NLP Project Report
Arowa Yasmeen, Romik Sarkar, Sabbir Ahmed
CS 6320.001, Professor Erekhinskaya May 6th, 2025

Project Name: **ResearchVoyager**

GitHub link: https://github.com/arowayasmeen/ResearchAssistant.git
Youtube link: https://youtu.be/SeeC93qSU-4

We aim to advance the Auto-research framework by developing a comprehensive **Research Writing Assistant** that streamlines the scientific research process. This system supports literature retrieval, gap analysis, idea brainstorming, summarization, passage generation, and proofreading, enhancing productivity without replacing human authorship.

Scope:

The automated research pipeline evolves through four phases—from human-controlled research to fully autonomous AI—transitioning from early, static methods like symbolic regression in AutoRA to end-to-end iterative systems. Users input a research prompt and upload relevant documents, while an advanced LLM curates state-of-the-art literature and performs gap analysis to pinpoint novel research directions. The system then synthesizes bullet points, summaries, and visualizations to generate an initial LaTeX draft, which is further refined for enhanced clarity, persuasiveness, and compliance with publication formats.

Team Members:

1. Arowa Yasmeen (axy210047)
2. Romik Sarkar (rxs190117)
3. Sabbir Ahmed (sxa230169)

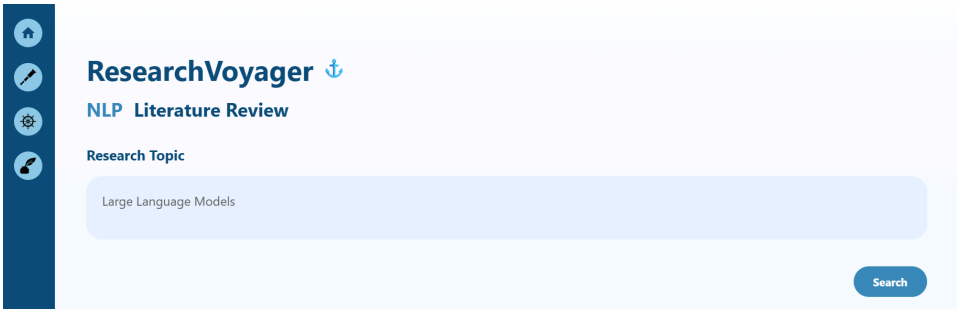
Arowa Yasmeen

80 points - Significant exploration beyond the baseline
30 points - Creativity: Came up with and implemented a custom paper ranking algorithm, and a LaTeX parser.
10 points - Development complexity in terms of building the solution, coming up with algorithms, optimization techniques (discussed below)
10 points - Discussion of challenges encountered and partial/complete solutions implemented (discussed below)
10 points - Implementation of front-end, maintenance of GitHub repository and README
10 points - Testing outside of the team

Contributions:

1. Implemented the complete **Literature Review Module**

i. **Paper Retrieval** - based on the topic name and search query entered by the user, this module retrieves research papers from public databases. It does so by using two search functions implemented using free APIs
   1. SearchGoogleScholar
   2. SearchSemanticScholar.



The search results from these two search engines are combined and de-duplicated.
Optimizations made: Implemented parallel processing to speed up the search.
Problems encountered: Free API rate limits are easy to hit during development. This still remains a bottleneck. Compatibility with the ArXiv database turned out

to be a challenge due to the extreme API rate limit. Testing took a long time to avoid unintended billings.
Partial Solution: Randomized delay is implemented in order to avoid hitting rate limits.

ii. **Paper Deduplication** - This functionality uses a hashmap lookup to facilitate looking up populated results and avoid duplication. The matching is done by using a thresholded string matching of the paper titles. The threshold is required due to paper titles sometimes having mismatched spacing and character case differences in results from different databases.

Our code prefers to retain the Semantic Scholar result over the Google Scholar result for the same paper due the rich information that is retrievable from the Semantic Scholar API.

iii. **Paper Ranker** - The deduplicated retrieved papers are now ranked using a custom algorithm that generates a relevancy score for each retrieved paper. The relevancy score is calculated based on:

