

Student research project II

Privacy focused personal assistant, powered by LLMs

Private smart second brain - Obsidian.md

submitted on 20 February 2024

Department Dual Study Computer Science Hochschule für Wirtschaft und Recht Berlin

Authors:	Leonard Heininger, Nicolas Kohl, Sebastian Paul
Company:	SAP
Department:	Technik
Field:	Computer Science
Course:	Computer Science
Class of:	2021
Evaluator:	Sebastian Fischer
Words:	7461 of 8000

Abstract

The paper presents a study on PAPA, a Personal Assistant, powered by Private AI, and a novel integration of Retrieval Augmented Generation (RAG) for personalized knowledge management (KM). It highlights the development and evaluation of a prototype that operates locally, ensuring data privacy and user control over personal knowledge bases. The research emphasizes the challenges of resource limitations and the current reliance on external cloud-based Large Language Models (LLMs). It also discusses the potential of RAG in enhancing personal KM through a sophisticated search mechanism rather than a fully-fledged assistant. The paper concludes with the optimism for future advancements in GPU technologies and AI systems to make powerful tools more accessible for localized use, aiming for a secure and self-reliant knowledge management ecosystem.

Contents

Table of Contents I						
A	crony	/ms	III			
1	Inti	roduction	1			
	1.1	Motivation	1			
	1.2	Problem Statement	1			
2	The	Theoretical Foundations 4				
	2.1	Knowledge Management	4			
	2.2	Large Language Models	5			
	2.3	Retrieval Augmented Generation	6			
3	Me	thodology	8			
	3.1	Design-Based Research	8			
4	Concept					
	4.1	Obsidian	10			
		4.1.1 Smart Connections	10			
	4.2	Personal LLM Responses	11			
	4.3	Architecture	12			
5	Imp	Implementation 15				
	5.1	Knowledge Base Indexing	16			
	5.2	Large Language Models	17			
	5.3	RAG Pipeline	18			
6	Results 2					
	6.1	Personalized Knowledge Management - Personal Focus	20			
	6.2	Local Operation and Model Flexibility - Privacy Focus	20			
	6.3	Browser-Based Operation and Independence	21			
7	Discussion		22			
8	Conclusion					
Bi	ibliog	graphy	27			
A	ffidav	vid	\mathbf{V}			

Acronyms

AI artificial intelligence

 $\ensuremath{\mathsf{DBR}}$ Design-Based Research

GenAl Generative AI

 $\ensuremath{\mathsf{GPT}}$ Generative Pre-trained Transformer

 ${\sf KM}$ Knowledge Management

 $\ensuremath{\mathsf{LCEL}}$ LangChain Expression Language

LLM Large Language Model

NLP natural language processing

 $\ensuremath{\mathsf{RAG}}$ Retrieval Augmented Generation

1 Introduction

1.1 Motivation

Generative AI (GenAI) is booming due to its remarkable capabilities. Open AI's ChatGPT is one of the most popular applications right now and more and more companies want to adapt artificial intelligence (AI) and integrate it into their business. [JCD+23] [Dua24] [AM23]

The demand for building AI-assistants in the business of Knowledge Management (KM) has significantly grown. This paper focuses on bringing the idea to a new level by allowing owners to fully utilize the capabilities of AI, customized and specific to their knowledge base. Instead of relying on general AI that tends to provide broad, vague responses, AI integrated in KM assists in accurately categorizing, retrieving, and analyzing information, ultimately improving decision-making and productivity within an organization. [Eli23]

However, there exist inherent challenges and concerns related to privacy and data ownership. 53% of Americans say they fear AI will damage the privacy of people who want to keep their data safe. It is important to acknowledge that a majority of AI capabilities are owned by big companies and third parties. Ensuring the privacy of an owner's data and respect for their data sovereignty becomes an urgent matter. Companies and individuals must carefully choose between using AI technologies for KM and maintaining their responsibility to protect their sensitive data. [TK23] [Kor23]

1.2 Problem Statement

In today's digital world, knowledge workers, business customers, and individuals are overwhelmed by an enormous amount of data that requires the use of effective KM strategies. This challenge not only involves the organization and retrieval of information but also ensures its confidentiality and integrity in a world increasingly reliant on digital solutions. Despite the increasing use of digital tools designed to improve KM and utilization, there is still a large gap in AI solutions that prioritize data protection while providing personalized, comprehensive, and interactive assistance. [KEC23] [BMR11]

This gap underscores a critical issue: the need for a system that can serve as a 'second brain', capable of not only aggregating and organizing information but also ensuring that data remains confidential and fully under the user's control. Current solutions are primarily based on cloud-based models, which carry risks for data protection and data sovereignty. While Large Language Model (LLM)s have emerged as a powerful tool for enhancing KM through services like Microsoft's integration of OpenAI's ChatGPT, they predominantly operate on cloud-based platforms. This reliance on external cloud components introduces significant concerns regarding data privacy, knowledge drain, and the potential leakage of intellectual property, particularly for critical decision-makers and top executives. The gap in current knowledge primarily revolves around the development of a system capable of leveraging the advanced capabilities of LLM while operating entirely on-premise, thus guaranteeing data privacy and security. [For22] [Cuo23]

This paper proposes the creation of PAPA (Personal Assistant, powered by Private AI), a privacy-focused, comprehensive, and interactive second brain. PAPA aims to operate securely on a local machine, offering a personalized and interactive second brain that enhances KM without compromising data confidentiality, keeping sensitive information under the user's control and away from cloud vulnerabilities. This solution intends to bridge the existing gap by providing seamless connectivity to one's knowledge base, offering a personalized experience that enhances data utilization without compromising on privacy. This research seeks to fill the existing knowledge gap by detailing the design, implementation, and evaluation of a system that leverages the power of LLMs in a user-centric, privacy-preserving manner, providing a blueprint for future developments in the field.

