
Optional instruction (e.g. humanize)
Extracted Files
CRP(EMSML)PAPER2023-24.pdf
4CSM1_CRP_Research_on_Python_Data_Visualization_Technolog
Extracted Arguments
Main Thesis
This paper presents a new method for solving large-scale dense linear equations using the Parallel LU Decomposition Algorithm, based on a divide-and-conquer strategy. The authors analyze the algorithm's speedup and efficiency, demonstrating its theoretical potential for improved problem-solving efficiency. The algorithm's application to a circle transportation problem is also explored.
Supporting Arguments
Parallel LU Decomposition Algorithm for Solving Large-Scale Dense Linear Equations
Claim: The Parallel LU Decomposition Algorithm, employing a divide-and-conquer strategy, improves the efficiency of solving large-scale dense linear equations.
Evidence: Analysis of the algorithm's speedup and efficiency (Sections II and III). Comparison with existing parallel algorithms (Introduction). Presentation of a specific implementation and results (Section IV).
Reasoning: The inherent parallelism in the LU decomposition method allows for efficient parallelization, reducing computational complexity and increasing speed.
Section: Sections II, III, and IV
Speedup and Efficiency of the Parallel Algorithm
Claim: The parallel algorithm demonstrates significant speedup and efficiency gains compared to serial methods.


Application to Circle Transportation Problem
Claim: The Parallel LU Decomposition Algorithm effectively solves a real-world problem, such as circle transportation.
Evidence: Detailed application of the algorithm to a specific circle transportation scenario (Section IV). Presentation of transport plans and associated linear equations.
Reasoning: The algorithm's capability to handle large-scale linear equations makes it suitable for complex transportation problems.
Section: Section IV
Counterarguments (if present)
Opposing View: The authors acknowledge that communication overhead can limit the scalability of parallel algorithms, especially as the number of processors increases (Section III).
Author Response: The authors demonstrate that, for sufficiently large problems, the speedup from parallelization outweighs the communication overhead.
Conclusion / Implications
The Parallel LU Decomposition Algorithm offers a theoretically more efficient and faster method for solving large-scale dense linear equations. Its application to the circle transportation problem demonstrates its practical utility. Future research could focus on optimizing the algorithm to further reduce communication overhead and improve scalability.
Save to Docs
Regenerate


CRP(EMSML)PAPER2023-24-ExtractedArguments
File Edit View Insert Format Tools Extensions Help
Share
Document tabs
Tab 1
Headings that you add to the document will appear here.
Main Thesis
> This paper presents a new method for solving large-scale dense linear equations using the Parallel LU Decomposition Algorithm based on a divide-and-conquer strategy. The authors analyze the speedup and efficiency of this algorithm, demonstrating its theoretical improvement in problem-solving efficiency. Furthermore, the proposed algorithm is applied to a circular transportation problem.
Supporting Arguments
1. **Parallel LU Decomposition Algorithm for Solving Large-Scale Dense Linear Equations**
- **Claim**: The Parallel LU Decomposition Algorithm improves the efficiency of solving large-scale dense linear equations.
- **Evidence**: Analysis of speedup and efficiency of the algorithm. Comparison with serial LU decomposition. Application to a circular transportation problem.
- **Reasoning**: The divide-and-conquer strategy inherent in the parallel algorithm reduces computational complexity and increases speed.
- **Section**: Abstract, Section III
2. **Analysis of Speedup and Efficiency**
- **Claim**: The Parallel LU Decomposition Algorithm demonstrates significant speedup and efficiency compared to serial methods, especially for large-scale problems.
- **Evidence**: Mathematical analysis of speedup and efficiency formulas, including discussion of communication overhead. Empirical results from the application to the transportation problem.
- **Reasoning**: Parallel processing reduces computation time. However, communication overhead may limit scalability for extremely large problems.
- **Section**: Section II, Section III.3


Choose File 4CSM1_CRP_Research_on_...Visualization_Technology.pdf
Extract Arguments
humanize this please!!
Extracted Files
CRP(EMSML)PAPER2023-24.pdf