FINAL YEAR PROJECT RESEARCH REPORT

Project Title: IGNIS, Vulnerability Management Personal Assistant

Created By

Khairul Amirin Bin Syahrean

Student Number: C00265680

4th Year (Hons) Cybercrime and IT Security

South East Technological University Carlow Campus

Supervised by

Richard Butler

October 27, 2023

Table of Contents

Abstract

This research report delves into the potential of employing an AI-powered personal assistant to enhance vulnerability management within organizations. It introduces artificial intelligence and its cybersecurity applications. The primary objective of the project is to illustrate the effectiveness of an AI chatbot assistant, integrated with a vector database efficient for data processing and provide users with data-driven insights into network threats and vulnerabilities.

The report goes through crucial components involved in developing this assistant, including choosing large language models as the base for the AI assistant, selecting a suitable programming language from selections such as Python Java and C++, and the choice of effective code editors like PyCharm and Visual Studio Code. The paper also evaluates network and application vulnerability scanning tools.

Going through this paper, it should help illuminate how AI can simplify and optimize vulnerability management processes by providing a user-friendly conversational interface. The findings compiled directly contribute to the final implementation of the proposed AI personal assistant.

Introduction

With the advent of ChatGPT, AI has been at the forefront of technological development, streamlining operations and minimizing resources needed to complete them.

I am creating an AI-powered personal assistant that assists cybersecurity members in vulnerability management of a company's IT security. It would mainly be a user-friendly personal chatbot allowing users to obtain insightful information on threats or vulnerabilities of the company's systems at a moment's notice. Chatbot workflow would be tailored individually depending on the user's wants and records it in the database for future sessions.

The purpose of this project is to showcase how AI can be integrated into vulnerability management of IT systems for businesses and organizations. As such, this research report will discuss the fundamental principles of Artificial Intelligence before delving into its application within the realm of cybersecurity. It then goes through how AI can be implemented into Cybersecurity.

Building this application requires a thorough examination and careful consideration of multiple impactful factors. The report goes through three aspects that significantly impacts the overall performance.

These aspects are:

- Large Language Model
- Developing Environment
- Programming Language
- Scanning Tools

AI in Cybersecurity

In this modern day and age, evolving communication technologies has massively improved our lives and overall civilization. One such technology is the Internet. With it, it allows people to share knowledge and provide social interaction, further improving the quality of life with relative ease.

There are however downsides to overreliance of the internet. Threat actors use this technology to take advantage of unsuspecting users and potentially breach organisations. Information security should be prioritised to prevent such cases from happening.

Information security has been a major focus for researchers within the field of Information Technology (Wiafe et al., 2023). Various methods and tools have been developed and modernized such as advanced firewalls, data encryption techniques, malware detectors, and intrusion detection and prevention systems. These techniques aim to protect systems and users against various types of threats.

There is a split in what exactly should be focused upon when refining Information Security techniques. Some believe that it's all about focusing on human behaviour, while others think that's not enough (Wiafe et al., 2023). Considering the amount of information most organizations handle, automation can't be avoided. As such, there needs to be a balance between man, machine, organizational policies on security.

This means that our cyber defence needs to:

- Get smarter,
- Be more adaptable,
- Strongly resist a wide range of threats.

Organizations in 2023 are turning towards AI tools to keep an eye on and fight against cyber-attacks and cybercrimes more effectively. Meaning, there's a need for researchers and anyone working in the field to keep up with the latest AI methods for cyber safety. Ignis, the AI Personal Assistant being developed is an example of an AI application that can be used for production within an organization.

What is Artificial Intelligence (AI)?

The human mind is the ideal model for autonomy. Artificial Intelligence itself is a simulation of the human mind through machine processing (Schroer, 2023). These processes can lead to operations mimicking human cognitive abilities such as identifying patterns and make optimal decisions (Laskowski, 2022). Our AI Personal Assistant aims to mimic a cybersecurity specialist capable of analysing vulnerabilities and give out suited remediations.

Cognitive abilities enabled by AI:

- Learning: AI programming is all about learning how to gather data and create rules on how to convert that raw data into usable info. These rules, which is often call algorithms, are just detailed instructions that guide computers on how to do certain tasks.
- Making decisions: This part of AI programming revolves around selecting the best algorithm to achieve what you want.
- Continuously improving: AI programming comes with a component that's designed to constantly tweak the algorithms, ensuring they give the most accurate results possible.
- Being innovative: AI programming also taps into the creative abilities, using neural networks, rule-based systems, statistical methods, and others to create brand new images, text, music, and ideas.

AI can be interpreted as a mix of specific hardware and software to create and train machine learning algorithms. Most programming languages can be used to create and develop AI for various uses. Python, Java, C++, and JavaScript are the popular choices among AI developers (Arora, 2023).

AI utilization can be seen in these operations:

- Natural language processing - this is the technology that lets programs understand and interpret human language, both spoken and written. Our Personal Assistant relies on this function.
- Expert systems – Trained software capable of providing knowledge and actions of specialized experts in a field.
- Speech recognition - Programs capable understanding spoken words and transform them into readable text.

- Machine vision - Allows a computer to 'see'. It involves one or more video cameras, changing signals from analogue to digital, and processing those digital signals. The data gathered is then sent to a computer or robot controller.

Ethical use of artificial intelligence

Before delving deeper into the application's purpose and features, any potential ethical concerns must be accounted for. With growing advancements in AI tools, apart from new opportunities for businesses emerging, but also a rise in ethical dilemmas (Laskowski, 2022). AI systems basically learn and evolve from what they are fed, which can either have positive or negative connotations. A typical AI's machine learning algorithm is only as intelligent as the data they're trained on. As humans are the ones who select this data, there's always a risk of bias in machine learning. It is crucial to keep a check on this potential bias, especially when using machine learning in real-time systems.

Regulatory compliance is another area where using AI can be difficult to regulate (Laskowski, 2022). In the US for example, financial institutions such as banks need to justify their credit-related decisions. Justification coming from an AI based algorithm would be difficult to accept. This is because these AI systems work by drawing out complex correlations from countless variables, which makes it quite challenging to explain the decision-making process. One of the personal assistant's core features is to give insight to historical data and vulnerabilities. Any conclusion drawn must be scrutinized to ensure it is reliable and relevant to the task.