The development of PAPA as a privacy-centered, interactive second brain provides a clear direction for the overall research work, which includes the following:

- 1. Designing a privacy-friendly architecture that enables LLMs local operation and ensures that the data remains under the user's control.
- 2. Ensuring the integration of PAPA with an existing knowledge base to enable a truly personalized user experience

3. Evaluate the effectiveness of PAPA in securely improving KM and utilization with a focus on data privacy.

2 Theoretical Foundations

2.1 Knowledge Management

KM is a relatively recent field that has evolved rapidly over the past few decades. Despite its young history, KM has become central to organizational strategy, as knowledge is increasingly recognized as one of the most crucial assets a company can possess. The ability to effectively manage, disseminate, and leverage knowledge within an organization can lead to significant competitive advantages, innovation, and improved decision-making processes. This recognition underscores the imperative for systems and methodologies that enable efficient knowledge handling. [VSD+20]

In the age of 'big data' companies collect tremendous amounts of unstructured data about customer contracts, product specifications, employee handbooks, processes, research and overall internal knowledge. This leads to scattered and incoherent data through out companies' systems. IDC predicts 90% of companies data are unstructured data. The problem of not being able to retrieve important and relevant information can not only cost many nerves but also cause immense costs for businesses as IDC estimates "that an enterprise employing 1,000 knowledge workers wastes at least \$2.5 to \$3.5 million per year searching for nonexistent information, failing to find existing information, or recreating information that can't be found." [FS01] [IDC22]

The rise of GenAI, including advanced LLMs, has significantly changed the KM landscape. By training LLMs on one's own data, users can apply GenAI to their own particular field and domain. GenAI technologies have introduced new methods for capturing, processing, and generating knowledge, transforming traditional approaches to KM. These technologies have made it possible to automate complex tasks related to knowledge processing, retrieve personalized information, and improve decision-making capabilities by providing insights into huge data sets that were previously unmanageable. The integration of GenAI into KM practice represents a critical shift that highlights the need for theoretical and practical pipelines to address the complexity of modern knowledge ecosystems. [MA24]

2.2 Large Language Models

The ambition to process natural language computationally has been a cornerstone of artificial intelligence research, leading to the development of natural language processing (NLP). The field of NLP set the stage for LLMs, aiming to enable computers to understand, interpret, and generate human language in a way that is both meaningful and contextually relevant. The evolution from rule-based systems to machine learning and deep learning models marked significant milestones in this journey. A pivotal moment in the history of LLMs was the publication of the 'Attention Is All You Need' paper in 2017 by Vaswani et al., which introduced the transformer architecture. This architecture, based on self-attention mechanisms, revolutionized the way models process sequences of data, enabling more effective handling of long-range dependencies in text and significantly improving the performance of NLP tasks. [VSP+17]

The field of LLMs has witnessed an unprecedented boom in technological progress since 2022. This period has seen the introduction of models with ever-increasing size and complexity, capable of generating human-like text and performing a wide range of NLP tasks with remarkable accuracy. The advancements in model architecture, training techniques, and data processing have not only enhanced the capabilities of LLMs but also expanded their applicability across various domains, from conversational AI to content creation and beyond. [Dea23] [Mar23a]

There are mainly two types of LLMs: decoder-only transformers and encoderdecoder-transformers. Even though the encoder-decoder architecture is the one originally proposed by by Vaswani et al., its application today is predominantly focused on generative tasks that require a close adherence to the input text, such as translation and summarization tasks. For broader generation tasks decoder-only models are used as the models are trained on such vast amounts of data that an explicit encoding of the input sequence is not needed. Furthermore, most LLMs todate come in two variants depending on the stage of training. The first variant are base models which are not fine tuned much and subsequently mainly suited for text completion after being provided with an initial prompt. The second variant are the instruct or chat models. Those are fine-tuned on instructions and conversational data which makes them a better fit for NLP tasks [Hug23].

However, to create those LLMs significant computational resources for training and inference are required, including vast amounts of data, processing power, and energy. This resource intensiveness has led to the predominant deployment of LLMs on cloud-based hyperscaler servers, where the necessary infrastructure can support the demands of these models. While cloud deployment facilitates the scalability and accessibility of LLMs, it also raises concerns regarding privacy, data sovereignty, and the environmental impact of computing at such a scale. [Ope18] [Hao19] [TK23] [Kor23] [CBMA+23]

One first approach is the quantization technique which reduce computational resources for model inference by lowering the precision of model parameters, typically from floating point to lower-bit integers, such as 4 or 3 bits. This process aims to minimize accuracy loss while speeding up inference and reducing model size. Despite a potential decrease in accuracy, quantization successfully preserve as much of the model's original performance as possible, balancing efficiency with effectiveness allowing to execute 175 billion-parameter models within a single GPU. Furthermore, with the usage of CPU-focused libraries such as GGML LLMs can even run on CPUs if quantized to int-3 or int-2. [FAHA22] [Ger23]

2.3 Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) is an advanced AI technology that improves the performance of LLMs by integrating real-time data queries from external knowledge bases. This process enriches LLMs with the ability to reference current and accurate information, increasing the relevance and reliability of their results. RAG addresses a major limitation of traditional LLMs by mitigating their tendency to produce results based exclusively on statistical correlations within their training data without an understanding of the actual content. By strategically incorporating external knowledge, RAG-equipped models can provide more accurate, reliable and interpretable answers, significantly increasing their utility in a wide range of applications. [LPP+21] [Mar23b]

The primary advantages of RAG include:

1. Enhanced Accuracy and Relevance: By drawing upon external databases, RAG ensures that the information generated by LLMs is grounded in the most recent and factual content. This feature is particularly beneficial for applications requiring up-to-date knowledge or specific data points that may not be contained within the model's initial training set.

2. Increased Transparency and Trustworthiness: RAG allows users to trace back the sources of the information provided by LLMs, offering a layer of transparency that

bolsters user trust. This capability is critical for verifying the accuracy of the generated content and understanding the basis of the model's responses.

