Project TLDR: A Desktop application for academic content summarization

Manu Hegde

A Capstone Project Proposal
submitted in partial fulfilment of the
requirements of the degree of

Master of Science in Computer Science & Software Engineering

University of Washington
2024-03-10

Project Committee:
Prof , Committee Chair
Name, Committee Member
Name, Committee Member

I. Introduction

This document outlines the development plan for a desktop application designed to summarize and rephrase academic papers, notes, and books using quantized Large Language Models (LLMs) optimized for performance on consumer-grade devices without the need for internet access.

II. Project Goals / Vision

A. Goal(s) of Completed Work
The primary goal is to develop an application that enables users, particularly those in academia, to easily understand complex academic materials in a manner that does not require sharing the materials or data with third parties over the internet.

Additionally, the goal also includes the ability to re-phrase and present information in an appealing and engaging manner that completely captures the users attention, rivalling that of works of artistic literature.

B. Identification of Problem and Opportunity
There is a significant need for tools that can summarize and interpret academic materials due to their often complex and dense nature. Current solutions either require internet access and share data with third-party providers or fail to deliver the latest information accurately.

While applications like ChatGPT [1], Bard, Claude enable users with these capabilities, in order to avail the functionalities, all the necessary data needs to be shared.
This sharing of data can be dangerous since some of this data might be re-used for fine tuning the model. And there is now proven research that these data can be extracted by using techniques like membership inference attack [2] and carefully crafted token sequence injection [3].

Hence opportunity lies in building a solution that avails the benefits on large language models without having the necessity to share any data with a 3rd party.

C. Stakeholders and Beneficiaries
The stakeholders are primarily the members of the academic community. While the intended primary beneficiaries are students and researchers, the usefulness of the software can also extend to professors and their teaching assistants as well.

III. Criteria

A. Level of Success

1. Minimum: The application can summarize and rephrase content from academic papers, notes, and books without internet access.

2. Expected: The application delivers context-specific information and communicates it in an engaging manner, utilizing models quantized for on-device performance that do not take more than 30% of resources of a standard M1 MacBook air

3. Aspirational: Ability to rephrase the content in very engaging and interesting manner, following the examples of books like HeadFirst Java that has engaging story telling for complex technical topics

B. Quality and Associated Measurement Metrics

Quality metrics include functional & qualitative, as well as non-functional metrics.
While the functional and qualitative metrics quantify the performance of the solution in delivering quality results, non-functional metrics help ascertain performance from resource consumption and speed of the software while achieving the functional goals.

Functional Metrics:

1. Summarization and presentation quality metrics:
   There is a need to express the summarization and presentation quality of the summary/result obtained. The quality can be expressed by using popular overlap calculation scores

   a. ROUGE: How many the words (and/or n-grams) in the *reference summary* appeared in the generated summary [4]

   b. BLEU: How many words (and/or n-grams) in the *generated summary* appeared in the reference summary [5]

   c. BertScore: Embedding based similarity score for each token in the generated sentence with each token in the reference sentence [6]

   These metrics will be used to make the following comparisons
   I. Compare against standard datasets
   II. Compare against online models

2. User satisfaction ratings:
   Software performance cannot be complete without capturing the feedback from the end user.

   a. NPS score: Net Promoter Score measures the likelihood of a user promoting the product to others within their circle

   2. Summarization quantity metrics:
      Summarization quantity metrics help in regulating the length of the output.

      a. Perplexity: To measure the exhibited verbosity to achieve certain content coverage

Non Functional Metrics
Non-functional metrics are critical to make sure that the application is still practically usable by the user. Even if the quality of results are above expectations but the software consumes

overwhelming resources or has extremely high response time, the software will be deemed un-usable
The thresholds how determining the usability however can only be obtained through experimentation and user feedback.