This leads to the next dilemma, that is who or what is responsible when an AI system's decision leads to negative outcomes, be it financial loss, health issues, or loss of life. The answer to this varies based on circumstance but is usually achieved through teamwork of variety of stakeholders - from lawyers and regulators to AI developers and ethical bodies. A prime example of this issue involves balancing the risks and benefits of automated driving systems, which might cause fewer accidents than human drivers, but still pose potential dangers.

There's also a risk of AI algorithms being misused or used for something entirely different from their original purpose. It's crucial to consider these scenarios from the get-go to reduce the risks and introduce effective safety measures.

Lastly, the ethics surrounding AI get even more complex with the development and widespread use of generative AI applications like ChatGPT and Dall-E (Laskowski, 2022). These applications are designed to generate new content based on existing ones, generating new ethical concerns related to misinformation, plagiarism, copyright violation, and content that could be harmful. Any projects involving LLMs including my application must be trained to avoid unnecessary and harmful information when generating text to the users.

Data Training

In most cases, AI systems improve by processing huge amounts of data that's been labelled, examining this data for any correlations or patterns, and then making predictions based on these patterns for what could happen in the future (Laskowski, 2022). A chatbot that's been given heaps of text examples can learn to sound like a human in chats. Similarly, an image recognition tool such as showing numerous images allows artificial minds to both identify and describe objects in pictures.

What is machine learning?

Machine learning focuses on how computer systems can learn and adapt from their experiences, even though they haven't been programmed to do so (Coursera, 2023). Unlike simple AI, where a programmer gives the machine instructions on how to react to certain situations, machine learning is about training a machine to learn by feeding it with a lot of data. In this case, the machine uses an algorithm, or a set of rules, to analyse and make conclusions from the data it's given. More data means better task execution or decision-making abilities.

Let's take Spotify as an example. The music streaming service uses machine learning to get to know what kind of music you prefer. Every time you listen to a song till the end or add it to your library, Spotify uses this info to tweak their algorithms and give you better song recommendations. Other platforms like Netflix and Amazon also make use of similar machine learning techniques to give personalized recommendations.

What is deep learning?

Think about deep learning as the next step in the evolution of machine learning. Deep learning is a type of machine learning that puts together a bunch of algorithms and computing units like neurons, into something we call an artificial neural network (Coursera, 2023). This network is heavily inspired by the structure of our own brains. Data travels through this interconnected web of algorithms in a not-so straightforward way, much like how our brains handle information.

Machine learning algorithms generally need a human touch to correct their errors, while deep learning algorithms can fix their mistakes through repeated attempts, all on their own. Machine learning can pick up things from a relatively small amount of data, but deep learning requires huge amounts of diverse data.

Fine-tuning

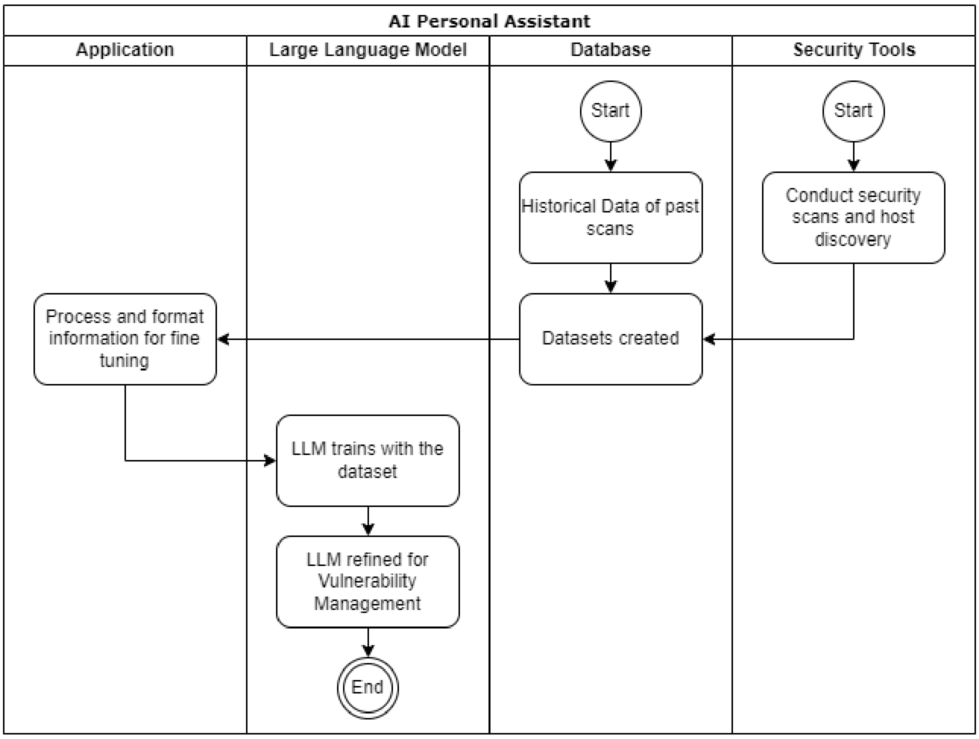
AI Personal Assistant
Application
Large Language Model
Database
Security Tools
Start
Start
Historical Data of past scans
Conduct security scans and host discovery
Process and format information for fine tuning
Datasets created
LLM trains with the dataset
LLM refined for Vulnerability Management
End

Figure 1: Fine Tuning

The diagram above is a chart demonstrating the technique for our AI Personal Assistant to obtain and learn new information regarding threat management. Our application processes the dataset and uses appropriate libraries for fine-tuning. OpenAI has functions designated for fine-tuning their LLMs.

Machine Learning and Deep Learning mostly applies to the creation of one model. Most Large Language Models have been pre-trained on a vast amount of text (OpenAI, 2023). To use the models effectively, we include instructions and sometimes several examples in a prompt. Using demonstrations to show how to perform a task is often called "few-shot learning."

Fine-tuning is similar to few-shot learning but is generally better as trains on many more examples than can fit in the prompt, achieving more accurate responses on a variety of tasks. **The key thing to note is a fine tune model needs less examples, if at all within its prompt to function properly (OpenAI, 2023).** This saves costs, token memory and enables requests for minimal data overhead.