For RAG to function effectively, knowledge is typically structured in the form of embeddings. Embeddings are high-dimensional vectors that represent data in a way that preserves semantic meaning, allowing for the efficient retrieval of information relevant to a given query. This does not strictly have to involve embeddings. However, their use is prevalent due to their efficiency in mapping semantic similarities within a high-dimensional space, facilitating the rapid retrieval of pertinent information from large datasets. Embeddings serve as a bridge between the query and the most relevant pieces of knowledge, enabling the generative component of RAG to produce precise, context-aware responses. [LPP+21] [WWC+19] [ZBC+22]

RAG leverages a dual-component architecture that first retrieves information relevant to a user query using a dense retriever, from an external knowledge base, like a vector store containing embeddings. Then augments a prompt combining the retrieved information and the initial user prompt and then generates a response based on this information using a generative model. This methodology allows RAG to automatically generate customized answers by tapping into a broad knowledge base, selecting the most relevant information, and crafting responses that are not only accurate but also tailored to the specific context of the query. The flexibility and adaptability of RAG make it a useful tool for applications requiring high levels of customization in answer generation, such as personalized assistance, and specialized knowledge dissemination. [LPP+21] [DM23]

RAG facilitates the construction of customized Generative Pre-trained Transformer (GPT) models by augmenting the generative capabilities of GPT with the targeted information retrieval of RAG. This approach enables the development of GPT models that can provide more specialized, information-rich responses. By integrating the retrieval mechanism, RAG allows for the dynamic expansion of GPT's knowledge base with specific datasets, effectively customizing the model's output to reflect the shades and depth of the targeted information domain. [LPP+21]

3 Methodology

3.1 Design-Based Research

Design-Based Research (DBR) is a flexible and systematic approach ideal for addressing real-world challenges, notably in dynamic fields such as educational technology and AI. It is an iterative process focused on linking theory with practical application to foster innovative solutions. DBR's core principle is that research should engage deeply with real problems, making it distinct from traditional theoretical models. In this study project, DBR guides the creation of a privacy-oriented interactive system by providing a structured yet adaptable innovation process. It is chosen for its capacity to yield practical insights essential for the design and development of complex AI systems, ensuring they are not only functional but also user-oriented, moving beyond concept to practically addressing privacy and data ownership for KM AI-solutions. [Sch17]

The analysis and exploration phase is the first and foundational stage of the DBR process, essential for understanding the problem and setting the groundwork for the project's development. Problem Identification: The phase starts by pinpointing the specific challenges in the AI market, with a focus on privacy issues in digital KM-assistants. This involves a contextual analysis within the wider AI development sphere and defining the problem's scope. Literature Review: A comprehensive literature review is conducted to assess previous solutions to similar problems, evaluate their effectiveness, and position the current project within the existing knowledge base, spanning theoretical and practical studies on AI privacy. Research and Design Requirements: Clear, measurable research and design requirements are developed to guide the design and testing phases, based on a deep understanding of the problem. This phase is critical for steering the project, ensuring the research is relevant and the intervention is theoretically sound, aiming for a significant impact on AI and privacy. [Sch17]

In the Design and Development phase of DBR, theoretical research transitions into the creation of a practical prototype. This is particularly crucial for the project. Prototype Development: Starting with design specifications rooted in the initial research, this phase involves crafting a prototype that incorporates user-centric design and KM-AI-functionalities, directly addressing privacy concerns. The process is for iterative prototyping. Feedback from these iterations refines the prototype, ensuring it aligns with project goals and integrates the requirements from the ground up. [Sch17]

This study project reached now this phase of DBR. Once a functional prototype is ready, it enters the Implementation and Testing phase, where its effectiveness and user experience are evaluated in simulated real-world environments. Data Analysis: Usage data and user feedback are collected and analyzed to gauge the prototype's performance. Iterative Testing: The DBR cycle continues with ongoing testing and refinement based on empirical evidence, ensuring the prototype meets its intended objectives and effectively bridges theoretical concepts with practical application. [Sch17]

This phase is a crucial part of DBR, analyzing the prototype's performance data to measure success against initial objectives through both formative and summative evaluations. Formative Evaluation: Involves ongoing assessment of the prototype's design and function, using feedback to pinpoint improvements that align with the theoretical framework. Summative Evaluation: Offers a conclusive overview of the prototype's effectiveness in enhancing privacy, contributing to the broader understanding of AI privacy solutions. Reflective Practice: The team critically reviews the entire DBR process, ensuring that findings lead to methodological enhancements and inform future research. [Sch17]

Future steps include leveraging evaluation insights for further refinements and continuous improvement. The DBR cycle's completion advances AI privacy technology and contributes to the theoretical and practical knowledge, guiding future privacycentered AI development and setting a precedent for research that melds theory with practice. [Sch17]

4 Concept

4.1 Obsidian

Obsidian is a KM tool that is designed to work as a second brain. It is a markdownbased note-taking application that helps users organize their thoughts, notes, ideas, and research in a highly interconnected, non-linear fashion. Obsidian uses Markdown for note-taking, which is a lightweight markup language with plain text formatting syntax. This makes it easy to write and format notes quickly. One of the core functionalities of Obsidian is the ability to link notes together. This helps in creating a web of knowledge, where ideas and information are interlinked, making it easy to navigate and discover relationships between different pieces of information. Obsidian also provides a graphical representation of one's notes and their interconnections through its Graph View. This visual map allows users to see the structure of their knowledge base and how notes are related to each other, facilitating easier navigation and organization. Another feature is the support of community-developed plugins, which extend the functionality of the software. These plugins can provide a wide range of additional features, from enhancing the user interface to integrating thirdparty services and automating tasks within the knowledge base. At its core, Obsidian is designed to be a personal knowledge base. It allows users to build a customized and highly individualized system for managing their knowledge, which can be both broad, covering a wide range of topics, and deeply interconnected. For this study project, Obsidian is used as a knowledge base and the basis to develop an AI-Chatplugin. Obsidian also supports the idea of a privacy-focused solution, as users run Obsidian locally and are the owners of all their notes, documents, and therefore data, ideas, and research. [Obs]

4.1.1 Smart Connections

The 'Obsidian Smart Connections' plugin, powered by OpenAI's GPT-4 and embeddings, enhances Obsidian by allowing users to chat with their notes, uncovering real-time relevant connections. It aims to democratize AI access, boosting productivity and creativity within the Obsidian community. Key features include Smart View for note suggestions and Smart Chat for interactive conversations with notes, making note management and insight discovery more intuitive and efficient. The Smart Chat feature is used as an inspiration for the plugin being developed in this study project. [Pet]

4.2 Personal LLM Responses

In the development of a privacy-focused interactive second brain utilizing LLMs, personalization of responses is crucial to creating a user-centric tool. Personalizing LLM responses can follow two approaches, each with its advantages and limitations in the context of privacy and resource utilization.