1. Resource usage metrics:
    - I. Main Memory(RAM) used in MB
    - II. Storage used – in MB
    - III. CPU usage – in percentage x clock rate x time
    - IV. Data throughput – in MB: Amount of data processed from source files in order to yield a given result
    - V. Resource efficiency: CPU Usage / Data Throughput

2. User interaction metrics:
    - I. Response time – in Seconds : Time taken to provide answers or summary based on a given set of text sources

C. Targets

a) Shortlisting top 2 open source models: Models to be considered based on recent developments are Gemma, Mistral-7B, Tulu2
b) Quantize the model weights
c) Run the model on the device, ideally, with not more than 30% of resource usage on a M1 Macbook Air (as a reference for benchmarking).
d) Pre-prompting to obtain desired results in terms of summary and creative re-phrasing
e) Finetuning based on results from the pre-prompting stage

IV. Positioning of Project

This project aims to fill the gap in the current academic toolkit by providing a private, efficient, and engaging way to digest academic content.

This will be a desktop and in future a mobile application that caters to the members of academia who work with large amount of literature on a regular basis

1. Existing similar solutions:
   Solutions like chatpdf.com exist (other than ChatGPT) but all notable existing systems are online and chat-gpt based

2. Key Value proposition: ChatGPT like interface tha runs on the device on collect set of documents

Contribution towards application of my academic learnings:
Further this project helps me apply various learnings from courses like High performance computing, Machine learning and parallel programming and make an impactful and useful project that could help students like myself and many more

V. Project Plan

A. Work to be Completed

1. Selection of open source model
2. Quantization of the selected model using techniques like such as LoRA [7]
3. Performance and resource optimization
4. Implementation of pre-prompting to improve communication style
5. Developing a desktop application with a user friendly graphical user interface

B. Milestones for Key Deliverables:

The project is optimistically aimed to be completed in a quarter (12 weeks) and is as follows

a) Week 1-2:
Shortlisting top 2 open source models: Models to be considered based on open source model leader board [8], which are open source and have shared detailed information on how they were trained (ex: Tulu2 [9], Mistral7B [10]).
Use functional-qualitative metrics to determine the best candidate model.

b) Week 3-6:
1) Quantize the model weights and check for base performance. When models are quantized, there is a certain amount of performance loss that is expected. The ideal amount of quantization can be figured out only after a number of trials.

2) Do pre-prompting to obtain desired results in terms of summary and creative re-phrasing

3) Optimize the model for lower resource usage (ideally, 30% or less RAM and CPU on a M1 Macbook Air)

c) Week 7-8:
Develop user interface for the software. Preferably use cross-platform frameworks that allow to use the same code to deploy software on different operating systems (ex: QT C++).

d) Week 9-10:

Start with user trials and pre-beta release. Share it with users and start taking feedback. Also measure non-functional metrics to determine usability along with user feedback. Start fine tuning the model based on user feedback.

e) Week 11-12:
Iterate based on user feedback and start sharing with more users

VI. Constraints, Risks, and Resources

A. Key Constraints

Technical Constraints

1. Resource constraints on the target device/consumer hardware

2. Persisting the quality of results after quantization

Non-Technical Constraints:

1. Time required for determining the model resource-performance trade-off
2. Cost of model finetuning (cost in terms of machine wear & tear, data acquisition & cleaning)

B. Resources Needed for Success

1. Diverse set of target hardware to test
2. Access to high quality academic materials for testing and refinement

C. Anticipated Risks

1. Technical challenges in model optimization – Optimizing the model for expected resource constraints while still retaining considerable performance includes risks of performance limitation

2. User acceptance and adaptation to new tools – Since this is a new approach to an existing problem, users may still have trust issues with LLMs based on their experience with other LLMs

VII. Scope and Requirements

Scope of functionality:

The software will contain the following modules:

1. Source content library: Contains all the source material based on which the language model will provide expected information
2. Summarizer model: The LLM and its wrapper layer that performs the summarization and Question & Answering