Training with historical security data such as past scans give the LLM artificial memory and knowledge on the organization's systems. LLM output would be more consistent and avoids hallucinating false information or knowledge outside the intended database.

Fine tuning can be conducted manually if the output of the LLM is dated. It is also possible to automate fine-tuning on a consistent schedule to ensure the LLM is up to date with latest information.

How to Improve Data Quality

Using Artificial Intelligence, especially in cybersecurity, requires high level data. Ignis, the AI assistant uses data training functions provided by OpenAI. When developing for production however, more refined methods are needed to fine tune models, especially if the LLM is not under OpenAI (GPT). Complications are expected when trying to find suitable datasets and getting them to match the fast pace and variety of data available. When AI is trained for cybersecurity, focus is typically given to its quality and understanding of surrounding context. In simpler terms, to make AI more useful, we should incorporate an understanding of context, and make use of recent event analysis (Wiafe et al., 2023).

Understanding context is key in cybersecurity.

Cybersecurity research often starts with cyber data, which has lots of low-level features. This data can be read using methods like data mining and machine learning, revealing clear patterns. But, more advanced contextual information, such as space and time relationships between events in a corporate network for example, or links and dependencies, can give a more accurate understanding of whether the data indicates suspicious actions (Wiafe et al., 2023). For instance, a single connection might seem harmless to cybersecurity experts, but different methods could identify them as DoS attacks. So, the many instances in the past where cybersecurity failed to foresee threats due to lack of contextual understanding demonstrate a clear weakness. This suggests that cybersecurity solutions that understand their context and can adjust accordingly could be a good area for future AI cybersecurity research.

Learning on the go and focusing on recent happenings

Machine learning models used for security often rely on historical data to make decisions. Nowadays however user behaviour and the strategies of opponents can change quite drastically over time (Wiafe et al., 2023). For tasks like data sorting, spam filtering, vulnerability classification, and mission planning, patterns of recent behaviour and machine learning rules could be more helpful than old ones. This indicates that successfully including analysis of recent events in research to better AI-related cybersecurity solutions could be beneficial.

Large Language Models - Culmination of AI Development

Referencing (Kerner, 2023), “A large language model (LLM) is an AI algorithm that processes large data sets through deep learning to understand, generate, and create new content. Recently generated text-based content has been gaining popularity amongst enterprises and the mass consumers, realizing their flexibility and wide range of applications.

The concept of language models is like spoken languages in that they are created to provide a platform for people to communicate and share ideas through designated semantics. (Kerner, 2023).

The first language model recorded is the Eliza language model, created in MIT all the way back in 1966. An LLM is the evolution of the language model where the data scale is exponentially larger than previous iterations (Kerner, 2023).

The variables stored within a model are called parameters. Most LLMs have a base parameter size of one billion. LLMs nowadays also utilize a new mechanic called transformers models which will be explained later. A combination of both these concepts laid the foundations for modern day language models. Newer language models such as Llama and GPT have been introduced with larger parameters and better Natural Language Processing, leading to faster and more accurate responses applicable for various functions. They all however follow the same principle which is to train against a dataset to generate an algorithm for reasoning and eventually generating content.

What is a transformer model?

A transformer model is commonly referred to as a neural network, a mechanism to mimic a human brain's cognition capability. The functionality can be inferred from its name in that it is good at transforming one type of data to another (Lawton, 2023). This idea first popped up in a 2017 Google article which found a nifty way to train a network to convert English into other languages in less time and more accurately (Turner, 2017).

Interestingly, this concept turned out to be more versatile than initially expected, so these transformer models became handy in creating text, visual images, and even step-by-step directions for robots (Lawton, 2023). They're also pretty good at making sense of data that comes in different forms - that's what we call multimodal AI. It makes tasks like converting language instructions into visuals or robot instructions possible.

Today, large language model (LLM) applications wouldn't be the same without transformers. They're behind popular technologies like ChatGPT, Google Search, Dall-E and Microsoft Copilot (Lawton, 2023). Currently, they are now part of any software that handles natural language processing, purely because they outperform older methods being able to help understand complex information like chemical structures, predict protein folding, and tackle big medical data.

What truly sets transformers apart is how they use a concept in AI called attention. This nifty little trick upsizes the importance of connected words to give better context to certain words or data, like parts of an image, protein structures, or speech sounds.

Use Cases for Large Language Models

Popularity of ChatGPT is proof Large Language Models (LLMs) can be successful commercially, seeing usage in a bunch of different Natural Language Processing (NLP) tasks (Laskowski, 2022).

This can range from:

- Text generation: LLMs can create text related to any subject they have been taught.
- Language translation: LLMs trained with a variety of languages can translate from one language to another.
- Summarising content: LLMs are great at shortening lengthy articles or text.
- Content rewriting: They have the capability to rephrase sections of text.

- Categorising and classification: LLMs can sort and group content.
- Sentiment exploration: LLMs can be used to grasp the tone of a content piece or a particular response.
- Conversational AI and chatbots: LLMs enable more nature-like conversations with users, stepping up from older AI technologies.

Chatbots stand as a familiar use-case for conversational AI, where users interact in a question-answer format. The most popular LLM-based AI chatbot is ChatGPT, created by OpenAI. It currently uses the GPT-3.5 model as its base, though newer GPT-4 LLM is available for paying users.