The first approach involves training the LLM directly on the user's personal data, leveraging what is known as parametric memory. This method integrates the user's data into the model's parameters through extensive training processes. The primary advantage of this approach is its ability to cover multiple domains comprehensively, as the LLM incorporates all the context provided by the user's data into its responses. However, this method is resource-intensive, requiring significant time, energy, and financial costs associated with the training process. The need for vast computational resources to retrain or fine-tune the model for personalization purposes presents considerable challenges, especially when considering scalability and accessibility for individual users.

The second approach, embodied by RAG, introduces personal data to the LLM through the prompts, utilizing non-parametric memory. This technique does not integrate personal data directly into the model's parameters but rather retrieves relevant information dynamically in response to specific queries. It avoids the need for extensive resources, making it suitable for local and offline applications. Also, the RAG method is inherently more flexible and dynamic, allowing for the swift incorporation of new information by simply adding it to the prompt. However, it is primarily effective for single-domain queries since it relies on the information retrieved specifically for each query, limiting the context to what is immediately available.

Given the primary importance of data privacy and ownership in the design of a personal assistant powered by private AI, the RAG approach aligns more closely with the project's objectives. RAG's suitability for local and offline use ensures that users retain control over their data, mitigating the risks associated with cloud-based storage and processing. This method facilitates a personalized experience while respecting the principles of privacy and data sovereignty, making it an optimal choice for integrating personalization into LLM-based applications without compromising user privacy or requiring significant computational resources. On top, it gives users the flexibility to choose their LLM-Chatmodel as the RAG process is decoupled from the actual LLM.

4.3 Architecture



Figure 1: Smart Second Brain Architecture

The architecture depicted in the provided image represents a sophisticated solution approach for a smart second brain, designed to enhance knowledge management and retrieval using a combination of multiple LLMs, a vector store, and processes to prepare the passed-through content. The term 'smart second brain' comes from the idea of turning Obsidian, which functions as a second brain, into a smart second brain by infusing AI capabilities. This RAG architecture can be categorized as a naive RAG. A naive architecture follows a traditional process that includes indexing, retrieval, and generation. It is also characterized as a "Retrieve-Read" framework. [YGW24]

Knowledge Base

At the foundation of this architecture lies the knowledge base, which is powered by Obsidian, a second-brain application for knowledge management. This base houses a diverse collection of notes, documents, and images, forming a rich repository of information that users can interact with.

Indexing

The process begins with the indexing of the knowledge base content. Documents within Obsidian are segmented into coherent sections, or 'chunks', to facilitate more granular analysis and retrieval. An embedding model then processes these chunks to generate embeddings, which are vector representations that capture the semantic essence of the text. These embeddings are subsequently stored in a vector store, a specialized database designed to handle high-dimensional data efficiently.

A vector store or vector database is a type of database that stores data as highdimensional vectors, which are mathematical representations of features. Each vector has a certain number of dimensions, which can range from tens to thousands, depending on the complexity and granularity of the data. The vectors are generated by applying an embedding function to the raw data, such as text, images, PDFs, and others. The embedding function can be based on various methods, such as machine learning models, word embeddings, or feature extraction algorithms. [Vec23]

The main advantage of a vector database is that it allows for fast and accurate similarity search and retrieval of data based on their vector distance or similarity. This means that instead of using traditional methods of querying databases based on exact matches or predefined criteria, you can use a vector database to find the most similar or relevant data based on their semantic or contextual meaning. [Vec23]

Retrieval

When a user then initiates a query, the retrieval process starts. The embedding model creates an embedding from the user's query, transforming the natural language input into a vector form that can be compared with document embeddings from the vector store. A vector similarity search is then performed to identify and retrieve documents from the vector store that are most relevant to the user's query. The system is configured to retrieve the 'top k' documents, where 'k' represents a predefined number of documents that best match the query and reach the set threshold of similarity with the user's query. If there are fewer documents found that reach the threshold than 'k', fewer documents will actually be retrieved.

Reduction

Occasionally, the breadth of retrieved documents may exceed the context capacity of the LLM being used. In such instances, a reduction step is necessary, wherein the documents undergo a summary reduction process. This step is crucial for filtering out the essence of the content, ensuring that the most salient information is retained while fitting within the LLM's contextual limitations.

Augmentation

With the document information condensed appropriately, the next phase is augmentation. The summarized information is integrated into a prompt template, ready for interaction with the LLM. This template not only includes the retrieved and summarized document information but also incorporates the user's initial query and any relevant chat history to provide context for the generation process.

Generation

Finally, the generation step takes place. The fully prepared prompt, including the user query, chat history, and relevant document information, is passed to the chat model. This generation model, synthesizes all the provided data to generate a coherent and contextually informed response. It leverages its trained understanding of language and the specific knowledge provided in the prompt to produce an output that is personalized and responsive to the user's informational needs from the initial query.

5 Implementation

Given that the foundational premise of our approach centers on the utilization of Obsidian as a KM system, our selection of technologies for the implementation is constrained.

First, as Obsidan is an electron app, all plugins developed for it must be written in JavaScript. This requirement restricts our options to languages within the JavaScript ecosystem. We chose TypeScript as the implementation language to ensure a type-safe and less error-prone developing experience. In addition, this approach streamlines the maintenance process, underscoring that this is a product, rather than merely a proof of concept. Finally, the process of understanding and extending the plugin is facilitated thereby lowering the barriers for third parties to contribute.

Second, the vector search engine used in the implementation must be compatible with the electron ecosystem. In the beginning, this posed a challenge. With the aforementioned hype around Artificial Intelligence, there was an increase in the development of vector search engines, so there is no sparsity on the market. However, most of the popular vector search engines do not run locally especially not in electron environments. For example the popular Faiss library [DGD+24] does not run inside of electron environments. In the end, we chose Orama as it satisfied our mentioned requirements [Riv22].