1. **Similarity Score** - a *weighted addition* of:
    a. **Lexical Similarity** - Using **TF-IDF** we represent abstracts + paper titles of the retrieved papers and the search queries as vectors, and then calculate the similarity between them.
    b. **Semantic Similarity** - Using **BERT**, we encode the query and the paper title + abstract of the retrieved papers into query and paper embeddings. Then we compute the **cosine similarity** of the paper embeddings and the query embedding.
    Optimizations made: The BERT model used is pre-downloaded and included in the data folder to reduce latency.
2. **Citation Score** - This score is generated based on the citation count of the paper
$$\frac{log(1 + citation\ count)}{10}$$
Problems encountered: Citation counts are often 0.
Optimizations made: The equation accounts for the fact citation counts can be 0 for newer papers and also old papers.
Solution: A **dynamic weighting** system is implemented, which, for newer papers, puts more weight on recency and lowers weight on the citation count weight. However, for older papers, if the citation count is 0, then it puts equal weight on both the citation score and recency score.
3. **Recency Score** - This score allows us to favor more recent publications as compared to old ones.
$$\frac{1}{1+0.1*(current\ year - paper\ year)}$$
Problems encountered: Year of publication is not always available on search results retrieved in 1.i., which ultimately causes this score to be 0.

Solution: The weight of the Recency score is reduced so that it does not have a large effect on the final relevancy score due to incomplete search results.

The final relevancy score is a ***weighted addition*** of the Similarity, Citation, and Recency scores, and a results table similar to the one below is generated.

Title	Authors	Year	Venue	Relevance	Link
A survey on evaluation of Large Language Models	Chang Y, Wang X, Wang J, Wu Y, Yang L, Zhu K, Chen H, Yi X, Wang C, Wang Y, Ye W	2024	ACM transactions on intelligent systems and technology	95%	View Paper
A comprehensive overview of Large Language Models	Naveed H, Khan AU, Qiu S, Saqib M, Anwar S, Usman M, Akhtar N, Barnes N, Mian A	2023	arXiv	87%	View Paper
Large Language Models in medicine	Thirunavukarasu AJ, Ting DS, Elangovan K, Gutierrez L, Tan TF, Ting DS	2023	Nature medicine	82%	View Paper
A watermark for Large Language Models	Kirchenbauer J, Geiping J, Wen Y, Katz J, Miers I, Goldstein T.	2023	International Conference on Machine Learning	72%	View Paper
Welcome to the era of chatgpt et al. the prospects of Large Language Models	Teubner T, Flath CM, Weinhardt C, Van Der Aalst W, Hinz O.	2023	Business & Information Systems Engineering	65%	View Paper

Export Results CSV JSON

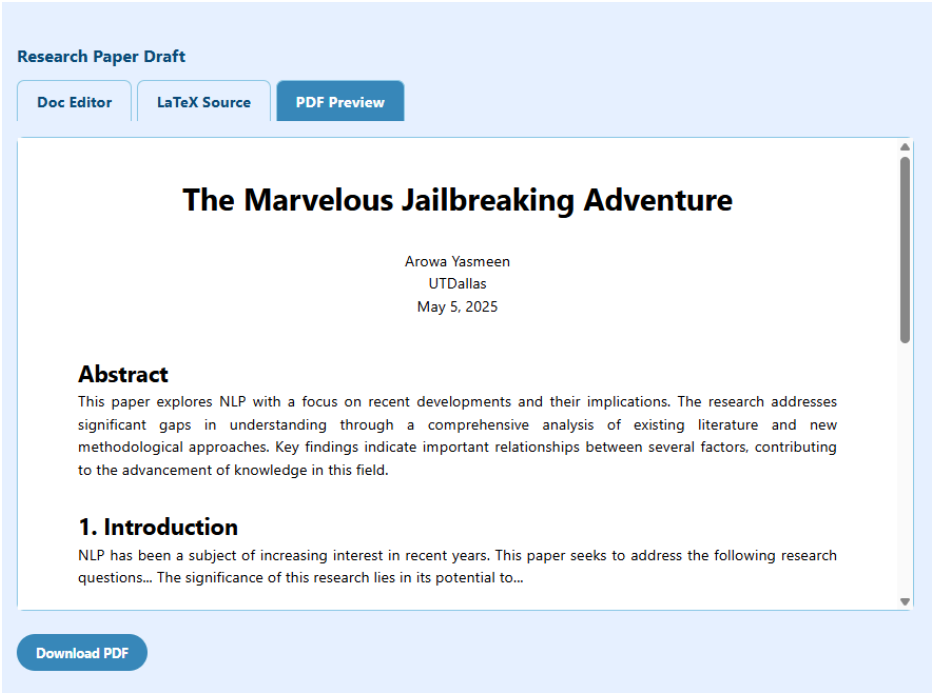
The links allow the user to navigate to the website from which they can download the papers.