LLM Implementation for the Project

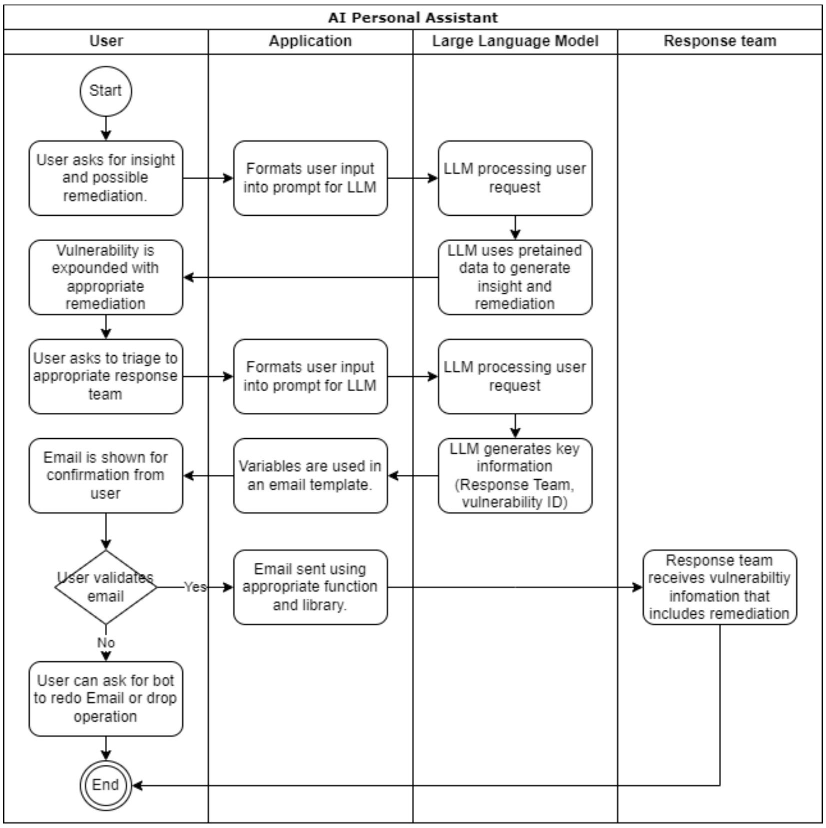
AI Personal Assistant
User
Application
Large Language Model
Response team
Start
User asks for insight and possible remediation.
Formats user input into prompt for LLM
LLM processing user request
LLM uses pretained data to generate insight and remediation
Vulnerability is expounded with appropriate remediation
User asks to triage to appropriate response team
Formats user input into prompt for LLM
LLM processing user request
LLM generates key information (Response Team, vulnerability ID)
Variables are used in an email template.
Email is shown for confirmation from user
User validates email
Yes
No
Email sent using appropriate function and library.
Response team receives vulnerabiltiy infomation that includes remediation
User can ask for bot to redo Email or drop operation
End

Figure 2: LLM Implementation

The chosen LLM explained later in the report would be tasked to do two things, act as a chatbot interface with Threat and Vulnerability Management members and conduct vulnerability management like existing security platforms such as Tenable or Qualys. The TVM member can prompt the bot to expound on detected vulnerabilities. The AI Personal Assistant should then provide insight based on previous scans on the targeted host and generate an appropriate remediation recommendation.

The application should have the capability of triaging the incident to the appropriate response team. The application prepares a notification email based on a template containing vulnerability ID, vulnerability description and possible remediations. The LLM should be able to decide on the best response team through its reasoning capability.

Challenges using Large Language Models

Big language models (LLMs for short) can be quite handy, but they do come with their set of challenges (Laskowski, 2022):

- Starting cost: LLMs, to function appropriately, need high-class graphics processing units which aren't cheap. Plus, the amount of data required for its operation is massive.
- Upkeep cost: Once an LLM is up and running, it can be expensive for organizations to keep it in check.
- Bias risks: Whenever AI is taught using unidentified data, it may keep persisting biases due it being almost impossible to remove them completely.
- Transparency problems: Users at times may find it tough to comprehend how an LLM came up with its outputs.
- Hallucination phenomenon: LLMs might sometimes come up with outputs disconnected from the data it was taught, a condition known as AI hallucination.
- Complication: Current LLMs, with their billions of elements, are incredibly detailed tools which are particularly challenging to troubleshoot.
- Glitch prompts: There has been a rise in uniquely designed prompts termed as glitch tokens. These prompts can sometimes lead an LLM to perform abnormally.

Choosing Large Language Models

ChatGPT

ChatGPT is an intelligent chatting robot which can provide a detailed response according to an instruction in a prompt. ChatGPT can perform tasks such as translation in multiple languages, story creation, code debugging. It can also do advanced reasoning, evidently shown through the ability to admit faults and refuse improper requests.

What makes ChatGPT stand out from earlier chatbots is that it can recall past parts of the conversation, leading to more seamless dialogue. It improved on itself when OpenAI released GPT-4 in March 2023, leading to a major upgrade in ChatGPT's abilities. Notably, users can now feed in both text and images simultaneously which enables it to tackle more complex multitasking jobs like summarising papers, creating image captions, and chart interpretation.

GPT-3.5 or GPT 4

GPT-4, the latest version of the open-source language model, is significantly more advanced and bigger than its earlier versions (Koubaa, 2023). The key differences between this version and the previous one, GPT-3, reveals just how the AI's potential has improved:

- Size: GPT-4 is a thousand times larger than GPT-3, having 170 trillion parameters compared to GPT-3's 175 billion. This huge jump shows the increased ability of GPT-4 to tackle complex language tasks.
- Modality: GPT-4 is not just text-oriented like GPT-3. It can also work with images.
- Context Window Length: This feature refers to the conversation history that GPT-4 uses to generate a response. The more it can remember, the better its responses are. The window length in GPT-4 can be up to 32768 tokens, 4 to 16 times larger than GPT-3. As a result, GPT-4 can digest longer conversations and provide a more relevant response.
- Output: GPT-4 can output up to 24000 words or 48 pages compared to GPT-3's 3000 words or 6 pages. This broadens the horizon for longer and more accurate text generation.

From these comparisons we can conclude that GPT-4 features a significantly larger architectural model size than its predecessors, including GPT-3 and its variants (Koubaa, 2023). The increased model size helps improve its natural language processing (NLP) capabilities, bringing more natural and responses relevant to context. Higher tier models bring more significant overhead such as processing and computing resource consumption. This leads to longer times for generating text.

Llama 2

Llama 2 released in the middle of 2023 is the second iteration of LLaMA (currently referred to as Llama 1), an LLM developed by Meta (Touvron et al., 2023). It is the culmination of the cooperation between Meta and Microsoft. Microsoft has also partnered with OpenAI making it a frontrunner corporation for AI development.

Llama 2 is currently available in 7B, 13B, and 70B parameter size. Unlike GPTs, Llama 2 is an open-source LLM, available for free for any business or research purposes.