Third, to address the requirement of enabling the use of LLMs locally on personal machines, a versatile solution was necessary. It must be compatible across various computer architectures and operate both on CPUs and GPUs. Furthermore, it is essential that the users can effortlessly install models of their choice. The most sophisticated solution in this domain currently is Ollama [Mor23]. Ollama is an open-source project designed to facilitate the management and selection of LLMs like how Docker handles containers. Ollama incorporates GGML quantized models hence allowing the models to also run on the CPU and simplifies the user experience by allowing for easy model management thereby democratizing access to LLMs [Ger23].

Finally, we use the LangChain framework. It is an open-source orchestration framework that streamlines the development of applications using LLMs by providing an overarching interface. Furthermore, it provides a debug capability for LLM outputs in LangChain applications [Cha23].

5.1 Knowledge Base Indexing

Obsidian exposes the getMarkdownFiles() function which returns all markdown files for a specific Obsidian vault, providing us with the entire knowledge base. To chunk the knowledge base we created the obsidianDocumentLoader(obsidianApp: App, files : TFile[]) function. This function takes markdown files as input and splits each file into smaller segments. We decided to segment the markdown files based on the headings contained within the document. With more specific units of meaning the embeddings will also be more precise and lead to better retrievals later on. The size of the chunks cannot be larger than the maximum token size of the embedding model. Otherwise, the model is not able to create a vector representation of this chunk. To address this case we decided to check the size of each heading and its corresponding paragraph. If the chunk is larger than the maximum token size of the embedding model, we split the chunk again and provided those chunks with an ID to indicate that they belong together.

After the chunking process, the chunks are processed by the embedDocuments(documents : Document[], indexingMode: IndexingMode = 'full') function. It aggregates the chunks into batches, with the batch size acting as a hyperparameter that determines the number of chunks to be embedded in a single execution. Opting for smaller batch sizes tends to reduce the efficiency of the process. However, it enhances the robustness as any errors encountered necessitate the re-execution of the entire batch. Each batch runs through two stages.

The first stage involves passing the chunks to a record manager. This record manager is a database designed to track the chunks that have been converted into vector representations within the vector database. This tracking capability provided by the record manager allows for the identification of chunks that have already been embedded. Consequently, embeddings are generated solely for those segments that have not yet been processed. In the event that the content of a chunk has been modified, only the corresponding entries in both the chunk embedding and the record manager are updated accordingly. If a chunk is already present in the record manager, the embedding process is skipped. This method significantly enhances the efficiency of the embedding process and, crucially for this project, conserves computing resources. The second stage involves the embedding process where the chunks are passed into the embedding model and turned into a vector representation of its content. The vectors are then stored in the Orama vector store with a hash of the chunk as a key and the vector itself as a value. Embedding models vary in the size of their embedding dimensions and in the manner they represent the meaning of the embedded text. Hence the dimensions of the Orama vector store have to be adjusted dynamically depending on which embedding model is used. Consequently, this necessitates the usage of distinct vector stores for each embedding model utilized to accommodate the unique dimensional specifications and representation methodologies of different models. This ensures that the retriever can always pull the relevant documents for each specific user query.

5.2 Large Language Models

Our goal with this application is to give the user as much freedom in choosing the model they want to employ as possible. As stated previously we chose Ollama to satisfy this requirement. As Ollama is a separate application we connect over the Ollama-provided REST API to Obsidian. The connection is only locally exposed and does not need an internet connection. The endpoints include but are not limited to generating a response from an input prompt with a provided model as well as generating embeddings for the provided text.

LangChain provides an integration for Ollama allowing us to seamlessly integrate these models into our application. An issue that arose was that LangChain uses the fetch() API as Ollama only allows cross-origin requests from 127.0.0.1 and 0.0.0.0 by default and Obsidian as an electron application uses its own URL. This resulted in constant CORS errors. To address this issue it is possible to add allowed origins to the Ollama application. However, this creates more overhead for the user as he has to change this variable manually [Mor23].

LangChain provides three distinct components associated with LLMs which are encapsulated by abstract classes: the BaseLLM class represents LLMs, the BaseChatModel class is designated for Chat Models, and the Embeddings class is allocated for Text Embedding Models. Ollama implements all three of these classes, enabling the utilization of instruct, chat, and embedding models within the LangChain framework. In the context of our current use cases, only the chat and embedding models are relevant. To implement those we use the provided class extensions OllamaEmbeddings and ChatOllama respectively. By providing the port on which the Ollama application is running and the model to be used we can create an instance of each class. The user is able to choose from all models installed with Ollama on his local machine which model he would like to use as an Embedding model or as a Chat model respectively. Employing distinct models for embedding and chatting, respectively, underpins our strategy to favor smaller, specialized expert models over larger, more generalized ones. Although this approach requires more storage space, it reduces the need for computational memory. This trade-off is particularly beneficial for the local usage requirement since storage space tends to be more affordable than computational resources.

Furthermore, a LangChain VectorStoreRetriever is created, wherein the chosen embedding model along with the corresponding Orama vector store is designated. The retriever processes an input string by embedding it using the given embedding model and then leverages Oramas vector search engine to perform a cosine similarity search on the vectorized query and stored vectors. Subsequently, it retrieves and returns the 'top k' documents that have the highest similarity to the input query.

5.3 RAG Pipeline

The LangChain Expression Language (LCEL) enables the construction of complex AI applications with features like optimized parallel execution for reduced latency and advanced mechanisms for retries and fallbacks to enhance system resilience. It also facilitates access to intermediate results, improving the transparency and debuggability of processes. Consequently, LCEL simplifies the orchestration of the aforementioned components into a cohesive and robust RAG pipeline.

The chain is configured to accept as input parameters the user's query and a history of previous messages, both provided as strings, along with the instantiated retriever and the chosen chat model. Concurrently, the user query and the message history are stored as parameters and the retriever fetches the most relevant documents related to the user's query. Additionally, we implemented the docsPostProcessor() function. Its purpose is to reassemble document chunks into their original structure when multiple chunks from the same document are retrieved. This process involves sorting related chunks according to their original headings structure and merging overly lengthy paragraphs based on their previously given index, ensuring the context provided is coherent and not merely a loose collection of retrieved information.