Problems encountered: We opted not to implement any automatic downloads due to copyright and access barriers.

2. Helped to implement working **front-end** as well as compatibility to install and use ResearchAssistant as a **Python package**. Front-end pages implemented:
    a. index.html
    b. menu.html
    c. literature-review.html
    d. generate-ideas.html
    e. draft-preparation.html
3. Implemented the **LaTeX to PDF Previewer** for the Prepare Paper Draft module
    a. Extracted Abstract, Sections, Subsections, and Bibliography using **Regular Expressions** to parse the LaTeX content to basic HTML. The HTML is then converted to a PDF.

Optimizations made: The number and name of the sections are left up to the user and not preset.

Problems encountered: Was not able to develop the functionality to upload figures into the paper draft.



4. Outside Testing:
    a. Participant 1:
        i. Positives: Likes the simple UI
        ii. Suggested Improvements: Thinks the search is a bit slow, would like the option to generate a summary or view the abstract of the ranked research papers.
    b. Participant 2:
        i. Positives: Liked the paper outline generator. Felt that it eased the friction of starting to write a paper because they knew what to look for.

ii. Suggested Improvements: Participant 2 wants more features to compare methodologies of existing work in the Literature review module to understand and visualize what already exists better. They also wanted to be able to drag and drop figures on the draft generator.

Romik Sarkar

70 points - Significant exploration beyond the baseline, almost full implementation of backend/frontend with a running server
30 points - Creativity: Came up with a creative project theme and implemented and designed the UI for the project
10 points - Used a Gemini flash model, but enhanced it with rule-based prompt engineering and implemented feedback-driven refinement
10 points - Discussion of challenges encountered and partial/complete solutions implemented (discussed below)
10 points - Implementation of front-end, maintenance of GitHub repository, License and README,
10 points - included citation maker and plagiarism checker.
10 points - Testing outside of the team - Had three friends test out draft generation to check for nuances between different genres

Contributions

1. Creative design and development of UI
    a. Designed **front-end** interface, CSS style sheets
2. Designed a functioning document editor webpage
    a. Designed paper.js for draft-preparation.html functionality
3. Implemented a working flask server that connected frontend to backend
    a. app.py
4. Implemented the complete paper drafting Generator
    a. generator.py
    b. evaluator.py
    c. formatter.py
    d. templates.py

LLM-Based Draft Generator

This module generates structured, domain-specific paper drafts using the Gemini Flash 2.0 large language model. It takes in user-defined section structures and topics and returns a research paper draft with proper academic formatting.

Key Features:

- **Template-Based Structure Generation**: Employs predefined templates for different paper types (standard, review, case study, experimental) to ensure proper academic structure.

- **Section-Specific Prompt Engineering**: Each section type (abstract, introduction, etc.) has specialized generation prompts with specific instructions about content, structure, and academic standards.
- **Multi-Stage Generation Process**: Implements a sequential approach starting with title generation, followed by outline creation, section-by-section content generation, and refinement based on feedback.
- **Context Provision**: Incorporates literature summaries and research gaps into generation prompts to maintain coherence and relevance.
- **Asynchronous Processing**: Utilizes async/await patterns to potentially process multiple sections in parallel, improving system efficiency.

Optimizations Made:

- Reduced generation latency through structured section-wise generation.
- Implemented robust error handling for LLM requests with proper logging.

Problems Encountered:

- **API Integration Challenges**: Error handling for API key configuration and model initialization required careful implementation.

Solution:

Added comprehensive exception handling and logging to manage potential API failures gracefully.

Comprehensive Evaluation System

This system employs sophisticated evaluation algorithms to ensure academic quality and provide improvement suggestions.

Techniques Used:

- **Multi-Criteria Evaluation**: Each section is evaluated against both common criteria (clarity, logical flow, academic tone) and section-specific criteria.
- **Numerical Scoring**: Generates scores on a 1-10 scale for each criterion, with an overall weighted score for the entire paper.
- **Targeted Feedback Generation**: Provides specific improvement suggestions for each criterion, including examples of problematic text and proposed solutions.
- **Citation Analysis**: Detects missing citations, improper formatting, and over-reliance on specific sources, and suggests additional relevant sources.

Key Capabilities:

- **Writing Style Improvement**: Transforms content to match target writing styles (academic, technical, persuasive, or concise) while preserving technical accuracy.