Criteria	Llama 2	GPT-3.5	GPT-4
Parameters	70 billion	154-175 billion	1-1.76 trillion
Max context	4096	4096 8001 16384	8192 32768
Modalities	Text only	Text only	Text and image
Accuracy **(few-shot, k=5 MMLU)**	68.9%	70%	86.4%
Complexity	Lower	Higher	Higher
Speed	Faster	Slower	Slower
Efficiency	More efficient	Less efficient	Less efficient

Table 1: LLM comparison (Touvron et al., 2023)

Speed & efficiency

Llama 2's smaller model size allows it to be the fastest for processing requests. So, if high speed and efficiency are critical, Llama 2 may be the best option.

Token limit

As seen in the table above, Llama 2 has the same token limit as the base variant of GPT-3.5-turbo, while the base GPT-4 doubles them. Both GPT models also are available in variants with even bigger token limits. Input and output length available is directly affected by the token limit imposed on each model. This implies that Llama 2 would be the least recommended choice.

Accuracy & task complexity

Benchmark (shots)	**GPT-3.5**	**GPT-4**	**PaLM**	**PaLM-2-L**	**LLAMA 2**
MMLU (5-shot)	70.0	86.4	69.3	78.3	68.9
TriviaQA (1-shot)	-	-	81.4	86.1	85.0
Natural Questions (1-shot)	-	-	29.3	37.5	33.0
GSM8K (8-shot)	57.1	92.0	56.5	80.7	56.8
HumanEval (0-shot)	48.1	67.0	26.2		29.9
BIG-Bench Hard (3-shot)	-	-	52.3	65.7	51.2

Table 2: LLM Benchmark (Touvron et al., 2023)

For this section MMLU will be used. MMLU (Massive Multitask Language Understanding) is a benchmark to measure knowledge during pretraining by evaluating models without it being pre-trained or is minimally trained. 5-shot or 5 sample trained MMLU benchmark shows that Llama 2 performs almost similar to GPT-3.5 (Touvron et al., 2023). GPT-4 performs better than Llama 2 and GPT-3.5 on the 5-shot MMLU benchmark. This makes it the top option for most complex tasks commanding the need for accurate and creative responses.

Chatbot Applications.

GPT-3.5

Ideal for medium to large enterprises. Being the forerunner of commercial LLMs, GPT-3.5 can handle human conversations and produce meaningful responses similar to or better than Llama 2. Token usage is properly priced proportionate to its performance levels (Luzniak, 2023).

GPT-4

High reasoning capacity leads to it being the best for mission critical applications. GPT-4 can resolve more complicated problems with concise yet elaborate responses while mimicking human-like communication (Luzniak, 2023). As an example, customers can be routed to a GPT-4 based chatbot for help desk services before needing to escalate to a human agent.

Llama 2

Llama 2 is open source making it ideal for smaller companies and businesses, allowing resources to be allocated to other maintenance costs. Smaller-scale inputs allows for Llama 2 to perform well (Luzniak, 2023).

Code Editors

Type of coding environment can greatly affect development speed and reliability of the final product.

Figure 3: PyCharm

Figure 4: Visual Studio Code

PyCharm and VS Code are great choices for Python focused developers. PyCharm created by JetBrains has high functioning features and interface, while Microsoft Visual Studio Code is highly flexible with customization options.

Criteria Listing for choosing the best developing environment (Luzniak, 2023):

Criteria	PyCharm	Visual Studio Code
User Interface	• Fairly polished and easy to use. • Has features useful for Python developers. • Has refactoring, code navigation, and code completion. • Package manager to manage and install new	• Simple and intuitive user interface • Has code completion and debugging ability. • Expansive library of plugins for version control systems, newer terminals, and debugging.

	packages for developing apps.	
Performance	• Powerful IDE requiring high memory and processing power, leading to IDE running slow when loading large files or conducting resource intensive tasks. • Has features related to enhancing coding speed such as smart indexing and caching, improving coding navigation and analysis.	• IDE is lightweight, therefore uses less system resources than PyCharm. It should be faster when loading projects and running intensive tasks. • Extra features like multi-process architecture allowing tasks and functions to be executed simultaneously for faster coding.
Ease of Use	• Provides a wide range of features and tools, making it suitable for experienced developers. • Steep initial learning curve. • Interface can be overwhelming	• Intuitive code editor designed for new developers, offering a range of features while ensuring its interface is easy to discern. • Hassle-free debugger.
Community Support	• Large user base, with vast resources allowing users to use IDE quickly. • Wide range of plugins and extensions developed and maintained by its community, allowing	• Large user community with resources available. • Its dedicated support team also offers documentation, forums, and tutorials for any issues users face. • Extensions and plugins created various users

	users to customize the IDE according to their individual requirements.	within VS Code community allows for wide customization.

Table 3: IDE Comparison

PyCharm is an IDE that is well-suited for Python development. As the AI Personal Assistant would likely be developed using Python code, PyCharm provides powerful and intuitive tools to effectively write, test, and debug the assistant's Python code.

Programming Languages

Python is the go-to language model for AI software building (Arora, 2023) and will be used to create the AI Personal Assistant. However, it is worth noting other languages are fully capable of building AI apps as well. They come with their own benefits and slight disadvantages when compared to each other.

Figure 5: Python

What is it: Python is a highly utilized, general-purpose programming language characterized by its relative ease of learning. Its inherent simplicity facilitates AI development which is why it has been universally adopted as the predominate language within the AI community.

Viability of Python for the project can be attributed to several notable factors (Arora, 2023):

- Ease of usage: Python features a simple syntax which includes straightforward words, symbols, and expressions necessary to create programs, thus enabling more time to be dedicated towards data analysis and model tuning.
- Platform Independent: All operating systems, be it iOS, Windows, or Linux, provide support for Python. Moreover, very little modification is required for a Python program to function across various platforms.
- Visualization tools: Python has a wide selection of data visualization libraries which are essential to AI development.
- Open-source capability: Python's underlying code can be adjusted, updated, or enhanced by virtually anyone. Consequently, numerous Python community members have developed frameworks and libraries to enable its application in virtually any machine learning or data science task, making it excellent for AI applications.