Depending on the length of the query, the length of the chat history as well as the value of 'k' used for the retriever, this can amount to a lot of tokens. Tokens in LLMs are the basic units of text, such as words or parts of words, depending on the tokenization method. [Tho23] If the text exceeds the maximum input token size of the embedding model we implement the docsReducePipe function which uses hierarchical tree summarization to reduce the size of the fetched context. Hierarchical tree summarization is a naive approach to reduce the size of the retrieved context with minimal information loss. For this purpose, the retrieved documents are split again into smaller chunks to fit into the maximum input token window. The model is then prompted to summarize the chunks to reduce the token size of the content. Subsequently, the summarized documents are merged and we check the amount of tokens. If the reduction was enough we are done but if the input size is still exceeded we repeat the process until the content fits into the token window.

The three parameters are passed to the next component, which utilizes a prompt template designed through prompt engineering techniques to encourage the production of relevant and coherent responses based on the provided parameters. The quality of the response generated by the chat model is directly proportional to the precision and structure of the prompt formulation, which augments the original user query with the retrieved documents and chat history, thereby enhancing the contextual relevance of the generated response. This prompt is subsequently passed to the chosen chat model which then generates its response. The prompt template further guides the chat model to disregard irrelevant augmented information and honestly state if lacks the information to address the query. The generation of the response is streamed to the user so that he can witness the creation of the answer in real-time.

6 Results

The results of this scientific study project are encapsulated in the development of a functioning prototype that embodies the integration of LLMs with a privacyfocused naive RAG-framework for personalized knowledge management on a personal knowledge base. The development of the prototype represents a significant step forward in the use of LLMs for personal and business purposes, with a focus on privacy, flexibility, and user-centered design. By enabling users to retain control over their data while harnessing the power of AI for personalized knowledge management, an approach is presented that could redefine the way individuals and businesses interact with their information in a secure, efficient, and personalized way.

6.1 Personalized Knowledge Management -Personal Focus

A cornerstone of our project is the ability to generate answers based on personal data, in other words, to create a personalized GPT. This means that all the functionality one would expect from an LLM is now available, with the added benefit of being able to use one's own data in real-time. Whether it is processing requests, generating contextualized responses, or retrieving documents, the prototype shows that it can do it all. The integration of RAG further enhances this capability and enables efficient document searches in the user's knowledge base. Therefore, the prototype is introduced as a new way for an AI-assistant in the KM-domain.

6.2 Local Operation and Model Flexibility - Privacy Focus

The prototype can operate locally, aligning with the core requirement of privacy focus. Users have the flexibility to choose the LLM they wish to integrate, allowing them to target specific business requirements, domains, and primary tasks. If users wish to only leverage the personal-focused part of the prototype, they could still leverage OpenAI's far advanced LLMs, but which would then remove the privacy aspect. This customizability goes so far that users can use their models and fine-tune them according to their specific industry, domain, or personal requirements. This feature ensures that the prototype can serve a wide range of applications, from personal assistance to domain-specific knowledge management, without compromising user privacy. The flexibility of freely choosing the LLMs allows for an even better user experience as a specific domain trained LLM could be integrated, that has domain and industry-specific general knowledge and real-time individual knowledge from the user's knowledge base.

6.3 Browser-Based Operation and Independence

A particularly interesting aspect of the prototype is its ability to run entirely within a browser, eliminating the need for server dependencies. Although the LLM component itself requires more robust computational resources beyond current browser capabilities, the developed package 'papa-ts' operates independently. This package can not only be integrated into applications such as Obsidian but can also be embedded directly into websites. This offers the possibility to create offlinecapable, serverless web applications that preserve user privacy and control over their data.

7 Discussion

In the discussion of this scientific paper, no presentable Key Performance Indicators (KPIs) for 'retrieval quality' [Rob24] could be established, as no local LLMs were found capable of generating sensible chat results before the deadline for submission but it is fair to say the used LLMs generate responses. So, one of the next challenges is to find LLMs that are suitable for the specific tasks to create actual comprehensive and contextually relevant responses and embed the knowledge in a way that makes it correctly retrievable. However, it is noted that the pipeline does function very effectively when interfaced with OpenAI's ChatGPT, which highlights the potential of the system under optimal conditions. One of the major successfully developed features which is a key benefit of using RAG is that the links to the notes work perfectly from where the content originates. Even though no KPIs can be presented, current known RAG limitations and possible solution ideas will be discussed.

RAG, in its present form, functions more as a search assistant within one's knowledge base rather than a fully-fledged personal assistant. It is tailored to find content that mirrors the user's query closely. A command like 'summarize all my notes' is theoretically feasible, but it would require the parameter 'top k' to encompass the total number of notes and the threshold for inclusion to be set to zero, which obliviously is not practical or efficient, as when the settings are set into this state all the notes will be retrieved for every query.

Another significant concern with RAG is the potential for information loss, primarily influenced by two factors: the 'k' value and the threshold setting. The 'k' value, the number of notes retrieved, is inherently limited, and while the proposed approach from this paper allows for the retrieval of all or a substantial number of notes, the necessary summarization could result in the loss of relevant details. The threshold setting poses a delicate balance. Setting it too low might include irrelevant information while setting it too high could result in missing critical data or not retrieving any data at all. The current developed state of the plugin requires manual adjustment of these parameters, which makes it laborious to work with. A future development goal is to automate the determination of optimal 'k' and optimal threshold values based on the intricacy and context of the query.

Another factor that could influence the KPI of 'retrieval quality' [Rob24] is the choice of LLMs. The accuracy of the embedding process is dependent on the specific LLM selected, and most are not explicitly trained for such retrieval tasks. Additionally, the exclusion of chat history in the retrieval process may adversely affect KPIs, as the chat context is not considered, potentially leading to less coherent or relevant responses over the course of an interaction.