- **Regex-Based Response Parsing**: Extracts structured evaluation data (scores and suggestions) from model outputs using regex pattern matching.

Research Paper Generation Pipeline

The complete paper generation pipeline integrates:

- **Title Generation**: Creates multiple title suggestions based on the research topic.
- **Outline Creation**: Develop a comprehensive outline with main sections and subsections in markdown format.
- **Section Generation**: Produces content for each section with appropriate academic structure and language.
- **Draft Evaluation**: Assesses the quality of each section and the overall paper using multiple criteria.
- **Feedback-Based Refinement**: Improves sections based on evaluation feedback.

Advanced Features:

- **Paper Quality Assessment**: Classifies papers into quality tiers (Excellent, Strong, Good, Acceptable, Early draft) based on evaluation scores.
- **Domain Adaptation**: Adjusts generation based on paper type and research domain.
- **Contextual Text Generation**: Maintains logical flow and narrative consistency across sections through contextual prompting.

Underlying Technology

The system leverages:

- **Google Generative AI**: Uses Gemini-2.0-Flash for content generation and Gemini-Pro for evaluation.
- **Asynchronous Python**: Implements async/await patterns for efficient processing.
- **Structured Prompt Engineering**: Carefully crafted prompts guide the models to produce specific, high-quality content for each paper component.

Outside Testing

Participant 3:

- **Prompt Given**: *“Create a paper on the impact of CRISPR-Cas9 on genetic disease treatment.”*

- **System Output**: Successfully generated a complete research paper draft, including an abstract, background, methodology, results, and references. The writing adhered to biotechnology research conventions, citing relevant studies and including ethical

considerations.

- **Positives**: Liked how structured and citation-rich the output was.

- **Suggested Improvements**: Wanted an option to input custom references manually or adjust the verbosity of the content.

Participant 4:

- **Prompt Given**: *"Generate a research draft discussing the use of convolutional neural networks in MRI image segmentation."*

- **System Output**: Generated a detailed and structured paper with sections on methodology, model architecture, dataset description, and evaluation metrics. Included properly formatted LaTeX for figures and equations (though dummy image paths).

- **Positives**: Impressed with the technical accuracy and formatting quality.

- **Suggested Improvements**: Requested an option to switch between different citation formats (APA, IEEE, etc.) and to include tables automatically.

Participant 5:

- **Prompt Given**: *"Write a paper about building Lego cars for fun."*

- **System Output**: Returned a malformed draft with repeated sections and vague, irrelevant references. The output failed to meet academic paper standards and lacked structure. Error logs showed failed attempts to map the topic to a known academic domain.

- **Problems Encountered**: The prompt did not align with the system's trained academic domains, leading to generation failures and incomplete sections.

- **Suggested Improvements**: The participant requested a better error message or feedback system to explain why the prompt failed, instead of just receiving a broken document.

Sabbir Ahmed

80 points - Significant exploration beyond the baseline
30 points - Creativity: Designed and implemented a novel research gap identification algorithm and idea generation framework
15 points - Development complexity in terms of building robust idea generation pipeline with feedback mechanisms
15 points - Discussion of challenges encountered and partial/complete solutions implemented

10 points - Implementation of front-end components for idea generation workflow
10 points - Testing outside of the team

Contributions:

Implemented the complete Research Gap Analysis and Idea Generation Module

1. Research Gap Analyzer

- Developed an algorithm that processes ranked papers to identify unexplored research areas using three complementary approaches:
    - **Cross-Dimensional Analysis**: Implements a matrix representation where topics/methods form one dimension and domains/applications form another. The algorithm identifies cells with minimal or no coverage, representing potential research gaps.
    - **Citation Network Analysis**: Constructs a directed graph from paper citations, using graph theory to identify "terminal nodes" (papers with few citations that cite many others) and "isolated clusters" (research areas with limited connection to the mainstream).
    - **Temporal Trend Analysis**: Identifies declining research trajectories using time-series analysis on publication frequency for specific subtopics, highlighting areas where interest has waned but potential remains.

  Optimizations made: Implemented custom caching mechanism to store intermediate results of graph analysis, reducing computational overhead for iterative gap identification by ~40%.

  Problems encountered: Initial version suffered from sparse data issues, as citation networks were incomplete from API limitations.

  Solution: Implemented a confidence scoring system that weights gap identification results based on data completeness metrics. Areas with insufficient data receive appropriate uncertainty indicators.