Disadvantages: Python does face restrictions in the execution of intricate mathematical and statistical functions. Moreover, its performance speed is comparatively slower than that of other languages such as C++ and Java.

Figure 6: C++

C++ plays an integral role in the development of versatile applications and is widely recognized as one of the leading programming languages. It serves as the backbone for various operating systems, including Windows, iOS, and Linux, and is utilized by a diverse range of applications (Ng, 2023). Additionally, prominent platforms such as YouTube, Spotify, Netflix, and various banking systems rely on C++ for their operations. Professionals working in cutting-edge fields like self-driving cars and robotics also consider it to be an indispensable tool.

Advantages: Unlike languages relying on an interpreter program, which can introduce additional processing overhead, C++ delivers swift and efficient programs (Ng, 2023).

C++ is often perceived as a challenging language to master due to its intricacies (Arora, 2023). Programming in C++ requires a substantial investment of time, with debugging often prolonging the development process. Furthermore, making modifications to a written program after adjusting hyperparameters can be time-consuming. It is also widely acknowledged that learning C++ comes with a steep learning curve.

Figure 7: Java

Java is a highly efficient and versatile programming language that is well-suited for constructing scalable AI infrastructure. Like Python, Java is popular and open source, with numerous frameworks and toolkits tailored specifically for machine learning and data science applications. However, Java possesses a longer history and therefore enjoys widespread adoption by various organizations (Ng, 2023). In contrast to Python, Java is characterized by its greater technical complexity, necessitating a more challenging learning process. Nevertheless, this added complexity allows for the more efficient execution of programs.

Advantages: Java excels at performing tasks comparable to Python, and in certain cases, it may even surpass Python's capabilities. For example, it offers frameworks for data science, classification, and deep learning, among others. Java's stringent rules make it less prone to code misuse or violations, making it adept at constructing large-scale back-end infrastructure for deploying machine learning models.

Disadvantages: Its learning curve is steeper when compared to Python, although not as steep as that of C++ (Ng, 2023). Additionally, writing programs in Java requires more time and debugging efforts due to its intricacy. These factors hinder the rapid prototyping of machine learning models. Moreover, Java's community is less prominent than Python's in terms of developing AI-focused tools, resulting in a narrower range of machine learning and data science applications where Java proves beneficial.

Figure 8: JavaScript

JavaScript is a high-level scripting language widely utilized to enhance user interaction on website. As the language of the web, it is universally supported by modern web browsers.

JavaScript excels in the field of web development and offers various functionalities such as manipulating the Document Object Model (DOM), creating intricate user interfaces, managing user events, and facilitating requests to online servers (Simplilearn, 2023).

Advantages: For AI application development, JavaScript provides access to libraries like TensorFlow.js (Arora, 2023), enabling the integration of machine learning capabilities directly into web browsers. Consequently, JavaScript becomes a valuable tool for harnessing AI and enhancing web user experiences, as well as for prototyping AI-driven web applications.

Disadvantages: Despite its numerous advantages, JavaScript's performance can be relatively slower compared to other programming languages as it relies on interpretation rather than compilation. Additionally, JavaScript exhibits certain 'habits', and beginners may find certain language features difficult to comprehend.

Vector Databases

Retrieval Augmented Generation

Our application will be utilizing Retrieval Augmented Generation or RAG for short. This technology allows for additional context to be provided to the large language model from a custom-made knowledge base (Merritt, 2023).

Initially to test if the large language model is functioning properly, we manually upload the scan results into the application. LLMs have limited context size making it difficult to process whole documents and carry out its prompted task. LangChain can split the text into chunks of text and pass them into embedding models.

Embedding models

Computers by themselves do not understand simple words or numbers. We therefore convert them into embeddings, a higher dimensional interpretation of an object, and the dimensions may be abstract and not meaningful to humans since they were produced by machine learning (Cloudflare, 2023).

Embedding models take the text chunks and vectorize it. In other words, it assigns a numerical representation to the text chunk which is vital for creating vector databases. Vector databases when configured properly allow users to search for the text chunk that most suits their query.

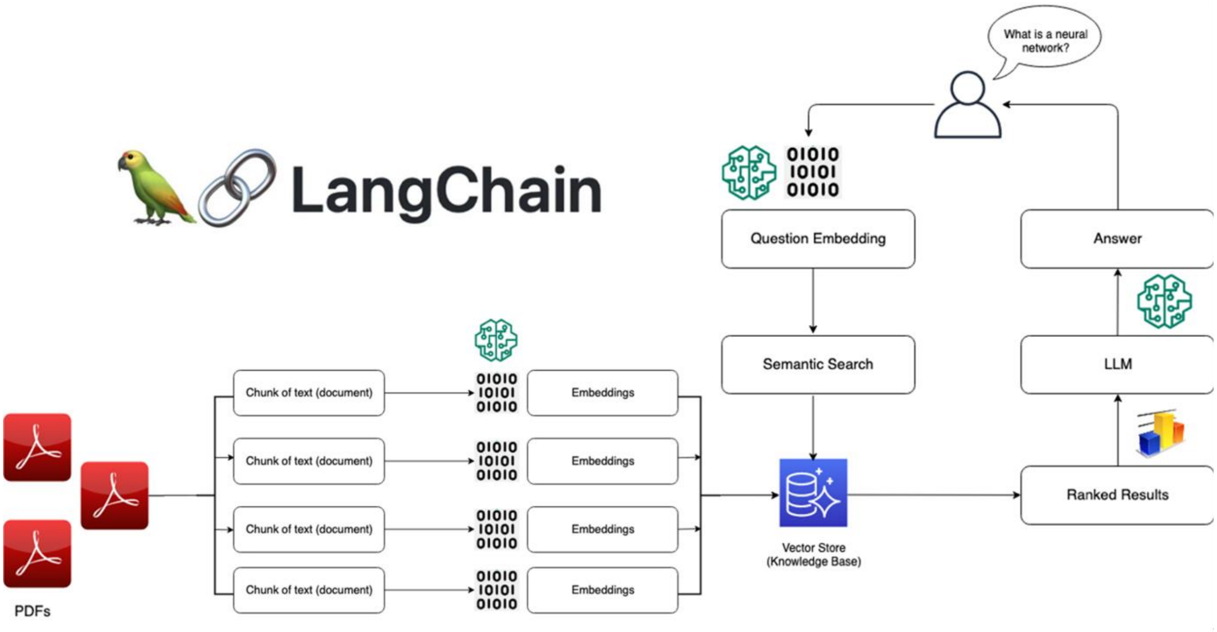

Figure 9: Visualization of semantic search (Amazon, 2023)

For the project the embedding algorithm used is OpenAI embedding. An example query would be if the user is asking for any latest critical vulnerabilities. The application would embed the question like the algorithm used on the scan results.