Their interaction is described in the following context diagram in Figure 7

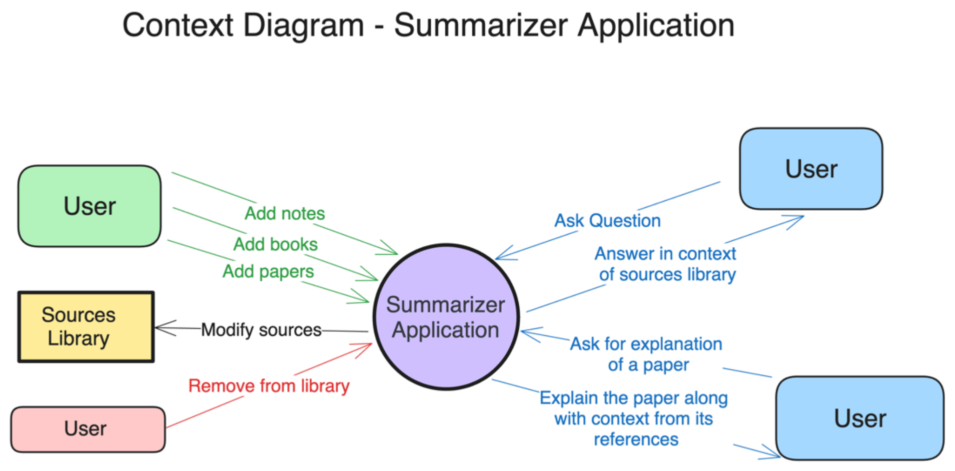
Context Diagram - Summarizer Application
User
Add notes
Add books
Add papers
Sources Library
Modify sources
Remove from library
User
Summarizer Application
User
Ask Question
Answer in context of sources library
Ask for explanation of a paper
Explain the paper along with context from its references
User

Figure 7.1 – Summarizer Software Context Diagram

This can further be broken into 1 authentication workflow and 3 usage workflows as described in the below Figure 7.2

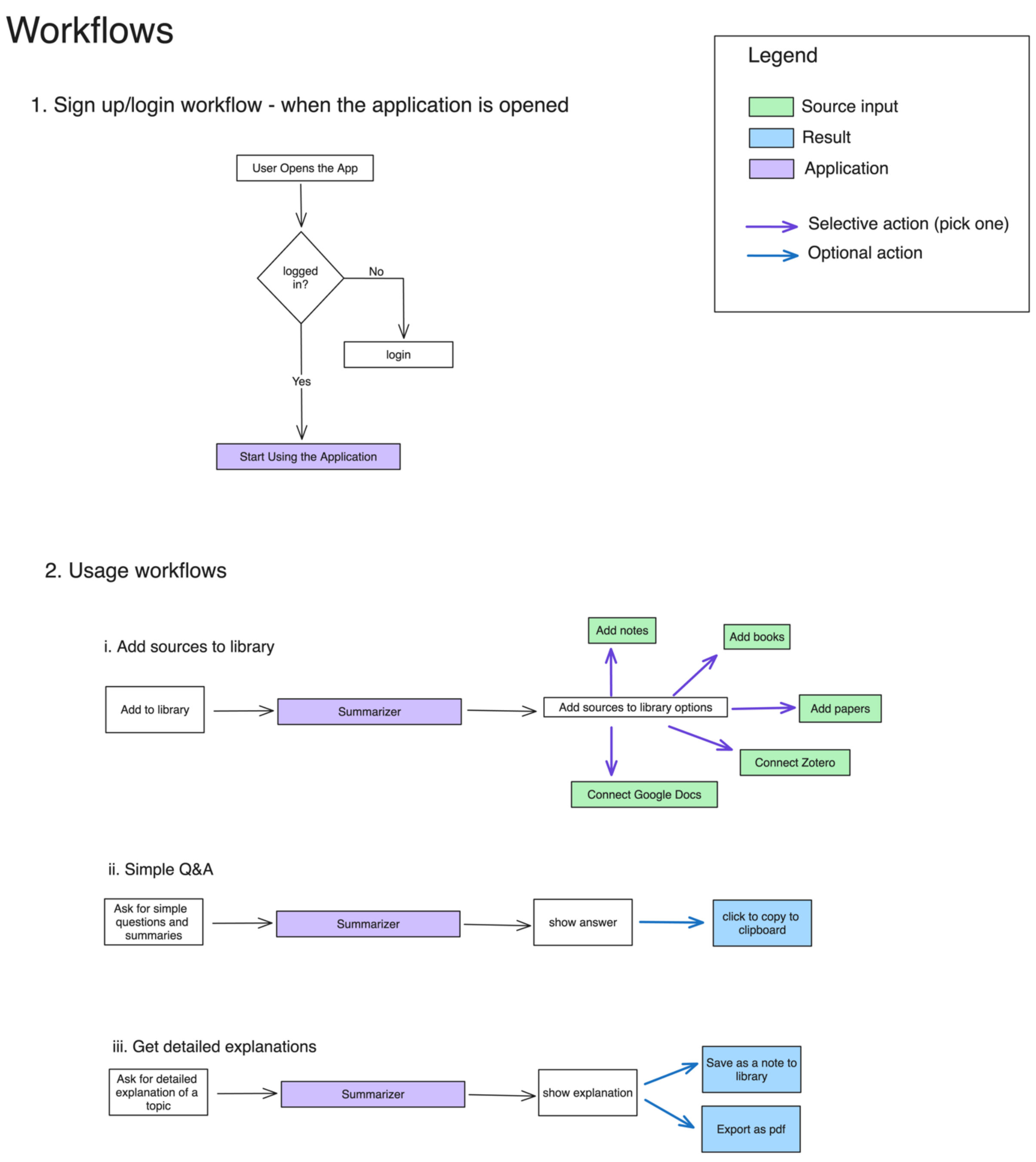
Workflows
Legend
Source input
Result
Application
Selective action (pick one)
Optional action
1. Sign up/login workflow - when the application is opened
User Opens the App
logged in?
No
login
Yes
Start Using the Application
2. Usage workflows
i. Add sources to library
Add to library
Summarizer
Add sources to library options
Add notes
Add books
Add papers
Connect Zotero
Connect Google Docs
ii. Simple Q&A
Ask for simple questions and summaries
Summarizer
show answer
click to copy to clipboard
iii. Get detailed explanations
Ask for detailed explanation of a topic
Summarizer
show explanation
Save as a note to library
Export as pdf