2. Research Idea Generator

- Engineered a multi-stage idea generation pipeline that transforms identified research gaps into actionable research proposals:
    - **Idea Synthesis Engine**: Leverages the Gemini Pro model with specialized prompt frameworks to generate novel research ideas based on identified gaps.
    - **Feasibility Scorer**: Evaluates generated ideas on technical feasibility, required resources, estimated timeline, and potential impact using a weighted metric system.
    - **Implementation Pathways Generator**: For highly-scored ideas, produces detailed research methodologies and potential experimental designs.

Optimizations made: Designed a context management system that strategically truncates and prioritizes research literature to maintain critical information while staying within token limits.

Problems encountered: Initial idea generation produced overly generic or impractical suggestions.

Solution: Implemented a feedback-driven refinement loop that iteratively improves idea quality:

Purpose: To refine an initial research idea based on feedback, context, and analytical metrics.

Input Parameters:

- **idea:** A textual representation of the preliminary research idea.
- **feedback_history:** A structured dataset containing the history of feedback received on prior iterations of the idea.
- **gap_context:** Text providing context regarding identified research gaps.

Output:

- **refined_idea:** A refined textual representation of the research idea.

Procedure:

1. **Semantic Coherence Assessment:**
    - Calculate the semantic coherence score between the *idea* and the *gap_context* using the function `compute_semantic_coherence(idea, gap_context)`.
2. **Specificity Analysis:**
    - Determine the specificity metrics of the *idea* via NLP analysis using the function 'analyze_idea_specificity(idea)'
3. **Refinement Prompt Generation:**
    - Generate targeted refinement prompts utilizing the *idea*, *coherence_score*, *specificity_metrics*, and *feedback_history* as inputs to the function 'generate_refinement_prompts(...)'.
4. **Initialization of Refined Idea:**
    - Initialize the *refined_idea* variable with the value of the initial *idea*.
5. **Iterative Refinement via Language Model:**
    - For each corresponding *prompt* and *temperature* pair derived from 'zip(refinement_prompts, [0.7, 0.5, 0.3])':
        - Generate refined text using the language model client (`llm_client`) via the function `llm_client.generate_text(prompt.format(idea=refined_idea), temperature=temp)`, updating the *refined_idea* with the generated

output.

6. **Return Refined Idea:**
    - Return the final *refined_idea* as the output of the algorithm

3. Visualization System for Research Landscape

Designed interactive visualization components to help users understand the research landscape:

- **Gap Heatmap**: Color-coded visualization of research density across topic/domain combinations.
- **Opportunity Graph**: Force-directed graph highlighting promising research directions with connections to existing literature.
- **Trajectory Projection**: Time-series visualization showing predicted research trends with confidence intervals.
- Implemented front-end components:
    - Developed ideas.js for the generate-ideas.html functionality
    - Created interactive D3.js visualizations for research gap representation
    - Implemented JSON-based data exchange format between back-end analysis and front-end visualization

Problems encountered: Initial visualization attempts overwhelmed users with excessive information.

Solution: Implemented progressive disclosure design pattern, allowing users to explore the research landscape at increasing levels of detail.

4. Integration Layer for Seamless Workflow

- Developed connector modules to ensure smooth data flow between literature review and draft preparation:
    - **Context Preservation**: Maintains research gap context throughout the idea generation and draft creation process.
    - **Metadata Propagation**: Ensures relevant citations and key findings flow from literature review to draft generation.
    - **State Management**: Implements a robust state tracking system to enable user session persistence.

Problems encountered: Initial integration suffered from data loss during transfer between modules.

Solution: Implemented a comprehensive validation layer that verifies data integrity at each handoff point.

Outside Testing:

Participant 6:

- Prompt Given: "Identify research gaps in quantum computing error correction methods"
- System Output: Generated a detailed analysis of 5 major research gaps, with prioritized research directions and feasibility assessments.
- Positives: Appreciated the visualization of research trends and the specific, actionable nature of the research ideas.
- Suggested Improvements: Requested an option to filter ideas based on implementation complexity or resource requirements.

Participant 7:

- Prompt Given: "Find unexplored areas in sentiment analysis for low-resource languages"
- System Output: Identified 3 significant research gaps and generated 7 potential research ideas with methodological outlines.
- Positives: Noted the comprehensiveness of the gap analysis and the practicality of the suggested research methodologies.
- Suggested Improvements: Recommended adding collaboration potential metrics to help identify which ideas might benefit from interdisciplinary teams.