The application performs a semantic search within the vector database to find which chunks of text are relevant to answer to that question based on similarity matrices (Cloudflare, 2023). The chunks that mostly matches the query embedding will be selected. These chunks are then sent to the LLM as context and continue conversing with the user. The user should be able to query the application multiple times, continuously querying the database until the user terminates the session.

Scanning Tools

While the focus of the project is to showcase how LLMs and AI in general can be utilized for managing and remediating vulnerabilities, data collection methods must be taken into consideration to guarantee quality and relevance to the model's purpose. Such data will be obtained through vulnerability scanners. Many organizations and developers are now using these scanners as testing tools that find problems in code and identify possible misconfigurations. To find these issues, scanners usually use a database filled with known problems such as National Vulnerability Database (NVD) created by the National Institute of Standards and Technology (NIST) (Khounborine, 2023).

The NVD is a library for Common Vulnerabilities and Exposures (CVE), each entry comes with an identification number, usually related to when the issue was discovered, a description of the vulnerability, and a list of software that the issue affects. To better understand how urgent the problem is, each CVE also has a Common Vulnerability Scoring System (CVSS) score that defines the severity of the issue.

A good vulnerability scanner does more than simply cross-checking the findings against the NVD. Scanners should collect data from different sources like software documentation sites to generate the vulnerability report.

These scanners have a key role in preventing cybersecurity threats and attacks. They prove handy when dealing with third-party providers or when using open-source code. When it comes to open-source code, there's no way to ensure all developers have used the best security practices. Consequently, it's tough to find weaknesses in the code or to confirm that there are none without running a vulnerability scanner (Khounborine, 2023).

But vulnerability scanners are not just made to scan code. They can do much more. They have more features that make them great tools for preventing cybersecurity issues.

Aspects of Security Scanning

Due to rapid advancements in security testing, there's all types of scanners designed to poke and prod at systems, each with different perks depending on the end goal, be it white box or black box testing.

White box testing views at the structure and code of a product, similar to a developer (Practitest, n.d.). Testers get a full-access pass to all the underlying structure of codes and find all the hidden issues. It is time-consuming dealing with all the information. The tester must also have solid knowledge of the technology used for the program.

Black box testing is the method not referring to the internal structure and design of the product, like a threat actor not knowing the internal functions (Practitest, n.d.). It is used to evaluate the external behaviour of the product, including any inputs and outputs produced.

Moving to authenticated versus unauthenticated testing. They are rather parallel to white and black box testing. To summarize, authenticated testing means logging in as a user login observe around to see what vulnerabilities that exist. Unauthenticated testing is coming at it from an outsider's perspective, not having credentials and to test how much impact can be done externally (Khounborine, 2023).

Types of Scanners

Application Scanners

Unlikely to be applicable to our bot but should be mentioned, application scanners go through an application to identify vulnerabilities and misconfigurations. These scanners employ different analysis techniques being:

- Software Composition Analysis (SCA).
- Static Application Security Testing (SAST)
- Dynamic Application Security Testing (DAST).

Each has certain advantages over the other. Therefore, it is best to implement a combination of said techniques where possible (Khounborine, 2023).

Network-based Scanners

As the primary data source for our chatbot model, network-based scanners identify possible attack vectors and vulnerabilities on targeted networks. Optimal usage can result these scanners being able to identify possible rogue or unknown devices (Khounborine, 2023). Given the nature of vulnerability management in most businesses and organizations, our AI Personal Assistant would be utilizing network scanners the most.

Network based scanners would utilize methods such as brute force scans, credentialed scans, and exploit scan (Khounborine, 2023).

Database Scanners

While not the primary focus for the project, database scanners can be considered for future developments. Database scanners can identify vulnerabilities in a database. One of the more well know vulnerabilities is SQL injection. The scanners should be able to identify this vulnerability. Another important vulnerability to monitor for is password decryption, where a threat actor uses a tool to generate combinations to gain access.

Database scanners can also check for suboptimal and exploitable configurations and inform whether the database and its information are secure. Scans are often intensive, going through internal and external spaces for vulnerabilities (Khounborine, 2023).

Security Scanning Software Comparisons

We will be looking at several scanning tools for the AI Personal Assistant to utilize through Python centric APIs.

- Nessus Scanner: This tool is one of the most popular vulnerability scanners in the world. It is designed to automatically identify vulnerabilities that hackers could use to break into your system, such as outdated software or weak passwords.
- Network Mapper (Nmap): Nmap is a powerful and flexible open-source tool used by network administrators to scan and map networks, monitor hosts, and perform a variety of other tasks. It can discover available hosts, their services, and the network topology effectively.
- Burp Suite: Burp Suite is a comprehensive web application security testing tool. It has a range of features to help penetration testers or bug bounty hunters manipulate web traffic and discover potential vulnerabilities in a web application.
- Zap (Zed Attack Proxy): This is an open-source web application security scanner, used for finding vulnerabilities in web applications. It is designed to be used by both those new to application security as well as professional penetration testers.
- OpenVAS: OpenVAS stands for Open Vulnerability Assessment System. It is a software framework of several services and tools offering vulnerability scanning and vulnerability management. All the components of OpenVAS are free software, and its global feed of Network Vulnerability Tests (NVTs) is updated regularly.