Looking ahead, the future outlook for this field is optimistic. With technological advancements leading to more powerful and cost-effective GPUs, there is potential for increased research and application of GPUs in diverse fields beyond their current uses like gaming, graphic editing, or even 'Bitcoin'-mining. The move towards more sophisticated and accessible AI applications is likely, provided the necessary computational resources become more widely available. [HB22]

Ultimately, for the system to be truly autonomous and local, running the LLMs entirely on-premise, significant resources are required. Until such a time when these resources are readily available, patience remains a virtue for local users, or reliance on cloud-based LLMs provided by external vendors will persist. The second option would be to decrease the size of LLMs to be able to run them locally without the need for a (personal) server. This underscores the need for ongoing development in computational efficiency for the democratization of AI technologies to make local, private AI a viable option for all users.

RAG Ecosystem RAG Prospect					
Downstream Tasks Technology Stacks	Challenges Modality Extension Ecosystem				
Dialogue Question answering Langchain LlamaIndex	Context Length Robustness Image Customization				
Summarization Eact verification ElowiseAL AutoGen	Hybrid Role of LLMs Audio				
	Scaling-laws for RAG Video Simplification				
The RAG Paradigm	Production-ready RAG Code Specialization				
Naive RAG Advanced RAG Modular RAG					
Chunk Optimization Iterative Retrieval Retriever Fine-tuning	Evaluation Aspects				
Query Rewriting Recursive Retrieval Generator Fine-tuning	Answer Relevance Noise Robustness				
Rerank Adaptive Retrieval Dual Fine-tuning	Context Relevance Negation Rejection				
*Key Issues of RAG	Answer Faithfulness Counterfactual Robustness				
What to When to How to use Retrieval	Evaluation Framework Benchmarks Tools TruLens RAGAS				

Figure 2: Summary of RAG ecosystem [YGW24]

This visual summary in figure: 2 from a scientific paper from 2024 about RAG for LLMs fabulously depicts the RAG ecosystem. In there, one can find many aspects

that were talked about in this paper but also many more aspects that did not fit in the scope of this study project. But it is a good visualization to show how broad the field of RAG already is, and that it can be used to make really significant steps in research.

It shows that with being a naive RAG the developed prototype is really at the beginning and at a very primitive state. In the 'Techniques for Better RAG' one can find the outlined ideas for improvement like the chunk optimization to automate the adjustment for 'top k' and the threshold. Or the retriever and generator fine-tuning, meaning to find better LLMs that fit the purpose of embedding and retrieving the right content.

This visualization also serves as good guidance on where the research can be continued when looking at the challenges, whereas the context length was already tackled through the summary reduction within the proposed RAG-pipeline.

8 Conclusion

One can draw the conclusion that it is possible to build a Personal Assistant, powered by Private AI aka PAPA. The developed RAG-pipeline allows users to interact with their personal knowledge on a private computer if enough resources are available.

In conclusion, this paper has addressed the pressing need for a privacy-focused, efficient personal knowledge management system by proposing a novel integration of RAG with LLMs. Despite the challenges posed by resource limitations, which prevented the generation of definitive KPIs and the deployment of a fully local LLM capable of producing high-quality chat interactions, the paper has laid the groundwork for significant advancements in the field of personalized knowledge management.

The RAG pipeline, when used in connection with OpenAI's ChatGPT, demonstrates the potential for such systems to significantly enhance the accessibility and utility of personal knowledge bases. The demonstration with ChatGPT is no scientific breakthrough but it can be used to showcase what is possible once fitting LLMs for retrieval tasks are found. While current limitations of RAG highlight its role as a sophisticated search assistant rather than a complete personal assistant, these insights provide valuable direction for future research and development efforts.

Furthermore, the discussion has clarified the critical factors influencing the 'retrieval quality' KPI, such as the 'top k' documents and threshold parameters, and that these may be optimized through automatic determination in future iterations of the system. The paper also underscores the importance of the selection of LLMs and the inclusion of chat history in improving the relevance and coherence of responses.

As we look to the future, with the anticipated advancement in GPU technologies and a broader application of AI systems, there is a strong indication that the focus will shift towards making these powerful tools more accessible for personal and localized use. This transition will address the current dependency on external cloudbased LLMs, driving the field towards a more secure and self-reliant knowledge management ecosystem. In essence, this paper not only contributes to the academic discourse on knowledge management and AI but also presents a tangible step towards realizing a system that aligns with the ideals of data privacy, user control, and local operation. The foundation laid by this research invites further exploration and innovation, setting the stage for future breakthroughs that will empower users to harness the full potential of their knowledge assets in a secure and personalized manner.

Bibliography

- [Obs] Obsidian overview. https://obsidian.md/.
- [Ope18] (2018). Research ai and compute. https://openai.com/research/ai-a nd-compute.
- [IDC22] (2022). Der ungenutzte wert von unstrukturierten daten.
- [Vec23] (2023). What is a vector database? https://learn.microsoft.com/en-u s/semantic-kernel/memories/vector-db.
- [AM23] Allioui, H. and Mourdi, Y. (2023). Unleashing the potential of ai: Investigating cutting-edge technologies that are transforming businesses. Volume 3(2).
- [BMR11] Bisong, A. and M. Rahman, S. S. (2011). An overview of the security concerns in enterprise cloud computing. *International Journal of Network Security* amp; Its Applications, 3(1):30–45.
- [Cha23] Chase, H. (2023). Langchain.js. https://github.com/langchain-ai/la ngchainjs.
- [CBMA+23] Cody Bumgardner, V., Mullen, A., Armstrong, S., Hickey, C., and Talbert, J. (2023). Local large language models for complex structured medical tasks. arXiv e-prints, pages arXiv-2308.
- [Cuo23] Cuomo, J. (2023). Exploring the risks and alternatives of chatgpt: Paving a path to trustworthy ai. https://www.ibm.com/blog/exploring-the-risks -and-alternatives-of-chatgpt-paving-a-path-to-trustworthy-ai/.
- [DM23] David Massey, Manav Gupta, M. C. C. K. P. N. (2023). Retrieval augmented generation. https://www.ibm.com/architectures/hybrid/genai-rag.
- [Dea23] Dean, J. (2023). Google research, 2022 beyond: Language, vision and generative models. https://blog.research.google/2023/01/google-resea rch-2022-beyond-language.html.
- [DGD+24] Douze, M., Guzhva, A., Deng, C., Johnson, J., Szilvasy, G., Mazaré, P.-E., Lomeli, M., Hosseini, L., and Jégou, H. (2024). The faiss library.

- [Dua24] Duarte, F. (2024). Number of chatgpt users (feb 2024). https://explod ingtopics.com/blog/chatgpt-users.
- [Eli23] Eliyahu, S. (2023). How generative ai is revolutionizing knowledge management. https://www.forbes.com/sites/forbestechcouncil/2023/08/23/ho w-generative-ai-is-revolutionizing-knowledge-management/.
- [FS01] Feldman, S. and Sherman, C. (2001). The high cost of not finding information. https://computhink.com/wp-content/uploads/2015/10/IDC20on20Th e20High20Cost200f20Not20Finding20Information.pdf.
- [For22] Forte, T. (2022). Building a Second Brain: A Proven Method to Organize Your Digital Life and Unlock Your Creative Potential. Atria Books.
- [FAHA22] Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. (2022). Gptq: Accurate post-training quantization for generative pre-trained transformers. arXiv preprint arXiv:2210.17323.
- [Ger23] Gerganov, G. (2023). ggml. https://github.com/ggerganov/ggml.
- [Hao19] Hao, K. (2019). The computing power needed to train ai is now rising seven times faster than ever before. https://www.technologyreview.com/2019/11/1 1/132004/the-computing-power-needed-to-train-ai-is-now-rising-sev en-times-faster-than-ever-before/.
- [HB22] Hobbhahn, M. and Besiroglu, T. (2022). Trends in gpu price-performance. Accessed: 2024-02-20.
- [Hug23] HuggingFace (2023). Llm prompting guide. Accessed: 2024-02-20.
- [JCD+23] Joublin, F., Ceravola, A., Deigmoeller, J., Gienger, M., Franzius, M., and Eggert, J. (2023). A glimpse in chatgpt capabilities and its impact for ai research.
- [KEC23] Klein, L., Earl, E., and Cundick, D. (2023). Reducing information overload in your organization. https://hbr.org/2023/05/reducing-information-ove rload-in-your-organization.
- [Kor23] Korte, K. (2023). Clone me: Data ownership in the age of ai. https: //www.forbes.com/sites/forbestechcouncil/2023/10/04/clone-me-dat a-ownership-in-the-age-of-ai/.

- [LPP+21] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., tau Yih, W., Rocktäschel, T., Riedel, S., and Kiela, D. (2021). Retrieval-augmented generation for knowledge-intensive nlp tasks.
- [Mar23a] Marr, B. (2023a). A short history of chatgpt: How we got to where we are today. https://www.forbes.com/sites/bernardmarr/2023/05/19/a-sho rt-history-of-chatgpt-how-we-got-to-where-we-are-today/.
- [Mar23b] Martineau, K. (2023b). What is retrieval-augmented generation? https: //research.ibm.com/blog/retrieval-augmented-generation-RAG.
- [MA24] Maryam Alavi, Dorothy E. Leidner, R. M. (2024). A knowledge management perspective of generative artificial intelligence. *Journal of the Association* for Information Systems, 25(1):1–12.
- [Mor23] Morgan, J. (2023). Ollama. https://github.com/ollama/ollama.
- [Pet] Petro, B. Obsidian smart connections. https://github.com/brianpetro/ obsidian-smart-connections.
- [Riv22] Riva, M. (2022). Orama. https://github.com/oramasearch/orama.
- [Rob24] Roberts, A. (2024). Top evaluation metrics for rag failures. https://towa rdsdatascience.com/top-evaluation-metrics-for-rag-failures-acb27d2 a5485.
- [Sch17] Schemme, D. Novak, H. (2017). Gestaltungsorientierte Forschung Basis für soziale Innovationen: Erprobte Ansätze im Zusammenwirken von Wissenschaft und Praxis. Bundesinstitut für Berufsbildung, W. Bertelsmann Verlag, Bielefeld.
- [Tho23] Thota, A. V. (2023). Explained: Tokens and embeddings in llms. Accessed: 2024-02-20.
- [TK23] Tyson, A. and Kikuchi, E. (2023). Growing public concern about the role of artificial intelligence in daily life. https://www.pewresearch.org/short-r eads/2023/08/28/growing-public-concern-about-the-role-of-artificia l-intelligence-in-daily-life/.
- [VSP+17] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In:

Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (Eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

- [VSD+20] Volini, E., Schwartz, J., Denny, B., Mallon, D., Durme, Y. V., Hauptmann, M., and Poynton, R. Y. S. (2020). The social enterprise at work: Paradox as path forward - 2020 deloitte global human capital trends. pages 63–70.
- [WWC+19] Wang, B., Wang, A., Chen, F., Wang, Y., and Kuo, C.-C. J. (2019). Evaluating word embedding models: methods and experimental results. APSIPA Transactions on Signal and Information Processing, 8:e19.
- [YGW24] Yunfan Gao, Yun Xiong, X. G. K. J. J. P. Y. B. Y. D. J. S. Q. G. M. W. and Wang, H. (2024). Retrieval-augmented generation for large language models: A survey. Accessed: 2024-02-20.
- [ZBC+22] Zarlenga, M. E., Barbiero, P., Ciravegna, G., Marra, G., Giannini, F., Diligenti, M., Shams, Z., Precioso, F., Melacci, S., Weller, A., Lio, P., and Jamnik, M. (2022). Concept embedding models: Beyond the accuracy-explainability tradeoff.

Affidavid

We declare on our honor:

- 1. that we created the student research paper on our own,
- 2. that we have marked the adoption of literal quotations from the literature as well as the use of thoughts of other authors at the appropriate places in the paper,
- 3. that we have not submitted this student research paper on any other exams.

We are aware that an untruthful declaration will have legal consequences.

Place, Date / Leonard Heininger

Place, Date / Nicolas Kohl

Place, Date / Sebastian Paul