Figure 7.2 – Usage Workflows

1. Authentication workflow: The user needs to sign up or sign in after opening the software for the first time after installation
2. Usage workflows:
    a. Add/Remove sources to library: Various sources of academic content like books, notes, papers (as pdf) or Zotero collections or google docs can be added at any point in time
    This will enable users to add new papers, notes and content as required and ensure the language model refers to the right set of content.
    b. Simple Q&A: Ask a simple question and get the appropriate precise answer, meant for quick doubt clearing use cases

c. Ask for detailed explanation: Obtain a detailed, engaging textual response on a technical topic based on the information available in the sources library

These workflows should result in an easy to use desktop application that can act as the go-to guide for understanding one or more technical papers, answering specific technical questions by referring to papers, notes and textbooks specifically present in the user-curated source library.

References

1. Chatgpt – https://chat.openai.com/
2. Mattern et al. (2023) - Membership Inference Attacks against Language Models via Neighbourhood Comparison, https://aclanthology.org/2023.findings-acl.719.pdf
3. Nasr et al. (2023) - Scalable Extraction of Training Data from (Production) Language Models, https://arxiv.org/pdf/2311.17035.pdf
4. Papineni et al. (2002) - BLEU: a Method for Automatic Evaluation of Machine Translation, https://aclanthology.org/P02-1040.pdf
5. Lin et al. (2004) - ROUGE: A Package for Automatic Evaluation of Summaries, https://aclanthology.org/W04-1013.pdf
6. Zhang et al. (2020) - BERTScore: Evaluating Text Generation with BERT
7. LORA
8. Model leaderboard - Open LLM Leaderboard by HuggingFace
9. Ivison et al. (2023) - Camels in a Changing Climate: Enhancing LM Adaptation with TÜLU 2, https://arxiv.org/pdf/2311.10702.pdf
10. Jiang et al. (2023) - Mistral 7B, https://arxiv.org/abs/2310.06825