Security Tool Comparisons

Criteria	**Security Tool**				
	Nessus Scanner	Network Mapper (Nmap)	Burp Suite	Zap	OpenVas
Type	Commercial	Open Source	Commercial	Open Source	Open Source
Scanning Target	Network	Network	Network	Web Application	Network
SAST	-	-	N	N	-
SCA	-	-	N	N	-
DAST	-	-	Y	Y	-

Authenticated Testing	Y	?	?	?	Y
Unauthenticated Testing	Y	?	?	?	Y
Supports many OS/multi-platform	Y	Y	?	Y	N
Generate vulnerability reports	Y	Y	Y	Y	Y

Table 4: Security Tool Comparison Table (Khounborine, 2023)

Nessus is the appropriate scanner for the AI personal assistant working on vulnerability management in an organization (Khounborine, 2023).

- Comprehensive Scanning: Unlike Burp Suite, Zap and Nmap which specialize in web application and network mapping respectively, Nessus offers broad vulnerability scanning across various networks and systems. This makes it more comprehensive and versatile.
- Detailed Reports: Nessus generates detailed vulnerability reports. An AI personal assistant could use this detailed information to accurately parse, prioritize, and manage vulnerabilities. This level of detail is particularly important in large and complex organizations where there may be a wide array of potential vulnerabilities to track and address.
- OS and Platform Support: Nessus has wide support for various operating systems and platforms. Considering the possible variability in an organization's system setup, this could be hugely beneficial. Nmap and OpenVAS fall short in this aspect.
- Authenticated Scanning: Unlike OpenVAS, Nessus supports both authenticated and unauthenticated scanning, providing a more thorough and accurate scanning result.
- Commercially Licensed: While it may be a drawback for some organizations working on a tight budget, the fact that Nessus is commercially licensed also suggests it likely gets more consistent updates, support and development when compared to open-source alternatives. Nessus Essentials is also free to use for testing purposes such as for our project.

Conclusion

The rapid development of AI has garnered attention of not just cybersecurity specialists but also threat actors wanting to leverage it for their own malicious purposes. I have made it a responsibility for myself to not only create an AI centric application to improve IT Security, but also use it to spread the benefits about AI to others in Vulnerability Management and the cybersecurity field in general.

After delving into the various factors, we can decide on the path to build the AI Personal Assistant. GPT 3.5 is the main language for developing the AI assistant. PyCharm will be the main IDE and code editor due to its various plugins, quality of life features and support specialized for AI application development. Python will be the main programming language, being robust and has capabilities for web application development makes it the best candidate.

Moving forward rigorous testing using datasets generated from Nessus as the main scanner will be done. Other scanning tools are also being considered and would be implemented if possible.

References

1. Amazon (2023) What Is LangChain? [online] Amazon. Available at: https://aws.amazon.com/what-is/langchain/
2. Arora, S. (2023). *Best Programming Language for AI Development in 2023*. [online] Hackr.io. Available at: https://hackr.io/blog/best-language-for-ai.
3. Cloudflare (2023). *What is a vector database?* [online] Cloudflare. Available at: https://www.cloudflare.com/en-gb/learning/ai/what-is-vector-database/
4. Coursera (2022). AI vs. Deep Learning vs. Machine Learning: Beginner's Guide. [online] Coursera. Available at: https://www.coursera.org/articles/ai-vs-deep-learning-vs-machine-learning-beginners-guide.
5. Lawton, G. (2023) What is a Transformer Model? [online] TechTarget. Available at: https://www.techtarget.com/searchenterpriseai/definition/transformer-model.
6. Kaur, R., Gabrijelčič, D. and Klobučar, T. (2023). Artificial intelligence for cybersecurity: Literature review and future research directions. Information Fusion, p.101804.
7. Kerner, S.M. (2023). *What is a large language model (LLM)?* [online] TechTarget. Available at: https://www.techtarget.com/ whatis/definition/large-language-model-LLM.
8. Khounborine, C. (2023) A Survey and Comparative Study on Vulnerability Scanning Tools.
9. Koubaa, A. (2023). GPT-4 vs. GPT-3.5: A concise showdown.
10. Laskowski, N. (2022). What is artificial intelligence (AI)? [online] TechTarget. Available at: https://www.techtarget.com/searchenterpriseai/definition/AI-Artificial-Intelligence.
11. Luzniak, K. (2023). 6 main differences between Llama 2, GPT-3.5 & GPT-4. [online] Neoteric. Available at: https://neoteric.eu/blog/6-main-differences-between-llama2-gpt35-and-gpt4/.
12. Merritt, R. (2023). *What Is Retrieval-Augmented Generation, aka RAG?* [online] Nvidia. Available at: https://blogs.nvidia.com/blog/what-is-retrieval-augmented-generation/
13. Ng, A. (2023) Five Important AI Programming Languages. [online] Available at: https://www.deeplearning.ai/blog/five-important-ai-programming-languages/.

14. OpenAI (2023). *Fine-Tuning*. [online] OpenAI Available at: https://platform.openai.com/docs/guides/fine-tuning.
15. Practitest (n.d.). *Black Box Vs White Box Testing*. [online] Practitest. Available at: https://www.practitest.com/resource-center/article/black-box-vs-white-box-testing/.
16. Salmon, P. (2023). PyCharm vs. VS Code: Which Python IDE Wins? [online] History-Computer. Available at: https://history-computer.com/pycharm-vs-vs-code-which-python-ide-wins/.
17. Schroer, A. (2023). What is artificial intelligence? How does AI work? [online] Builtin. Available at: https://builtin.com/artificial-intelligence.
18. Simplilearn (2023). *10 Practical Applications of JavaScript and Tips for JavaScript Professional*[online] Simplilearn.com. Available at: https://www.simplilearn.com/applications-of-javascript-article.
19. Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S. and Bikel, D. (2023) Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288.
20. Turner, A. (2017). *Google I/O 2017: Google Translate talks the talk thanks to AI neural networks*. [online] The Sydney Morning Herald. Available at: https://www.smh.com.au/technology/google-io-2017-google-translate-talks-the-talk-thanks-to-ai-neural-networks-20170517-gw6fnx.html [Accessed 8 Dec. 2023].
21. Wiafe, I., Koranteng, F.N., Obeng, E.N., Assyne, N., Wiafe, A. and Gulliver, S.R. 2020. Artificial intelligence for cybersecurity: a systematic mapping of literature. IEEE Access, 8, pp.146598-146612.

Appendix

Figures:

Tables: