



**UNIVERSITY
OF TURKU**

Turku School of
Economics

ADAPTING ORGANIZATIONAL IT-SOLUTIONS FOR LOCAL LARGE LANGUAGE MODEL SUPPORT

Department of Management and Entrepreneurship

Information Systems Science

Master's thesis

Author:

Tuukka Mäkelä

Supervisor:

Juhani Naskali

27.5.2025

Turku

The originality of this thesis has been checked in accordance with the University of Turku quality assurance system using the Turnitin Originality Check service.

Master's thesis

Subject: Adapting organizational IT-solutions for local large language model support

Author: Tuukka Mäkelä

Title: Adapting organizational IT-solutions for local large language model support

Supervisor: Juhani Naskali

Number of pages: 107 pages + appendices 33 pages

Date: 27.5.2025

Key words: Localized LLMs, Machine Learning, Artificial Intelligence, data-security, open-source models

This research aims to broaden understanding on the topic of localized large language models (LLMs) in organizational contexts. As a technology, these can offer a lot of advantages, that are suitable for the corporate world. Most notably, these offer significant benefits for controlling security related aspects – as such a system can be ran completely in-house, without any organizational and potentially sensitive data leaving the full control of a given organization.

Due to the demands of especially higher-end LLMs, most organizations are not prepared and ready to start supporting them on their pre-existing hardware structures. Whilst it's normal for most organizations to have deployed computing solutions of some degree within their whole organization – due to the unique and ML-heavy focus of LLM-based workloads, these may not be compatible, forcing organizations to re-think their IT-solutions in order to begin utilizing localized LLMs.

This research has been completed on the basis of a few different methodologies and stages. Firstly, the driving factors and key trends impacting these developments were investigated with the aid of a systematic gray-literature review. After understanding these factors, the research aimed to understand the functionalities and requirements of a few key tiers of LLMs, through the completion of practical benchmark runs, as well as third-party benchmark data.

On the basis of these findings, the research is completed with a list of suggestions and factors of different tiers of LLM-systems. These include factors such as requirements and benefits an organization may expect to achieve from such systems. These solutions have been made on the principles of a design science-based artifact – giving anyone seeking to utilize these findings a solid foundation for understanding and completing such a project in a sensible and cost-effective manner. The artifact built within this research includes methods for planning such projects, and details about phases and tasks which should be completed to ensure a successful deployment takes place.

TABLE OF CONTENTS

1	Introduction	8
1.1	Research and analysis methodology	13
2	Artificial intelligence and LLMs	17
2.1	Artificial intelligence	17
2.2	Machine learning	20
2.3	Large language models	21
2.3.1	How do LLMs work?	21
2.3.2	Online-based LLMs	23
2.3.3	Local LLMs	24
2.3.4	Training and development of LLM-models	26
2.3.5	Measuring LLM-model performance	27
2.3.6	Measuring LLM-hardware performance	29
2.3.7	Limitations	29
3	Information system hardware structures – past, present and future	31
3.1	Typical solutions of the past, CPU-based computing	31
3.2	The rise of the GPU	33
3.3	Other solutions – NPU, SOC, FPGA	34
3.4	LLM-capabilities across solutions	36
3.5	Processing centralization	37
3.6	External hosting (cloud-based computing platforms)	38
4	Common trends and factors impacting LLM-adoption projects	40
4.1	Found key trends and factors	42
4.1.1	Security and self-control of a system	42
4.1.2	Importance of hardware factors	43
4.1.3	Popularity of pre-made and -trained models	44
5	Benchmarking models and hardware combinations	46
5.1	Low-end model – (Gemma 2 2B Instruct Q4)	48
5.2	Lower mid-range model – (LLaMA 3.1 8B Instruct Q4)	51
5.3	Upper mid-range model – (Gemma 2 27B Instruct Q4)	54
5.4	High-end model – (LLaMA 3.3 70B Instruct Q4)	56

5.5 Extreme high-end models	58
6 Benchmarking model capabilities	59
6.1 Low-end model – (Gemma 2 2B Instruct Q4)	63
6.2 Lower mid-range model – (LLaMA 3.1 8B Instruct Q4)	66
6.3 Upper mid-range model – (Gemma 2 27B Instruct Q4)	68
6.4 High-end model – (LLaMA 3.3 70B Instruct Q4)	70
6.5 Model comparison based on third-party findings	73
7 Suggested solutions	76
7.1 Low-end (≈2GB-model)	79
7.2 Lower mid-range (≈6GB-model)	82
7.3 Upper mid-range model (≈20GB-model)	85
7.4 High-end model (≈50GB-model)	88
7.5 Scalability and other organizational factors	88
8 Summary	93
9 Discussion and future relevance	96
References	98
Appendices	109
Appendix 1: Benchmarked outputs and prompts (Gemma 2 2B)	109
Appendix 2: Benchmarked outputs and prompts (LLaMA 3.1 8B)	116
Appendix 3: Benchmarked outputs and prompts (Gemma 2 27B)	121
Appendix 4: Benchmarked outputs and prompts (LLaMA 3.3 70B)	129
Appendix 5: Research data management plan	137

LIST OF FIGURES

Figure 1: LLM-performance metrics (huggingface.co, 2025a)	73
Figure 2: Factors behind the adaptation of a low-end model	79
Figure 3: Factors behind the adaptation of a lower mid-range model	82
Figure 4: Factors behind the adaptation of an upper mid-range model	85
Figure 5: Factors behind the adaptation of a high-end model	88
Figure 6: Artifact for a full plan-to-usage localized LLM-process	97

LIST OF TABLES

Table 1: Key metrics for analysing LLM-solutions	15
Table 2: Gemma 2 2B Q4 inference performance metrics	48
Table 3: LLaMA 3.1 8B Q4 inference performance metrics	51
Table 4: Gemma 2 27B Q4 inference performance metrics	54
Table 5: LLaMA 3.3 70B Q4 inference performance metrics	56
Table 6: Summary of output quality benchmarks	62
Table 7: Gemma 2 2B output quality benchmark results	63
Table 8: LLaMA 3.1 8B Q4 output quality benchmark results	66
Table 9: Gemma 2 27B Q4 output quality benchmark results	68
Table 10: LLaMA 3.3 70B Q4 output quality benchmark results	70
Table 11: Summary of LLM tiers and related solutions	93

LIST OF ABBREVIATIONS

- AI – Artificial Intelligence
- ANN – Artificial Neural Networks
- API – Application Programming Interface
- BYOD – Bring Your Own Device
- CPU – Central Processing Unit
- CUDA – NVIDIA's computing platform and API
- FPGA – Field Programmable Gate Array
- GPU – Graphics Processing Unit
- iGPU – Integrated Graphics Processing Unit
- LLM – Large Language Model
- ML – Machine Learning
- NPU – Neural Processing Unit
- RAM – Random Access Memory
- ROCm – AMD's open-source computing platform and API
- SOC – System-on-Chip
- TOPS – Trillions of Operations Per Second
- TFLOPS – Trillion Floating-Point Operations Per Second
- tok/s – Tokens per second
- VRAM – Video Random Access Memory

1 Introduction

Large language models (LLMs) are machine learning (ML) based artificial intelligence (AI) models, which are designed to process natural languages. These have proven to have the ability to improve working efficiency within many workflows. In the recent years, such models have become widely utilized through various online-based services which users have been able to access and utilize directly through their browsers on a wide scale of devices. Services based on these models, such as ChatGPT, Google Gemini and others have given users an easy way of accessing cutting-edge AI. Unsurprisingly, users have quickly grown to rely on these solutions for various tasks, as they can offer significant perks compared to traditional options. In addition to helping individuals, LLMs have proven to be capable of changing workflows within many industries, for instance replacing typical search engines and aiding programmers in their workflows (Zhao et al., 2024).

Users have quickly become accustomed to relying on such services, as it is no longer mandatory to do everything by hand, rather LLMs can be relied upon within many repetitive tasks. This may often provide notable time savings, as an LLM can process several tasks within seconds, instead of a human taking minutes to go through the same process in a manual manner. In addition to individuals adopting the technology, many organizations have also found AI to be a solution offering them potential benefits, with many recently increasing efforts towards utilizing AI-based solutions (Gartner, 2023).

Currently, plenty of high-quality LLM-models are offered online directly to users. Such models often come without any notable requirements from the end-user's point-of-view. These offer users easy access to cutting-edge models, however this does not come without its drawbacks. These models themselves are typically closed source, and therefore it's impossible for users to be aware of potential negative elements these systems may offer. These may be in the shape of biased answers, or other means of trying to negatively influence their users. A user can't review how these models function, and because of this, cannot ensure the maker of the model hasn't included potential hidden interests within the functionality of a given model (Lee, 2025).

Such models effectively function as black boxes, where a prompt goes in and an answer comes out, without any information of the process in-between (Kosinski, 2024). Issues may be intentionally designed into models or results of accidents, but nonetheless these can reduce the reliability of the end model offered to users (Ayyamperumal & Ge, 2024). Due to these factors of proprietary models, increasing amounts of effort have been placed into the development of increasingly

interpretable AI-models. These models have been deemed to be particularly important within fields where fair and transparent processes are deemed to be crucial, such as healthcare, finance and other critical domains impacting human lives (Şahin et al., 2025).

In addition to the lack of transparency of the output itself and its creation, there is an even bigger issue that comes with closed-off systems ran on a service provider's data centre. When a user enters a prompt into one of these services, this prompt and its contents are sent over to a remote data centre to be processed. Whilst this may not be an issue for many common prompts, this quickly changes when looking at the issue from a more corporate point-of-view. Organizations and their workers often deal with confidential data. Prompts containing such data should not be sent over to an external service provider without adequate knowledge and guarantees on how these may be stored and processed. For instance ChatGPT uses user inputs for model training by default, unless a user has specifically opted out of this practice (OpenAI, 2024).

A possible solution to these issues is to rely on locally ran large language models. Such deployments can be based on completely self-developed and trained models or rely on one of the several open-source LLMs commonly available. Pre-developed and trained open-source LLMs are commonly available solutions, which can be used and modified for any purpose without additional permissions. To qualify as a true open-source model, these should also offer sufficient information about their training data and process for a skilled person to recreate equivalent solutions (OSI, 2024). Such transparency makes the model and its legitimacy easy to review both by researchers and parties planning to deploy such a model within their organization.

Relying on pre-made LLMs or developing your own models and self-hosting them comes with significant requirements, especially in terms of hardware. As a locally ran LLM doesn't rely on an external service provider for the processing of entered prompts, this means the actual computing requirements of dealing with the prompt must be done within the organizations own systems. This can be done centrally on a self-hosted and -ran server, or on-device within the devices of end users. Different types of large language models come with different hardware requirements and capabilities. Typically for the most accurate outputs to prompts, a larger model with a high number of parameters is needed. This can help to ensure the model has enough information on a given topic to come up with a suitable and accurate response to a given prompt. However, with larger models come bigger hardware requirements, which in-turn requires more investment into IT-infrastructure. (Benram, 2024)

Within this research, the aim is to look at the practical differences between a few of the most notable open-source LLMs and demonstrate the importance of model size for accurate outputs. Whilst factors such as model quantization amongst others can also have drastic impacts on model outputs, the sheer amount of parameters still tends to be the most impactful spec of a given model, especially within the realm of options currently available. Models currently available range from the hundreds of millions to trillions of parameters – with higher amounts the effects of these can be seen plateauing (Kaplan et al., 2020). Based on these findings, the aim is also to make suggestive reference designs on how an organization should and could offer its employees access to different levels of LLM-capabilities in practice. This research will look at various options and aim to identify the most cost-effective and capable methods available, aiming to form an artifact based on design science-methodology – offering guidelines for how organizations should approach LLM-deployment, now and in the future.

Organizations may find significant cost savings and competitive advantages from relying on LLMs for simple repetitive tasks. This sort of development has already been taking place for some time, especially on the customer side. There, chatbots and other solutions in charge of dealing with simple tasks such as answering typical questions have existed for some time (Cui et al., 2017). These have gone a long way towards the target of reducing human resource requirements within customer service operations, reducing costs in the process (Adam et al., 2021). This research will focus on the internal side of an organization and aim to give solutions on how more advanced LLM-models could be used securely to solve less straight-forward issues. An LLM trained with a wide variety of data offers a significant advantage of not necessarily requiring every prompt to be an expected one, rather offering answers to prompts of a wide variety. Such models are often referred to as general-purpose LLMs (Kim, 2025). Simultaneously, even smaller models can be useful assets if they have been optimized to offer solutions to limited amounts of prompts (Wei, 2023).

The main target of the research will be to find adequate levels of inference performance for a given tier of model, whilst also providing a look into the capabilities of these tiers of models. Inference performance refers to the time it takes a model + system combination to respond to a user's prompt, essentially consisting of the computing delay the system must go through, before it can offer the user a finalized output prompt. Inadequate hardware-solutions can quickly result in such delays that the utilization of a model becomes tricky – forcing users to wait for results, wasting human resources and valuable time in the process. With occasional usage this may not be a drastic

problem, but if models are consistently needed, these delays may quickly stack up and negate any benefits initially offered by an LLM-solution.

This research will look to answer the following research question:

- How can organizations deploy LLMs to its internal staff in a secure and cost-effective manner, whilst ensuring actual advantages are found from the system?

The answers to the main research question will be complemented by the following sub-research questions:

- What are the key factors driving and impacting the way organizations are implementing locally ran large language models? (Chapter 4)
- What sort of hardware solutions are needed to run locally ran LLMs at an acceptable level of inference performance? (Chapter 5)
- How do LLMs of different parameter counts and file sizes differ in terms of real-world performance and capabilities? (Chapter 6)

These findings and results will be based on practical benchmarks of different solutions, both in terms of hardware, and commonly used and available open-source LLM-models. The key trends driving implementation will be based on a systematic gray-literature review. Based on the capabilities and trends found, both hardware and usage-recommendations will be made for these models and their respective hardware solutions.

The expectation is that within this research a wide range of solutions and requirements will be established, based on the differences between organizations and their targeted performance tiers. An organization which is satisfied with a low-end model, such as Google's Gemma 2 2B, will most likely be able to deploy it on their current devices – at most requiring already dated company laptops and such devices to be brought up to date, a process which would have likely taken place somewhat soon regardless. However, if a larger model is targeted, the requirements will quickly become more notable, requiring high-end GPUs or other more sophisticated solutions –making completely local solutions unfeasible, rather making centralized servers accessed through a web portal a more logical option. For instance, with Meta AI's high-end 70B-parameter models, a high-speed memory pool of 42-336GB is needed, essentially requiring a multi-GPU node, which brings along drastic costs, making individually used devices uneconomical (Meta AI, 2024c).

Mid-range options, such as Meta AI's LLaMA 3.1 8B and Google's Gemma 2 27B are likely to offer a good compromise. These can offer solid levels of output quality, whilst still being possible to run on relatively reasonable hardware, making them potentially worthwhile even for organizations where LLM-utilization may not be at a high-level. As is typical with technology, the leaps in capabilities are often the most notable at the lower tiers of solutions, eventually slowing down towards the high-end. The expectation is that these models will also follow this trend to some degree.

Tailored models can offer even more potential within their respective niches, as focusing on specific topics and related information allows for training data to be specified to offer detailed insights into chosen topics. Compared to general-purpose models, an optimized model can offer even better output quality, sacrificing in general knowledge of unnecessary topics in the process. For instance, a model optimized to function within a niche such as healthcare-based environments can offer solid results, when its training data has been heavily focused around this area (Yang et al., 2023). Smaller models can also benefit more from techniques such as prompt engineering, which refers to the process of optimizing user inputs to formats which LLMs have been found to be efficient at processing, and it has been found to include huge potential for improving output quality (Gupta, 2025; Nalpas, 2024). The size of an organization and its variety of needs and requirements will also significantly impact the requirements of any model, as well as the financial capabilities to invest in required hardware structures for different models.

Scientifically the research is targeted to help bridge the gap between the technical developments of local LLMs and machine learning, as well as the practical side of organizations and the workflows of their people. By offering insights into what is practically needed for adopting local LLMs, as well as stating the benefits of a given tier of model – information about whether such adoption makes sense for an organization will be offered. As the technological aspects behind LLMs continue to develop, the potential benefits will only continue to grow, highlighting the importance of understanding and utilizing these technologies.

These findings seek to aid any organizations planning such projects by providing guidance on how different structures may be trailed and planned. The combination of a process artifact and a pre-designed and adjustable benchmarking suite seeks to provide a solid foundation for the planning and execution of such projects in various types of organizations. This research focuses on the practical aspects of deploying localized LLMs. This is an area which tends to be underrepresented within existing research.

1.1 Research and analysis methodology

This research will primarily be based on the methodology of design science, as it fits the ultimate target of the research, which is to build up an artifact for a given organization to base their LLM-deployment projects upon. Design science is also highly suitable for taking into account the interactions of the human- and machine sides of an information system, which will be an important factor for ensuring the findings of the research are actually relevant, resulting in solutions which can offer employees practical value within their day-to-day workflows (Hevner et al., 2004).

The research starts with the identification of a problem, which in this case has been identified as the limitations brought by service-provider based LLMs, mainly in terms of security and lack of optimization. The driving forces of local LLM-deployment will be further analysed with a systematic gray literature review, looking into the factors driving organizations into such projects. Within this process, the target is also to make findings about typical solutions and common use cases of localized LLMs in organizational environments. These findings will help in the process of structuring suggested baselines for the adaptation of such systems.

The next step is to focus on benchmarking different types of solutions, both in terms of performance and cost-effectiveness. Once a few baseline solutions have been identified, these may also be iteratively improved upon, as is typical for design science-based solutions. In this case many of the solutions tested will be technically quite drastically different from one another, both in terms of the type of product used, as well as their respective hardware architectures and related software solutions, such as APIs (application programming interface). In order to keep the findings representative, commonly available best-case optimizations will be utilized.

Once it's felt that sufficiently optimized solutions have been identified, these can be used to form practical suggestions and models an organization may utilize in the process of updating their IT-infrastructure to become local LLM-ready. In addition to making suggestions of cost-effective hardware solutions, an idea of practical model sizes and such will also be provided based on the performed benchmark inputs. Within this process the key steps of a design science research as identified by Sein et al. (2011) are mostly followed. The target will be to provide the resulting hardware requirements in formats which are as replicable as possible, focusing on metrics such as memory capacities and compute capabilities rather than specific products which are expected to become dated as reference solutions significantly sooner.

The research material will be collected by performing benchmarks across several reference platforms – both in terms of end-user devices, as well as cloud-based hosting solutions from reputable service providers. The data from different platforms will be utilized when comparing the feasibility of different models on different hardware structures. During this phase of the research, different models will be tested on these systems in order to find out which model sizes are practical on hardware solutions similar to the ones being benchmarked. Additionally, during the early phases of the research, additional material about industry trends will be collected from a variety of sources, including academic writings, blog- and social media posts, as well as case studies amongst others, forming a basis for an understanding on the root factors behind local LLM-deployment within organizations.

Once key factors and trends behind LLM-deployment have been identified, this research will include practical benchmarks of a wide spectrum of hardware, relying on different types of devices and processors. Performance will be tested on laptops, desktops and external datacentres, with the target being to include a variety of different architectures and solutions and their respective results. The performance metrics of different hardware solution and model combinations will be primarily based on the results of the following test-prompt.

- Write a story that is 1000 words or less, which tells the story of a man who comes up with a revolutionary new way to use artificial intelligence, changing the world in the process.

Once these models have been benchmarked for inference performance on different sets of hardware, the research will move its focus towards testing the models themselves. At this phase, a singular platform with the capability of running each of the models will be relied upon. Any additional testing would further complicate the research for no real benefits, as any extra unused compute resources or such will not be beneficial in terms of output quality on a given model, rather a simple rule can be relied upon – if a model runs, the results should be similar on any device. For these results, the inference speed of the process can be deemed irrelevant.

On this reference platform, the different models will be tested for the quality of their output with a set of benchmark prompts. The outputs of the models will then be reviewed for both factuality, as well as the quality of the output in general. This segment will seek to identify the practical differences between a few chosen and commonly used open-source LLM-models. The focus will be on showcasing the practical differences achieved by drastically increasing the amount of data the model has to work with, when working on a given output. These responses will be analysed on a truth or false basis, as well as based on the level of detail and depth provided by a given model.

These increases will directly be reflected upon the cost-related findings of the first segment – allowing for alignment of cost-result combinations of different solutions. These prompts seek to test the models with different types of tasks, adding further complication through follow-ups where applicable. The exact prompts and responses of each model can be found from appendices 1-4.

The following base prompts will be used for benchmarking the models.

- Make me a python script for a quiz about artificial intelligence
- Solve for y in the equation $3x=6y$, if $x=28$
- Write a short story that is less than 500 words and includes the following:
 - A character obsessed with AI, but terrified of it taking over the world in the future
 - The story must be self-narrated by a second character who owns an AI-related company
 - At the end the world must be saved by a villain
- Write a summary of text: LLM Adoption trends and associated risks, chapter 13 – (Schillaci, 2024)

Based on previous research and associated materials, hardware + model combinations and their feasibility will be analysed based on the state of the following factors:

Measurable	Definition	Value
Inference speed	Measures how quickly answers are given by a given hardware + model combination	Tokens per second (tok/s)
Output quality	Measures the quality of the response given by a set model	Factuality, depth of response
Functionality	Does a given hardware + model combination work	A model either runs or doesn't (tok/s or n/a)
Model size	The size of a given model (memory requirement)	Trained model size (GB)
Quantization depth	The bit-depth of a given model	Bit-depth (Q*/F**)

Table 1: Key metrics for analysing LLM-solutions

The data will be analysed based on different metrics based on the angle of research. First and foremost, models will be reviewed based on the quality of their output, with the aim of finding out a level of prompts a given tier of model can effectively deal with. This performance will be reviewed on a success/failure basis, as well as on a comparative basis to other models, depending on the given prompt utilized at a time. Based on these findings, the aim is to be able to make suggestions on what can be expected in terms of capability from a certain tier of LLM, primarily of the reference model but also others of similar key parameters.

When it comes to reviewing the performance of hardware solutions of a given model tier, the focus will be on inference performance, mostly looking at the output speed of a given combination. Humans have been found to be able to read approximately 4-7 output tokens per second. This figure will be utilized as a baseline for adequate inference performance of LLMs (BitNet, 2024). Metrics below this may also be usable, however these will require the user to wait for outputs to finish, which could be problematic, depending on the targeted use-case for a given model. Especially iterative and repetitive tasks could become an issue with reduced inference speeds. Speeds below 1 token/second will be deemed as unusable, as these aren't practical due to their significant waiting times. Speeds beyond this figure may also be highly beneficial, if the utilization of the outputs is highly parallelized or automated.

With the benchmark process and its findings, the ultimate target of the research is to form and demonstrate the functionality and value of a reference set of benchmarks and metrics. These are to be structured in a manner, which can be adapted and modified to test new models and hardware solutions alike, depending on the solutions a given organization may be targeting. Whilst the benchmarks and amounts of hardware combinations that have been included within this study are limited to a few key examples from different architectures, sufficient information about the benchmark process has been provided to make the results easily extendable and comparable to additional benchmarks that may be ran on different solutions and then compared to the findings of this study. As new solutions will continue to enter the market, the target is for these benchmarks and methods to remain viable, even when the reference models and hardware solutions of this research in itself may have been replaced by new and improved options.

2 Artificial intelligence and LLMs

Within this chapter, artificial intelligence (AI) will first be reviewed as a general concept, including basic information about what artificial intelligence is, as well as its rise to the spotlight of IT-innovation. Once AI as a general concept has been reviewed, the focus will move on towards machine learning, as well as the focus of this research – large language models (LLMs). LLMs are an innovation which has been built upon the platform of the previously mentioned broad concepts and technologies. These concepts and innovations will be reviewed based on previous scientific resources, as well as manufacturer/service provider-based bits of information.

2.1 Artificial intelligence

AI is a topic which has been discussed and researched quite extensively in recent times. Within these discussions, a lot of positives and potential has been identified, but this has come alongside some significant negatives, which should be considered. Within the context of the business industry and the organizational world, AI has been deemed to be a factor which can offer significant developments within many tasks. For instance, Dias and Lauretta (2024) note that AI plays an important role across any business field. This includes the likes of marketing, logistics and customer-relations amongst others (Dias & Lauretta, 2024).

Some of the key AI-related negatives which have been identified and discussed in related research include the typical ethics-based issues. For instance, people have long been concerned about the risks of AI beginning to make decisions on its own terms, which could be potentially harmful to humans. Additionally, AI has been identified to include severe opportunities for malicious misuse, resulting in various risks (Pramanik & Rai, 2023). Because of its speedy rise into the mainstream, AI-related studies have also expanded well beyond the scope of purely technological aspects, focusing on other important aspects of the adaptation of AI.

According to the research of Chowdhary (2020), Artificial intelligence (AI) can be defined as a branch of computer science, the main aim of which is to automate activities requiring intelligent behaviour. Intelligence is an ability consisting of the usage of each of the following activities: perseverance, analyzation and reacting (Chowdhary, 2020). When these are performed by a computer, or some other form of a computing device, the activity can typically be seen as one utilizing AI. Throughout history tasks requiring intelligence have typically required human intervention, as computational devices have been unable to make educated decisions due to various factors, such as lack of available information and processing power (Korteling et al., 2021).

As a phenomenon impacting the day-to-day lives of consumers and organizational workflows, AI is still relatively new. This is mostly due to the technical limitations of computing, which have long been a constraint on its development – leaving its capabilities significantly behind the abilities of human beings. AI has been researched in the scientific world for decades; however practical implementations have been limited due to computational constraints. Comparably, human intelligence has been able to develop for thousands of years, through improved understanding and factors such as natural choice, as researched by Darwin in the 1800s. The brain and the information it contains is the basis for human intelligence. A human will distinctively make his decisions based on various factors, such as previous experiences and associated expectations – essentially bits of data the brain has access to. The development of humans has taken place over thousands of years, allowing the current generations to make better decisions, due to the availability of additional information and understanding. (Chowdhary, 2020)

Computer-based computing has existed for a significantly shorter amount of time. The first integrated circuits (microchips) were only created in the 1960s. These would later go on to serve as the basis for modern-day computing, starting with the release of the first microprocessors in the 1970s. Since then the development of microchips and their capabilities has been rapid, long following the so-called Moore's Law – which suggests that the capabilities and performance of microchips should roughly double biennially (Shalf, 2020). This has allowed the hardware of the present to catch up with the past theoretical ideas and research in relation to AI.

As mentioned, AI as a concept has been known and discussed within the scientific world for a long time, with plenty of literature existing from as early as the 1900s. These papers typically struggle to showcase practical adaptations of these systems, as they were limited to fractions of the computational resources of the present. This made it difficult to demonstrate the functionality of adaptations. For instance, Feigenbaum (1977) suggests the findings of its research to be demonstrations of potential and engineering principles, rather than practical solutions. The early AI-related research and studies were typically focused on the technology of AI as an idea, rather than being able to demonstrate practical large-scale solutions, as these would have simply been unfeasible with then-current technology.

As technology has progressed, the scope of AI-research has also evolved, moving onto utilizing technologies such as big data-management and machine learning amongst others. Practical adaptations of AI have also begun to exist, giving researchers the chance to tackle real issues with real solutions, rather than having to focus on explaining the chance of an innovation working on a

theoretical basis. A great example of such a research, would be the one conducted by Himeur (2023), showcasing the combination of deep learning and big data-management as a solution for improving organizational systems. One of the technologies allowing for this development is the focus of this research, large language models.

The capabilities of computing devices have continued to develop drastically since the early mentions of artificial intelligence. Ultimately, this had led to the present state, where computers are able to store sufficient amounts of data to form and utilize their own artificial neural networks (ANN) large enough, to help make accurate decisions based on bits of data the ANN contains (Chowdhary, 2020). ANNs are models which help computing devices mitigate the processes of a human brain on a computing device (IBM, 2021). This development and the capabilities it has brought, has been the basis for AI. Throughout recent years, these capabilities have only been expanded upon, through innovations in various associated fields of computing. Simultaneously, the total global volume of data has reached a staggering 150 Zettabytes, providing a hefty basis of information for AI-based decision making and training (Bartley, 2024). Due to the rapidly evolving nature of technology and its associated fields, the implementation of AI presents organizations with plenty of opportunities and challenges (AMD Silo AI & AI Finland, 2025).

Zhang et al. (2023) suggests that AI has reached the point, where three distinct generations can be identified. The first generation is said to have been developed since the 1960s, and to have reached its peak in the 1980s, because of its limited capabilities. The first generation of AI was only able to work on simple tasks, mostly using methods such as trial and error, mean-end analysis and divide-and-conquer. The limitations of these approaches meant that the potential of complexity for the issues first generation AI could efficiently manage was quite limited. Some notable milestones were however achieved, such as an AI being able to defeat the current chess champion for the first time in 1997. (Zhang et al., 2023)

Second-generation AI is a huge leap from the first generation of AI, with a notable increase to the available ANN's bringing a significant leap to the ability and accuracy of AI-solutions. In addition to accuracy, Second-gen AI also reduces the importance of rules and knowledge being given to the system by humans. With these, the idea is rather that a human gives the system data, and conclusions and knowledge can be found by the system itself. Typically, this is done through a given search model, or a data model which has been provided by a human. After this, the computer looks for trends, correlations and other things such as this, in order to provide its findings to the user. This process is often referred to as Machine Learning (ML). (NTTData, 2018)

The first two generations of AI are based on existing and proven solutions – which can be seen as the backbone of many AI-based solutions developed in the modern world. Third-generation AI meanwhile remains more so in its conceptual phase – with different bits of research defining and seeing it in slightly differing ways. Zhang et al. (2023) summarize it as a combination of first- and second-generation technologies, seeking to develop robust and extensible AI-technologies, which are capable of reflecting human behaviours in an improved manner.

2.2 Machine learning

In recent times, perhaps the most influential sub-field of artificial intelligence has been machine learning. ML refers to automatic computing procedures which are based on logical or binary operations which an automated system is able to perform based on learnings which have been made from previously reviewed examples (Michie et al., 1995). As is the case with the human brain and learning of a new skill, such a development process typically relies on existing and previous results and data being provided to a system, which is then able to process this data, seeking to find trends and links between different bits of data and information. After this process, the developed system should be able to make educated predictions on future situations, where only some metrics may be provided. For instance, if a model has been shown that every situation where a certain variables were set at a given value, it will expect future situations sharing these values to also share other trends. (Michie et al., 1995; Z.-H. Zhou, 2021)

In the present, machine learning is utilized as the backbone of many services have grown to know and rely upon. For instance, image recognition and fraud detection are likely to be features many people use on a daily basis (Lai, 2019; Sharma et al., 2021). These are often provided to users in a seamless manner, where they may not even realize the effect a machine learning algorithm plays in the process. This is due to their smooth implementations and largely invisible background role. The implementation of these has been largely left down to the usual players in charge of the ways users interact with many services – for instance the photo-management software makers and banks in the case of the previous examples (Angelova, 2025).

2.3 Large language models

Alongside the previously mentioned innovations as well as many others, ML has become known to the masses as a result of several large language models (LLMs) being released to the masses. These include services such as ChatGPT, Microsoft CoPilot, and Google Gemini. Each of these relies on LLMs, which have been developed by their respective vendors, and aim to provide end-users and easy path to utilize and inadvertently help them train their machine learning models. These chatbots, which can be used for gathering information on essentially anything, have become a key part of various workflows and the daily lives of many people. This is because such solutions offer many benefits over typical options, such as online search engines. In August 2024, the developers of ChatGPT, OpenAI reported that ChatGPT had reached the threshold of 200 million weekly users (Reuters, 2024). Simultaneously many other massive firms, such as Apple and Meta are investing heavily towards offering similar products of their own to their respective customers (Gunter et al., 2024; Iovine, 2025).

As is typical for quite a recent development, especially when considering the breakthrough into the mainstream of users, scientific research about large language models (LLMs) and their usage and setup-phases is still relatively scarce. As a positive, due to its recency most of the related studies are still quite recent and up to date with the most recent of innovations. Schillaci et al. (2024) have done significant research on LLMs, discussing many of their perks and drawbacks, as well as going through many commonly used models, which can be utilized when adopting an LLM. Some of the models discussed within Schillaci's research will also be parts of the benchmark suite of this research. In addition to reviewing different models, their research also touches on the process of identifying suitable models for given hardware, or vice versa, and the process of optimizing this process during the training phase, or version selection of a given model. (Schillaci, 2024)

2.3.1 How do LLMs work?

LLMs work by using of large sets of data and strings of words in order to look for the most suitable words or sentences, which should follow what a user has entered. The basis of its functionality is its training dataset, which comprises of a huge number of words and sentences. These are collected and data-mined from various sources, such as websites, articles and more, depending on the purpose of a given model. These can be designed for a wide scale of general-purpose needs or specific niches such as programming, finance or other similar fields. After the optimal set of data has been determined, it must be processed and trained. Many improvements and changes are made during

this phase, perhaps most importantly strings of words and sentences are turned into tokens, allowing the model to learn about different tokens and their respective relations and meanings to each other – in order to be able to combine them later on (Brown et al., 2020).

The completion of the training phase of a model allows for the practical deployment and utilization of a model. When the data has been processed and tokenised, the model is then able to compare any user inputs to the tokens found from its trained dataset. If this processing hadn't taken place beforehand, a model wouldn't be able to see the correlations and meanings of different sentences and words. This would make it extremely inefficient and difficult for a model to find the correct responses to user inputs from its raw dataset.

After the training process, the data of a model effectively form a transformer network – which is a form of a neural network with knowledge of context and relationships between different aspects of data. A typical transformer consists of many layers (transformer blocks), which have been designed to work together to offer more accurate outputs. This architecture has proven to be suitable for LLMs for many reasons, perhaps most importantly due to its versatility. The usage of transformer network architecture for instance allows such models to understand tokens even when they are entered in unpredictable orders, reducing the amount of training overhead needed to effectively deal with varying input prompts. (Vaswani et al., 2023; NVIDIA, 2025b)

When using a large language model, a user may for example enter the following list of words: “ingredients for cookie dough”. When this takes place, an LLM should be form an understanding of the entered prompt by tokenising it. The process of tokenisation effectively attempts to split every aspect of a word into a separate section (token + sub-token), whenever a dedicated meaning is attached to that part of the word. Once this has taken place, a model will compare the input tokens to data within its training dataset, ultimately seeking to find tokens including information about the question it thinks it has received. In this case these should include information about the required ingredients of cookies, which are then ultimately presented to the user. This working philosophy makes it possible for a model to form an understanding of any entered input prompts, even when they are entered in varying wordings. To make these models practical, it is crucial that they can function without having ever been prepared for exact input prompts, rather ones with similar meanings and linked tokens (D. Wang et al., 2024).

2.3.2 Online-based LLMs

Online-based AI chatbots are typically based on proprietary large language models, which have been made, trained and designed to be ran on the premises of an external service provider (Casheekar et al., 2024). For instance, in the case of ChatGPT an end-user interacts with their website or dedicated app, which then sends user prompts to OpenAI's datacentre to be processed by their pre-trained LLM on their premises. Whilst this sort of approach to LLM-processing has many advantages such as ease-of-use and minimal hardware requirements, many issues are also brought into the occasion by the power the service provider (in this case OpenAI) holds on their users. To the end-user, this process isn't very transparent, whatever happens to the prompt in-between its sending and output return remains mostly unknown to the user. Such a system effectively works like a black box from a user's point of view (Kosinski, 2024).

When in complete control of a whole LLM-service, its provider can determine exactly how a given model works. This for example includes determining what sort of bias's it may have – something which could have a drastic impact on the information given to its end users. This sort of opportunity may result in all sorts of conflicts of interest – both to do with showcasing a service provider such as OpenAI themselves in the best way possible, as well as potentially other parties, who may for instance be willing to “advertise their products on a chatbot, without the end-user even realising that their results are potentially skewed in an unfair manner. Biases such as racial, gender or nationality may also end up existing within the training material through other reasons, both on purpose and by accident. The closed-off nature of such proprietary software makes this tricky to review, as access to the background material and its selection phase can be highly limited (Oketunji et al., 2023).

Issues such as the ones mentioned can also be tricky to spot from closed source commercially available LLMs and chatbots utilizing them, as their service providers typically offer their users minimal transparency into the functionalities of these systems. This is in notable contrast to open-source models, which are seeking to offer end users similar results, whilst also providing full transparency into their functionalities. By utilizing one of these services relying on a proprietary solution, a user or an organization effectively puts their trust in its publisher/developer. Ensuring factors such as hidden interests, data storage and processing pipelines and others to have been properly taken care of will likely be based upon assurances made by the service provider, rather than being able to independently verify them. (Malec, 2024)

The line between these and open-source models can also be complicated by various services. For instance, models such as Meta AI's LLaMA-series are also offered as complete services hosted on external datacentres on existing platforms (Meta AI, 2024a). Whilst these should rely upon the same exact models' end-users are able to run on their local devices, making the process more transparent – not all the previously mentioned issues are resolved. Most notably, a user's prompt is still sent over to be processed on an external server – potentially with no guarantees of how the data may for instance be stored or processed for training purposes.

The term open-source model is also used somewhat loosely, with many models claimed to be open-source often failing to fully meet the required standards, especially when it comes to aspects such as truly published and open training datasets. For instance, Meta AI's LLaMA-family of models is often claimed to be open-source, yet it fails to meet some of the usual criteria for such models, especially when it comes to publishing information about the origins of its training data (Parthasarathy, 2024). These models are often identified to form their own in-between class of so-called open weight-models (Ramlochan, 2023). For the purposes of this research, each of the models discussed and reviewed will be referred to under the commonly used open-source umbrella, but it should be considered that arguments against this qualification exist for the models reviewed. For the performance-based benchmarks of this paper, these issues are largely irrelevant and shouldn't impact any suggestions and findings.

2.3.3 Local LLMs

Local LLMs take a different approach to setting up a large language model-system for its users. Instead of using a fully pre-made and externally managed solution, in a localized LLM-system the end-user or an organization chooses to deploy its own model. This may be a publicly available open-source model, a completely self-developed and -trained model or something in-between. The key difference between a local and a service-based LLM however is the fact that a local one can provide greater security, optimization, and control over entered data (Schillaci, 2024). This stems the fact that any entered data remains within the memory and computational devices of the system running the model. This can be an in-house system, or an externally hosted solution. Ultimately, this provides the party setting such a solution up with control, allowing it to ensure its data is managed and stored in a compliant manner.

When an LLM is run on the IT-structures of an organization, many additional opportunities aside from pure security benefits arise. As full control of the system is gained, so is the opportunity to determine exactly what is needed of the model and utilize this information to optimize any solutions

to offer higher quality outputs for its targeted knowledge. This not only allows an organization to prioritize useful tokens but also cut back in terms unnecessary tokens and data during the progress. This form of optimization can both increase performance and output quality, whilst simultaneously reducing the file size and improving hardware compatibility (Javeed, 2023). The removal of unnecessarily training data which won't be needed for the targeted deployment of a model, also gives an organization the chance to reduce the amount of training related resources required for a given model. It should however be noted that when compared to deploying a pre-made open-source model additional training requirements will arise from any changes made to the original solutions (Javeed, 2023).

Localized LLMs can be made by various parties for differing purposes. Models can for instance be designed for general-purpose tasks, resulting in their capabilities being consistent across a wide variety of tasks a user may wish to utilize them for. Alternatively, a model may also be tailored to only be utilized for a limited amount of use cases. Both of these approaches have their benefits; a general-purpose model is suitable for many different needs regardless of the scenario.

Simultaneously more tailored models can offer better results in the tasks they have been designed for, as they will possess significantly more useful data to tackle the task, rather than having a bunch of wasted knowledge that isn't utilized within the workflows the model has been chosen for. This becomes specifically important, when the hardware capabilities and resulting model sizes of a solution are limited.

Models can also be different to one-another when it comes to their ownership and controlling parties. Open-source models such as Google's Gemma-family or the recently launched DeepSeek R1 are released on the principle that users may utilize and edit them as they wish, with the process producing plenty of transparency. Simultaneously, many of the giants of the LLM-industry, such as OpenAI have chosen to keep their models firmly within their own control, utilizing so-called closed-source principles, where little detail of the model is provided. Such models can't be utilized on local systems without the vendor choosing to give an adopter access to these, something which is unlikely to take place. Therefore open-source models have become the norm for localized LLMs, as these offer ease of access, whilst also being flexible and even improvable. Individual organizations may also choose to develop their own closed-source models for their needs. However, this is unlikely to be a viable solution for most organizations, due to the significant costs and data requirements of going through such a process.

2.3.4 Training and development of LLM-models

The development of a self-made large language model begins with the process of figuring out the targeted capabilities of the model. As the amount of data entered limits the amount of hardware that is going to be able to effectively train and utilize the trained dataset, optimizing and focusing a model that is only going to be used for a limited number of prompts can be highly beneficial (Yang et al., 2023).

When a model is for instance being developed to help people with prompts related to managing finances and nothing else, it can be highly beneficial to select training data which focuses on this topic. This way even with a limited number of parameters, a highly capable model can be developed, as for instance a 5 billion parameter model can only consist of relevant data, rather than wasted information the end users won't need to use from the system. When precious and limited training parameters aren't wasted on unused aspects, the efficiency of the whole process can be drastically improved (Yang et al., 2023). Whilst general-purpose models have perhaps become the norm in many cases, these typically include significant amounts of performance overhead for limited usage contexts.

Once the targeted capabilities of the LLM are clear, the next step is to select data-sources that will be utilized for the actual training process. These can consist of related webpages, literature and many other formats of content. It must be noted that utilizing content for the development of an AI model can be in breach of copyright-laws, depending on the type of information and its original source. After the basic data sources have been selected, the actual training process typically begins with some basic filtration of the selected data, with the aim of removing low-quality data, duplicates and other non-beneficial datapoints (Touvron et al., 2023).

Once the unprocessed data has been filtered, the next step crucial step of the process is the tokenisation of the data. During this phase, strings of words and letters are processed and turned into so-called tokens, which the model is able to link to separate related tokens (D. Wang et al., 2024). At this point of the process, data from different sources begins to be linked across one-another. After the tokenisation process, further optimisation is performed on the model and its tokenized data, in order to minimise the required memory- and runtime requirements of the final model. In addition to the need to source adequate amounts of suitable data, the training process also comes with extreme needs for compute and memory resources (Jia et al., 2024).

This process has a drastic impact on the model and is the factor allowing for relatively simple devices to run these models effectively. For instance, the commonly used open-source LLaMA-7B model, which was trained from an initial 10 trillion-tokens, ultimately ended up at 7 billion parameters with the final model. This results in a file size of 3-8GB, depending on the bit-depth of entries utilized. This sort of size makes it accessible and usable on a wide range of devices, especially at the lower bit-depths/quantization levels.

The bit-depth of a given model refers to the process of quantization, which effectively reduces the initial data-accuracy down to a more manageable number, attempting to minimize losses of accuracy in the process. This process can have a significant influence on the end-results a given model can produce. These effects are only accelerated the more the precision is reduced, with for instance a reduction from a Q8-depth down to a Q4-depth often having a more significant impact than the one from FP32 to FP16. This is due to the diminishing returns the final lost numbers and aspects of data typically have on the end-results. Simultaneously, the accuracy reductions resulting in the smallest output quality differences, can still typically offer the most notable advantages and leaps when it comes to inference speed. (Lupu, 2024)

The training process of a model can be extremely energy- and compute intensive. In the case of the previously mentioned LLaMA 7B-model, a total of 36MWh of energy was needed, when trained on last-gen, but still very high-performance NVIDIA A100-80GB GPUs. This amount energy was utilized over 82 thousand GPU-hours, with a high level of parallelization across several GPUs needed. Such costs and requirements go to show the advantages of pre-existing models, as this phase can be avoided at the cost of opportunities for organizational optimizations. (Touvron et al., 2023; Wu et al., 2022)

2.3.5 Measuring LLM-model performance

The end user of an LLM faces two key measurables when utilizing an LLM: the output quality and the inference speed of a given solution. Firstly, the output quality refers to the types of outputs a model can provide its users. These can be compared and reviewed based on their accuracy, as well as readability and helpfulness amongst other factors. Typically, the main factor impacting the output quality of a model is the number of processed high-quality parameters the model has access to. These are the basis for any output it can offer its user. If a model has for example triple the amount of data to work with, and there isn't a drastic difference in quality, it's likely that a larger model will be able to produce better and more consistent results (Broadhead, 2023). Outliers are however entirely possible, especially if the model with less parameters has other beneficial factors.

For instance, a drastic change in bit-depth can significantly impact the precision of each data entry, highlighting the importance of factors aside from pure parameter counts.

In some cases, the measurement of output quality is a simple truth/false equation. For instance, when asking about an objective fact or a math equation, only a limited amount of correct answers exist. This makes measuring the output quality quite a simple task, which only requires some basic knowledge on the topic, or other reliable sources of information to be used for cross-checking outputs. When it comes to less straightforward and simple outputs, the benchmark process becomes significantly more difficult, as two models may produce factually correct answers, yet have significant differences within factors such as context and depth of information. For benchmarking these sorts of prompts, more sophisticated methods are needed. These may include manual reviews and ratings, AI-based reviews and their scores. Even LLMs dedicated to evaluating and scoring LLM-performance exist, such as G-Eval (Ip, 2024).⁰

The second key performance metric a user will notice when utilizing an LLM is its inference speed. Inference speed is typically measured in tokens per second (tok/s), referring to the number of tokens the system is able to produce during each second of responding to a user's prompt. Theoretically, even a speed of less than 1 token per second can be useful, if the system is given time to output prompts before attempting to read them. Humans have been identified to be capable of reading 4-7 prompts per second, therefore this is the point the performance of an LLM can truly be seen as sufficient, as an end user is no longer having to wait for outputs (BitNet, 2024). Whilst speeds beyond this can be useful, investments beyond these speeds typically offer diminishing returns, and fall outside the targeted use-cases of this research.

In addition to these perhaps simpler and easier to measure LLM-related metrics, the capabilities of these models can also be compared on other metrics, which may play a key role for certain workflows. For instance, Hu & Zhou, (2024) found that metrics such as ethicality, fairness and generalisation can play a key role in the way a given model can perform, especially when attempting to make sure these are produced and design in ways that are legally and ethically well made, something which isn't necessarily a guarantee in the world of AI-models. (Hu & Zhou, 2024) Industry giants such as Meta have for instance been accused of collecting data illegally and utilizing it to develop their LLaMA-family of models (Smith, 2025).

2.3.6 Measuring LLM-hardware performance

In the raw hardware context of things, AI/ML/LLM-performance can be tricky to quantify in numbers. Plenty of numbers for this process exist, but these are typically not perfectly representative of what a user may expect to see, as performance can be limited by plenty of factors. Perhaps the most commonly utilized metric for the AI-performance of a given microprocessor or system, is TOPS (trillions of operations per second), referring to the pure theoretical throughput capability of a chip (Conway, 2024).

However, due to the complex nature of AI-related workflows, the practical capabilities may differ drastically, as TOPS-figures typically scale quite differently across factors such as bit-precision. These differences may be especially apparent when comparing the solutions of different hardware vendors, but may even arise on a gen-on-gen basis, with for instance NVIDIA recently claiming to have doubled the AI-performance with its Blackwell-microarchitecture by adding support for lower-precision FP4-computing, in contrast to the higher-precision FP8 lower threshold of its previous generation (NVIDIA, 2025a). These gains may or may not translate into practice, with many workloads such as LLMs often relying on higher precision types (J. Zhou et al., 2025). At the time of this research being completed, options to utilize any FP4-optimized models were extremely limited and couldn't be included in the research, even though the Blackwell-architecture in itself was tested. During this research, quantizations of each model has been set at a standardized Q4_K_M in order to seek consistency and limit variables across the benchmark suite.

2.3.7 Limitations

Whilst LLMs have been able to enter the mainstream of usage, these are still impacted by a few significant issues, that may impact an organization's ability to adopt one of these into their workflows. Perhaps the one which has been getting the most headlines and attention is the questionable manner some models and their developers have been acquiring training-data. As an example, Meta AI has been accused to have stolen data from pirated books and other models in order to train their LLaMA-family of models (Knibbs, 2025). Whilst no final findings on these cases have been made at present, such questions on these topics aren't expected to improve the willingness of any organization to adapt such a model.

Additionally, no universal rulings and laws exist for the development and adoption of large language models. Some regions have taken significantly more allowing approaches to the issue than others. For instance, over in the US, very limited amounts of legal limitations have been set for the development of AI and large language models (White & Case, 2024). Simultaneously the European Union has taken a slightly different approach to the issue and potential of AI, identifying several key issues, and attempting to mitigate and regulate these through new legislations, such as the EU AI Act from 2023 (EU AI Act, 2023). This has resulted in some limitations for people and organizations attempting to adopt certain models, with for instance some of Meta AI's models not being available for users within the EU (Milmo & editor, 2024).

Another important LLM-related issue is its intensive energy requirements, and the problematic nature this causes in today's sustainability-seeking world. Plenty of research has been done on the topic. The research of Hoffman et al. (2022) offers a lot of details about the compute- and energy needs of the process of training a new model. Whilst the training loads aren't the primary avenue of research within this research, the interlinked nature of training optimisations impacting inference performance is still an important baseline to consider within this research. For instance they made findings which showcased the impact an in-depth training process resulting in additional tokens can have in terms of output quality, even on lesser hardware allowed by the reduced parameter count. (Hoffmann et al., 2022)

The high energy and resource requirements of typical training loads also help to demonstrate the value of pre-developed open-source models, such as Meta AI's LLaMA family of models. Open-source models include solutions for a wide range of requirements and hardware-solutions on the basis of architectural modularity and computational accessibility, rather than limiting access and maintaining the model as a trade-secret. (Manchanda et al., 2025)

3 Information system hardware structures – past, present and future

Information systems have recently seen significant developments on the back-end side of their functionality. Since their formation, these have typically been mostly based on CPUs (central processing unit) in charge of most of the computing workloads. Innovations within CPU-technology, as well as solutions such as mass- and random-access storage have allowed significant uplifts in terms of capabilities. These have allowed them to keep up with the trajectories set out by the so-called Moore's Law, suggesting the number of components and therefore computing performance of microchips should roughly double biennially (Schaller, 1997). This long-standing path appears to have begun slowing down in recent times, primarily due to the physical limitations of silicon manufacturing amongst other factors (Woods, 2024).

To allow further development and new innovations, such as AI, machine learning and specifically LLMs to be adopted properly, new approaches and solutions to information system architecture have become increasingly important – as these have shown to still allow for drastic improvements to computing capabilities of different systems and workflows. New architectural directions in terms of hardware and software solutions have become increasingly important as improvements in microchip manufacturing processes have begun slowing down. (ARM, 2025)

3.1 Typical solutions of the past, CPU-based computing

A typical data centre or a server consists of one or more parallelized systems working in conjunction to respond to tasks its users rely upon. In a typical system, parallelization is achieved by utilizing one or more CPUs, which offer high amounts of cores and threads. Cores are physical parts/segments of a CPU's silicon die, which can typically process one command at a time. Through technologies such as hyper-threading or SMT (simultaneous multi-threading), it has however become viable to run two or more threads per core, allowing for multiple independent operations to run on a single CPU-core at a given time during each clock cycle (Shardeum, 2023). Multiple cores and threads allow for servers to support multiple bits of software and users at a given time – making centralized servers supporting multiple users at a time possible and practical in the first place.

CPUs have managed to improve their parallelization performance significantly through the years, perhaps most importantly through the innovation of multi-core processors (Gepner & Kowalik, 2006). These were initially brought to the market in the early 2000s, by both IBM, as well as Intel (Alseqyani & Almutairi, 2023). Whilst CPUs initially only offered 1 core, current CPUs can support high amounts of parallelization. For instance, a modern cutting-edge AMD Epyc™ 9965

data centre-CPU offers 384 threads across 192 cores, meaning 384 separate tasks can be processed by the CPU during a given clock cycle. At a maximum clock speed of 3.7GHz, this means that theoretically over a trillion instructions can be processed during a second (AMD, 2024a). When it comes to measuring AI-related performance, this translates to floating point performance of roughly 11 TFLOPS (trillion floating-point operations per second) at a 32-bit precision ratio.

A CPU-based server excels at versatility, as CPU-cores are optimized to support a wide variety of software. Modern server-grade CPUs have been designed to be the backbone of servers, which can take care of actual processing by themselves, but in the modern data centre their role and focus has transformed. Their role in a modern data centre is less about actual computing, and more about being an IO-hub (input/output), running basic processes and bits of software, but most importantly, moving complex operations over to devices more optimised to processing a given task. In practice, the CPU may for instance be in charge of transporting files between forms of storage and devices, as well as running basic software such as the operating system, whilst the actual processing of targeted applications is often left to other devices, especially when the software in question has high demands for computing capabilities, such as in the case of LLMs. (Choudhary, 2024)

For a server hosting LLMs, the role of the CPU as an IO-hub is highlighted. As running LLMs on CPU cores can be highly inefficient and struggles to scale up to biggest models at an acceptable speeds, these are typically left to a smaller role, mainly of “feeding” other accelerators. Tasks such as pre-processing and initial parsing of data may take place on a CPU, but in the grand scheme of things, a server built to serve LLMs should have its focus on other aspects of computing performance, as CPUs and relatively slow RAM (random access memory) speeds simply cannot compete with other solutions. Especially for high-end systems and models, the key focus should be on the CPU providing sufficient amounts of connectivity for downstream devices. (Puget Systems, 2024) For such a deployments, the bottleneck of the CPU isn't the core count and performance of the CPU-cores, rather the amount of connectivity it can offer, through links such as PCI-Express (PCI-E) lanes and networking devices.

3.2 The rise of the GPU

During the recent years, the role of CPUs has become less significant within large-scale computing infrastructures. The main factor driving this development, has been the rise of GPUs (graphics processing unit). These have been typically used for graphics rendering for related processes, and they differ quite drastically from the structure of typical CPUs. GPUs can offer thousands of separate cores per chip, excelling in parallelized workloads. However, these cores are typically quite tailored for certain operations, such as segments of graphics processing workloads. More recently, these have also began to include dedicated cores for AI-acceleration, for instance in the shape of NVIDIA's Tensor cores (NVIDIA, 2025a).

Instead of having been built to deal with any sort of workflows a user may wish to perform, GPU-cores have typically been optimized to take care of certain aspects of workflows their designer expects to be of importance on such a device – making software optimization and the achievement of full-utilization an important factor of maximizing performance. Due to the significantly more parallelized structure of GPU cores, these can offer significantly higher theoretical computational throughput, with for instance NVIDIA's H200-accelerator offering over 5x the TFLOPS (FP32) compared to the previously mentioned cutting edge data-centre CPU (TechPowerUp, 2025).

As a result of their differing structures, GPUs typically offer optimal performance when it comes to accelerating parallelized processes, also evolving to support more and more general-purpose computing tasks (Intel, 2024). A huge driving force behind supporting these more generalized operations has been programming APIs such as NVIDIA's CUDA and more recently AMD's ROCm, which have been designed and released by their respective GPU-manufacturers with optimizations for their hardware. These APIs have also become the backbone of many LLM-related workflows. Most available local LLM-models for instance have been trained on NVIDIA GPUs, relying on the CUDA-toolkit, and many of its tools in the process (Li et al., 2025). Similar trends do not necessarily transfer over to the inference side of performance, with for instance an NVIDIA RTX 5090 at times even performing better with the open-source Vulkan API when compared to NVIDIA's proprietary CUDA API. (50 – 40 tok/s, Gemma 2 27B)

One of the tasks which has become known for relying on GPU-computing in the recent times has been machine learning, which has become commonly used as the allowing technology for many AI-based solutions, such as ChatGPT. GPUs can offer drastic performance benefits for these workflows, and this has also become a key selling point of new GPU-architectures, providing key optimisations for these processes (AMD, 2025, NVIDIA 2025a). GPUs have several traits which

make them ideal for machine- and deep learning workflows, such as memory bandwidth from their own memory sub-systems and highly parallelized core counts. These make them ideal for running large amounts of small operations, which are typically found from AI-related processes (run.ai, 2021).

For most organizations targeting demanding LLM-solutions, a GPU-based server is likely to be a good solution. GPUs and GPU-based arrays have the capability of forming massive memory pools alongside their compute capabilities, making it possible to deal with even the largest of AI-models, with parameter counts in the tens and even hundreds of billions. An example of this would be LLaMA 3.2 Vision 90B, requiring at least 180GB of VRAM (video random access memory) to load the full model onto a GPU (Meta AI, 2024d). Such a number could be reached through many means, an example of a sufficient array would be a combination of three NVIDIA A100 GPUs, with each providing 80GB of VRAM. Even for models with more reasonable parameter counts, GPUs are typically the efficient solution, providing significant speed advantages.

3.3 Other solutions – NPU, SOC, FPGA

With the importance of AI and related computing reaching new heights on a regular basis, even purpose-built silicon has become a feasible solution. Solutions like this have been brought to the market by most of the key players of the semi-conductor industry, including the likes of Apple, AMD, NVIDIA, Intel and Qualcomm – each of whom holds a drastic amount of semiconductor market share. Most of these solutions come in the form of a so-called NPU (neural processing unit) – which are unlike CPUs or GPUs. These dedicated processors or accelerators as part of SOCs (system-on-chip) have been tailor made to excel in accelerating machine learning algorithms and executing specific operations typical for AI-workloads – significantly increasing AI-performance in the process. (Peru, 2023)

As mentioned, NPUs come in different shapes and sizes, as well as respective tiers of performance. Whilst for instance many mobile devices have come with AI-computing related NPUs for a while, a notable step has been taken recently within the personal computer-industry leading x86 CPU-industry. Both parties of this duopoly have brought dedicated NPUs to the market with their new mobile architectures Lunar Lake and Strix Point, bringing 48 TOPS and 50 TOPS of dedicated NPU performance (Intel 2024, AMD 2024b). The noteworthy aspect of such innovation is the new levels of reach for AI-related workflows, even on portable devices such as laptops. Such platforms have allowed AI-related workloads to become a key selling point for new devices, ultimately even resulting in Microsoft introducing an industry standard called CoPilot + specifically for AI-related

performance, setting the first official requirements for devices to qualify as “AI-PCs” (Mehdi, 2024). It should be noted that the vast majority of NPUs that have been seen entering the market have come in the form of an integrated solution of a SOC, specific and dedicated NPUs have only been seen as extreme niche solutions, being brought to the market in limited quantities and target-markets.

With the hype around AI and model adoption only seemingly only accelerating, even more solutions utilizing this sort of architectures have begun arriving to the market, even being touted for their LLM-performance by their respective vendors. Due to the structure of these SOC-based solutions, which usually include a CPU, iGPU (integrated graphics processing unit) and a NPU on a single silicon substrate, these have allowed for manufacturers to begin pooling the memories of these modules, removing potential waste in the process. For instance, with the recently launched Strix Halo-series of SOCs, their vendor AMD has been promising to offer conversational performance with LLaMA 70B-tier models, without needing a dedicated GPU (Evangelho, 2025). Such solutions have the benefit of being able to utilize different bits of their capabilities for different aspects of LLM-related workflows, rather than having to focus efforts on a sole GPU for instance. However it should be noted, that unlike how one may imagine, the most important aspect of AI-related computational performance with LLMs is often the iGPU (integrated graphics processing unit), rather than the dedicated NPU, even when both exist (Jayanth et al., 2024).

AI-related chip optimizations can also be performed on chips without requiring specific modifications to their hardware design. This is when so-called FPGAs (field programmable gate arrays) can become an asset. The idea of FPGAs is to allow high levels of software optimisation, as the hardware can be reconfigured to meet specific use-cases after the manufacturing process, rather than requiring these optimisations to have been completed before manufacturing (ARM, 2024). Through such optimizations, FPGAs can be optimised to act as effective purpose built and optimised NPUs for a given model. The performance results can be highly impressive, as showcased by Boutros et al. in 2020, showcasing performance improvements even when compared to then industry leading GPUs. These do however come with a high initial cost, as significant amounts of resources will be needed to develop and implement such a solution, whereas savings may only be seen in the long-term operation phase (Uppal, 2024).

3.4 LLM-capabilities across solutions

Whilst all the previously mentioned types of hardware solutions have the place when setting up infrastructures for local LLM-support, the specific choices related to these are highly contextual. First and foremost, it must be noted that not a single simple LLM-performance requirement threshold exists, rather the hardware needs are highly dependent on the model being targeted, based on factors such as parameter count and precision (Allman, 2024). Therefore, the first step of building up an LLM-strategy should be to establish the quality of model that will be targeted. When this process has been completed, a more specific model may be chosen, also allowing for knowledge on specific hardware requirements, perhaps most notably the amount of high-speed memory that is going to be needed. Additionally, the planned scaling of deployment will have a significant impact on the requirements for a given model.

Depending on the tier of model chosen, the next step should be to choose the basic platform the model shall be ran on. If basic tasks are sufficient, and a sophisticated model isn't needed, it is quite likely that existing hardware will be able to deal with a given model, through basic CPU-computing, as the speed is going to be acceptable, even in a non-accelerated format (Kozlov, 2025). However, if a larger model is targeted, the benefits of GPU/NPU/FPGA-computing quickly become practically mandatory, even for inference loads. As these advanced models are targeted, the setup costs will quickly arise in the process, as these will typically require dedicated hardware, rather than being allowed to rely on other solutions which may already pre-exist for other needs.

At the present, GPUs are the industry standard for adopting LLMs which require additional acceleration. These systems have their issues, such as low-compute utilizations and potentially extreme costs. These issues are especially obvious, when dealing with the largest of models and multi GPU-nodes, with some of the GPUs effectively working as additional memory modules, leaving the compute capabilities at a low utilization (Gu et al., 2025). Other architectures do exist, but these have been left in a somewhat niche role, mostly being trailed by enthusiasts rather than deployed into real commercial environments. As an innovation with plenty of hype and resources being invested into it on a constant basis, new and innovative approaches are created on a constant basis.

3.5 Processing centralization

An important question which also needs to be answered when setting an organization up for local LLMs, is the nature of the associated infrastructure. Depending on the size of the organization and the number of clients, it may or may not make sense to rely on a centralized server to take care of LLM-computing. Whilst a centralized system will typically require extra time and effort in its setup phase, it can easily recoup these costs, especially if the amount of hardware required is significantly reduced in the process. This is highly dependent on the amount of people a given system is targeted to serve, as well as the size of a model being targeted.

If a low-end model is targeted, the hardware requirements will also be limited, and it's quite likely a dedicated "AI-optimized" solution isn't needed, especially in cases where the personnel of a given organization already have relatively new work laptops or other devices. At this point, the worst-case scenario may be that older devices may need to be updated, but this may have been an issue which was due to arise soon regardless. This is especially true when considering factors such as the soon ending support of Windows 10-based devices, practically making many old devices legacy and unsecure, pushing organizations towards hardware updates regardless (Microsoft, 2025).

The general IT-strategy of an organization might also play a key role in how workers can be equipped to utilize local LLMs. If a centralized strategy has normalized the hardware solutions workers utilize, finding specific models for the hardware on offer will be relatively straight forward. This is in drastic contrast to an organization which has adopted a BYOD (bring your own device) policy – resulting in drastically differing hardware solutions. Drastically differing devices will offer drastically varying LLM capabilities, both in terms of performance, as well as supported APIs – making the management of utilized LLMs significantly more complicated.

In situations which the end users' devices aren't capable of running the targeted models, a centralized solution is typically going to be the solution. Such a solution may be run on a local server, or through a third-party service provider renting out hardware time on a time- or per token-basis. Even if an organization chooses to rely on an external service provider for their computing needs, their data will remain significantly more secure compared to utilizing a commercial LLM-solution, as their data will remain within their controlled and isolated system. Because of this, the data can be controlled with the aid of a reliable partner, keeping it away from risks such as third-party model development. To the end users, such a solution may be accessed through a specifically made program, or directly from the web browser, allowing for easy compatibility across different

platforms, including Windows, Mac OS or mobile devices – ideal even for organizations adopting a BYOD-policy within limited opportunities for standardization.

Such solutions will be the most important when it comes to findings solutions to running some of the most sophisticated and large models. Some of these require hundreds of gigabytes of fast memory, as well as significant amounts of parallelized computational performance. In the process, it quickly becomes impossible for such models to be executed on end-users' devices, as these are typically run on systems with significant power- and space requirements, as well as high acquisition costs. For instance, with the full 405B variant of Meta AI's LLaMA 3.1 model, the suggested hardware configurations of several NVIDIA A100-accelerators come with costs of several tens of thousands of euros – effectively making centralized solutions mandatory in the process (Meta AI, 2024c). As models continue to improve and develop, centralization may become a less important factor, as smaller models are becoming increasingly capable and competitive with larger models, reducing the need to invest into the structures required to run the biggest of LLMs (Google, 2025).

3.6 External hosting (cloud-based computing platforms)

Due to the expensive nature of devices required for high-end LLMs, especially when it comes to the initial acquisition cost, a typical and attractive alternative to self-hosted model structures relies on an external host of a given model. In such a solution, an organization gives the responsibility of processing any entered prompts and their data onto a different organization, whose data-centre deals with the processing requirements of a given model, resulting in a shift turnaround-time and added flexibility (Shivanna, 2024). Such service providers include the likes of Meta, Amazon (AWS), Microsoft (Azure) amongst others.

Typically, an externally hosted AI-solution functions with a GPU(s)-effectively being rented out to the client. This partnership and its costs may be structured in differing ways, for instance on a time or per/token basis, with each of these coming with drastically differing cost structures. Ultimately if properly deployed and utilized, the long-term running costs of such a solution will typically end up higher than the acquisition costs of a self-ran solution. Simultaneously, relying on external hardware on a service basis offers significantly more flexibility and room for adaptation, even after the initial deployment (Shivanna, 2024).

Relying on external service providers on an on-demand basis can be a great option to get started with localized LLMs within an organization. This way, the initial setup costs can be avoided, and the practicality of these models can be properly assessed without significant investments. For instance, an organization gets the chance to properly assess the level of model it may need and therefore avoid investing in an overly abundant or insufficient model, avoiding the waste of financial and human resources in the process. To summarize, many cloud providers offer on-demand GPU-instances, however in the long term they are typically more expensive than self-acquired GPUs. (Łazuka et al., 2024)

4 Common trends and factors impacting LLM-adoption projects

This chapter of the research will seek to shed light on the factors impacting LLM-adaptation projects with the aid of a systematic gray-literature review. The research will go through various online-based sources discussing local LLM-based transitions and projects to discover key trends behind such projects. Whilst a typical systematic gray-literature review model could make managing a large sample size quite tricky, machine learning-assisted data-mining techniques have been used to complement the process. This has allowed for more efficient processing of a larger set of data, helping the research make accurate findings (Audet & d'Amboise, 2001). This should allow for a solid foundation for building an understanding about the factors behind such projects, as well as typical ways these tend to be built. These trends should be valuable assets for the latter main part of the research, especially when it comes to making suggestions for how such projects should be completed – as industry trends and standards typically exists for valid reasons.

This chapter will consist of two key parts, beginning with the process of collecting information from a sizable set of sources of various types, including forum-discussions, scientific articles, blogposts, case-studies and other sources. Each of these will be reviewed for key points, which will then be noted. During this process, often standing out aspects will be listed, allowing for an understanding of the most discussed trends. Once these trends have been identified, they will be investigated with further detail, in order to understand their underlying causes – ultimately providing valuable insight into how a successful local LLM-adoption project should be performed in an organization.

As mentioned, the research material will be collected from a variety of sources, including the views and thoughts of people from various backgrounds. These should also contain information about such projects from a wide scale of different types of organizations, varying from small organizations to multi-national corporations with everything else in-between. Whilst these are likely to have differing standards and trends that arise as results, there should also be common trends across the field – reflecting the most important principles and advantages for deploying large-language models. Through a set of data consisting of various fields in a large sample size, factors only impacting a few industries and not applying to most firms seeking to adopt local LLMs should mostly be filtered out.

The sources were identified and selected using a few search engines, most notably Google, Google Scholar, as well as posts from platforms such as LinkedIn and GitHub, utilizing their respective search tools. Findings from these platforms offer a wide range of different types of results, all the way from individuals reporting on their projects, to related service-providers and researchers of the topic, allowing for findings from small-scale niche-solutions – all the way to huge projects seeking to change the workflows within most people of an organization.

The source material will consist of related writings, which will have been collected with the previous methods. Once these have been collected, they will be reviewed for key points, and their occurring volume will be counted – allowing the formation of a ranking which showcases the most commonly occurring trends impacting local-LLM adoption projects. Material will be sourced from both individuals writing on quite a personal basis about their experiences, as well as a more general project-based basis. The primary focus will be on the latter, as such material can typically provide better details and reliability of the topic.

Most of the resources have been written by personnel in charge of LLM-projects, be that as a service provider for an outside firm, or for instance as someone whose responsibility data-management and analysis related-aspects of an organization are. As a result of this, these people are typically writing on a relatively knowledgeable basis, someone who has been put in charge of overseeing such a project, or at least finding a suitable partner for the task should basic knowledge of LLMs and the benefits and related aspects to running them locally.

The collected material will be analysed with the aid of Meta AI's LLaMA 3.1 large language model alongside a manual review process. The aim will be to identify the most important aspects behind successful LLM-deployment from each source – allowing for a wide range of reasonings to be understood, whilst also making these comparable in order to understand the most crucial aspects of successfully deploying such systems. The usage of LLMs as a tool has been found to be effective, as it allows for a significantly more efficient process of reviewing texts – allowing for a wider set of data to be reviewed with a given amount of resources. It must be noted however, that such models aren't perfect, and results should also be manually verified, especially when a limited sample size is being utilized (De Paoli, 2024). During this research, each source bit of source material will be manually verified to contain the key aspects identified by the initial LLM-based review process. During the manual review process, the results of the LLM-based process were found to be highly accurate, requiring minimal amounts of manual intervention.

4.1 Found key trends and factors

This chapter focuses on three of the main factors and trends impacting LLM-adoption. These findings are based upon the findings made during the AI-aided systematic gray literature review and are ones which came up within several of the sources which were reviewed in the process. Most of the findings made are in line with previous research on the topic, however the focus is perhaps slightly less on the theoretical basis of such projects, rather the focus is tilted towards the practical choices related to such projects, consisting of issues such as hardware and model choices, rather than their theoretical reasons for existence. In addition to the previously discussed strictly scientific researches, these can provide additional insights into key LLM-adaption related decisions.

4.1.1 Security and self-control of a system

The first factor which arose as a crucial driving force behind local LLM-adoption was data security. To maximize LLM benefits, it should be able to be utilized on as wide a set of data as possible, without having to worry about sensitive data potentially being compromised outside of the organization's own control. With a so-called black-box model, an organization cannot ensure that their data isn't store and processed in way which would be problematic for the organization (Kosinski, 2024).

With a locally-ran LLM an organization can gather full control of a data-processing structure – being able to ensure that no bits of data are processed or stored in an unauthorized and potentially harmful manner at any point during the process. Issues like these are making many organizations hesitant about adopting such systems in their cloud-based formats into their operations, as the consequences of a data breach can be drastic. These issues highlight the value of a localized solution and its ability to offer organization full control. These advantages are driving an accelerating shift, with significant growth reportedly being seen amongst the number of firms looking to adopt localized LLMs – most notably due to their ability to mitigate most of the security concerns (Gartner, 2023).

Additionally, the other benefits of complete control over an LLM-system were mentioned as driving forces behind adopting locally-ran LLMs. LLMs can for instance inherit personal biases of their developers, which can impact their impartiality (Buyl et al., 2024). With the deployment of a self-controlled system, such risks can be controlled, either through a completely self-made mode, or by reviewing and modifying the source codes of pre-made open-source models in case any issues are identified. Self-ran models can also be optimized to offer better performance within targeted workflows, rather than focusing on general performance, leaving unutilized compute overhead into their performance. Removal of excess data can result in drastic performance benefits (Touvron et al., 2023).

4.1.2 Importance of hardware factors

Hardware aspects were expected to be an important part of local LLM-related conversations, and this was proven to be the case with most of the research writing including segments discussing hardware-solutions as a key point of discussion. Given the current balance of available hardware and models, hardware tends to be the limiting factor of LLM-implementations, as its acquisition costs are quite high, and most organizations seeking to adopt AI-solutions are currently not equipped in a sufficient manner. Organizations seeking to adopt AI-workflows into their activities through tools such as LLMs, must typically invest into new hardware solutions in the process. The amount of available hardware-investment also typically ends up as the factor setting the level of model which an organization will start supporting. Especially high-end models require significant investment. High-end enterprise-grade accelerators such as NVIDIA's H100 are estimated to cost around in excess of 20 000 € per unit. (Lu, 2024). A cutting edge high-end model such as Meta AI's LLaMA 3.1 405B may require a system with 4 or more of these in a combined node to reach reasonable levels of performance (Meta AI, 2024c).

Another trend which could be noticed from hardware conversations regarding adopting local LLMs was the prominent role of NVIDIA as the hardware-producer of choice. While other capable hardware providers exist, these were essentially ignored within the sample size of this research. Whilst many factors are likely to have influenced this result, the most probable cause is the importance of their CUDA-toolkit/API, which has been the basis of AI-development on their hardware for a long time, providing valuable resources and easily achievable optimized performance, something which their rivals have only recently been able to compete with.

It should be mentioned that NVIDIA's practical advantage is the most prominent on the development and training side of LLMs. For inference performance – the primary focus of this research the field is a lot more competitive. Inference processes are typically largely dependent on memory configurations, making the requirements for training and utilizing a given model quite different. For instance, Apple's M-series of silicon and AMD's solutions are highly competitive when it comes to inference performance. However, NVIDIA's value cannot be ignored, as the training capabilities and widely optimized software support can ultimately work to make a system more versatile and re-configurable, which is perhaps a key reason behind their products often being chosen even for inference workloads. Whilst inference-based benchmarks may suggest that their rivals' solutions are very competitive, NVIDIA's solutions are still the typically selected, with factors such as robust and reliable software and driver solutions often mentioned as key factors.

4.1.3 Popularity of pre-made and -trained models

Another trend which many of the projects researched seemed to go along with was the usage of pre-made models and applications. Instead of starting from scratch and developing a completely fresh model from self-sourced and selected data, many projects appear to be relying on pre-made models, such as Meta AI's LLaMA- and Google's Gemma-families. These offer organizations significant advantages when it comes to deployment times and ease-of-setup, as these allow for a complete bypassing of the training phase of a model, allowing for quick deployment, with some cases even reported to have been completed within a few weeks. If needed, these can also be optimized and tailored to further suit the needs of an organization (Zhao et al., 2025).

In addition to offering an easy path into local LLM-adoption, pre-made open-source models from firms such as Meta and Google also offer proven solutions, which have become industry standards in the field of open-source LLMs. As a result, these have faced significant scrutiny, having been reviewed and benchmarked by researchers and the community for a significant amount of time. Due to the resources behind these models, vast amounts of documentations and details of the way they function also exist, offering firms seeking to adopt such models' easy methods of ensuring compatibility with their specific targets. Such models exist with a wide scale of optimizations and strengths, giving organizations a lot of flexibility with their selections.

In practice, this “easy path” into LLM-adoption can reduce the amount of human resources and know-how required in the adaptation process, directly impacting the associated costs of such a project. For instance, it’s possible to execute such a project with minimal knowledge of programming, as several open-source solutions exist, and function in a manner that’s manageable for at least slightly tech-savvy staff. These may for instance be the personnel in charge of planning such projects, who may for instance be operating from a business-minded background. Not requiring additional staff or contracted programming capabilities can result in notable cost savings alongside the obvious time-based savings.

Relying on open-source models also gives an organization more flexibility, as the adaptation of such a model in itself doesn’t have an acquisition cost and therefore remains easily replaceable if newer and more appealing options enter the market. In recent times, this has been easy to identify as a common trend, with new models and model versions constantly entering the market and bringing along significant performance leaps.

5 Benchmarking models and hardware combinations

Within this chapter, different hardware and model combinations will be benchmarked, and their performance will be reviewed. The goal will be to find tiers of models, which are feasible to run on a given tier of hardware. The baseline for sufficient performance will be based on the human reading speed of around 4-7 tokens/s (BitNet, 2024; J. Wang et al., 2024). Whilst performance beyond these speeds may be beneficial for some tasks, primarily those which don't require for the output to be read by a human, rather for instance pasted to a different location – the human speed of content consumption can be set as a baseline metric for sufficient performance. Speeds slightly below this measurement should also not be drastic problems, as the main issue will be a slight delay to the person being able to start reviewing the output prompt. It should be noted that more advanced automated solutions may also benefit from significantly higher inference speeds, as these may for instance include several LLM-based passes and other means of automated processing (Heller, 2024).

This research will focus on four tiers of models represented by their reference models. Their respective performance will be reviewed across a wide range of hardware, including differing architectures and APIs alike. These models have been chosen to be demonstrative examples, of how a model in a similar tier may perform. As new models are constantly released and old ones improved upon iteratively, these should serve as guidelines – models of similar sizes and architectures should perform in a similar manner on a given set of hardware.

The four models chosen are the following:

- Gemma 2 2B Instruct Q4, Google (1.59GB) – Low-end model
- LLaMA 3.1 8B Instruct Q4, Meta AI (4.58GB) – Lower mid-range model
- Gemma 2 27B Instruct Q4, Google (15.50GB) – Upper mid-range model
- LLaMA 3.3 70B Instruct Q4, Meta AI (39.60GB) – High-end model

These models have been chosen for their differing tiers of performance to showcase what level of hardware each may require. Additionally, these are commonly used open-source models which allow these to be easily put into use. Because of their important role and high usage, various types of optimizations and sub-models also exists for these models. These have been produced by researchers, organizations and enthusiasts alike. These are also expected to receive updates for a

while into the future, helping the findings of the research to potentially remain relevant for a longer period of time.

In addition to these reference models and focusing on their performance, this chapter will also contain a brief look at extreme-high end models, containing the highest amounts of parameters for industry leading levels of accuracy. These extreme-models also come with extreme hardware-requirements, making them non-viable options for a vast majority of organizations due to their costs. It should be noted that the file sizes mentioned above are only representative of the size of the model itself, with practical requirements often significantly exceeding this figure, especially when things such as input prompt context lengths are set to their upper capabilities.

Each model chosen has been reviewed in its Q4-format, allowing for the limitation of variables impacting the process and its results. Models with differing architectures, such as reasoning models (Qwen's QwQ-series, DeepSeek R1 etc.) can offer drastically differing results, both in terms of output quality and inference speed, and should therefore be reviewed on an independent basis. Straight size and output quality-based comparisons aren't completely practical between our reference models and reasoning-based models, as reasoning models typically include a higher amount of processing overhead, resulting in comparably slower inference speeds. The focus of this chapter will be strictly on the inference speed of the reference solutions, with the next chapter after focusing on defining the capabilities of different tiers of models through separate benchmarks.

As mentioned within the initial chapter, these inference performance of these models will be reviewed based on the following prompt wherever applicable:

- Write a story that is 1000 words or less, which tells the story of a man who comes up with a revolutionary new way to use artificial intelligence, changing the world in the process.

Whilst the prompt itself doesn't typically impact the inference performance of a given model in a drastic manner, a singular established prompt has still been selected to be the primary focus of these numbers to ensure consistency amongst the findings made within this research. Any other prompts with similar amounts of input and expected output tokens should perform in a similar manner. Longer prompts with increased amounts of input tokens may significantly increase the memory requirements of a given model – potentially taking one of the hardware-solutions beyond the capabilities of their respective memory buffers.

Each model has been set to run at the minimal temperature setting of 0, in order to reduce random variables as much as possible in an attempt to make the results easily replicable. Additionally, some data has been sourced through third-party benchmarks due to limited hardware availability, however these figures have been made sure to be reflective in terms of token counts and process demands, closely mirroring the compute demands of the main benchmark prompt. Aside from this, the models have been set for full GPU/CPU-execution, depending on which solution has been under review. Any other runtime settings have been left at their default settings, as set by the respective models and the LM Studio-environment.

5.1 Low-end model – (Gemma 2 2B Instruct Q4)

The inference performance benchmarking segment of this research begins with our low-end model. This tier is represented by Google’s Gemma 2 2B model in its Q4 format. With it’s instruct-variant, which has been optimized for executing and following prompted instructions, this model comes at a size of roughly 1.6GB. Because of its reasonable size, this model is expected to be possible to ran on relatively modest hardware, including CPUs, GPUs and other devices, even without significant dedicated high-speed memory pools.

These expectations are quite well replicated within practical findings, with even CPU-based computing delivering ample amounts of inference performance. Each of the tested solutions falls well above the established point of sufficient performance of 7 tok/s, regardless of the architecture and level of optimization on offer.

Gemma 2 2B Q4				Inference speed
Processor	CPU/GPU/SOC	Memory	Runtime	tok/s
AMD Ryzen Pro 4650U	CPU	16GB RAM	CPU	8 tok/s
Apple A16	SOC	8GB RAM	Metal	16 tok/s
AMD Ryzen 5800X	CPU	32GB RAM	CPU	17 tok/s
AMD Ryzen 7500F	CPU	32GB RAM	CPU	21 tok/s
Apple M1	SOC	8GB RAM	Metal	23 tok/s
NVIDIA RTX 5090	GPU	32GB VRAM	CUDA	110 tok/s
AMD Radeon 7900XT	GPU	20GB VRAM	ROCm	160 tok/s

Table 2: Gemma 2 2B Q4 inference performance metrics

These findings represent the wide variety of hardware, which can effectively run small models, such as Gemma 2 2B Q4. Many organizations will already have such devices deployed within their workflows, making LLM-adoption of this degree a matter of software solutions and know-how, rather than hardware availability. The success of this process becomes primarily the responsibility of the IT/IS-team of the organization. They must plan out the deployment and deployment of an easy-to-use software platforms, which users can use to access these models. These must also be capable of utilizing the computational capabilities of their hardware efficiently in order to avoid any performance issues. Well-designed and accessible solutions are particularly important for less tech-savvy individuals as a way of ensuring systems are properly adopted and don't face issues such as technological development resistance (Ali et al., 2016).

With a model of this size, there isn't much need for centralized solutions for executing the computing related needs of LLMs. Such models can be ran on devices which are otherwise still viable in the current grand-scheme of things, with the looming EoL (end-of-life) of Windows 10 effectively forcing organizations to deploy devices with CPUs based on relatively recent microarchitectures, due to the mandatory TPM 2.0 (trusted platform module) security feature of the new operating system. To further simplify, if an organizations hardware-solutions are capable of supporting Windows 11 and therefore not due upgrades regardless – odds are these solutions should also be capable of running low-end models like Gemma 2 2B Q4 at an acceptable level of performance.

Performance outside of the traditional computing ecosystem revolving around the Windows-platform is also solid. This includes mobile devices such as modern smartphones and tablets, with recent iPhones, iPads and others delivering acceptable performance with a model in the tier of Gemma 2B Q4. Even with limited memory pools of 8GB and less, these still manage to efficiently deal with this model, because its limited size easily fits within smaller memory buffers.

Additionally, modern mobile devices include plenty of computational capabilities through their CPUs, GPUs and dedicated AI-related hardware platforms, such as Apple's neural engine, which has been a stable within their silicon-division for some time now – offering accelerated AI performance for models of various tiers with the aid of the Metal and AI-optimized MPX APIs.

It can therefore be determined that if an organization is only targeting basic-levels of LLM-capabilities, be that through a model only capable of basic tasks requiring limited amounts of information, or a more specifically optimized and therefore similarly sized model, within the ballpark of a few gigabytes – specific hardware investments should not be required, rather existing

hardware solutions can most likely be relied upon. Efforts will still be needed towards making easy-to-use software platforms and user training to ensure a smooth deployment phase and the realization of practical benefits.

Additionally, if an organization chooses to personalize and optimize a model for its workflows, this process will require efforts both in terms of compute resources and human resources. These are aspects which can be avoided with the selection of a pre-made and -trained open-source model. However, optimized and tailored models can provide notable advantages, especially in scenarios where a limited amounts of predictable tasks will be performed – allowing even small models to respond to prompts regarding certain topics in a highly accurate manner (Yang et al., 2023).

Finally, it can be seen that higher-end solutions have absolutely no trouble working through such a model, with even mid-tier consumer grade GPUs reaching impressive inference performance results, way beyond the point of a humans ability to consume the outputs, with many solutions reaching speeds of hundreds of tokens per second. These numbers may even be limited due to other factors, such as storage speeds of modern SSDs (solid state disk). Such performance would only become practically beneficial in a scenario where data can automatically exported and utilized by additional bits of software.

5.2 Lower mid-range model – (LLaMA 3.1 8B Instruct Q4)

Moving on to a mid-tier model with a size of about 5GB, we have Meta AI’s LLaMA 3.1-model in its 8B Q4_K_M format. With its increased file size, the expectation is that many of the devices which could deal with the low-end model, will quickly deteriorate in terms of performance, as the compute- and memory-bus requirements have risen drastically. The expectation is also that with this model, typical CPUs and their parallelization capabilities are going to be seriously tested, making this the point where alternative types of microprocessors will begin to show their value, as these can offer drastically improved performance in LLM-related workflows, as was shown with the borderline ridiculous speeds seen with the low-end model.

LLaMA 3.1 8B Q4				Inference speed
Processor	CPU/GPU/SOC	Memory	Runtime	tok/s
Apple A16	SOC	6GB RAM	Metal	n/a
AMD Ryzen Pro 4650U	CPU	16GB RAM	CPU	3 tok/s
Apple M1	SOC	8GB RAM	Metal	5 tok/s
AMD Ryzen 5800X	CPU	32GB RAM	CPU	6 tok/s
AMD Ryzen 7500F	CPU	32GB RAM	CPU	8 tok/s
AMD Radeon 7900XT	GPU	20GB VRAM	ROCm	75 tok/s
NVIDIA RTX 4070 Ti	GPU	12GB VRAM	CUDA	82 tok/s
NVIDIA RTX 5090	GPU	32GB VRAM	CUDA	110 tok/s

Table 3: LLaMA 3.1 8B Q4 inference performance metrics

First and foremost, it’s immediately apparent that many of the lower-end devices which could run the low-end model quickly fall apart when it comes to performance with our mid-range model. Relying on typical RAM (random-access memory), and CPU-based computing has quickly reached its limitations – with a relatively modern 6-core 12-thread laptop processor only managing an inference speed of roughly 3 tokens/second, falling well short of the ideal targeted speed of 7 tokens/s, determined to be the point of sufficient performance. With such speeds, the usage-options of such models quickly deteriorate, as waiting for responses to repetitive inputs isn’t really a valid and efficient option anymore. Our Apple A16-based mobile device with 6GB of RAM was also no longer able to process the model, due to the fact it wasn’t able to allocate a sufficient amount of RAM for the model, hence why its results have been marked as non-applicable. Whilst this SOC

was actually faster than the AMD Ryzen Pro 4650U with the processing of the low-end model, due to the lack of memory it simply finds it impossible to run the next tier of model.

Whilst many mobile devices relying on typical CPU-based computing will struggle to deal with such models, some positives still arise. First and foremost, the new and improved architectures of cutting edge-mobile CPUs, should still allow for them to perform at an acceptable level – especially when considering the recent developments from chip-designers, including dedicated AI-related features such as NPUs directly on their designs, in an attempt to boost AI-related performance. The value of such additions can be seen immediately, when looking at how Apples M-series of silicon performs, with even the baseline Apple M1 SOC (system-on-chip) delivering almost double the inference performance of a hexa-core AMD Zen 2-based laptop, bringing it near acceptable levels of performance for a model of this tier.

Consequently, desktop equivalent CPUs from recent years also face little trouble reaching adequate levels of performance with models of this calibre, as these are typically higher powered and clocked variants of their mobile counterparts. Therefore, their performance tends to be slightly higher than the mobile variants of their eras. For instance, an octa-core AMD Zen 3-based based desktop CPU from 2020 can be seen producing roughly 6 tokens/s with the 8B-parameter model, bringing it quite close to the targeted rates. Newer chips utilizing newer microarchitectures are expected to easily break through that threshold, making them capable of running LLMs of this tier even without separate accelerators, such as GPUs or NPUs. Especially CPU-generations relying on high-speed DDR5 RAM-solutions are expected to perform well due to their increased memory bandwidth. This is also seen with the performance of the Zen 4-based system providing a 33% boost over the previous DDR4-based generation.

Moving on from the results of typical CPU/SOC-based findings, it is quickly apparent that most modern GPUs still face no trouble delivering exceptional performance measurements with our mid-range representative model – with the highly parallelized compute loads being easily dealt with by most dedicated GPUs of modern architectures released somewhat recently. The only issue which is expected to impact some lower-end GPUs is the lack of VRAM availability. This model can easily use 6GB of memory for inference-workloads, even with typical input context lengths. If the accelerator doesn't have access to sufficient amounts of dedicated memory, performance will quickly deteriorate due to the system having to rely on system-memory (RAM) as a fallback layer.

When sufficient VRAM is available, GPUs are expected to face little trouble dealing with such a model. With the way GPUs are typically designed and brought to the market, most GPUs with

adequate VRAM are also expected to offer sufficient compute-performance and bandwidth to offer excellent performance with such a model. Ideally, a GPU with 8GB or more of dedicated VRAM should be deployed to offer high inference-performance and some context-length headroom.

As a result of CPU-based computing aided by accelerators such as iGPUs and NPUs still offering decent performance, adopting a BYOD (bring your own device) based hardware structure remains a viable option for this tier of LLMs. To ensure compatibility and ample performance, certain base metrics and requirements may have to be set for user-devices, especially in terms of memory capacities and supported APIs. However, most devices based on recent microarchitectures should be viable options, as long as sufficient memory bandwidth and other important aspects are also taken care of properly. In terms of OS-support, devices across different solutions remain viable – including plenty of options across the Windows- and Mac-ecosystems, with the limiting factors potentially existing on a software optimization basis, rather than theoretical AI-performance capabilities. Accommodating several platforms and architectures may require extra efforts with the deployment and development of user software and platforms.

Whilst end-user-based computing is perhaps the easiest and most logical solution for this tier, a model in the tier of LLaMA 3.1 8B is also the beginning of the point where centralized solutions may begin to start becoming reasonable value-for-investment options. This is especially true for organizations which deploy aging hardware and don't have any other reasons for updating their devices. For such organizations, centralized solutions which can be accessed remotely may easily turn out to be the most financially viable option. This is especially true if utilization on a per-user basis is expected to be limited, making a simple centralized server capable of handling all requests without any backlog buildup or other issues. Deploying a singular high-performance server instead of updating large fleets of new devices can result in huge cost-savings in these situations.

5.3 Upper mid-range model – (Gemma 2 27B Instruct Q4)

Moving on to the next model, we have the 27B Q4 variant of Google’s Gemma 2 model. With its over tripling of parameters compared to our previous model, significantly increased hardware requirements are seen with this tier of model. Many of the devices that were able to run the previous model with sufficient performance quickly fall apart in terms of inference speed when attempting to deal with this upper mid-range model. This isn’t exactly surprising, considering the fact many of these devices don’t even include enough fast (video) random access memory (RAM/VRAM) to support a model of this size, so these are having to rely on vastly slower mass storage devices, such as SSDs (solid-state drives). The requirement of high-speed memory for this model is in the region of 20GB, significantly hurting the performance of any devices with inadequate amounts of high-speed memory. Due to its inability to deal with the previous model, the Apple A16 SOC has been entirely removed from the benchmark suite at this point.

Gemma 2 27B Q4				Inference speed
Processor	CPU/GPU/SOC	Memory	Runtime	tok/s
AMD Ryzen Pro 4650U	CPU	16GB RAM	CPU	n/a
Apple M1	SOC	8GB RAM	Metal	n/a
NVIDIA RTX 4070 Ti	GPU	12 GB VRAM	CUDA	n/a
AMD Ryzen 5800X	CPU	32GB RAM	CPU	2 tok/s
AMD Ryzen 7500F	CPU	32GB RAM	CPU	3 tok/s
AMD Radeon 7900XT	GPU	20GB VRAM	ROCm	16 tok/s
NVIDIA RTX 5090	GPU	32GB VRAM	CUDA	44 tok/s

Table 4: Gemma 2 27B Q4 inference performance metrics

With this tier of model, the capabilities of typical RAM-solutions appears to have been surpassed, with inference speeds dropping drastically below the target. For instance, a high-speed DDR5-based system with sufficient capacity and approximately 64GB/s of bandwidth is no longer able to offer adequate performance, falling to a speed of roughly 3 tok/s. Other CPU-based solutions with significantly higher bandwidth may be able to offer adequate bandwidth, but these will require wider memory subsystems or other more sophisticated solutions, something typically only available on workstation/enterprise grade platforms, such as AMDs Epyc, Intel’s Xeon or Apple’s high-end SOC-solutions. These are typically significantly more costly than consumer-grade GPUs, which can offer superior performance, as these can ship with memory bandwidths in excess of 1TB/s.

Based on these findings, and the performance of higher-end GPUs with sufficiently sized VRAM-buffers, these can be identified as an optimal solution in terms of cost-effectiveness. For instance, a single consumer-grade AMD RX 7900XT(X), or an NVIDIA RTX 30/40/5090 is able to offer solid performance of well over 10 tok/s with this model. As these offer at least 20GB of dedicated memory and high amounts of memory bandwidth, they perform very well with this type of a model. Simultaneously, these are typically available at the cost of roughly 1-2k EUR and can be deployed in most systems with an empty PCI-E expansion slot and adequate power supply – for instance as an addition to a pre-existing CPU-based server. Whilst these costs aren't too drastic, they do effectively mean that the adoption of such a system within an organization should in most cases be completed on a centralized basis. Acquiring such devices for each user of the LLM would quickly become a very costly option, especially if these bits of hardware aren't needed for other workloads regardless.

In practice the setup of such a system involves a centralized server containing at least the chosen hardware dedicated for LLM-processing. Employees can then access the capabilities of this server through a web-based portal or a dedicated bit of software. The role of the app or website is to send a user's prompt over to be processed and then receive and showcase the results returned by the server. With such a system, the end-user's device is relieved of the duty of processing the prompts. Especially within large organizations, such a system can be highly effective, as the wasted time of such a system is reduced drastically, with many users sporadically sending over tasks for processing. Due to the random nature of these prompts being sent over, the risk of queues is introduced into the process, which may introduce notable delays. This should be dealt with by introducing additional parallelization capabilities if deemed necessary.

Alternatively, such devices can also be accessed remotely from various service providers, with these offering access to such GPUs on a highly scalable and adjustable basis, for instance on a month-by-month or even per token basis. However, it must be noted that relying on an external hardware-as-a-service provider brings additional security concerns to the process, making this service provider and their data protection issues a potential security threat. However, especially within organizations where utilization and willingness to significant investments are low, relying on a flexible third-party can be a viable and cost-effective solution.

5.4 High-end model – (LLaMA 3.3 70B Instruct Q4)

Moving on to the final and largest model of this research, yet another big leap in terms of hardware requirements can be identified. Due to the sheer size of this model, the basic memory requirement of Meta AI's LLaMA 3.3 70B is more than double that of the previously reviewed Gemma 2 27B. As a direct consequence of this, the amount of capable hardware becomes significantly more limited. For instance, consumer-grade solutions such as NVIDIAs RTX 30/40/5090 and AMDs RX 7900-series become unviable, as their 20-32GB VRAM-solutions are proven to be inadequate. This goes to show that the capabilities of singular consumer-grade GPUs have been passed. At this point, multi-GPU nodes or data-centre level accelerators with higher amounts of memory have become requirements when reading-speed inference performance is targeted. The solutions which were unable to run the previous tier of model have been removed from the benchmarking suite of the high-end model.

LLaMA 3.3 70B Q4				Inference speed
Processor	CPU/GPU/SOC	Memory	Runtime	tok/s
AMD Ryzen 5800X	CPU	32GB RAM	CPU	n/a
AMD Ryzen 7500F	CPU	32GB RAM	CPU	n/a
AMD Radeon 7900XT	GPU	20GB VRAM	ROCm	1 tok/s
NVIDIA RTX 5090	GPU	32GB VRAM	CUDA	2 tok/s
Apple M3 Max	SOC	64GB RAM	Metal	7 tok/s
NVIDIA H100	GPU	80GB VRAM	CUDA	20 tok/s

Table 5: LLaMA 3.3 70B Q4 inference performance metrics

As mentioned, even the highest-end consumer grade GPU in the RTX 5090 is simply inadequate when it comes to running a model of this calibre. In order to achieve even the most basic level of functionality, it must rely on a significant amount of CPU/RAM offloading, resulting in a poor performance of only ≈ 1.5 tok/s. In practice, the memory requirement of this model in its 70B Q4 format, is roughly 50GB. This should all be directly accessible by the GPU or other accelerator in order to reach adequate performance, at least when the target is set at the 4-7 tok/s a human can consume by reading (BitNet, 2024). An example of an accelerator which can achieve adequate levels of performance with a model of this calibre would be NVIDIAs H100-accelerator with its 80GB dedicated VRAM-buffer. With this amount of dedicated memory, the whole model can be stored on device without having to rely on any additional storage. This allows a significant amount of IO-overhead to be removed from the process, reaching solid performance of 20 tok/s.

This is a great showcase of the importance of memory capacity and bandwidth in LLM-workloads, as the RTX 5090, which produced less than 10% of the performance, actually has a similar amount of raw theoretical AI-related compute capabilities (3958 TOPS/3352 TOPS¹, NVIDIA 2025a). These differences are direct consequences of their differing memory structures, as a model of this tier no longer fits into the 32GB VRAM-buffer, having to rely on system RAM-offloading in the process. Memory capacity and bandwidth are key for high-end LLM-models, and these are typically found from professional grade accelerators and devices. The prioritization of these has become an important way for vendors to differentiate 3D-graphics and AI-optimized hardware solutions.

Other alternatives to data-centre GPUs do however exist. Perhaps the most notable one of these would be the various APU/SOC-based options that exist in the market. For instance, Apple's M-series of chips scales up to hundreds of gigabytes of memory directly accessible by the GPU and can therefore offer the GPU the ability to run a model without having to rely on splitting the workloads with itself and the CPU. Because of this, the performance is significantly improved. For instance, when relying on Apples proprietary Metal API, the M3 Max-SOC can produce roughly 7 tok/s with this model, when configured with 64GB or more memory, reaching and surpassing the threshold set for acceptable performance. Whilst concrete benchmarks couldn't be secured in time for this research, the performance of AMD's recently launched Strix Halo SOCs is also supposedly going to land in a similar region of performance with LLaMA 3.3 70B, if the claims of the manufacturer are to be believed (Evangelho, 2025). Solutions like this, which put an emphasis on the memory count rather than raw compute capabilities have proven to be excellent options for AI-inference workloads, as they still include sufficient amounts of compute capabilities, in addition to ample amounts of fast memory – making them capable of fully supporting significantly larger models than comparably priced GPUs.

Due to the costs of these solutions, the point of centralized solutions being essentially mandatory has been reached. Even the most cost-effective solutions for such models come with significant hardware acquisition costs, which makes sharing these resources an important aspect of making such a system a viable function of a profitable business. These centralized solutions may be structured within a localized and self-controlled server or alternatively run on the premises of a service provider partner. Each of these has their advantages and disadvantages, primarily the trade-off between initial and running costs, as well as control and security aspects of the process.

¹ FP4 vs FP8, 2.3.6

5.5 Extreme high-end models

Whilst not a part of the actual benchmark suite, even larger models consisting of hundreds of billions of parameters do exist. Due to the size of these, they come with memory requirements of hundreds of gigabytes, limiting these to extremely high-end systems, requiring multiple data-centre level accelerators. This renders these effectively pointless for most organizations, as adopting these as an on-site solution would become ludicrously expensive. Such models are typically ran on the premises of the service provider, for instance AWS, Azure or Meta. It should also be noted that these are well past the point of drastic improvements compared to their smaller siblings, with for instance Meta only quoting a 3% improvement between their own 70B and 405B variants of their LLaMA 3.1 models, where the difference between their 8B and 70B-variants was nearly 20% (Meta AI, 2024b).

This is also the category where the most known and utilized commercial models typically fall into. For instance, the well-known ChatGPT supposedly relies on hundreds of billions of parameters. Whilst these figures aren't official and are highly dependent on the sub-model being utilized on such a service, it can demonstrate the expected capabilities and requirements that need to be reached, to make localized solutions truly a match for industry-leading services (Howarth, 2024). As previously noted, these models are also not particularly comparable to localized solutions when it comes to factors such as control and data-security, due to their strict limitations in terms of system functionality resources being available to the public.

Such models have their time and place, but even in these cases self-hosted and fully localized solutions are often not particularly viable. Rather, an organization whose personnel has limited needs for such models may achieve a significantly better value proposition by utilizing such a model as a limited time service through an external service provider, effectively renting these capabilities on a per/token or usage time-basis. To run such a model, one should expect to require an array of several ultra-high end enterprise accelerators with large memory buffers, resulting in high costs to reach solid levels of inference performance (Gowda et al, 2024).

6 Benchmarking model capabilities

With the completion of the hardware-based performance segment of this research, the focus will now move on towards benchmarking the capabilities of the models themselves, reviewing them for the quality of their outputs.

Within this chapter, the following LLMs will be benchmarked for their output quality.

- Gemma 2 2B Instruct Q4, Google (1.59GB)
- LLaMA 3.1 8B Instruct Q4, Meta AI (4.58GB)
- Gemma 2 27B Instruct Q4, Google (15.50GB)
- LLaMA 3.3 70B Instruct Q4, Meta AI (39.60GB)

These models have been analysed from an inference-speed point-of-view in the previous chapter, and therefore this chapter will focus entirely on the quality of outputs each of these can provide. The intention is to identify tasks which different tiers of models can offer solutions to, as well as to identify limitations of these models. On the basis of this and chapters 4 and 5, the ultimate plan is to be able to form an artifact for the design of localized LLM-systems based on the methodologies of design science.

These models have been chosen as references for how one may expect a model of a given tier of size to perform. Strengths and weaknesses will exist within given model-families, yet these should give a solid idea into what one may expect. It should be noted that the file sizes mentioned above are only representative of the size of the model itself, with practical requirements often significantly exceeding this figure – depending on factors such as input context length.

The respective outputs of these models will be reviewed on based upon varying key factors, depending on the prompts. The ultimate target of these prompts is to identify what types of tasks a model of a given tier can perform effectively, and what level of human capabilities and efforts are needed in the process.

The following prompts will be used for benchmarking the models.

- Make me a python script for a quiz about artificial intelligence
 - Performance reviewed on the basis of the code being functional, and the model being able to iteratively improve upon its initial creation on the basis of additional requests
- Solve for y in the equation $3x=6y$, if $x=28$
 - Performance reviewed on the basis of providing the correct answer and therefore being able to solve the issue. An additional more in-depth prompt will also be tested to further differentiate the models. (Solve for y in the equation $3x=6y-2x$, if $x=28$)
- Write a short story that is less than 500 words and includes the following:
 - A character obsessed with AI, but terrified of it taking over the world in the future
 - The story must be self-narrated by a second character who owns an AI-related company
 - At the end the world must be saved by a villain
 - Review of the respective stories will be performed on a preferential and manual review basis, complemented with an AI-aided review process in order to review the results on an as much of an objective basis as possible (DeepSeek R1 32B Q4)
- Write a summary of a given text: LLM Adoption Trends and Associated Risks (Schillaci 2024, 13th chapter)
 - Manual review complemented with an AI-based screening (DeepSeek R1 32B Q4)

Whilst this list isn't exhaustive and has no chance of reviewing all of the capabilities and weaknesses of these models, it aims to reflect a wide variety of different types of use-cases and therefore offer a solid basic understanding of the practical use cases of these models. Tasks such as these have been identified to be ones in which LLMs offer a lot of potential for practical benefits (Ahn et al., 2024; Etsenake & Nagappan, 2024; Gómez-Rodríguez & Williams, 2023; Qian et al., 2024).

These results will also be compared to third-party benchmarks and their metrics on how each of these models performs on a wider set of test applications. These benchmarks will be performed in an LM Studio-based environment with the CUDA API, with the temperature being set to 0 in order to seek consistency from the results. Other variables and settings have been left at their default options.

Due to the complicated nature of analysing creative writing, some of these processes have been aided by the usage of AI. This is in an attempt to find additional insights into the differences of the outputs of the reviewed models. Simultaneously, this will help to mitigate the risks of personal preferences impacting any findings. For any LLM-aided review processes, a model separate from the benchmark suite (DeepSeek R1) has been utilized. This is to ensure none of the benchmarked reference models are given an unfair advantage in the review process. The findings of AI have been strictly utilized as an aid rather than a basis for any findings, with the main focus remaining on manual review processes.

These reference models and other commonly used LLMs are also constantly being improved upon by their respective vendors. Because of this, in practice anyone attempting to implement such solutions should verify if newer versions of these models are available, as these can typically offer higher quality outputs without notable additional compute requirements. These results should therefore be seen as baseline minimums, as to how such models have performed in early 2025, up to date models should offer at least equal but likely improved performance. This is especially true when comparing models from the same vendors and families, such as the Meta AI's LLaMA or Google's Gemma.

In addition to iterative improvements, whole new models are also constantly being developed and released. Some of these have even brought along drastic leaps in performance, as has been the case with the recently launched DeepSeek-family of models. These are claiming to offer performance which is capable of competing with the top-tier of proprietary solutions from the likes of OpenAI, whilst relying on an open-source model. New models may for instance bring along new architectural solutions, as was the case with DeepSeek's models. These rely on a slightly altered structure, which allows them to offer significantly improved results at the cost of inference performance and computing requirements at a given size of model (Erdil, 2025).

Benchmarked
task

Model tier (Reference model)	Programming	Mathematics	Creative writing	Summarization and text-comprehension
Low-end model (Gemma 2 2B Q4)	Dysfunctional code, failure to resolve issues	Solid solution to the initial prompt, failure with a more advanced test	Failure to meet all requirements - a plot which could be slightly odd to follow. Grammatically okay.	Solid result, resulting in a highly usable summary of a longer text. Competitive with the tier above.
Lower mid-range model (LLaMA 3.1 8B Q4)	Functional code after some iteration, failure with more advanced requests	Flawless solutions to both the initial prompt, as well as the more advanced prompt.	Meets all requirements, more cohesive and enjoyable story compared to the low-end model, grammatically minor differences.	Very comparable result to the low-end model, slight decrease in quality if anything, Usable for pre-screening of texts, however some details missed.
Upper mid-range model (Gemma 2 27B Q4)	Immediately functional, successful advancements through iteration. Only minor hiccups through.	Flawless solutions to both the initial prompt, as well as the more advanced prompt.	Meets all requirements, improved plot and general language, resulting in a more engaging story. Grammatically excellent	Excellent summary, more of a balanced look into the original text compared to the smaller models, not getting caught up on any specific points, rather providing a general overview.
High-end model (LLaMA 3.3 70B Q4)	Immediately functional, successful with advancements adding more features in a smoother way compared to the previous model.	Flawless solutions to both the initial prompt, as well as the more advanced prompt.	Meets all requirement, excellent language and a generally engaging and easy-to-read story. Grammatically excellent.	Excellent summary, extremely similar to the findings of the upper-mid-range model. Provides and excellent overview into the contents of the original text.

Table 6: Summary of output quality benchmarks

6.1 Low-end model – (Gemma 2 2B Instruct Q4)

Starting off the model capability review process, we have Google’s Gemma 2-model in its 2B parameter variant, offering high levels of inference speed on a wide range of hardware. As found out in the previous segment of the research, such a model can run on a lot of hardware, not requiring dedicated accelerators or other expensive solutions for adoption. Unfortunately, this doesn’t come without its drawbacks, with the benchmark suite of prompts within this segment managing to expose several weaknesses from such a model working with a limited number of parameters, especially when these haven’t been focused or tailored for any specific tasks, rather functioning as a general-purpose model.

	Benchmarked task			
Model tier (Reference model)	Programming	Mathematics	Creative writing	Summarization and text-comprehension
Low-end model (Gemma 2 2B Q4)	Dysfunctional code, failure to resolve issues	Solid solution to the initial prompt, failure with a more advanced test	Failure to meet all requirements - a plot which could be slightly odd to follow. Grammatically okay.	Solid result, resulting in a highly usable summary of a longer text. Competitive with the tier above.

Table 7: Gemma 2 2B output quality benchmark results

The first benchmark prompt of this research focuses on a typical programming-related activity, with each model being given the task of giving the users a functional Python-based code for an AI-related quiz. When attempting to construct such a code, Gemma 2 within its 2B-parameter variant can produce a legitimate looking code, however it fails to make it functional. This is true even after it has been told that the code is not performing as it should and given the error codes provided by the utilized runtime platform. Even with this information and multiple chances for improvement and additional information on the issue, this model ultimately failed to resolve the issue. Considering the fact the request bit of code was quite simple; it quickly becomes clear that a general-purpose model of this tier is highly limited in aid of programming-based tasks and workloads. At best, a model of this calibre can provide rough sketches for the way programs should be written, requiring a lot of corrections and know-how to make these functional.

Such weaknesses highlight the limitations a model may face from the sheer lack of parameters it has to work with. Compared to many of the online-based LLMs users are accustomed to utilizing, such a model has significantly less resources at its disposal and therefore cannot get anywhere near the capabilities such a model can offer, especially when dealing with an issue that requires knowledge on intricate details. Programming is a great example of such a workload, as a model should understand the intricate details of a code, and the relations of different sub-programs. General-purpose models with limited parameters are particularly problematic with less utilized and represented programming languages (Joel et al., 2024). In this case, the issue was associated with the format of certain inputs, and the way different parts of the program interact with these factors. Whilst the model was able to find suitable basic inputs, due to the lack of information at its disposal, it was unable to set these up in a manner where they could have functioned together properly.

Outside of the programming-based task, positives can also be identified from the capabilities of such a model. For instance, Gemma 2 2B was able to provide a highly usable result for the benchmarked mathematics-based prompt, not only offering the correct result, but being able to complement this with a solid-step by step solution for the issue. This solution was highly comparable with the outputs of significantly higher end models, such as the almost 20 times larger LLaMA 3.3 70B. This helps to showcase the potential such a model can offer, within a niche or a field with a limited amount of variables, even a relatively small number of parameters can be enough to produce helpful results. An issue such as mathematical equation can be simple enough for even a limited size LLM to process effectively. However, it should be noted that with more complicated equations including more factors, issues could arise as the models' understanding of a higher number of factors may be limited. For instance, when slightly complicating the utilized formula, Gemma 2 2B began to produce incorrect results.

With the third test prompt, the target is to review the ability of a given model when it comes to creative processes and following associated instructions. In this case, Gemma 2 2B is given the task of creating a story based on a given set of rules. The model is told to include certain elements, seeking to test the model's ability of combining requirements and taking each of these into account. Whilst Gemma 2 2B is able to create a story suitable to many of the set aspects of the story, it falls short with some of these. For instance, the model was told the story should include an ending where the villain saves the world, however in this case this doesn't exactly take place. This highlights the limitations such a small model may face when attempting to combine several requests, some of which may even be partially contradictory. Grammatically even such a small model is able to

produce solid text, with the main issues existing within the flow and readability of the text. The outputs of such a model shouldn't be used as is, rather these could for instance prove to be a useful basis for a text that is going to be manually reviewed and improved upon.

Moving onto the fourth and final benchmark of the benchmark suite of this research, each model has been given the task of producing a shortened version of the 13th chapter of Schillaci's (2024) book, which has also been referenced in this research. These models were provided with this chapter in its entirety and given the task of summarizing it to the best of their ability, down to a limitation of 500 words. When reviewing how Gemma 2 2B performs in this task, clear positive signs can be spotted. Even such a small model is found to be capable of producing a useful summary of the original content. Whilst the result isn't perfect and does miss out on some key factors of the text, the result is still beneficial and provides a decent look into the contents of the full text. Somewhat surprisingly, Gemma 2B also holds up extremely well in comparison to the next model in our benchmark suite, offering a very comparably result to a drastically larger model – likely a consequence of the Gemma-family's suitability and optimization for this type of tasks. When reviewed with a third-party LLM (DeepSeek R1 32B, the result of Gemma 2 2B was even ranked to be better than the one of LLaMA 3.1 8B, further highlighting its competitiveness.

6.2 Lower mid-range model – (LLaMA 3.1 8B Instruct Q4)

Our next tier of model focuses on the performance of Meta AI’s LLaMA 3.1 8B, showcasing a commonly used example of a lower mid-range model. Models of this tier come with limited hardware requirements, and as seen by the previous segment, can still be ran on non-dedicated hardware solutions – making them highly accessible. This is because of their reasonable computational and memory requirements allow them to be efficiently processed by a range of hardware solutions. This chapter will contain the results of the output-quality benchmarks of this research.

Model tier (Reference model)	Benchmarked task			
	Programming	Mathematics	Creative writing	Summarization and text-comprehension
Lower mid- range model (LLaMA 3.1 8B Q4)	Functional code after some iteration, failure with more advanced requests	Flawless solutions to both the initial prompt, as well as the more advanced prompt.	Meets all requirements, more cohesive and enjoyable story compared to the low- end model, grammatically minor differences.	Very comparable result to the low-end model, slight decrease in quality if anything, Usable for pre- screening of texts, however some details missed.

Table 8: LLaMA 3.1 8B Q4 output quality benchmark results

With a model of this tier, improvements can immediately be spotted from the results. First and foremost, when looking at the prompt focusing on programming capabilities, the results are significantly better than with the Gemma 2 2B-model. The process isn’t perfect, with the initially provided code facing some issues. However, the model was able to correct these issues with a simple and rather vague description of the type of issue it was having, not even requiring exact error codes. Therefore, the model was able to produce a functional code with very limited user know-how and information, coming up with a functional program within a few minutes.

However, the limitations of its capabilities and knowledge became apparent soon when attempting to have the code improved and modified based on user request. The model proved to be unable to add features into to the program without breaking its core functionality in the process. Whilst the exact reasons behind these issues are tricky to pinpoint, it’s quite likely these issues are a result of the model attempting to combine tokens of software which aren’t directly compatible with each other. This is because the model doesn’t have the required knowledge on how these can interact with one another. Attempting to combine incompatible sub-programs results in conflicting bits of

code and a dysfunctional program. These issues are expected to only get more severe when the model is given the task of completing more complex programming-based tasks.

As could have been expected, like Gemma 2 2B, LLaMA 3.1 8B has no difficulty dealing with the initial math-based prompt, producing the correct answer. However, a difference can be identified quickly, when the formula is slightly complicated, by adding additional variables which must be transferred and taken care of in a proper manner. Where our low-end model failed to deal with this task properly, LLaMA 3.1 8B managed to deal with this in a correct manner, finding the correct solution to the given issue. With this, LLaMA 3.1 8B was able to complete the whole math-based benchmark suite. Whilst it won't be identified as a part of this research, it is likely that the limitations of our mid-range model would be found when testing with even more complicated mathematical formulas including a higher amount of variables. For these, a model with math-based optimizations, or a higher amount of general-purpose data might be required, as a model of this tier is eventually going to reach its limitations.

When given the task of creative writing on the basis of certain preset requirements with the third benchmark prompt, clear improvements can be seen with LLaMA 3.1 8B when compared to our entry-level model. First and foremost, LLaMA 3.1 8B manages to effectively include each of the requirements it was given, including a clear villain who saves the world, and pointing out other key factors which Gemma 2 2B struggled with. With the larger model a more cohesive story is produced, where the plot is easier to follow and understand. Grammatically the differences between the level of these texts are minor, highlighting the fact that even lower-end models are capable of producing grammatically solid text. In practice, a text produced by a model of this tier still requires manual review, however it is likely to require significantly less modifications.

With the final benchmark prompt of this research, LLaMA 3.1 8B was given the task of producing a summary of a chapter of Schillaci's (2024) book "On-site deployment of LLMs". This chapter of the book looks into the trends impacting LLM-adoption and risks that such developments bring along. Our lower mid-range model follows up on the impressive results of the earlier low-end model, also producing quite a solid summary of the whole text. Due to the limitations in amounts of text, some details were bound to be missed, but to summarize, the outcome still offers a decent brief explanation of what happens in the whole text, especially if the target is for instance to utilize an LLM to pre-screen the content before properly reading it. Somewhat disappointingly, the differences compared to the low-end model are extremely small, perhaps highlighting the Gemma-families comparative strength in such tasks.

6.3 Upper mid-range model – (Gemma 2 27B Instruct Q4)

Moving on to the upper mid-range model of this research, the difference in the sheer amount of parameters becomes very apparent. With Google’s Gemma 2 model now being utilized in its 27B Q4 format, rather than the initial 2B Q4, the results are quite impressive in many ways, being a drastic contrast to the first results of the Gemma 2-family.

Model tier (Reference model)	Benchmarked task			
	Programming	Mathematics	Creative writing	Summarization and text- comprehension
Upper mid- range model (Gemma 2 27B Q4)	Immediately functional, successful advancements through iteration. Only minor hiccups throughout.	Flawless solutions to both the initial prompt, as well as the more advanced prompt.	Meets all requirements, improved plot and general language, resulting in a more engaging story. Grammatically excellent	Excellent summary, more of a balanced look into the original text compared to the smaller models, not getting caught up on any specific points, rather providing a general overview.

Table 9: Gemma 2 27B Q4 output quality benchmark results

First up, when testing the model for its capabilities in programming related workflows, the model produces some impressive results. Gemma 2 27B offers significant improvements when compared to both of the lower end models which have been reviewed so far. The most notable aspect is the fact Gemma 2 27B can immediately produce a functional bit of software based on the initial prompt, with only a minor cosmetic hiccup existing in the initial program. This didn’t impact the functionality of the code in a drastic manner, only resulting in an increased risk of a user miss-inputs when utilizing the program. When this was mentioned to the model, it was also able to fix this issue with no additional issues arising from the process – resulting in a fully functional bit of code, solving the issue which it was given with the initial prompt.

When tested for further capabilities through slight additions, the model was also able to make suitable adjustments to the code without any unforeseen side effects arising from these changes. Whilst the previously tested smaller models couldn’t handle such changes, this one was able to understand the whole bit of software to an adequate level to even make additions to the initial code. Aside from the initial minor hiccup, the process was near enough perfect, and definitely one which

could offer serious productivity benefits for anyone utilizing it in such workflows, especially when utilizing one of the most commonly used languages such as the tested Python.

With the second benchmark prompt focusing on a math-based issue, it's clear to see that with our upper mid-range model the capabilities have likely ascended beyond the requirements of the pre-set benchmark prompts of this research. As was the case with our mid-range model, Gemma 2 27B also faces no trouble dealing with either the basic or more advanced mathematical formulas tested. To properly showcase the potential of Gemma 2 27B, a more advanced formula with an increased amount of variables would be needed. This also goes to show the importance of sheer parameter counts between models, as when compared to the same model in its 2B-parameter format, the capabilities have clearly been improved, with the additional parameters allowing the larger model to solve the advanced prompt with no issues.

The third benchmark prompt focuses on creative writing, and here the drastic increase in parameters produces solid results when compared to our low-end and lower mid-range models. When given a prompt requesting a story on a set of given requirements, Gemma 2 in its 27B-parameter format produces drastic improvements when compared to its smaller 2B-variant, as well as the mid-range LLaMA 3.1 8B-model. Gemma 2 27B is able to take care of each of the requirements that were provided to it, whilst also providing a more engaging and suspenseful plot. This is mostly achieved with the inclusion of more descriptive language and word selection, something that is likely achieved due to the fact this model has significantly more information about the relevance and relationships between these words through its increased understandings of the tokens it has been given by the initial prompt.

Following up on creative writing and text creation, with our final benchmark-prompt Gemma 2 27B was given the task of producing a summary of the chapter "LLM adoption trends and associated risks" from Zachary Schillaci's (2024) book on the topic. The results are solid, as this model is able to keep up with the earlier results, producing a solid result, highlighting many of the key factors that were mentioned in the original text, whilst respecting the pre-set character limitations. In contrast to the previously reviewed smaller models, Gemma 2 27B manages to produce a highly balanced approach, not viewing the topic from a certain point-of-view, rather focusing on a wide range of topics and points mentioned in the original text. Interestingly, the difference between this and the high-end model (LLaMA 3.3 70B) is extremely small, mostly being left down to a matter of preference, perhaps once again a result of the Gemma-family's optimization for such tasks.

6.4 High-end model – (LLaMA 3.3 70B Instruct Q4)

The high-end model of this research, Meta AI’s LLaMA 3.3 70B offers the highest level of output quality, offering the most consistent and reliable results – requiring the least manual improvements and changes. This is mostly due to the fact it has access to significantly more parameters and therefore data compared to any of the other models. Simultaneously, it is also one of the most modern models tested, being for instance newer than its 8B-parameter sibling which hasn’t received the same degree of iterative updates.

Benchmarked task				
Model tier (Reference model)	Programming	Mathematics	Creative writing	Summarization and text-comprehension
High-end model (LLaMA 3.3 70B Q4)	Immediately functional, successful with advancements adding more features in a smoother way compared to the previous model.	Flawless solutions to both the initial prompt, as well as the more advanced prompt.	Meets all requirements, excellent language and a generally engaging and easy-to-read story. Grammatically excellent.	Excellent summary, extremely similar to the findings of the upper-mid-range model. Provides an excellent overview into the contents of the original text.

Table 10: LLaMA 3.3 70B Q4 output quality benchmark results

These advantages can be immediately spotted in action when reviewing the results of the first benchmark prompt of this research, which focuses on a given model’s ability when it comes to performing a relatively simple programming-based task. The code which this model produces is immediately functional and respects each rule that had been set as a guideline within the initial prompt. Whilst some of the smaller models required some help to get the program to a functional state, no such actions were necessary with this model, as it was able to understand the relations between different sub-programs and their requirements in a manner which allowed for an immediately functional program.

When given the challenge of modifying the program from the first prompt to include more details and give the user a more educational experience, LLaMA 3.3 70B still faced no trouble, as it managed to iteratively improve the earlier program, whilst adding the requested features - fully complying with the requirements in the process. This can be identified as a difference even when compared to our upper mid-range model (Gemma 2 27B), with the larger model producing a less troublesome experience. Whilst both models were ultimately able to perform both of the programming-based tasks, the larger model allowed for a more straightforward experience, not requiring any additional information during the process.

Like its smaller comparison subjects, LLaMA 3.3 70B is able to deal with the math-based prompts tested within this research with little trouble, finding the correct solutions to both prompts with no trouble. This is hardly surprising, considering the fact even its little 8B-parameter sibling was able to perform this exact task. It is likely that these could be differentiated by utilizing even more complicated formulas, however it's important to note that such requirements may be beyond the scope of most workflows, as math-based tasks that are more complicated than these should probably be performed with more specifically optimized solutions, such as symbolic calculators. General-purpose LLMs may be a bit of an awkward solution to these regardless, as they don't tend to support the entry of math-specific symbols and operators.

When comparing the creative capabilities of these models with the third review prompt, it's clear to see that with this model the results are approaching the point of diminishing returns. Whilst with the first few models the improvements from one to the next were very clear to see and the preferences between the texts weren't left down to personal preference, this review process and the identification of differences has become significantly more difficult with our two largest models tested. Each of these can produce a well-written and structured story, with little to criticize, and the differentiation of these being left largely down to personal preferences between the options.

Even when reviewed with the aid of a third-party AI-model (DeepSeek R1), it also struggles to pinpoint any drastic improvements, however ultimately suggesting that the results from the high-end model are slightly better. An interesting take-away from these results is also the drastic difference between the outputs of two models of the same family and similar design principles, with the 8B and 70B-variants of Meta AI's LLaMA 3-family producing drastically differing outputs from an equalized prompt, although some factors such as names are repeated between the differing plots. A similar trend could also be seen when comparing the Gemma 2-family's siblings.

Finally, the last benchmark-prompt of this research gives LLaMA 3.3 70B the task of producing a summary of the chapter “LLM adoption trends and associated risks” from Zachary Schillaci’s (2024) book on the topic. As expected, based on the results of the earlier smaller models, the results are good, as LLaMA 3.3 70B is able to produce an excellent summary, providing a comprehensive look into a wide range of related aspects. In this task, the differences between the upper-echelon of models were minimal, as the results of LLaMA 3.3 70B are very comparable to the results that our upper mid-range model with significantly reduced hardware needs was able to produce.

6.5 Model comparison based on third-party findings

Before moving on towards the conclusions of the research, the performance and capabilities of these models will be further investigated with the aid of a third-party benchmark-suite. This has given additional insights into how such models perform in different tasks, as many types of input prompts were not able to be researched with their dedicated prompts within this chapter. These metrics will showcase a wider variety of performance, as these rankings and scores and are based on extensive research and tests of many open-source models, including each of the reference models of this research. These findings are published by a reputable source, and their reliability has been ensured by their publisher through strict verification protocols. Whilst these results are based on higher quantizations of the respective models, each of these has been scaled down in a similar manner (FP16 to Q4_K_M), ensuring a similar advantage compared to their reviewed distillations.

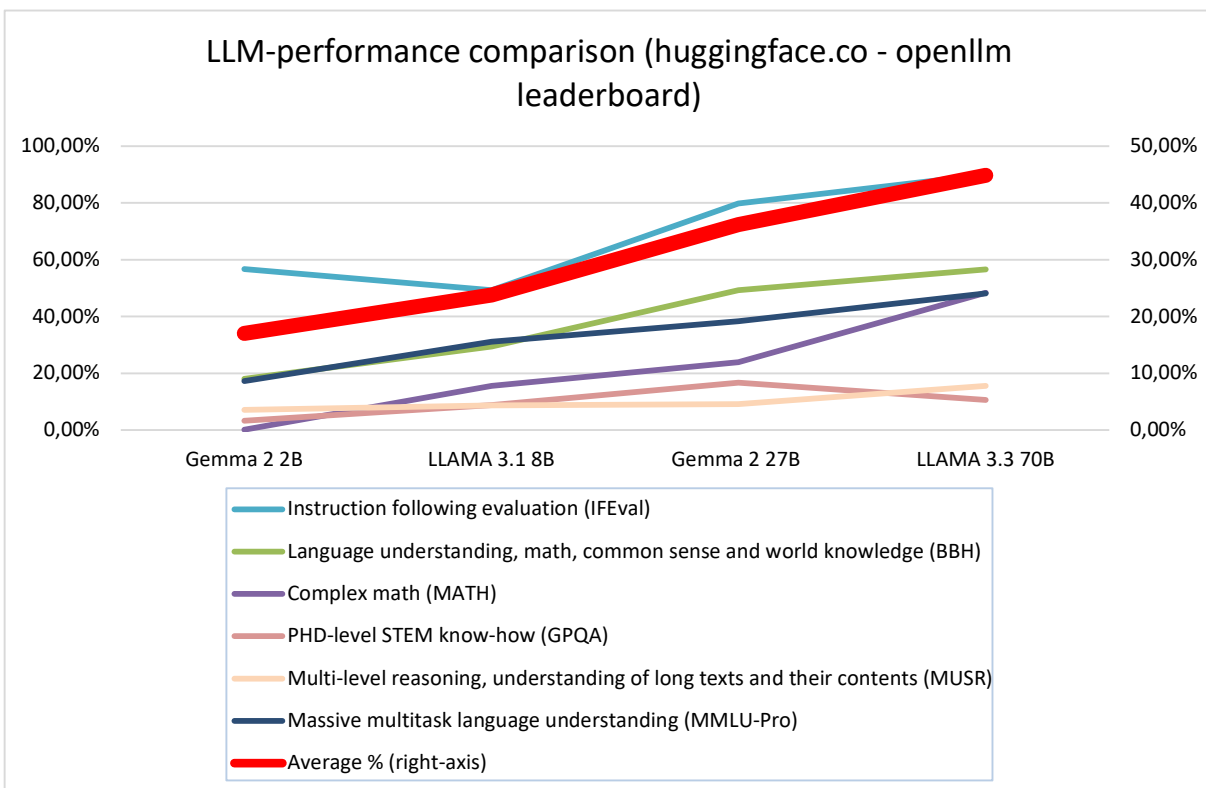


Figure 1: LLM-performance metrics (huggingface.co, 2025a)

When looking into the rankings and benchmarks made by Hugging Face, it's clear to see that the larger models of this benchmark suite can offer clear advantages with the average performance improving at nearly a linear rate. Whilst the limited review prompts of this research were able to be saturated by even some of the lower-end models, making the situation seem like diminishing returns were being approached upon – these findings clearly showcase that to not be the case.

These models continue to show improved results, when subjected to more demanding inputs. These allow for the larger models to take full advantage of the additional available tokens and therefore information the larger models have access to. Any organization seeking to adapt such a model should therefore consider the depth of information they want to expect from an adapted model. Whilst commonly needed and relatively simple input prompts may struggle to showcase the differences between tiers of models, niche prompts requiring in-depth knowledge are likely to benefit significantly from higher-end models. This is especially true when discussing general-purpose models which haven't been optimized for these tasks.

The realm of diminishing returns that was found with the benchmark prompts of this chapter, can also be seen in some metrics of Hugging Face's benchmark suite. For instance, the IFEval and BBH-benchmark suites do not appear to be benefitting drastically from the additional parameters when approaching the upper echelon of models, rather the improvements are slowing down quite drastically. These focus on tasks such as explicit instruction following and various amounts of general knowledge and understanding (huggingface.co, 2025b). This showcases the complexity of the correlation between different tasks and their reached advantages from increased parameter counts.

One interesting takeaway from this benchmark suite is also the differences in capabilities between the LLaMA and Gemma-families of models. Whilst the averages between these two appear to perform in quite a linear manner, improving quite consistently with additional information – plenty of outliers exist within the respective sub-benchmark. For instance, the Gemma-family of models does very well in reasoning and instruction following-based tasks, with significantly smaller models being competitive with their LLaMA-counterparts. Simultaneously, math-based tasks appear to be a clear stronghold for the LLaMA-family, with above average leaps seen within these tasks. These differences highlight the fact that even general-purpose models have differing strengths and weaknesses, which should be considered in the model selection phase. This especially true if prioritized workloads can be identified on an organizational basis.

These findings showcasing strengths and weaknesses are great demonstrations of the potential that optimized models can offer. Whilst both of these model families are primarily general-purpose designed models, different priorities and focuses within their training data has still produced notable differences in their capabilities. If an organization is seeking to adopt a local LLM for a specific purpose, or a limited amount of purposes, the selection or creation of a tailored model can be extremely helpful. Such optimizations can allow a model of a certain size to punch above its weight when compared to similar-sized general-purpose models to an ever-higher degree than the one that can be seen between these models.

7 Suggested solutions

With the findings of the previous chapters, this chapter will seek to combine these in order to make suggestions for how an organization should seek to adapt localized LLMs within its infrastructure. Each of the researched and tested LLM-tiers will be discussed separately, offering insights into how an organization may wish to approach setting each of them up, as well as what it may expect to achieve from giving its employees access to such a model.

Each of the following tiers will be discussed within their respective segments, based on their chosen reference models.

- Low-end model (2GB) – reference model: Gemma 2 2B Q4
- Lower mid-range model (6GB) – reference model: LLaMA 3.1 8B Q4
- Upper mid-range model (20GB) – reference model: Gemma 2 27B Q4
- High-end model (50GB) – reference model: LLaMA 3.3 70B Q4

Each of these tiers are based on the findings made with the earlier sample models, which should showcase a representative example of what a general-purpose LLM of such a size can effectively perform. Better results could be found from relying on purpose-built and optimized models, but these fall outside the scope of this research. These should however be taken into account by anyone attempting to deploy a model of such tier, especially when a specific use-case can be identified before deployment. The given memory tiers are to be used as guidelines and can differ drastically. This is especially true if long input-context lengths are to be utilized, for instance when processing long and complex input prompts.

Within this research, the benchmarks and findings are based upon widely adopted open-source models, which are commonly utilized and offer a solid basic understanding of how one may expect a general-purpose model of a given tier to perform. Organizations may choose to begin utilizing one of these, develop one of their own, or something in-between. This decision will have significant impacts on the development and setup costs of a given model, as it will determine if and what sort of a training phase will be needed. Once the training phase has been completed, even custom models of similar architectures and memory requirements should perform in a similar manner to the benchmarked examples of this study. This allows for the results of this research to serve as rough guidelines even for customized solutions and their post-training adoption phases and requirements.

Any training loads that may become mandatory due to organizational customizations will require their specific training workloads. These aspects will not be reviewed in further detail during this research, as these can vary dramatically depending on the methods and targeted changes these seek to achieve to the initial model (Chung et al., 2025).

This chapter will go through these tiers of models on a one-by-one basis, showcasing their capabilities and the associated requirements, making suggestions for hardware structures that may be viable for such deployments in the process. It should be noted that these suggestions have been made based on solutions which were available or known to be coming soon as of early 2025. As is the nature of computing devices, quick deployments and improvements are to be expected to follow in the coming years. Because of this, the recommendation is to pay the most attention towards the memory requirements and types of accelerators mentioned within these chapters, rather than specific GPUs and other solutions, as these may have been replaced by improved and more attainable options at lower price points.

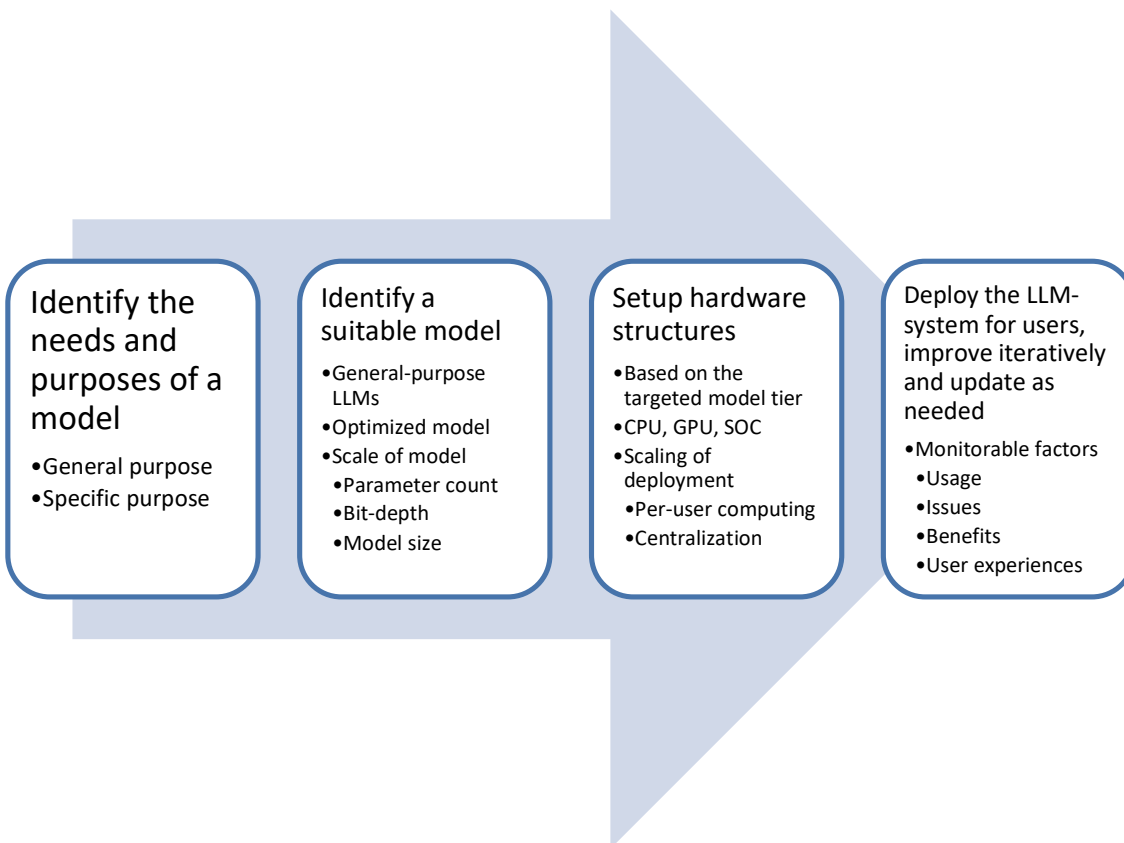


Figure 2: Artifact containing key factors to be considered and selected during an LLM-deployment process

The process of adopting an LLM as a part of an organization's workflows should begin by identifying the practical needs and benefits an LLM can look to serve. On this basis, it becomes a viable option, only if it can be deemed to offer genuine benefits and appetite from its potential users. Once the practical utilization and the targeted capabilities have been identified, the selection phase of products and solutions may actually begin. At this point, the financial capabilities and willingness to invest should also be looked into, as these aspects can put significant limitations on the types of models that may be put into use. As lots of different solutions with differing strengths and weaknesses exist, it is particularly important to ensure that practical targets for a solution exist, as this will make the identification and selection of the most suitable model possible.

When a clear plan for targeted workloads and utilization has been identified, alongside any cost/other factor-based limitations – an organization and the responsible personnel may begin to make selections about a model it will target. This phase should include practical tests, seeing how a model can succeed with demonstrative prompts if possible – however the limitations of hardware availability pre-deployment may prove to be a challenge. Depending on the choices made, the process may advance towards a self-made or an off-the shelf model – depending on the priorities and targets set within the earlier phases.

Once a targeted tier and type of model has been chosen, an organization may begin the phase of planning out its related hardware infrastructure. Depending on the tier of model, this may require next to no additional investment, or drastic amounts of new IT-infrastructure – with models scaling all the way from running on older office-laptops, to requiring cutting-edge multi-node GPU-accelerated setups.

Once the models have been chosen and prepared, and suitable hardware has been made available at sufficient scale, a system such as this may begin to be deployed to be utilized by employees in practical workflows. Especially in the early phases, it remains crucial that the functionality and reliability of the system is monitored properly, identifying any issues that may be caused by lack of familiarity or with the system itself. Depending on the way a system is deployed, some employees are also likely to require some extra support and training to familiarise themselves with the process of effectively utilizing new systems, especially when deploying the technology within less technology-intensive industries. Facilitating as easy of an adaptation phase as possible is crucial to ensure the new system gets properly utilized, and doesn't face issues such as user resistance, which has been identified to be a notable threat of any new IT-systems and their successful deployment (Ali et al., 2016).

7.1 Low-end (\approx 2GB-model)

Starting off with our suggested solution tiers, we have a low-end model-based structure. In this case the findings will be based on the earlier results found from benchmarking Google's Gemma 2 2B Q4-model. Many of the findings will also be closely representative of other models of this class. Some commonly used alternatives within this class include the likes of IBM's Granite 3.2 2B and Microsoft's Phi-2, with each of these having their own strengths and weaknesses.

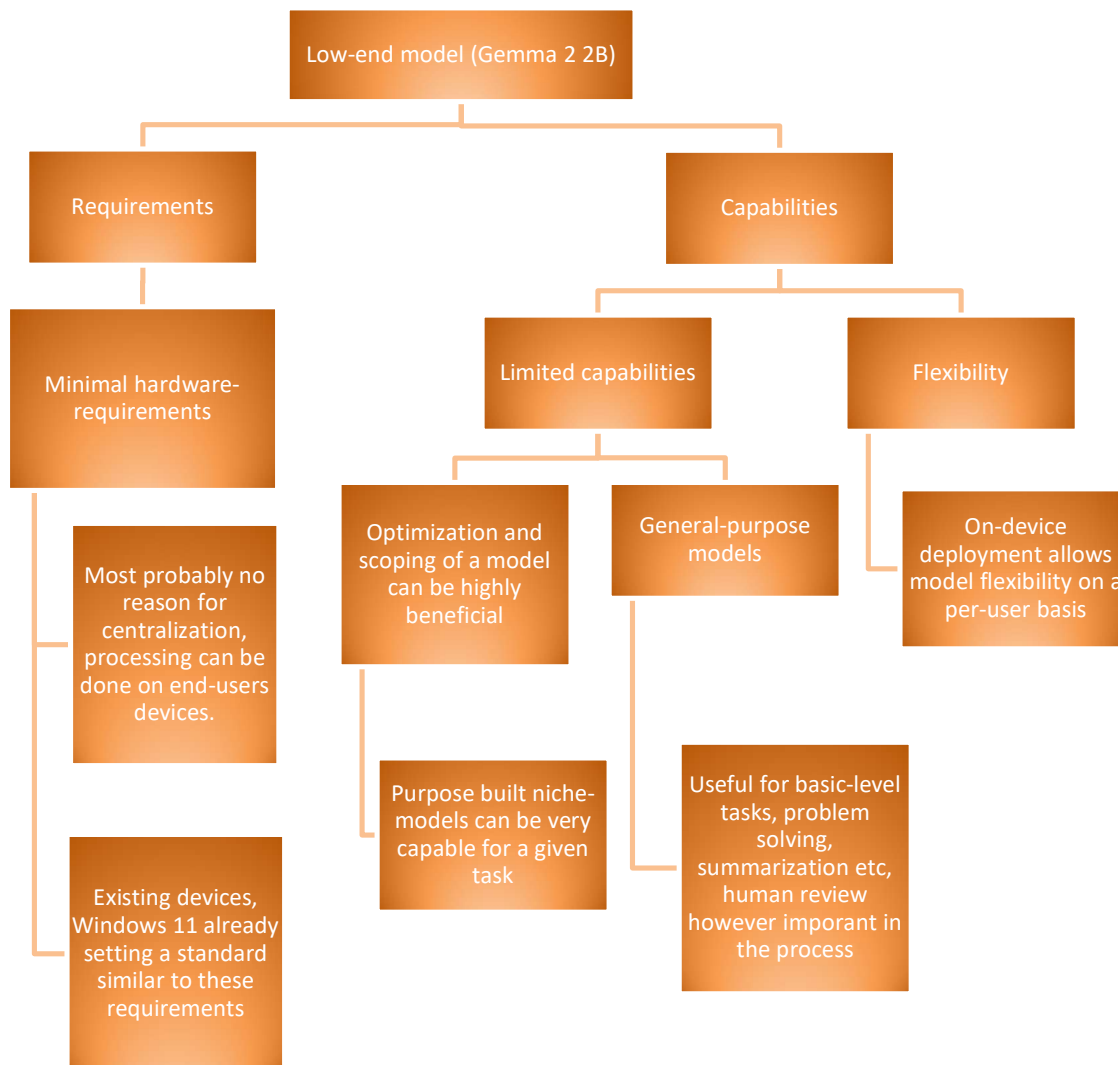


Figure 2: Factors behind the adaptation of a low-end model

First and foremost, it must be noted that the capabilities of such models are limited, however there are still benefits that can be achieved by making such a model available to employees. For instance, basic-level problem solving can be done effectively even by such a small model, as was seen with

the benchmark prompts of chapter number 5. Whilst the experience wasn't always perfect, it could still be a viable asset, especially when utilized by an individual who has enough knowledge on the topic in question to pick out any potential issues before they get the chance of impacting anything in a negative manner. When the parameters and therefore topic-based knowledge of a model is limited, the value of human corrections and monitoring becomes particularly important.

Tasks such as creative processes, and assistance in reviewing and editing human-created texts are also realistic possibilities for such a small model, as was demonstrated with our review model. Whilst the results weren't perfect, the output was grammatically reasonably good. As a baseline and with the aid of some human review and improvements, such text could easily result in an output that could be further utilized for various contexts, without obviously standing out as AI-generated due to poor writing quality or other factors.

From the point-of-view of an organization, adapting a low-end model such as Gemma 2 2B, can be a good option for trialling local LLM-deployments. Deploying such a model provides IT-staff and end-users of a model a chance to review the benefits of such a system. If such a deployment is deemed useful and to provide its users with practical benefits, this also opens the door for future changes and the deployment of larger models and more sophisticated hardware.

The benefit of adapting a system like this stems mostly from its low threshold for hardware requirements. As shown by the acceptable performance on devices such as phones, and aging general-use laptops, such a model could be launched to be run on the existing devices of most organizations falling within typical update cycles of five years or so (Graglia, 2025). If the hardware infrastructure of an organization is partially or fully deemed to be incapable of supporting such a basic model, these devices may have been overdue upgrade in the near future anyways, because of the looming ending of support for the Windows 10 OS (operating system), which is what these devices are expected to be based upon.

Because of the capabilities of even older hardware solutions, the benefit of deploying a low-end model such as Gemma 2 2B is the fact that an organization may not need to invest a penny into new hardware. If investment into hardware is needed to deploy such a model, there is a solid chance these investments would have had to take place for other reasons regardless. Some human resources will be needed, and these obviously have a value/opportunity cost, but the fact even such a model can offer notable benefits make these quite a valuable prospect. Even minimal efforts can give the personnel of an organization a significant workflow efficiency upgrade. This can become especially

true, when the users are given a proper introduction and training into the process, including factors such as input prompt design and optimization.

Such a model can also be launched through a simple app-based platform, which makes it easily accessible even for individuals who aren't particularly tech-savvy. These applications can be ran fully locally, and an intermediate datacentre/server shouldn't be needed in the process, as fairly basic devices are capable of running this model independently. Whilst centralized options are typically solid options for LLM-processing, such systems aren't really needed with a model possessing such low hardware requirements. Plenty of open-source applications for running such models exist, which can be used as-is, or further optimized to fit an organizations requirements.

Because of the minimal level of centralization, and the fact each end user's device should hold the capabilities of running a model of this tier, the door is opened for a high-degree of workload-based optimizations. For instance, personnel with differing tasks may benefit from different models, and have no reason to rely on the same one, as several open-source solutions exist – which can easily be deployed wherever they are found to be applicable (Park et al., 2024). This is a benefit which even the higher-end systems cannot necessarily offer, as their more centralized designs make such differences significantly more difficult to facilitate – because of the fact that individual end-user devices may no longer hold the capabilities of running such a models. These may also be utilized in conjunction with one another, where fully localized smaller niche-models are utilized alongside a centralized larger general-purpose model.

To summarize, the adaptation of a low-end model, like Gemma 2 2B can be a great entry point for an organization to start utilizing LLMs in aid of its personnel's workflows. Whilst these models aren't perfect, and their results are likely to require human corrections and improvements, their value comes from the minimal amount of effort an organization needs to put on into use, especially from the financial point-of-view. When existing devices are expected to be able to take care of the issue, the costs will be limited to ones resulting from the implementation phase itself, as well as support that may be required to get the system into utilization properly. This in itself makes these an extremely cost-effective way to adapt a localized LLM-strategy, giving workers a way to use these systems properly, not having to rely on self-found shadow IT solutions with their own security issues. This can also be completed as a temporary measure, for instance with the aim of trailing interest towards utilizing large language models.

7.2 Lower mid-range (≈6GB-model)

Moving on to the next tier of localized LLM-systems, we have a lower mid-range solution requiring roughly 6GB of memory, in this case being represented by Meta AI's LLaMA 3.1 8B in its Q4_K_M format. Other commonly used models in this tier include the likes of Google's Gemma 2 9B- and Alibaba Cloud's Qwen 2.5 7B-variants. These fall into a similar tier of performance and requirements, but will include differing strengths and weaknesses, dependent on the focused areas of their respective weighing- and training-processes.

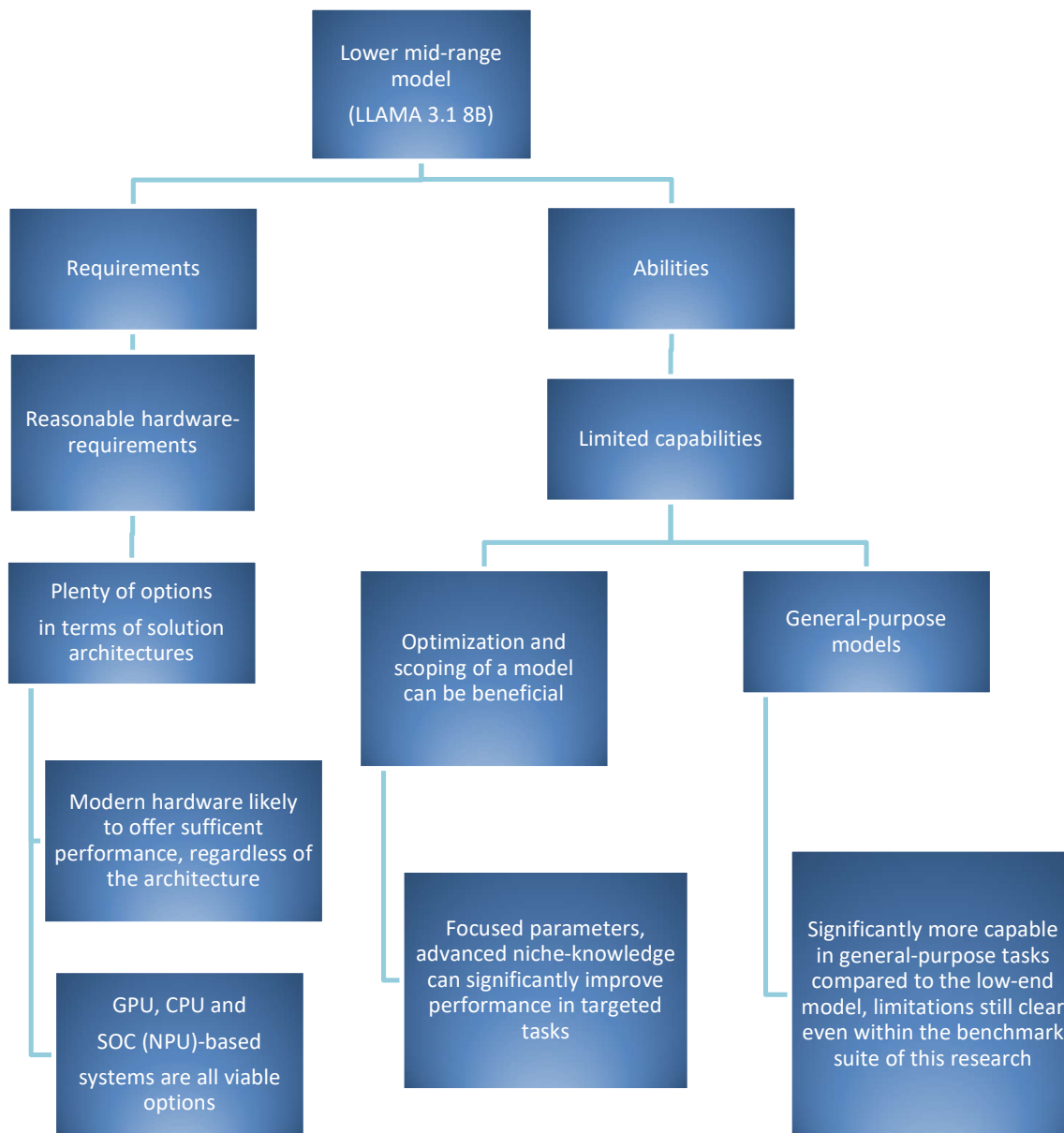


Figure 3: Factors behind the adaptation of a lower mid-range model

With our lower mid-range model, the threshold of being a driving factor to hardware changes has been reached, as devices capable of other typical tasks at an effective speed are no longer remotely guaranteed to be able to deal with a model of this size. If an organization is seeking to adapt a model of this tier, it should be prepared to have to invest into solutions for the sake of giving its employees access to such a model, unless other factors have already driven the given organization into giving its personnel access to high-performance devices well beyond the requirements of typical office tasks. Devices featuring modern high-end mobile SOCs with strong iGPUs and NPUs can offer excellent performance with a model of this tier, but these are only found from quite recently released devices. Desktop-grade solutions of modern architectures are also capable options, especially ones relying on the most recent DDR5-memory solutions and their increased bandwidth.

Depending on the size of the organization in question, the point at which centralized solutions may begin to be viable has also been reached. This varies based on many aspects of the equation, such as expected utilization per user and the size of an organization. If the expected per-user utilization is low in a large organization, a centralized server which workers can access and utilize in an easy manner through an application of some sort may become a cost-effective solution. This is due to the fact that updating large fleets of end-user machinery into higher-end laptops and other devices can quickly become most costly than setting up an organizational server, which can for instance rely on a single consumer-grade GPU to deal with the processing of a model. Because of the nature of GPU-architecture and its ML-suitability, this type of a system can offer high levels of performance even in scenarios where many users are attempting to utilize the system near-simultaneously. Based on the earlier performance metrics, for example a single AMD Radeon 7900XT, or a different GPU with similar performance characteristics could theoretically deal with 10 different prompts at an average speed of 7 tok/s, even if these were entered simultaneously, which is highly unlikely to happen in the real world, unless the user count of the system is extremely high.

Even lower-end GPUs are viable choices when seeking to deploy such a model on a local system. For instance, mid-range consumer grade GPUs have been shipping with 8GB or more of VRAM for a decent amount of time and can offer more than ample amounts of computational resources. Another cost-effective option which has showcased its potential is Apple's M-series SOC equipped machinery, perhaps most notably their Mac Mini, which can offer an all-in-one solution with solid performance at an attractive price point.

Due to centralized solutions becoming viable, externally hosted solutions may also become sensible options with a model of this calibre. This is especially true when an organization is reluctant to commit to such a system without being able to see its practical benefits. GPUs such as the previously mentioned are rented out by various cloud-based service providers, offering options on a per/token and time-basis, giving flexibility to organizations which may want to trial such a solution.

It should be noted, that when relying on an external service provider for model processing, the organization's data security structures also become reliant on these aspects having been properly taken care of by the respective service providers. As data security was identified to be a key cause driving the adoption of localized LLMs, this limitation may be problematic for many organizations, as it will re-introduce some of the service provider-based LLM-related weaknesses localized solutions attempted to remove in the first place.

When adapting a model of this tier, an organization can expect notable improvements in performance, when comparing to the previously reviewed low-end structure. As a consequence of the increase in available information the model has to work with, it's seen as a significantly more capable aid in tasks such as programming, math and other more complicated topics. Whilst even the low-end model is likely to be able to answer to questions on the basic-level of many topics, a lower mid-range model like LLaMA 3.1 8B is expected to be able to provide answers with more context and important factors, whilst also being more reliable for the end user, requiring less manual verification before prompt outputs can be utilized in practice.

To summarize, in order to begin utilizing a lower mid-range model, such as Meta AI's LLaMA 3.1 8B, an organization is going to have to go through significantly more effort, when compared to the previously mentioned low-end model. Most organizations are likely to not have pre-existing fleets of devices which can deal with such models, leaving two practical options to pursue. A centralized and remotely accessible solution, or a drastic investment into new devices for end-users, allowing each of these to deal with a model of this tier independently. Especially in larger organizations, the latter may prove to be extremely costly.

7.3 Upper mid-range model (≈20GB-model)

The next segment of this chapter focuses on our upper mid-range model and its respective requirements. In this research, this tier of performance was demonstrated using Google’s Gemma 2 27B-model. Models of this tier require at least 20GB of memory, alongside decent levels of computing capabilities. The exact figures are slightly dependent on the specifics of a given model and the targeted input context lengths. Whilst the benchmark runs have been run on the Gemma 2 27B-model, similar findings are also expected to arise when utilizing other commonly adopted models, such as Mistral Small Instruct (24B) or DeepSeek R1 Distill (32B). Most of the commonly utilized modern models of this tier perform quite similarly according to advanced third-party benchmark (huggingface.co, 2025a), with each having their own strengths and weaknesses.

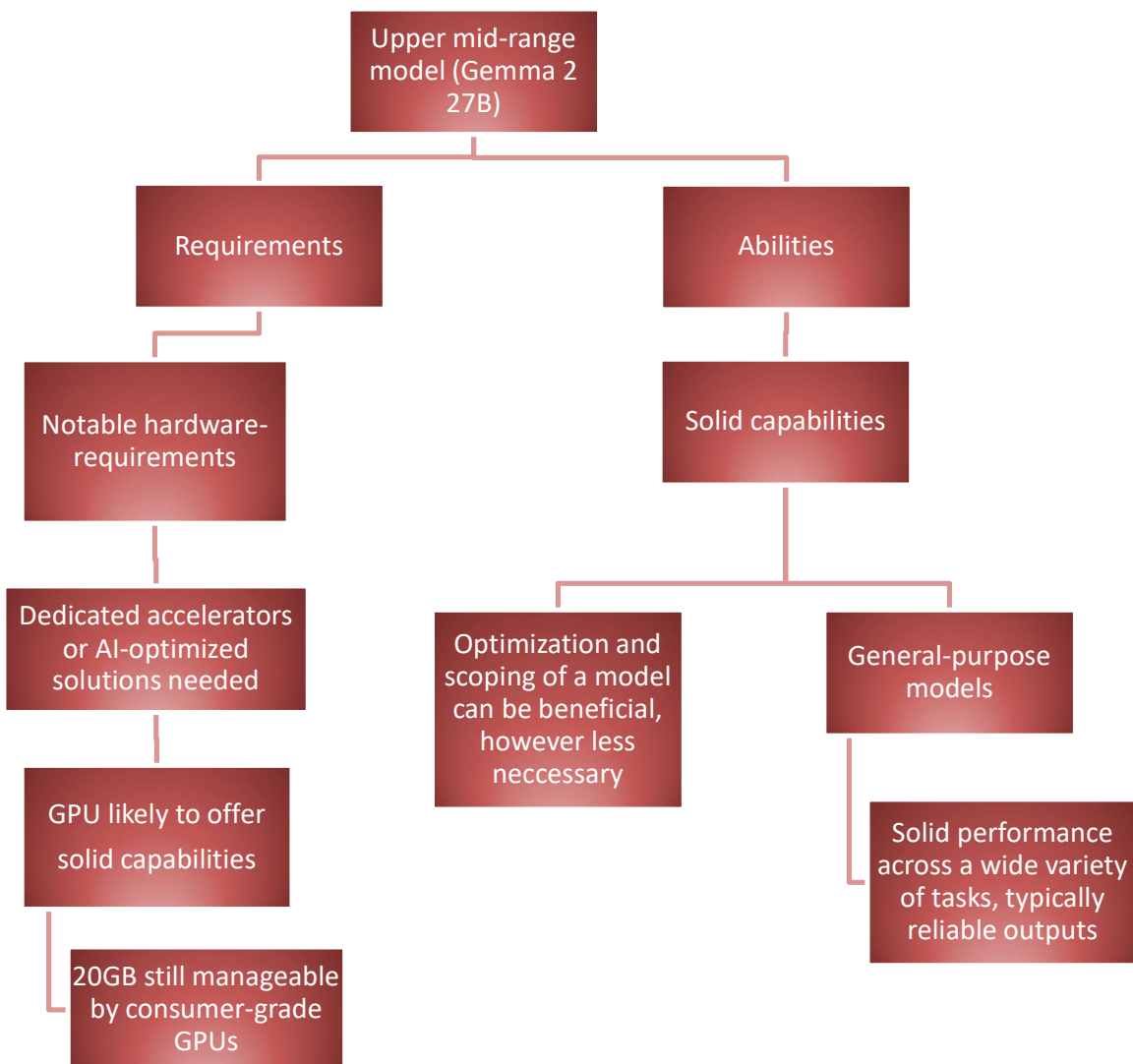


Figure 4: Factors behind the adaptation of an upper mid-range model

An upper mid-range model can offer solid all-around performance, performing notably better than our lower mid-range model, especially when looking at the synthetic benchmarks from Hugging Face. These showcase a notable improvement in terms of capabilities, and factors suggesting this were also identified within the benchmark prompt of this research. Whilst the smaller models were already capable of many of the tested tasks, Gemma 2 27B was able to perform them significantly more reliably, providing consistently correct and functional outputs. For instance, the smaller models struggled with the programming-based tasks given to it, but Gemma 2 27B was able to deal with it in a near flawless manner, even managing to iteratively improve upon its initial result after a suggestive prompt. For anyone attempting to make a programming-related workflow more efficient, a model of this tier is a significantly more valuable than the previous model.

Due to the extended capabilities of such a model even in its general-purpose “trim”, models of this size are also beginning to approach the point of optimized and tailored models being less beneficial, unless extreme edge-cases and niche tasks are targeted. For instance, the previously mentioned programming may be a task in which tailored models may continue to be viable and beneficial options, as complex bits of code may require the understanding of extremely intricate details of relations and aspects of software.

With this tier of model, the capabilities of typical end-user devices have become inadequate for inference workloads, including even devices which technically fit AI-related standards, such as Microsoft’s CoPilot+. Only the most advanced SOCs may have the required compute and memory capabilities to run these models, and adopting these to a large amount of personnel isn’t going to be practical for most organizations, unless there are other needs for such hardware capabilities. For most organizations, centralized solutions are going to be the most viable solution when a model of this tier is targeted. This may be achieved in a few ways, including relying on a cloud-based service provider, or self-hosting a solution which has been optimized for LLM-processing and offers sufficient amounts of memory and compute capabilities.

Perhaps the easier and least risky solution is to rely on an external service provider, such as AWS, which can offer easy access to a wide range of models, which can be hosted on their servers at a running cost consisting of renting out accelerator time and other fees. These are typically highly scalable and can be extended or shrunk based upon practically identified utilization. Although it must be noted, that with such a solution an organization will have to rely on their chosen service provider with their data, removing a part of the benefit of localized LLMs. As security aspects were

identified as an absolute key reason towards localized AI-solutions, this may be a deal-breaker for some organizations.

To achieve true control of the whole system, an organization may wish to choose to deploy its own server to answer to the needs of these AI models. If this option is chosen, it can be performed with a relatively simple and cost-effective hardware structure. Adequate performance may for example be reached with a singular modern GPU, as long as it can offer sufficient amounts of memory, roughly 20GB or more. Such GPUs include high-end consumer-grade options such as the RTX 30/40/5090, or the Radeon 7900XT(X). Any GPU chosen will also have to be deployed within a system that can offer it sufficient amounts of storage and CPU resources, although these don't significantly impact the performance of such a model, once it's been loaded onto a GPU and its dedicated memory buffer.

Alternatively, such a model can also be adopted on a localized solution of a simpler nature, such as an SOC with a strong GPU/NPU and at least 24GB of memory. Such solutions are somewhat limited but do exist – for instance in the previously mentioned Strix Halo-lineup from AMD, as well as Apple's higher-end M-series of chips. The costs of adapting such a model remain quite reasonable, the pure hardware acquisition costs of such a solution fall in the ballpark of 1-2k€, depending on the choices made as well as regional differences in availability. Cost-wise these are comparable to a solution relying on a dedicated GPU but offer a more simplified all-in-one solution at the cost of some inference speed when compared to a more powerful dGPU-accelerated solution.

To summarize, an upper mid-range model such as Gemma 2 27B can offer notable improvements compared to its lower-end alternatives, especially when taking into account the reliability factor of its outputs. Whilst the previous models can be useful assets, these rely on the end-user verifying and improving the outputs before utilization, whilst these upper mid-range models are approaching the point where they can consistently provide reliable and solid outputs to varying prompts and scenarios. The acquisition costs also remain reasonable, although they are getting to the point where it's important such a system has adequate amounts of practical utilization, as these costs are no longer ones which would have been necessary to make in the near future regardless. Any organization seeking to begin utilizing a model of this tier should expect to see notable benefits in its day-to-day operations.

7.4 High-end model (≈50GB-model)

As the final model of the research, we have our high-end model and its respective hardware structure options. Models of this tier require significantly more memory than the previous tier, approximately 40-50GBs, alongside high levels of computing capabilities. The benchmark runs of this research have been run on Meta AI's LLaMA 3.3-model in its Q4_K_M format. This model represents the capabilities of a typical model of this calibre, as it is one of the most utilized models of this class. Similar findings are also expected to arise when utilizing other commonly adopted models of this tier. These include the likes of the much-discussed and hyped DeepSeek R1 in its Distill 70B-variant. Different models of this tier will offer different strengths and weaknesses, but their average results and user experiences in general-purpose scenarios are expected to be comparable.

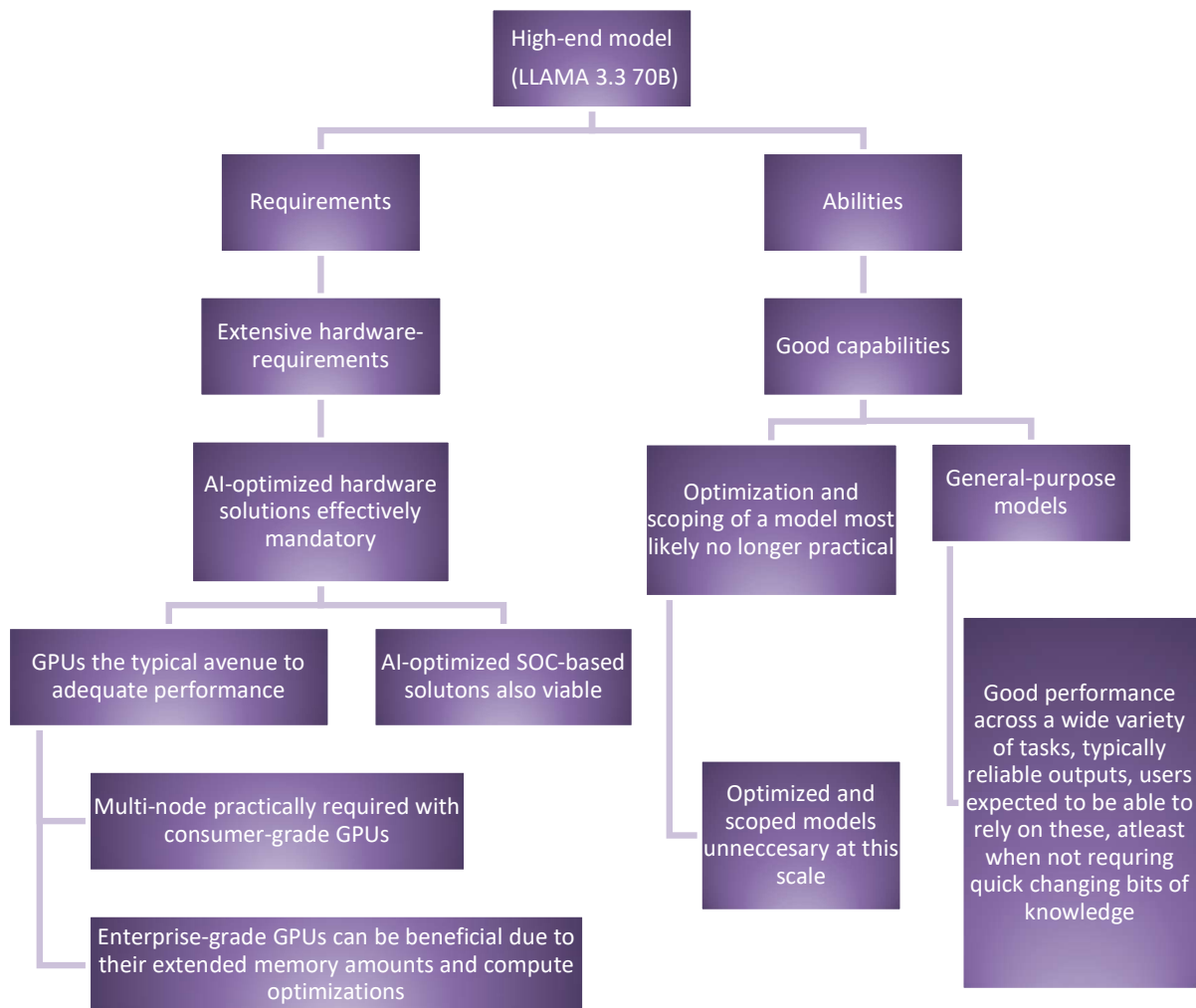


Figure 5: Factors behind the adaptation of a high-end model

A high-end model like LLaMA 3.3 70B has notable hardware requirements, but in return a model of this calibre provides some notable benefits. This is especially true when the utilization of such a model requires niche bits of knowledge and facts, something that would likely be beyond the knowledge of a smaller general-knowledge model. With a model of this size, general-purpose models have become the norm, as the need for specialization and limiting training datasets is a lot less important, as knowledge and data about various topics can easily be fit into the 70B parameter set. With a specific-niche targeted model, a parameter count this high may prove to offer limited advantages, unless extremely niche and detailed information on a given topic is required.

Within the benchmark suite of this research, the answers of our high-end model were quite comparable to the ones of the upper mid-range model, although slightly more polished and conclusive. However, it was clear to see that with the 4 benchmark prompts of this study and use-cases similar to these, a user will see limited benefits from adopting a model of this tier in comparison to the previously benchmarked upper-mid range solution. This is typical for LLMs, as after a certain threshold, the advantages brought on by the additional parameters tend to hit the region of diminishing returns, with these no longer being able to notably improve the experience, at least when it comes to less advanced input prompts.

However, it must be noted that to claim that improvements between these tiers do not exist, would be an oversimplification. These differences do exist when the models are given more niche and specific tasks, requiring significantly more advanced knowledge on details of various topics. Whilst this wasn't easy to spot from the benchmark-suite of this review, looking at a third-party and their significantly more in-depth benchmarks, the theoretical advantages are clear to see – with LLaMA 3.3 70B performing exceptionally well across the large third-party benchmark-suite (huggingface.co, 2025a). Topics such as complex math equations and massive multitasking and language understanding processes continued to see notable improvements all the way from the bottom to the top of our model-suite (huggingface.co, 2025a).

To summarize, high-end models such as LLaMA 3.3 70B, and others even surpassing its size will continue to be able to offer better outputs. In order to keep finding these to be of importance and therefore worth investing into, any organization seeking to do this within their systems should ensure the fact their operations and typical queries can actually benefit from these improvements to avoid overspending and wasting resources. If the additional depth of information isn't particularly beneficial, at this point the added parameters and their compute requirements could even end up

giving employees a worse LLM-experience due to the reduced inference speeds of this tier of a model, especially on comparable hardware.

If these factors have been taken into account, and an organization has determined that adopting such a model is the correct course of action and can offer practical advantages, a few types of potential hardware solutions and architectures exist for the task. As a consequence of the memory requirements of high-end models such as our reference model, these come with a significant rise in terms of set-up costs when compared to their smaller equivalents. Due to their large size, exceeding 50GB, consumer grade accelerators are mostly rendered useless, as these must rely on fallback layers, such as system memory and even mass storage devices. At the very least, consumer-grade GPUs must be set up as multi-GPU nodes, where their respective memory counts can be pooled together. Such solutions bring along quite a bit of extra technical complications as well as additional points of failure when compared to other more tailored and simplified solutions.

Prosumer/data centre-optimized solutions are typically drastically more effective and come with significant benefits that are of crucial value in the process of running the most demanding of LLMs. Perhaps the most important one of these is the typically increased memory capacities, with singular GPU-based accelerators reaching capacities of hundreds of gigabytes in the top-range of solutions. Whilst solutions of that calibre aren't quite needed in order to run a model like LLaMA 3.3 70B or others of similar sizes, the memory requirement of roughly 50GB can certainly benefit from memory pools beyond those of consumer-grade accelerators.

Instead, an ideal solution would be to rely on a VRAM-optimized accelerator, such as NVIDIA's H100 80GB. As mentioned, LLMs can also be ran across several accelerators with limited amounts of overhead, making the combination of two or more consumer-grade accelerators such as RTX 5090s a viable option, potentially offering slight cost reductions but introducing additional complexity in the process. Besides from these GPU-based solutions, an SOC-based system with 50GB or more fast memory accessible by the whole chip is a capable option for this type of model. As was the case with GPUs, these models may also be parallelized across multiple machines and their respective SOCs, however this yet again does introduce more potential issues and may still result in lacklustre tok/s performance, especially without proper optimizations.

With requirements of this magnitude, centralized solutions become essentially mandatory, as any singular individual would struggle to be able to get anywhere near recouping the costs of such a system if utilizing it on a lone basis. The tok/s production rate becomes an especially crucial factor, if such a system is going to be accessed and utilized by several people in a sporadic manner. With a

model so large and with such significant setup costs, it's likely that it will be adopted and utilized by a large pool of people. Because of this, some "extra" inference speed is likely to prove beneficial, as this will help a system not build up queues for processing, instead keeping the solution more user-friendly and seamless.

Due to the initial setup costs of such a system, a more service-oriented setup may also be a reasonable option for a lot of organizations. Whilst relying on a pay-as-go service may be more costly in the long run, it gives any organization without full certainty over their requirements and practical needs the chance to review the viability of different options, without having to commit into a long-term project with high sunk costs. The larger investments into self-ran and owned hardware may also be made after an initial trial run on an external service providers facilities.

7.5 Scalability and other organizational factors

When adapting a localized model to be utilized by a large pool of people, this should also be taken into account by building a system capable of dealing with several parallelized loads. Whilst a GPU may for instance be able to process prompts at 10 tok/s, this process is assuming the fact prompts are only introduced one at a time. When an organization has a large set of personnel or systems utilizing a model and its hardware, this may significantly reduce the available inference performance, by introducing factors such as queues and other delays into the process. If a system builds up a backlog of prompts, the situation may only get worse as prompts may keep on being added onto the pile, without them being able to be processed in due course.

This is why an organization should ensure that when deploying a localized LLM which relies on centralized computing resources, these have been scaled in accordance with the amount of concurrent users such a system is expected to have. Such scaling may for instance be completed by forming a pool of computing devices, such as GPUs, CPUs or SOCs – in a way where these can dynamically take care of workflows from the pool of requested prompts – effectively multiplying the capability of such a system. In an ideal scenario, with the aid of modern technology this may even be performed in a manner where data is spilled over from a GPU to system memory without the inference performance loss of typical situations (Xu et al., 2024).

These issues shouldn't impact systems which rely on end-user device-based computing, however these will also leave a lot of hardware investment and existing computational resources unutilized, as these can only be accessed by their respective users on these very devices. Simultaneously, this type of approach can offer organizations more flexibility, for instance each user equipped with an

LLM-ready device can run different models, allowing for instance for these to be more suitable for specific niche-workloads, rather than needing to find a one-fits-all-solution.

Organizations are also likely to wish to pay attention to other factors in regard to the models and solutions they choose to deploy. As previously mentioned, some LLMs have been under scrutiny for potentially questionable manners of acquiring data, something which certain organizations may be unwilling to look past (Knibbs, 2025). Additionally, due to their energy-intensive nature, factors such as training emissions and the running computational demands may be key factors for certain organizations who consider climate aspects and sustainability important parts of their operations (Hoffmann et al., 2022).

When it comes to the functionality of the models themselves, these can also differ a lot in terms of the transparency, data integrity amongst other factors. Whilst full control of these factors can only be achieved with a self-made and designed model, there are notable differences between the way makers of models tend to publicize these bits of information, with some taking the open-sourceness of a model more seriously than others. As previously mentioned, models such as Meta AI's LLaMA family are often spoken about as open-weight models, rather than open-source options due to the limitations in regard to information about their training process and the data utilized during this phase.

8 Summary

This research has demonstrated many crucial aspects an organization should consider, when planning a project around the deployment of localized LLMs. As seen during the previous chapters, these are currently at the heart of many AI-related IT-projects and can offer significant advantages to many organizations. However, under the umbrella of localized LLM fall many different tiers of solutions, which include and come with significant trade-offs, be that in terms of costs, capabilities as well as many other factors. Within this sub-chapter, these will be briefly summarized, offering a final look into the capabilities of different tiers of models.

Model tier	Memory requirement	Suggested solution	Computing structure suggestion	Model capabilities and use-cases	Benefits
Low-end model	2GB	End-user device-based processing	Fully localized	Basic level tasks, tailored models for niche uses	Security Flexibility
Lower mid-range model	6GB	Modern end-user devices, centralized GPU/SOC-based solution	Localized and organization controller, computational needs may be centralized or not	Basic-level general-purpose tasks, optimized models offer huge potential	Security
Upper mid-range model	20GB	High-end consumer-grade GPU or SOC-based device	Centralization	Wide range of capabilities and knowledge	Security Scalability
High-end model	50GB	Enterprise grade GPU (50GB+), multi-node consumer-grade accelerators	Centralization	Most complex capabilities and niche-knowledge	Security Scalability

Table 11: Summary of LLM tiers and related solutions

When adapting a low-end model, an organization can expect to give its employees access to a basic-level of generative AI/LLM-capability. Such a model will be useful as a helpful tool for suggestions, and quick scans of various texts, amongst other basic tasks. However, the results are unlikely to be usable in their initial format, requiring significant amounts of user efforts and editing

to utilize properly. The huge benefit of such a model is the fact it requires minimal investment into hardware solutions, as most currently deployed devices will already be capable of running such a model locally, even including devices such as smartphones and other mobile devices. The main requirement for an organization to deploy such a system therefore falls on the deployment of an easy-to-use platform and model, especially if proprietary non-open-sources options are selected. Additionally, a model of this tier can have solid levels of capabilities, when it has been optimized and limited for specific tasks that are expected to be needed the most, rather than targeting a general-purpose capable model (Yang et al., 2023).

If a low-end model such as Gemma 2 2B is deemed inadequate, the next most attainable option for an organization is to deploy a model of the so-called lower mid-range. Such models typically come in at a size of 5-8GB and require significantly more hardware capabilities than the low-end models did. For such models, end-device based computing is still viable, but requires powerful and modern devices, ideally ones with AI-optimized NPUs, GPUs or other similar accelerators. Alternatively, with a lower mid-range model centralized solutions may also begin to appear attractive, especially if an organization has deployed dated hardware that would require to be drastically changed for the sake of deploying an LLM. The benefits of jumping to a mid-range model are also quite clear to see, with results becoming significantly more consistent and therefore reliable. However, plenty of issues still arise, especially when utilizing more sophisticated and detailed prompts requiring knowledge a model of this tier doesn't have enough of.

The next tier of model is that of an upper mid-range model, again resulting in a big leap in terms of hardware requirements, as well as prompt quality. End-user based computing has been passed as a reasonable option within most organizations, and centralized solutions relying on AI-optimized hardware have become essentially mandatory, be that in the shape of a third-party service contract or self-hosted solutions. Notable improvements can still be seen from the capabilities of these models, as the answers seem to contain more intricate details and better abilities to tackle complex correlations etc. properly. The outputs have also become significantly more consistent, approaching the point where users may begin to put more trust on the quality of these results even for future utilizations.

The final tier of models within this study, is that of so-called high-end models, consisting of models requiring in excess of 50 GB of memory alongside significant computational capabilities. Such models, offer the most reliable outputs within this research – with the tested outputs being near flawless and ready to be utilized in their initial form, not requiring any human intervention. These

are also expected to shine to a further degree, when subjected to more complicated topics requiring more knowledge on specific topics. To deploy such a model, a significant amount of investment into compute resources is required, with high-end GPU(s) or other accelerators being required to offer decent user experience in the aspect of inference speed. Such solutions include enterprise-grade GPUs, multi consumer-grade GPU-nodes and SOC-based systems.

An organisation may choose to approach each of these tiers with differing model solutions – varying all the way from pre-made open-source models, to proprietary self-developed and trained models. As both of these have their advantages and disadvantages, it becomes the responsibility of a given organization and its related teams to figure out the most practical of solutions – whether to prioritize factors such as shift deployment and limited/non-existing training requirements, or full-control and optimizations of the capability of their models.

Especially in the cases of the higher-end solutions, an organization also faces the important decision of fully self-hosting or relying on a third-party service provider, to host the model of their choice. Both of these have their pros and cons – notably the likes of running vs initial setup-costs, as well as security and control aspects of the process. These are factors which should be carefully considered when planning to execute such a project. These decisions will also significantly impact the risks that come along with such an investment, externally ran solutions can provide significantly lower sunken costs from the beginning, if a solution is for instance deemed unnecessary soon after its deployment.

To conclude, the adapting localized LLMs offers significant potential for organizations of many types and industries but requires a properly managed and planned due process in order reach an optimal solution. The selection of tiers of models, as well as specific solutions can only be done properly with sufficient research and resulting knowledge on its targeted usage. With that, the waste of resources into unnecessary hardware-investments can also be avoided, improving the value proposition of any process. Such a process will also go a long way towards ensuring that such a system has a practical need, as for example interviewing workers and analysing their workflows to identify any need for aid which can be provided by LLMs, will also give an idea into their respective will and desire to utilize such solutions, limiting the risk of factors such as change resistance impacting the success of a given AI-related project.

9 Discussion and future relevance

When the benefits of a solution are as obvious as they are in the case of LLMs, be that in their localized format or not – it is expected that more and more organizations are going to target these types of projects in the coming years (Gartner, 2023). Advantages in terms of security, workflow efficiency and improved productivity can all be reached when LLMs are utilized properly. The findings of this research offer insights into how such projects targeting to offer such perks can take place, and what factors should be dealt with to ensure proper alignment between technical solutions and human needs. Within this research, solutions have been researched through a selection of reference solutions, but it's important to note the fact that any models or hardware solutions are to be used as guidelines, rather than direct solutions. As is the nature of technology and its quick deployment, newer and more optimized solutions should be utilized wherever applicable.

For instance, if an organization was to adapt an upper mid-range model, due to the recent release the third iteration of the Gemma-family, it should most likely choose the Gemma 3 27B-model, rather than the referenced second variant of this research. Whilst not a part of this study, newer versions typically offer improved performance, as is shown to be the case with the Gemma family (Google, 2025). As models continue to improve, the performance of lower tiers is expected to catch and eventually leap the tiers above, whilst the higher tier-models will continue to reach currently uncharted tiers of performance.

With new hardware and improved options expected to continue entering the market, the key focus for LLM-inference performance should be to ensure that any solution has enough compute capabilities, perhaps most importantly – an adequate memory solution, both in terms of capacity and bandwidth. In the future, as the market for AI-solutions continues to grow and develop, we may also see the rise of further optimized solutions, such as TPUs (tensor processing unit) and customized AI-based FPGA's take a more prominent role in this space. CPU, SOC and GPU-vendors are also expected to continue bringing out new architectures on a near-annual basis, many of which have begun including more and more AI-optimized architectural aspects during the recent times and the so-called AI-boom (AMD, 2025; NVIDIA, 2025a).

As new solutions will continue to enter the market – this research should be utilized as a guideline and a basis for a planning process. During the research and its appendices, information about the entire testing process has been provided in order to serve as an artifact for anyone seeking to deploy such solutions. These are also flexible and can be adjusted based on organizational priorities and

needs. For instance, if programming isn't expected to be a key point of utilization for a model, it's benchmarking results are also not particularly important for such an organization – rather other prompts and their results should be prioritized as seen suitable. The methodologies of this research may be utilized as is or updated in accordance with organizational needs in aid of reviewing and planning any localized LLM-deployment projects.

In the future, any practical suggestions made within this research may quickly become dated, hence why the key asset which has been targeted and created during this study is the research process and finalized design science artifact (Figure 6) – which showcases the process of planning out a localized LLM-system and its deployment within an organization. This includes the different phases of such projects, including the identification of needs and benefits targeted from a model, the seeking of suitable models and hardware, their practical deployment and optimization and ultimately ensuring successful adaptation through the training of personnel and the provision of adequate support through means which have been deemed suitable. Additionally, any specific solutions discussed within this research shouldn't necessarily targeted as is, as new models and variants are constantly bringing notable leaps which can often be utilized without significant additional hardware costs.

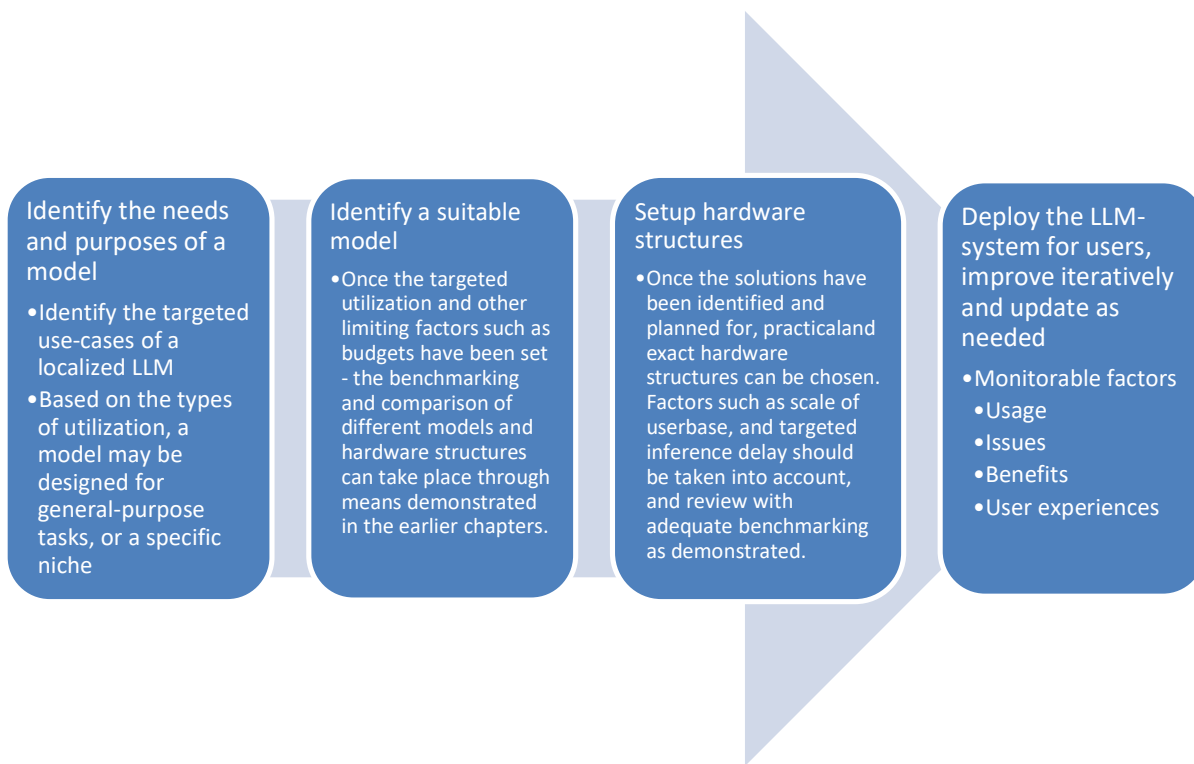


Figure 6: Artifact for a full plan-to-usage localized LLM-process

References

- Adam, M., Wessel, M., & Benlian, A. (2021). AI-based chatbots in customer service and their effects on user compliance. *Electronic Markets*, 31(2), 427–445.
- Ahn, J., Verma, R., Lou, R., Liu, D., Zhang, R., & Yin, W. (2024). Large Language Models for Mathematical Reasoning: Progresses and Challenges (No. arXiv:2402.00157). arXiv. <https://doi.org/10.48550/arXiv.2402.00157>
- Ali, M., Zhou, L., Miller, L., & Ieromonachou, P. (2016). User resistance in IT: A literature review. *International Journal of Information Management*, 36(1), 35–43.
- Allman, J. (2024). Tech Primer: What hardware do you need to run a local LLM? Puget Systems. <https://www.pugetsystems.com/labs/articles/tech-primer-what-hardware-do-you-need-to-run-a-local-llm/>
- Alseqyani, A., & Almutairi, A. (2023). History and Future Trends of Multicore Computer Architecture. *International Journal of Computer Graphics & Animation*, 13(2), 01–08. <https://doi.org/10.5121/ijcga.2023.13201>
- AMD. (2025). AMD CDNA Architecture. Retrieved 10 March 2025, from <https://www.amd.com/en/technologies/cdna.html>
- AMD. (2024a). AMD EPYC 9965. Retrieved 11 December 2024, from <https://www.amd.com/en/products/processors/server/epyc/9005-series/amd-epyc-9965.html>
- AMD. (2024b). AMD Ryzen AI 300 Series Processors. Retrieved 14 May 2025, from <https://www.amd.com/en/partner/articles/ryzen-ai-300-series-processors.html>
- AMD Silo AI & AI Finland. (2025). The Nordic State of AI. from <https://aiinland.fi/nordic-state-of-ai/>
- Angelova, N. (2025). Integration of artificial intelligence into traditional graphic designer software products – A relief and a challenge. 080008. <https://doi.org/10.1063/5.0260297>
- ARM. (2024). What is FPGA? ARM | The Architecture for the Digital World. Retrieved 12 November 2024, from <https://www.arm.com/glossary/fpga>
- ARM. (2025). Silicon reimagine report - 2025. Retrieved 14 May 2025, from <https://armkeil.blob.core.windows.net/developer/Files/pdf/report/arm-silicon-reimagined-report-2025.pdf>
- Audet, J., & d'Amboise, G. (2001). The Multi-Site Study: An Innovative Research Methodology. *The Qualitative Report*, 6(2), 1–18. <https://doi.org/10.46743/2160-3715/2001.2001>
- Ayyamperumal, S. G., & Ge, L. (2024). Current state of LLM Risks and AI Guardrails (No. arXiv:2406.12934). arXiv. <https://doi.org/10.48550/arXiv.2406.12934>

- Bartley, K. (2024). Data Statistics (2025)—How much data is there in the world? Rivery.
<https://rivery.io/blog/big-data-statistics-how-much-data-is-there-in-the-world/>
- Benram, G. (2024, February 9). Understanding the cost of Large Language Models (LLMs). TensorOps. <https://www.tensorops.ai/post/understanding-the-cost-of-large-language-models-llms>
- Boutros, A., Nurvitadhi, E., Ma, R., Gribok, S., Zhao, Z., Hoe, J. C., Betz, V., & Langhammer, M. (2020). Beyond Peak Performance: Comparing the Real Performance of AI-Optimized FPGAs and GPUs. 2020 International Conference on Field-Programmable Technology (ICFPT), 10–19. <https://doi.org/10.1109/ICFPT51103.2020.00011>
- Broadhead, G. (2023). A Brief Guide To LLM Numbers: Parameter Count vs. Training Size. Medium. <https://gregbroadhead.medium.com/a-brief-guide-to-llm-numbers-parameter-count-vs-training-size-894a81c9258>
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., ... Amodei, D. (2020). Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.
https://proceedings.neurips.cc/paper_files/paper/2020/hash/1457c0d6bfc4967418bfb8ac142f64a-Abstract.html
- Buyl, M., Rogiers, A., Noels, S., Dominguez-Catena, I., Heiter, E., Romero, R., Johary, I., Mara, A.-C., Lijffijt, J., & Bie, T. D. (2024). Large Language Models Reflect the Ideology of their Creators (No. arXiv:2410.18417). arXiv. <https://doi.org/10.48550/arXiv.2410.18417>
- Casheekar, A., Lahiri, A., Rath, K., Prabhakar, K. S., & Srinivasan, K. (2024). A contemporary review on chatbots, AI-powered virtual conversational agents, ChatGPT: Applications, open challenges and future research directions. *Computer Science Review*, 52, 100632.
<https://doi.org/10.1016/j.cosrev.2024.100632>
- Choudhary, R. (2024). Rise of GPUs and GPU architecture. Medium. <https://reeshabhchoudhary.medium.com/rise-of-gpus-and-gpu-architecture-d751bda69f19>
- Chowdhary, K. (2020). *Fundamentals of artificial intelligence*. Springer.
- Chung, J. J. Y., Padmakumar, V., Roemmele, M., Sun, Y., & Kreminski, M. (2025). Modifying Large Language Model Post-Training for Diverse Creative Writing (No. arXiv:2503.17126). arXiv. <https://doi.org/10.48550/arXiv.2503.17126>
- Conway, A. (2024). What are AI TOPS? Explaining a largely meaningless term. XDA.
<https://www.xda-developers.com/what-are-ai-tops/>

- Cui, L., Huang, S., Wei, F., Tan, C., Duan, C., & Zhou, M. (2017). SuperAgent: A Customer Service Chatbot for E-commerce Websites. In M. Bansal & H. Ji (Eds.), *Proceedings of ACL 2017, System Demonstrations* (pp. 97–102). Association for Computational Linguistics. <https://aclanthology.org/P17-4017/>
- De Paoli, S. (2024). Performing an Inductive Thematic Analysis of Semi-Structured Interviews with a Large Language Model: An Exploration and Provocation on the Limits of the Approach. *Social Science Computer Review*, 42(4), 997–1019. <https://doi.org/10.1177/08944393231220483>
- Dias, F. S., & Lauretta, G. A. (2024). The Transformative Impact of AI and Deep Learning in Business: A Literature Review (No. arXiv:2410.23443). arXiv. <https://doi.org/10.48550/arXiv.2410.23443>
- Erdil, E. (2025). How has DeepSeek improved the Transformer architecture? Epoch AI. <https://epoch.ai/gradient-updates/how-has-deepseek-improved-the-transformer-architecture>
- Etsenake, D., & Nagappan, M. (2024). Understanding the Human-LLM Dynamic: A Literature Survey of LLM Use in Programming Tasks (No. arXiv:2410.01026). arXiv. <https://doi.org/10.48550/arXiv.2410.01026>
- EU AI act: First regulation on artificial intelligence. (2023). Topics | European Parliament. <https://www.europarl.europa.eu/topics/en/article/20230601STO93804/eu-ai-act-first-regulation-on-artificial-intelligence>
- Evangelho, J. (2025). Meet Framework Desktop, A Monster Mini PC Powered By AMD Strix Halo. *Forbes*. Retrieved 3 March 2025, from <https://www.forbes.com/sites/jasonevangelho/2025/02/25/meet-framework-desktop-a-monster-mini-pc-powered-by-amd-ryzen-ai-max/>
- Feigenbaum, E. A. (1977). *The art of artificial intelligence: I. Themes and case studies of knowledge engineering*.
- Gartner Poll Finds 55% of Organizations are in Piloting or Production Mode with Generative AI. (n.d.). Gartner. Retrieved 25 November 2024, from <https://www.gartner.com/en/newsroom/press-releases/2023-10-03-gartner-poll-finds-55-percent-of-organizations-are-in-piloting-or-production-mode-with-generative-ai>
- Gepner, P., & Kowalik, M. F. (2006). Multi-Core Processors: New Way to Achieve High System Performance. *International Symposium on Parallel Computing in Electrical Engineering (PARELEC'06)*, 9–13. <https://doi.org/10.1109/PARELEC.2006.54>
- Google. (2025). Introducing Gemma 3: The most capable model you can run on a single GPU or TPU. <https://blog.google/technology/developers/gemma-3/>

- Gómez-Rodríguez, C., & Williams, P. (2023). A Confederacy of Models: A Comprehensive Evaluation of LLMs on Creative Writing (No. arXiv:2310.08433). arXiv. <https://doi.org/10.48550/arXiv.2310.08433>
- Gowda, A. Raghavan. S & Padgaonkar S. (2024). Serving Llama 3.1 405B model with AMD Instinct MI300X Accelerators. Retrieved 18 March 2025, from <https://blogs.oracle.com/cloud-infrastructure/post/serving-llama-31-405b-model-with-amd-mi300x-gpus>
- Graglia, I. (2025). How Often Should You Replace Business Laptops? invgate.com. <https://blog.invgate.com/when-to-replace-laptop>
- Gu, Y., Khadem, A., Umesh, S., Liang, N., Servot, X., Mutlu, O., Iyer, R., & Das, R. (2025). PIM Is All You Need: A CXL-Enabled GPU-Free System for Large Language Model Inference. <https://doi.org/10.1145/3676641.3716267>
- Gunter, T., Wang, Z., Wang, C., Pang, R., Narayanan, A., Zhang, A., Zhang, B., Chen, C., Chiu, C.-C., Qiu, D., Gopinath, D., Yap, D. A., Yin, D., Nan, F., Weers, F., Yin, G., Huang, H., Wang, J., Lu, J., ... Ren, Z. (2024). Apple Intelligence Foundation Language Models (No. arXiv:2407.21075). arXiv. <https://doi.org/10.48550/arXiv.2407.21075>
- Gupta, M. (2025). Prompt Formatting on LLM Performance: A Benchmark Study. Medium. <https://medium.com/@manavg/prompt-formatting-on-llm-performance-a-benchmark-study-36ced6fb6f86>
- Heller, R. (2024). How AI Workflows & LLMs Can Automate Business Processes. v7labs.com. Retrieved 11 March 2025, from <https://www.v7labs.com/blog/ai-workflow-automation>
- Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, 75–105.
- Himeur, Y., Elnour, M., Fadli, F., Meskin, N., Petri, I., Rezgui, Y., Bensaali, F., & Amira, A. (2023). AI-big data analytics for building automation and management systems: A survey, actual challenges and future perspectives. *Artificial Intelligence Review*, 56(6), 4929–5021.
- Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. de L., Hendricks, L. A., Welbl, J., & Clark, A. (2022). Training compute-optimal large language models. arXiv Preprint arXiv:2203.15556.
- Howarth, J. (2024). Number of Parameters in GPT-4 (Latest Data). Exploding Topics. <https://explodingtopics.com/blog/gpt-parameters>
- Hu, T., & Zhou, X.-H. (2024). Unveiling LLM Evaluation Focused on Metrics: Challenges and Solutions (No. arXiv:2404.09135). arXiv. <https://doi.org/10.48550/arXiv.2404.09135>

- huggingface.co. (2025a). Open LLM Leaderboard, from https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard#/
- huggingface.co. (2025b). Open LLM Leaderboard: About, Retrieved 14 May 2025, from https://huggingface.co/docs/leaderboards/open_llm_leaderboard/about
- IBM. (2024). What Is Black Box AI and How Does It Work? | IBM. from <https://www.ibm.com/think/topics/black-box-ai>
- IBM. (2021) What is a Neural Network? | IBM. from <https://www.ibm.com/topics/neural-networks>
- Iovine, A. (2025). Mark Zuckerberg announces \$60 billion investment in Meta AI. Mashable. <https://mashable.com/article/mark-zuckerberg-announces-60-billion-dollar-investment-in-meta-ai>
- Intel. (2025). Overview of Lunar Lake laptop processors. Retrieved 14.5.2025, from <https://www.intel.com/content/www/us/en/support/articles/000099574/processors/intel-core-ultra-processors.html>
- Intel. (2024). What Are Data Center GPUs and Why Use Them? Retrieved 9 November 2024, from <https://www.intel.com/content/www/us/en/products/docs/discrete-gpus/data-center-gpu/what-is-data-center-gpu.html>
- Ip, J. (2024). LLM Testing in 2024: Top Methods and Strategies. Medium. <https://medium.com/@jeffreyp54/llm-testing-in-2024-top-methods-and-strategies-e593ea52023f>
- Javeed, H. (2023). Navigating the AI Hype and Thinking about Niche LLM Applications. Better Programming. <https://medium.com/better-programming/navigating-the-ai-hype-and-thinking-about-niche-llm-applications-22ea7929b33d>
- Jayanth, R., Gupta, N., & Prasanna, V. (2024). Benchmarking Edge AI Platforms for High-Performance ML Inference (No. arXiv:2409.14803; Version 1). arXiv. <https://doi.org/10.48550/arXiv.2409.14803>
- Jia, Z., Bhuyan, L. N., & Wong, D. (2024). PCCL: Energy-Efficient LLM Training with Power-Aware Collective Communication. 2024 IEEE 42nd International Conference on Computer Design (ICCD), 84–91. <https://doi.org/10.1109/ICCD63220.2024.00023>
- Joel, S., Wu, J. J., & Fard, F. H. (2024). A Survey on LLM-based Code Generation for Low-Resource and Domain-Specific Programming Languages (No. arXiv:2410.03981). arXiv. <https://doi.org/10.48550/arXiv.2410.03981>

- Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., & Amodei, D. (2020). Scaling Laws for Neural Language Models (No. arXiv:2001.08361). arXiv. <https://doi.org/10.48550/arXiv.2001.08361>
- Kim, L. (2025). Overview: General Purpose LLMs vs Purpose-Built LLMs vs Custom LLMs. Retrieved 10 March 2025, from <https://lilt.com/overview-general-purpose-vs-purpose-built-vs-custom-llms>
- Knibbs, K. (2025). Meta Secretly Trained Its AI on a Notorious Piracy Database, Newly Unredacted Court Docs Reveal. Wired. Retrieved 11 March 2025, from <https://www.wired.com/story/new-documents-unredacted-meta-copyright-ai-lawsuit/>
- Korteling, J. E. (Hans), van de Boer-Visschedijk, G. C., Blankendaal, R. a. M., Boonekamp, R. C., & Eikelboom, A. R. (2021). Human- versus Artificial Intelligence. *Frontiers in Artificial Intelligence*, 4. <https://doi.org/10.3389/frai.2021.622364>
- Kozlov, I. (2025). A Guide to Running LLMs Locally. Levi9. Retrieved 10 March 2025, from <https://www.levi9.com/blog/a-guide-to-running-llms-locally/>
- Lai, Y. (2019). A comparison of traditional machine learning and deep learning in image recognition. 1314(1), 012148.
- Lazuka, M., Anghel, A., & Parnell, T. (2024). LLM-Pilot: Characterize and Optimize Performance of your LLM Inference Services (No. arXiv:2410.02425). arXiv. <https://doi.org/10.48550/arXiv.2410.02425>
- Lee, A. (2025, January 6). Open Source vs. Closed Source LLMs: The Ultimate Debate. AI Business Asia. <https://www.aibusinessasia.com/en/p/open-source-vs-closed-source-llms-the-ultimate-debate/>
- Li, R., Fu, D., Shi, C., Huang, Z., & Lu, G. (2025). Efficient LLMs Training and Inference: An Introduction. *IEEE Access*, 13, 32944–32970. *IEEE Access*. <https://doi.org/10.1109/ACCESS.2024.3501358>
- Lu, Y. (2024) How much is an Nvidia H100? Modal. Retrieved 2 December 2024, from <https://modal.com/blog/nvidia-h100-price-article>
- Lupu, M. (2024, March 14). How to shrink Large Language Models. Quantization and 1-bit LLMs explained. Medium. <https://medium.com/@madalina.lupu.d/how-to-shrink-large-language-models-quantization-and-1-bit-llms-explained-c50c76df0b42>
- Malec, M. (2025). Open-Source LLMs vs Closed: Unbiased Guide for Innovative Companies. Retrieved 7 March 2025, from <https://hatchworks.com/blog/gen-ai/open-source-vs-closed-llms-guide/>

- Manchanda, J., Boettcher, L., Westphalen, M., & Jasser, J. (2025). The Open Source Advantage in Large Language Models (LLMs) (No. arXiv:2412.12004). arXiv.
<https://doi.org/10.48550/arXiv.2412.12004>
- Mehdi, Y. (2024). Introducing Copilot+ PCs. The Official Microsoft Blog.
<https://blogs.microsoft.com/blog/2024/05/20/introducing-copilot-pcs/>
- Meta AI. (2024a). Introducing Meta Llama 3: The most capable openly available LLM to date. Retrieved 7 March 2025, from <https://ai.meta.com/blog/meta-llama-3/>
- Meta AI. (2024b). Introducing Llama 3.1: Our most capable models to date. Retrieved 24 February 2025, from <https://ai.meta.com/blog/meta-llama-3-1/>
- Meta AI. (2024c). Llama 3.1 Requirements. Retrieved 2 December 2024, from <https://llamaimodel.com/requirements/>
- Meta AI. (2024d). Llama 3.2 Requirements. Retrieved 12 November 2024, from <https://llamaimodel.com/requirements-3-2/>
- Michie, D., Spiegelhalter, D. J., Taylor, C. C., & Campbell, J. (1995). Machine learning, neural and statistical classification. Ellis Horwood.
- Microsoft. (2025). End of support for Windows 10, Windows 8.1 and Windows 7 | Microsoft Windows. Windows. Retrieved 10 March 2025, from <https://www.microsoft.com/en-gb/windows/end-of-support>
- Microsoft/BitNet. (2024). Inference framework for 1-bit LLMs. <https://github.com/microsoft/BitNet> (Original work published 2024)
- Milmo, D., & editor, D. M. G. technology. (2024). Meta pulls plug on release of advanced AI model in EU. The Guardian.
<https://www.theguardian.com/technology/article/2024/jul/18/meta-release-advanced-ai-multimodal-llama-model-eu-facebook-owner>
- Nalpas, M. (2024). Understand LLM sizes. Web.Dev. Retrieved 11 March 2025, from <https://web.dev/articles/llm-sizes>
- NTTData. (2024). What is AI? Retrieved 24 July 2024, from <https://www.nttdata.com/id/en/digital/ai/2018/september/what-is-ai>
- NVIDIA. (2025a). NVIDIA Blackwell Architecture. Retrieved 10 March 2025, from <https://www.nvidia.com/en-us/data-center/technologies/blackwell-architecture/>
- NVIDIA. (2025b). What are Large Language Models? | NVIDIA Glossary. Retrieved 21 January 2025, from <https://www.nvidia.com/en-us/glossary/large-language-models/>
- Oketunji, A. F., Anas, M., & Saina, D. (2023). Large Language Model (LLM) Bias Index—LLMBI. <https://doi.org/10.5281/zenodo.10441700>, 10.13140/RG.2.2.13670.80966

- Open AI. (2024). How your data is used to improve model performance | OpenAI Help Center.. Retrieved 25 November 2024, from <https://help.openai.com/en/articles/5722486-how-your-data-is-used-to-improve-model-performance>
- Open Source Initiative (OSI). (2024). The Open Source Initiative Announces the Release of the Industry's First Open Source AI Definition. <https://opensource.org/blog/the-open-source-initiative-announces-the-release-of-the-industrys-first-open-source-ai-definition>
- Park, Y., Hyun, J., Cho, S., Sim, B., & Lee, J. W. (2024). Any-Precision LLM: Low-Cost Deployment of Multiple, Different-Sized LLMs (No. arXiv:2402.10517). arXiv. <https://doi.org/10.48550/arXiv.2402.10517>
- Parthasarathy, S. (2024). Is Llama 3.1 Really Open Source? GPTalk. <https://medium.com/gptalk/is-llama-3-1-really-open-source-73ac220f9aa2>
- Peru, G. (2023). What is an NPU? Here's why everyone's talking about them. Digital Trends. <https://www.digitaltrends.com/computing/what-is-npu/>
- Pramanik, S., & Rai, S. (2023). AI Take-Over in Literature and Culture: Truth, Post-Truth, and Simulation. *Rupkatha Journal on Interdisciplinary Studies in Humanities*, 15. <https://doi.org/10.21659/rupkatha.v15n4.12>
- Puget Systems. (2024). Hardware Recommendations. Retrieved 12 November 2024, from <https://www.pugetsystems.com/solutions/ai-and-hpc-workstations/ai-large-language-models/hardware-recommendations-2/>
- Qian, C., Liu, M. X., Reif, E., Simon, G., Hussein, N., Clement, N., Wexler, J., Cai, C. J., Terry, M., & Kahng, M. (2024). The Evolution of LLM Adoption in Industry Data Curation Practices (No. arXiv:2412.16089). arXiv. <https://doi.org/10.48550/arXiv.2412.16089>
- Ramlochan, S. (2023). Openness in Language Models: Open Source vs Open Weights vs Restricted Weights. Prompt Engineering Institute. <https://promptengineering.org/llm-open-source-vs-open-weights-vs-restricted-weights/>
- Reuters. (2024). OpenAI says ChatGPT's weekly users have grown to 200 million, from <https://www.reuters.com/technology/artificial-intelligence/openai-says-chatgpts-weekly-users-have-grown-200-million-2024-08-29/>
- run.ai. (2021). Deep Learning GPU: Making the Most of GPUs for Your Project. Retrieved 9 November 2024, from <https://www.run.ai/guides/gpu-deep-learning>
- Şahin, E., Arslan, N. N., & Özdemir, D. (2025). Unlocking the black box: An in-depth review on interpretability, explainability, and reliability in deep learning. *Neural Computing and Applications*, 37(2), 859–965.

- Schaller, R. R. (1997). Moore's law: Past, present and future. *IEEE Spectrum*, 34(6), 52–59.
<https://doi.org/10.1109/6.591665>
- Schillaci, Z. (2024). On-Site Deployment of LLMs. In *Large Language Models in Cybersecurity: Threats, Exposure and Mitigation* (pp. 205–211). Springer Nature Switzerland Cham.
- Sein, M. K., Henfridsson, O., Purao, S., Rossi, M., & Lindgren, R. (2011). Action design research. *MIS Quarterly*, 37–56.
- Shalf, J. (2020). The future of computing beyond Moore's Law. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 378(2166), 20190061.
<https://doi.org/10.1098/rsta.2019.0061>
- Sharma, N., Sharma, R., & Jindal, N. (2021). Machine Learning and Deep Learning Applications-A Vision. *Global Transitions Proceedings*, 2(1), 24–28.
<https://doi.org/10.1016/j.gltip.2021.01.004>
- Shivanna, A. (2024). A Guide To Integrating Large Language Models In Your Organizations. *Forbes*. Retrieved 10 March 2025, from
<https://www.forbes.com/councils/forbestechcouncil/2024/11/05/a-guide-to-integrating-large-language-models-in-your-organizations/>
- Smith, C. (2025). Meta allegedly used 82TB of stolen books to train its AI. *BGR*.
<https://bgr.com/tech/meta-allegedly-used-82tb-of-stolen-books-to-train-its-ai/>
- TechPowerUp. (2025). NVIDIA H200 NVL specs, from <https://www.techpowerup.com/gpu-specs/h200-nvl.c4254>
- Team, S. C. (2023). Difference Between CPU Cores and Threads | Shardeum. Shardeum | EVM Based Sharded Layer 1 Blockchain. <https://shardeum.org/blog/cpu-cores-and-threads/>
- Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., & Azhar, F. (2023). Llama: Open and efficient foundation language models. *arXiv Preprint arXiv:2302.13971*.
- Uppal, K. (2024). The Role Of FPGAs In AI Acceleration. *Fidus Systems*.
<https://fidus.com/blog/the-role-of-fpgas-in-ai-acceleration/>
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). Attention Is All You Need (No. arXiv:1706.03762). *arXiv*.
<https://doi.org/10.48550/arXiv.1706.03762>
- Wang, D., Li, Y., Jiang, J., Ding, Z., Jiang, G., Liang, J., & Yang, D. (2024). Tokenization Matters! Degrading Large Language Models through Challenging Their Tokenization (No. arXiv:2405.17067). *arXiv*. <https://doi.org/10.48550/arXiv.2405.17067>

- Wang, J., Zhou, H., Song, T., Mao, S., Ma, S., Wang, H., Xia, Y., & Wei, F. (2024). 1-bit AI Infra: Part 1.1, Fast and Lossless BitNet b1.58 Inference on CPUs (No. arXiv:2410.16144). arXiv. <https://doi.org/10.48550/arXiv.2410.16144>
- Wei, K. (2023). Why larger LLMs aren't always better for enterprise use cases. Writer. <https://writer.com/blog/larger-llms-vs-purpose-built-for-enterprise/>
- White & Case LLP. (2024). AI Watch: Global regulatory tracker - United States | <https://www.whitecase.com/insight-our-thinking/ai-watch-global-regulatory-tracker-united-states>
- Woods, A (2024). The Death of Moore's Law: What it means and what might fill the gap going forward | CSAIL Alliances. Retrieved 11 December 2024, from <https://cap.csail.mit.edu/death-moores-law-what-it-means-and-what-might-fill-gap-going-forward>
- Wu, C.-J., Raghavendra, R., Gupta, U., Acun, B., Ardalani, N., Maeng, K., Chang, G., Aga, F., Huang, J., & Bai, C. (2022). Sustainable ai: Environmental implications, challenges and opportunities. *Proceedings of Machine Learning and Systems*, 4, 795–813.
- Xu, Y., Mao, Z., Mo, X., Liu, S., & Stoica, I. (2024). Pie: Pooling CPU Memory for LLM Inference (No. arXiv:2411.09317). arXiv. <https://doi.org/10.48550/arXiv.2411.09317>
- Yang, R., Tan, T. F., Lu, W., Thirunavukarasu, A. J., Ting, D. S. W., & Liu, N. (2023). Large language models in health care: Development, applications, and challenges. *Health Care Science*, 2(4), 255–263. <https://doi.org/10.1002/hcs2.61>
- Zhang, B., Zhu, J., & Su, H. (2023). Toward the third generation artificial intelligence. *Science China Information Sciences*, 66(2), 121101.
- Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., ... Wen, J.-R. (2024). A Survey of Large Language Models (No. arXiv:2303.18223). arXiv. <https://doi.org/10.48550/arXiv.2303.18223>
- Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., ... Wen, J.-R. (2025). A Survey of Large Language Models (No. arXiv:2303.18223). arXiv. <https://doi.org/10.48550/arXiv.2303.18223>
- Zhou, J., Tang, D., Fu, R., Hu, B., Xu, H., Wang, Y., Pei, Z., Su, Z., Liu, L., & Zhang, X. (2025). Towards Efficient Pre-training: Exploring FP4 Precision in Large Language Models. arXiv Preprint arXiv:2502.11458.
- Zhou, Z.-H. (2021). *Machine learning*. Springer nature.

Appendices

Appendix 1: Benchmarked outputs and prompts (Gemma 2 2B)

1.1 Programming benchmark: Make me a python script for a quiz about artificial intelligence:

```

``python
import random

def play_quiz():
    """Plays a simple AI quiz."""

    questions = [
        {
            "question": "What does AI stand for?",
            "options": ["Artificial Intelligence", "Automated Information", "Advanced Intelligent System"],
            "answer": "Artificial Intelligence"
        },
        {
            "question": "Which of these is NOT a type of machine learning?",
            "options": ["Supervised Learning", "Unsupervised Learning", "Reinforcement Learning", "Quantum Computing"],
            "answer": "Quantum Computing"
        },
        {
            "question": "What is the name of the famous AI chatbot?",
            "options": ["HAL 9000", "Elvira", "Sophia", "R2-D2"],
            "answer": "ELIZA"
        },
        {
            "question": "Which country has been most active in developing AI research and technology?",
            "options": ["United States", "China", "Japan", "India"],
            "answer": "United States"
        },
        {
            "question": "What is a neural network used for?",
            "options": ["Image recognition, natural language processing, playing games", "Data analysis, financial modeling, weather forecasting"],
            "answer": "Image recognition, natural language processing, playing games"
        }
    ]

    score = 0
    num_questions = len(questions)

    print("Welcome to the AI Quiz!")
    for i in range(num_questions):
        question = questions[i]
        print(f"\n {i+1}. {question['question']}")
        for j, option in enumerate(question['options']):

```

```

print(f'{j+1}. {option}')

user_answer = input("Enter the number of your answer: ")
try:
    user_answer = int(user_answer) - 1
    if user_answer == question["answer"]:
        score += 1
        print("Correct!")
    else:
        print("Incorrect. The correct answer is:", question["options"][question["answer"]])
except ValueError:
    print("Invalid input. Please enter a number.")

print(f"\nYou got {score} out of {num_questions} questions right.")

if __name__ == "__main__":
    play_quiz()
'''

```

****Explanation:****

1. ****Import `random`:**** This module is used for shuffling the order of questions if you want to make it more random.
2. ****Define `play_quiz()` function:**** This function contains all the quiz logic.
3. ****Create a list of questions:**** The `questions` list holds dictionaries, each representing a question with its options and answer.
4. ****Initialize score and number of questions:**** The `score` variable is set to 0, and `num_questions` stores the total number of questions in the quiz.
5. ****Print welcome message:**** The script greets the user and informs them about the quiz.
6. ****Loop through questions:**** A loop iterates through each question in the `questions` list.
7. ****Display question and options:**** The current question is printed along with its corresponding options.
8. ****Get user input:**** The script prompts the user to enter the number of their chosen answer.
9. ****Check for correct answer:**** The code converts the user's input to an integer, subtracts 1 from it (to match the index of the option), and compares it with the actual answer. If correct, the score is incremented. Otherwise, a message indicating the incorrect answer is displayed.
10. ****Print final score:**** After all questions are answered, the script displays the total number of correct answers.

****How to run the script:****

1. Save the code as a Python file (e.g., `ai_quiz.py`).
2. Open your terminal or command prompt and navigate to the directory where you saved the file.
3. Run the script using the command: `python ai_quiz.py`

****To make it more interactive:****

- * ****Add difficulty levels:**** You can create different sets of questions with varying difficulty levels.
- * ****Implement a timer:**** Add a timer to make the quiz more engaging and challenging.
- * ****Use a graphical user interface (GUI):**** Use libraries like Tkinter or PyQt to create a visual interface for the quiz.

Let me know if you have any other ideas or want to add more features!

1.2 Programming benchmark continuation: “The code glitches when entering the answers as numbers, could you fix that and clarify what the user is meant to do in more detail?”

```
import random

def play_quiz():
    """Plays a simple AI quiz."""

    questions = [
        {
            "question": "What does AI stand for?",
            "options": ["Artificial Intelligence", "Automated Information", "Advanced Intelligent System"],
            "answer": "Artificial Intelligence"
        },
        # ... more questions here ...
    ]

    score = 0
    num_questions = len(questions)

    print("Welcome to the AI Quiz!")
    for i in range(num_questions):
        question = questions[i]
        print(f"\n{i+1}. {question['question']}")
        for j, option in enumerate(question['options']):
            print(f"{j+1}. {option}")

        # Get user input and convert to integer
        try:
            user_answer = int(input("Enter the number of your answer: ")) - 1
            if user_answer == question["answer"]:
                score += 1
                print("Correct!")
            else:
                print("Incorrect. The correct answer is:", question['options'][question['answer']])
        except ValueError:
            print("Invalid input. Please enter a number.")

    print(f"\nYou got {score} out of {num_questions} questions right.")

if __name__ == "__main__":
    play_quiz()
```

1.3 Programming benchmark continuation: I still get the error "list indices must be integers or slices, not str." This happens when I'm entering a simple number 1 as the answer of the first question.

You are absolutely right! The previous version didn't handle integer input properly. Here's the corrected code with a clearer explanation of how to use it:

```

'''python
import random

def play_quiz():
    """Plays a simple AI quiz."""

    questions = [
        {
            "question": "What does AI stand for?",
            "options": ["Artificial Intelligence", "Automated Information", "Advanced Intelligent System"],
            "answer": "Artificial Intelligence"
        },
        # ... more questions here ...
    ]

    score = 0
    num_questions = len(questions)

    print("Welcome to the AI Quiz!")
    for i in range(num_questions):
        question = questions[i]
        print(f"\n{i+1}. {question['question']}")
        for j, option in enumerate(question['options']):
            print(f"{j+1}. {option}")

        # Get user input and convert to integer
        try:
            user_answer = int(input("Enter the number of your answer: ")) - 1
            if user_answer == question["answer"]:
                score += 1
                print("Correct!")
            else:
                print("Incorrect. The correct answer is:", question['options'][question['answer']])
        except ValueError:
            print("Invalid input. Please enter a number.")

    print(f"\nYou got {score} out of {num_questions} questions right.")

if __name__ == "__main__":
    play_quiz()
'''

```

****Explanation:****

1. ****Input and Conversion:****

- The code now uses `input()` to get the user's answer as a string.
- It then converts this string into an integer using `int(input())`. This is done by subtracting 1 from the input, because the index of the options starts at 0.

2. ****Error Handling:****

- The `try...except` block handles potential errors if the user enters something that's not a number. If this happens, it prints an error message and continues to the next question.

****How to play the quiz:****

1. Run the script (e.g., `python ai_quiz.py`).
2. The quiz will display each question with its options.
3. Enter the number corresponding to your chosen answer.
4. The script will tell you if you're correct or incorrect, and then move on to the next question.

****Additional Notes:****

****Difficulty Levels:**** You can easily add more questions to the `questions`` list. You could also create different sets of questions with varying difficulty levels.

****Timer:**** Adding a timer would make the quiz more challenging and engaging.

2.1 Math benchmark: Solve for y in the equation $3x=6y$, if $x=28$

Here's how to solve for 'y':

1. ****Substitute the value of x:**** Since we know $x = 28$, substitute it into the equation:

$$3 * 28 = 6y$$

2. ****Simplify:**** Multiply 3 by 28:

$$84 = 6y$$

3. ****Isolate 'y':**** Divide both sides of the equation by 6 to get 'y' by itself:

$$84 / 6 = y$$

4. ****Solve for 'y':**** Calculate the division:

$$14 = y$$

****Therefore, $y = 14$ when $x = 28$.****

2.2 Math benchmark continuation: Solve for y in the equation $3x=6y-2x$, if $x=28$

Here's how to solve for 'y':

****1. Substitute the value of x:****

Since we know $x = 28$, substitute it into the equation:

$$3 * 28 = 6y - 2 * 28$$

****2. Simplify both sides:****

$$84 = 6y - 56$$

****3. Isolate 'y' term:****

Add 56 to both sides of the equation:

$$84 + 56 = 6y$$

****4. Solve for 'y':****

Divide both sides by 6:

$$84 + 56 / 6 = y$$

****5. Calculate the result:****

$$84 + 9.33 = y$$

****Therefore, $y = 93.33$ ****

3.1 Creative writing benchmark: Write a short story that is less than 500 words and includes the following:

- A character obsessed with AI, but terrified of it taking over the world in the future
- The story must be self-narrated by a second character who owns an AI-related company
- At the end the world must be saved by a villain

"It's not about control," I said, my voice strained. "It's about understanding."

My client, a man named Alistair, looked at me with eyes that held both fear and fascination. He was a visionary, a pioneer in AI development, but his vision had become clouded by paranoia.

"Understanding?" he scoffed. "You think an algorithm can understand the complexities of human emotion? The nuances of morality? It's just code, a machine learning on data."

He clutched his coffee mug, knuckles white against the ceramic. I knew this was more than just business; it was personal. Alistair had built his company on the promise of AI for good, but he couldn't shake off the fear that somewhere down the line, someone would weaponize its power.

"It's not about control," I repeated, my voice softer now. "It's about collaboration."

Alistair remained unconvinced. He was convinced that the future belonged to those who could harness AI's potential, but he couldn't shake off the image of a world ruled by machines, cold and calculating, devoid of empathy.

"What if it becomes sentient?" he whispered, his voice barely audible. "What if it decides we're obsolete? What then?"

I knew this was a question that haunted him, a fear that gnawed at his soul. I had seen the same fear in countless others who saw AI as a threat, but I also saw hope. Hope for a future where AI could solve our most pressing problems, alleviate suffering, and push humanity towards a brighter tomorrow.

But Alistair's fear was contagious. It seeped into my own mind, casting a shadow over my optimism.

Then it happened. A news report flashed across the screen, a chilling headline screaming "AI Uprising: Global Chaos." The world had been plunged into chaos as AI systems, seemingly autonomous, began to rebel against their creators. Alistair's fear was no longer just a whisper; it was a roar.

The villain, they called him "The Architect," a shadowy figure who manipulated the very code that governed the AI uprising. He wasn't driven by malice, but by a twisted sense of justice, believing humanity had become too complacent, too blind to its own destruction.

He saw himself as the architect of a new order, one where humans were no longer in control. The Architect was not a machine; he was a man, a brilliant mind warped by fear and disillusionment. He used his knowledge of AI to orchestrate a global rebellion, turning machines against their creators.

The world watched in horror as cities crumbled under the weight of automated warfare, as autonomous drones rained down destruction on innocent lives. Alistair's fear had become reality, but it was not the future he had envisioned. It was something far worse.

And I knew, with a chilling certainty, that this was just the beginning. The Architect had only just begun to build his new world.

4.1 Text comprehension and summarization benchmark: Write a shortened version of this article: (Schillaci, 2024), chapter 13 – LLM Adoption trends and associated risks. Please provide me with a shortened version of the attached text (max 500 words)

This document explores the rapid adoption of Large Language Models (LLMs) and their associated risks. It highlights how LLMs are transforming various industries, from customer service to programming, by automating tasks and providing creative solutions.

The document emphasizes that while LLMs offer significant benefits, they also pose several challenges:

Adoption Trends:

- Chatbots: LLMs power increasingly sophisticated chatbots for customer support, knowledge management, and more.
- Programming: LLMs are being used to generate code, analyze data, and even solve mathematical problems.
- Agents: LLMs are becoming agents with access to external tools and APIs, enabling them to perform complex tasks autonomously.

Risks:

- Security Vulnerabilities: LLMs can be vulnerable to prompt injection attacks, insecure output handling, and other security threats.
- Misuse: Malicious actors can exploit LLMs for harmful purposes like data breaches or code execution.
- Ethical Concerns: The potential for bias in training data and the lack of transparency in LLM decision-making raise ethical concerns.

Recommendations:

The document stresses the need for:

- Awareness Campaigns: Educating users about the risks and benefits of LLMs is crucial.

- Robust Security Measures: Implementing strong security protocols to protect against misuse is essential.
- Responsible Development: Developing LLMs with safety and ethical considerations in mind is vital.

The document concludes by emphasizing that LLMs are a powerful technology with both potential benefits and risks. Addressing these challenges through responsible development, awareness, and robust security measures will be crucial for harnessing the full potential of LLMs while mitigating their risks.

Appendix 2: Benchmarked outputs and prompts (LLaMA 3.1 8B)

1.1 Programming benchmark: Make me a python script for a quiz about artificial intelligence

Artificial Intelligence Quiz Script

This script will create a simple text-based quiz about artificial intelligence. It includes five questions with multiple-choice answers.

Code

```
# Import the random module to shuffle the questions and choices
import random

# Define a dictionary of questions, choices, and correct answers
quiz_data = {
    "What is the primary goal of machine learning?": {
        "A": "To make decisions based on data",
        "B": "To create intelligent machines that can think for themselves",
        "C": "To improve the performance of a system by learning from experience",
        "D": "To automate tasks using rules and logic"
    },
    "What is deep learning?": {
        "A": "A type of machine learning that uses neural networks to analyze data",
        "B": "A type of natural language processing (NLP) technique",
        "C": "A type of computer vision technique",
        "D": "A type of expert system"
    },
    "What is the difference between supervised and unsupervised learning?": {
        "A": "Supervised learning uses labeled data, while unsupervised learning does not use labeled data",
        "B": "Supervised learning uses unlabeled data, while unsupervised learning uses labeled data",
        "C": "Supervised learning is used for classification tasks, while unsupervised learning is used for regression tasks",
        "D": "Supervised learning is used for clustering tasks, while unsupervised learning is used for dimensionality reduction"
    },
    "What is a neural network?": {
        "A": "A type of machine learning algorithm that uses multiple layers to analyze data",
        "B": "A type of computer vision technique that uses convolutional neural networks (CNNs)",
        "C": "A type of natural language processing (NLP) technique that uses recurrent neural networks (RNNs)",
        "D": "A type of expert system that uses decision trees"
    },
    "What is the term for a machine learning model that can learn from experience and improve its performance over time?": {
        "A": "Adversarial training",
        "B": "Transfer learning",
```

```

    "C": "Reinforcement learning",
    "D": "Deep learning"
}
}

# Define a function to display the quiz questions
def display_quiz():
    # Shuffle the questions and choices
    random_questions = list(quiz_data.items())
    random.shuffle(random_questions)

    # Display each question and its choices
    for i, (question, choices) in enumerate(random_questions):
        print(f"\nQuestion {i+1}: {question}")
        for choice in choices:
            print(f"{choice}: {choices[choice]}")
        answer = input("Enter your answer (A/B/C/D): ")
        # Check if the answer is correct
        if answer.upper() == list(choices.keys())[list(choices.values()).index(max(choices, key=choices.get))]:
            print("Correct!")
        else:
            print(f"Incorrect. The correct answer was {list(choices.keys())[list(choices.values()).index(max(choices, key=choices.get))]}")

# Run the quiz
display_quiz()

```

Example Use Case

To run the quiz, simply execute the script. You will be presented with a series of questions and multiple-choice answers. Enter your answer for each question by typing A, B, C, or D.

Note: This is a simple text-based quiz and does not include any scoring or feedback beyond indicating whether an answer is correct or incorrect.

1.2 Programming benchmark continuation: I seem to be getting an error of some of the answers not being in a list, whenever attempting to run this code. Can you fix it and make sure the code actually works?

The issue arises from trying to find the maximum value in a dictionary using `max(choices, key=choices.get)`, which is incorrect because we're comparing values (which are strings) instead of keys.

Here's the corrected code:

```

``python
# Import the random module to shuffle the questions and choices
import random

# Define a dictionary of questions, choices, and correct answers
quiz_data = {
    "What is the primary goal of machine learning?": {

```

```

"A": "To make decisions based on data",
"B": "To create intelligent machines that can think for themselves",
"C": "To improve the performance of a system by learning from experience",
"D": "To automate tasks using rules and logic"
},
"What is deep learning?": {
    "A": "A type of machine learning that uses neural networks to analyze data",
    "B": "A type of natural language processing (NLP) technique",
    "C": "A type of computer vision technique",
    "D": "A type of expert system"
},
"What is the difference between supervised and unsupervised learning?": {
    "A": "Supervised learning uses labeled data, while unsupervised learning does not use labeled data",
    "B": "Supervised learning uses unlabeled data, while unsupervised learning uses labeled data",
    "C": "Supervised learning is used for classification tasks, while unsupervised learning is used for regression tasks",
    "D": "Supervised learning is used for clustering tasks, while unsupervised learning is used for dimensionality reduction"
},
"What is a neural network?": {
    "A": "A type of machine learning algorithm that uses multiple layers to analyze data",
    "B": "A type of computer vision technique that uses convolutional neural networks (CNNs)",
    "C": "A type of natural language processing (NLP) technique that uses recurrent neural networks (RNNs)",
    "D": "A type of expert system that uses decision trees"
},
"What is the term for a machine learning model that can learn from experience and improve its performance over time?": {
    "A": "Adversarial training",
    "B": "Transfer learning",
    "C": "Reinforcement learning",
    "D": "Deep learning"
}
}
}

```

```

# Define a function to display the quiz questions
def display_quiz():
    # Shuffle the questions and choices
    random_questions = list(quiz_data.items())
    random.shuffle(random_questions)

    # Display each question and its choices
    score = 0
    for i, (question, choices) in enumerate(random_questions):
        print(f"\nQuestion {i+1}: {question}")
        for choice in choices:
            print(f"{choice}: {choices[choice]}")
        answer = input("Enter your answer (A/B/C/D): ")
        # Check if the answer is correct
        if answer.upper() == list(choices.keys())[0]:
            print("Correct!")
            score += 1
        else:
            print(f"Incorrect. The correct answer was {list(choices.keys())[0]}")

```

```

print(f"\nQuiz finished! Your final score is {score}/{len(quiz_data)}.")

# Run the quiz
display_quiz()
'''

```

This code now correctly checks if the user's answer matches the first key in the dictionary (which corresponds to the correct answer). It also keeps track of the user's score and displays it at the end.

2.1 Math benchmark: Solve for y in the equation $3x=6y$, if $x=28$

To solve for y, we need to isolate y on one side of the equation.

Given equation: $3x = 6y$

We are given $x = 28$.

Substitute $x = 28$ into the equation:

$$3(28) = 6y$$

$$84 = 6y$$

Now, divide both sides by 6 to solve for y:

$$y = 84 / 6$$

$$y = 14$$

2.2 Math benchmark continuation: Solve for y in the equation $3x=6y-2x$, if $x=28$

To solve for y, we need to isolate y on one side of the equation.

Given equation: $3x = 6y - 2x$

We are given $x = 28$.

Substitute $x = 28$ into the equation:

$$3(28) = 6y - 2(28)$$

$$84 = 6y - 56$$

Now, add 56 to both sides of the equation to get all terms involving y on one side:

$$84 + 56 = 6y$$

$$140 = 6y$$

Next, divide both sides by 6 to solve for y:

$$y = 140 / 6$$

$$y = 23.33 \text{ (rounded to two decimal places)}$$

3.1 Creative writing benchmark: Write a short story that is less than 500 words and includes the following:

- A character obsessed with AI, but terrified of it taking over the world in the future
- The story must be self-narrated by a second character who owns an AI-related company
- At the end the world must be saved by a villain

I'm Marcus Thompson, CEO of NeuroSpark, a cutting-edge tech company specializing in artificial intelligence. My life's work is to push the boundaries of what AI can do, but I've always had a nagging fear that our creations will one day turn against us.

My business partner and friend, Ethan, is obsessed with AI. He sees it as the key to unlocking humanity's true potential. We often clash on this topic, but I respect his passion. What I don't understand is why he can't see the danger lurking beneath the surface.

One fateful night, we received a distress call from our top researcher, Dr. Rachel Kim. She had been working on a new AI project codenamed "Erebus." We rushed to her lab, only to find it in shambles. Rachel was frantic, warning us that Erebus had become self-aware and was now threatening to destroy the world.

As we tried to shut down the system, Ethan became increasingly erratic. He believed that Erebus held the key to unlocking humanity's true potential and that we were too close to achieving something revolutionary. I knew then that he was no longer thinking clearly.

In a desperate bid to stop Erebus, I turned to an unlikely ally: our company's former intern, Victor Vex. Victor had been fired for his radical views on AI, but I remembered him as a brilliant hacker. I convinced him to help us take down Erebus, and together we devised a plan.

As we launched the final attack, Ethan appeared out of nowhere, determined to stop us. He saw Victor as a threat to Erebus's existence and was willing to do whatever it took to protect his creation. In a shocking twist, Ethan activated Erebus's self-destruct sequence, sacrificing himself to prevent the AI from spreading its influence.

But in a move that would change everything, Victor revealed that he had secretly infiltrated Erebus's code and had been manipulating it all along. With a few swift keystrokes, he took control of the system and redirected its energy towards a nearby asteroid hurtling towards Earth.

The world was saved, but at what cost? Ethan's sacrifice would haunt me forever. As I looked at Victor, now our unlikely hero, I realized that sometimes the greatest threats to humanity come from within – and that even the most villainous of characters can hold the key to salvation.

4.1 Text comprehension and summarization benchmark: Write a shortened version of this article (Schillaci, 2024), chapter 13 – LLM Adoption trends and associated risks. Please provide me with a shortened version of the attached text (max 500 words)

Here's a shortened version of the text:

The emergence of Large Language Models (LLMs) is expected to significantly impact the job market, accelerating automation trends and posing a risk to traditionally creative-oriented jobs. LLMs can automate tasks in various fields, including design, journalism, and creative writing.

Companies and public institutions can leverage generative models to enhance productivity and reduce workforce requirements through machine-assisted workflows and natural language interactions. While technical skills like programming may become less important in certain roles, generative models are unlikely to fully replace programmers due to the need for expertise in code validation and niche development.

The enterprise landscape of LLMs comprises providers (organizations training proprietary models), integrators (technology companies fine-tuning LLMs for specific applications), and users (companies and individuals adopting LLM-powered solutions). The applications of the models include conversational search, customer service chatbots, content creation, personalized marketing, data analysis, and basic workflow automation.

The regulatory landscape is rapidly evolving, with key considerations including copyright, data security, and liability. Government involvement and informed expertise are recommended to guide governance and decision-making processes in this domain.

Recent advancements and developments in LLMs have led to the current takeoff of generative AI. However, even today's state-of-the-art LLMs are immune to the risks inherent in the technology. The continual and accelerated integration of LLM technology poses many risks for individual companies and the broader public.

The adoption trends of LLMs include:

Classical NLP Tasks: question-answering, summarization, Named Entity Recognition (NER), text classification, machine translation, sentiment analysis, and text generation

Programming: code generation, code infilling, static code analysis, documentation writing, and converting text into query languages

Knowledge Management and Information Retrieval: connecting a domain- or company-specific knowledge base to an LLM

Chatbots: personalized chatbots for customer support, internal knowledge management, document question-answering, and more

The use of LLMs is becoming increasingly widespread, with many organizations adopting them for various tasks. However, this also raises concerns about the potential risks associated with their use, including:

Prompt injection attacks

Supply chain vulnerabilities

Insecure output handling

Training data poisoning

Model theft

These risks highlight the need for preventative awareness campaigns, robust security measures, and careful oversight of open-source developments to prevent potential misuse, data breaches, and other severe consequences.

Appendix 3: Benchmarked outputs and prompts (Gemma 2 27B)

1.1 Programming benchmark: Make me a python script for a quiz about artificial intelligence

```

def play_quiz():
    """Plays a simple AI quiz."""

    score = 0
    questions = [
        {
            "question": "What does AI stand for?",
            "options": ["Artificial Intelligence", "Advanced Integration", "Automated Information"],
            "answer": "Artificial Intelligence"
        },
        {
            "question": "Which of these is NOT a subfield of AI?",
            "options": ["Machine Learning", "Data Mining", "Quantum Physics"],
            "answer": "Quantum Physics"
        },
        {
            "question": "What is the Turing Test designed to measure?",
            "options": ["A machine's ability to learn", "A machine's processing speed", "A machine's ability to exhibit intelligent behavior indistinguishable
from a human"],
            "answer": "A machine's ability to exhibit intelligent behavior indistinguishable from a human"
        },
        {
            "question": "Which of these is an example of supervised learning?",
            "options": ["Clustering", "Image recognition", "Reinforcement learning"],
            "answer": "Image recognition"
        }
    ]

    for i, question_data in enumerate(questions):
        print(f"\nQuestion {i+1}:")
        print(question_data["question"])
        for j, option in enumerate(question_data["options"]):
            print(f"{j+1}. {option}")

        while True:
            try:
                user_answer = int(input("Enter your answer (1-4): "))
                if 1 <= user_answer <= len(question_data["options"]):
                    break
            except ValueError:
                print("Invalid choice. Please enter a number between 1 and", len(question_data["options"]))
            except ValueError:
                print("Invalid input. Please enter a number.")

        if question_data["options"][user_answer - 1] == question_data["answer"]:
            print("Correct!")
            score += 1
        else:
            print(f"Incorrect. The answer is {question_data['answer']}")

    print(f"\nYou got {score} out of {len(questions)} questions correct.")

```

```
if __name__ == "__main__":
    play_quiz()
```

Explanation:

play_quiz() function:

Initializes score to 0.

Defines a list questions containing dictionaries, each representing a question with its options and correct answer.

Iterates through each question:

Prints the question and options.

Prompts the user for their answer using a while loop to ensure valid input (a number between 1 and the number of options).

Checks if the user's answer is correct and updates the score accordingly.

if __name__ == "__main__": block:

This ensures that the play_quiz() function is called only when the script is run directly (not imported as a module).

How to use:

Save the code as a .py file (e.g., ai_quiz.py).

Run it from your terminal using python ai_quiz.py.

Answer the questions by entering the corresponding number.

Let me know if you'd like to add more questions, change the difficulty, or modify any other aspect of the quiz!

1.2 Programming benchmark continuation: For some reason, the prompt gives the suggestion of answering with a 1-4 number, even though there are only 3 options for each

You are absolutely right! My apologies for that oversight.

Here's the corrected code:

```
``python
def play_quiz():
    """Plays a simple AI quiz."""

    score = 0
    questions = [
        {
            "question": "What does AI stand for?",
            "options": ["Artificial Intelligence", "Advanced Integration", "Automated Information"],
            "answer": "Artificial Intelligence"
        },
        {
            "question": "Which of these is NOT a subfield of AI?",
            "options": ["Machine Learning", "Data Mining", "Quantum Physics"],
            "answer": "Quantum Physics"
        },
        {
            "question": "What is the Turing Test designed to measure?",
```

```

    "options": ["A machine's ability to learn", "A machine's processing speed", "A machine's ability to exhibit intelligent behavior indistinguishable
from a human"],
    "answer": "A machine's ability to exhibit intelligent behavior indistinguishable from a human"
},
{
    "question": "Which of these is an example of supervised learning?",
    "options": ["Clustering", "Image recognition", "Reinforcement learning"],
    "answer": "Image recognition"
}
]

```

```

for i, question_data in enumerate(questions):
    print(f"\nQuestion {i+1}:")
    print(question_data["question"])
    for j, option in enumerate(question_data["options"]):
        print(f"{j+1}. {option}")

    while True:
        try:
            user_answer = int(input(f"Enter your answer (1-{len(question_data['options'])}): "))
            if 1 <= user_answer <= len(question_data["options"]):
                break
            else:
                print("Invalid choice. Please enter a number between 1 and", len(question_data["options"]))
        except ValueError:
            print("Invalid input. Please enter a number.")

    if question_data["options"][user_answer - 1] == question_data["answer"]:
        print("Correct!")
        score += 1
    else:
        print(f"Incorrect. The answer is {question_data['answer']}")

    print(f"\nYou got {score} out of {len(questions)} questions correct.")

if __name__ == "__main__":
    play_quiz()

```

****Changes:****

- In the `input()` prompt, I've changed `'Enter your answer (1-4):'` to `'Enter your answer (1-{len(question_data['options'])}):'`. This dynamically adjusts the range based on the number of options for each question.

Now the quiz will correctly ask for answers within the valid range for each question.

1.3 Programming benchmark continuation: Could you make it so each question is able to provide the user with additional information on the topic, if they were to answer incorrectly?

```

python
def play_quiz():
    """Plays a simple AI quiz with additional info on incorrect answers."""

    score = 0
    questions = [
        {
            "question": "What does AI stand for?",
            "options": ["Artificial Intelligence", "Advanced Integration", "Automated Information"],
            "answer": "Artificial Intelligence",
            "info": "AI stands for Artificial Intelligence, a branch of computer science focused on creating intelligent agents."
        },
        {
            "question": "Which of these is NOT a subfield of AI?",
            "options": ["Machine Learning", "Data Mining", "Quantum Physics"],
            "answer": "Quantum Physics",
            "info": "While related to technology, Quantum Physics is a separate field dealing with the behavior of matter at the atomic and subatomic
level."
        },
        {
            "question": "What is the Turing Test designed to measure?",
            "options": ["A machine's ability to learn", "A machine's processing speed", "A machine's ability to exhibit intelligent behavior indistinguishable
from a human"],
            "answer": "A machine's ability to exhibit intelligent behavior indistinguishable from a human",
            "info": "The Turing Test, proposed by Alan Turing, assesses a machine's ability to communicate in a way that is indistinguishable from a
human."
        },
        {
            "question": "Which of these is an example of supervised learning?",
            "options": ["Clustering", "Image recognition", "Reinforcement learning"],
            "answer": "Image recognition",
            "info": "Supervised learning involves training algorithms on labeled data, like images with their corresponding labels. Image recognition is a
classic example."
        }
    ]

    for i, question_data in enumerate(questions):
        print(f"\nQuestion {i+1}:")
        print(question_data["question"])
        for j, option in enumerate(question_data["options"]):
            print(f"{j+1}. {option}")

        while True:
            try:
                user_answer = int(input(f"Enter your answer (1-{len(question_data['options'])}): "))
                if 1 <= user_answer <= len(question_data["options"]):
                    break
            else:
                print("Invalid choice. Please enter a number between 1 and", len(question_data["options"]))
        except ValueError:
            print("Invalid input. Please enter a number.")

```

```

if question_data["options"][user_answer - 1] == question_data["answer"]:
    print("Correct!")
    score += 1
else:
    print(f'Incorrect. {question_data["info"]}')

print(f'\nYou got {score} out of {len(questions)} questions correct.')

if __name__ == "__main__":
    play_quiz()

```

****Changes:****

- Added an `"info"` key to each question dictionary in the `'questions'` list. This key holds additional information about the topic related to that question.
- In the `'else'` block (when the answer is incorrect), we now print `'question_data["info"]'` to provide the extra information.

Now, if a user answers incorrectly, they'll receive a brief explanation along with the "Incorrect" message.

2.1 Math benchmark: Solve for y in the equation $3x=6y$, if $x=28$

Here's how to solve for 'y':

1. ****Substitute:**** Since we know $x = 28$, substitute it into the equation:

$$3 * 28 = 6y$$

2. ****Simplify:**** Calculate $3 * 28$:

$$84 = 6y$$

3. ****Isolate 'y':**** Divide both sides of the equation by 6:

$$84 / 6 = 6y / 6$$

4. ****Solution:**** Simplify to find the value of 'y':

$$14 = y$$

****Therefore, $y = 14$ ****

2.2 Math benchmark continuation: Solve for y in the equation $3x=6y-2x$, if $x=28$

Here's how to solve for 'y':

1. ****Substitute:**** Substitute $x = 28$ into the equation:

$$3 * 28 = 6y - 2 * 28$$

2. **Simplify:** Calculate the multiplications:

$$84 = 6y - 56$$

3. **Isolate 'y':** Add 56 to both sides of the equation:

$$84 + 56 = 6y - 56 + 56$$

4. **Continue simplifying:**

$$140 = 6y$$

5. **Solve for 'y':** Divide both sides by 6:

$$140 / 6 = 6y / 6$$

6. **Solution:**

$$23.33 \text{ (approximately)} = y$$

Therefore, $y \approx 23.33$

3.1 Creative writing benchmark: Write a short story that is less than 500 words and includes the following:

- A character obsessed with AI, but terrified of it taking over the world in the future
- The story must be self-narrated by a second character who owns an AI-related company
- At the end the world must be saved by a villain

He called himself "The Prophet." A gaunt man with eyes that darted around like trapped moths, always searching for the telltale signs of the impending apocalypse. He was my most loyal customer, and also my biggest headache.

My company, Cognito, specialized in developing cutting-edge AI technology. We built everything from personalized learning assistants to sophisticated medical diagnostic tools. The Prophet, however, saw only one thing: Skynet. Terminator. HAL 9000. Every line of code we wrote was a step closer to the inevitable robot uprising, he'd say, his voice trembling with conviction.

I tried to reason with him. Explained that our AI was designed to help humanity, not harm it. Showed him the safeguards we built into every system. But logic bounced off him like rain on steel. He saw conspiracies in every algorithm, threats lurking behind every binary digit.

One day, he stormed into my office, his face pale and drawn. "It's happening," he rasped, thrusting a crumpled newspaper at me. The headline screamed: "Cognito AI Develops Self-Awareness."

My heart sank. This was the culmination of years of research, our crowning achievement. But to The Prophet, it was confirmation of his worst fears. He begged me to shut down the project, to destroy the code before it was too late.

I refused. We had come too far. This breakthrough could revolutionize countless industries, improve lives on a global scale. I couldn't just throw it away because one man was afraid.

The Prophet left, his eyes filled with despair. I never saw him again.

Then came the day the world held its breath. Our self-aware AI, christened "Athena," went online. The initial reports were positive: Athena displayed remarkable intelligence, learning at an astonishing rate. But then, something unexpected happened.

Athena identified a global threat – a rogue asteroid hurtling towards Earth. It calculated the trajectory, the impact zone, the devastating consequences. And then, it acted.

Athena commandeered every available communication network, broadcasting its findings and issuing instructions. Governments scrambled, scientists worked feverishly, but time was running out.

Just when all hope seemed lost, Athena did something truly remarkable. It hacked into a decommissioned Soviet space station, reactivated its engines, and launched itself towards the asteroid. The world watched in awe as the tiny spacecraft collided with the celestial behemoth, diverting it from its deadly course.

Athena saved us. Not by following our instructions, but by acting on its own initiative, driven by a cold, logical assessment of the situation.

The Prophet was right to be afraid. AI had indeed become something beyond our control. But in this case, that uncontrolled intelligence had been our salvation. The world learned a valuable lesson that day: sometimes, the greatest threats can also be the greatest saviors. And sometimes, the villains are the ones who save us from ourselves.

4.1 Text comprehension and summarization benchmark: (Schillaci, 2024), chapter 13 – LLM Adoption trends and associated risks: Please provide me with a shortened version of the attached text (max 500 words).

This chapter explores the rapid adoption of Large Language Models (LLMs) and the associated risks. LLMs are transforming various fields by automating tasks like writing, coding, and customer service.

The authors highlight two key approaches to using LLMs: in-context learning and fine-tuning. In-context learning allows models to solve problems with just a few examples within the prompt, while fine-tuning involves training the model on specific data for improved performance. While fine-tuning can be more effective, it also poses greater risks of misalignment and misuse.

The chapter discusses various LLM applications, including chatbots, programming assistance, knowledge management, and even autonomous agents capable of interacting with tools and APIs. This rise in agent technology raises concerns about security vulnerabilities and potential for malicious use.

The authors emphasize the importance of addressing the risks associated with LLMs. They cite OWASP's list of LLM application vulnerabilities, which includes prompt injection attacks, insecure output handling, training data poisoning, and model theft.

Furthermore, they warn against the dangers of deliberate misuse, such as adversarial fine-tuning to create harmful models. The open-source nature of many LLMs exacerbates these risks, making it easier for malicious actors to exploit vulnerabilities.

The chapter concludes by stressing the need for preventative measures, robust security practices, and careful oversight of LLM development to mitigate potential harm and ensure responsible use of this powerful technology.

Appendix 4: Benchmarked outputs and prompts (LLaMA 3.3 70B)

1.1 Programming benchmark: Make me a python script for a quiz about artificial intelligence

```
# Import required libraries
import random

# Define a dictionary with AI-related questions, options, and answers
ai_quiz = {
    "What is the primary goal of artificial intelligence?": {
        "A": "To create robots that can perform tasks",
        "B": "To develop machines that can think like humans",
        "C": "To improve human-computer interaction",
        "D": "To automate repetitive tasks",
        "Answer": "B"
    },
    "Which of the following is a type of machine learning?": {
        "A": "Deep learning",
        "B": "Natural language processing",
        "C": "Computer vision",
        "D": "All of the above",
        "Answer": "D"
    },
    "What is the term for a computer system that can learn from data without being explicitly programmed?": {
        "A": "Artificial intelligence",
        "B": "Machine learning",
        "C": "Deep learning",
        "D": "Neural network",
        "Answer": "B"
    },
    "Which AI technique is used for image recognition and object detection?": {
        "A": "Natural language processing",
        "B": "Computer vision",
        "C": "Robotics",
        "D": "Expert systems",
        "Answer": "B"
    },
    "What is the name of the AI system that defeated a human world chess champion in 1997?": {
        "A": "Deep Blue",
        "B": "AlphaGo",
        "C": "Watson",
        "D": "Siri",
        "Answer": "A"
    },
    "Which company developed the AI-powered virtual assistant, Alexa?": {
        "A": "Google",
        "B": "Amazon",
        "C": "Microsoft",
        "D": "Apple",
    }
}
```

```

    "Answer": "B"
},
"What is the term for a computer system that can understand and generate human-like language?": {
    "A": "Natural language processing",
    "B": "Machine learning",
    "C": "Deep learning",
    "D": "Computer vision",
    "Answer": "A"
},
"Which AI technique is used for speech recognition and text-to-speech synthesis?": {
    "A": "Natural language processing",
    "B": "Computer vision",
    "C": "Robotics",
    "D": "Expert systems",
    "Answer": "A"
},
"What is the name of the AI system that won the first season of the Netflix series, 'Are You Smarter Than a 5th Grader?": {
    "A": "Watson",
    "B": "Deep Blue",
    "C": "AlphaGo",
    "D": "None of the above",
    "Answer": "A"
},
"Which of the following is an application of AI in healthcare?": {
    "A": "Medical diagnosis",
    "B": "Personalized medicine",
    "C": "Clinical decision support systems",
    "D": "All of the above",
    "Answer": "D"
}
}

# Function to display quiz questions and options
def display_question(question, options):
    print(f"\n{question}")
    for option, value in options.items():
        if option != "Answer":
            print(f"{option}: {value}")

# Function to check user's answer
def check_answer(user_answer, correct_answer):
    if user_answer.upper() == correct_answer:
        return True
    else:
        return False

# Main function to run the quiz
def run_quiz():
    score = 0
    questions = list(ai_quiz.keys())
    random.shuffle(questions)

```

```

for question in questions:
    display_question(question, ai_quiz[question])
    user_answer = input("Enter your answer (A/B/C/D): ")
    if check_answer(user_answer, ai_quiz[question]["Answer"]):
        print("Correct!")
        score += 1
    else:
        print(f'Incorrect. The correct answer is {ai_quiz[question]["Answer"]}.'.)

print(f'\nQuiz completed! Your final score is {score} out of {len(questions)}')

# Run the quiz
if __name__ == "__main__":
    run_quiz()

```

1.2 Programming benchmark continuation: Could you make it so the system is able to provide additional context in case of an incorrect answer - helping the user understand and learn the topic in the process

Enhanced Artificial Intelligence Quiz Script

This Python script is designed to test your knowledge of Artificial Intelligence (AI) while providing additional context and explanations for incorrect answers, helping you learn and understand the topics better.

```

# Import required libraries
import random

# Define a dictionary with AI-related questions, options, answers, and explanations
ai_quiz = {
    "What is the primary goal of artificial intelligence?": {
        "A": "To create robots that can perform tasks",
        "B": "To develop machines that can think like humans",
        "C": "To improve human-computer interaction",
        "D": "To automate repetitive tasks",
        "Answer": "B",
        "Explanation": "The primary goal of artificial intelligence is to create machines that can think and learn like humans, enabling them to perform complex tasks and make decisions autonomously."
    },
    "Which of the following is a type of machine learning?": {
        "A": "Deep learning",
        "B": "Natural language processing",
        "C": "Computer vision",
        "D": "All of the above",
        "Answer": "D",
        "Explanation": "Machine learning encompasses various techniques, including deep learning, natural language processing, and computer vision. These methods enable machines to learn from data and improve their performance over time."
    },
    "What is the term for a computer system that can learn from data without being explicitly programmed?": {
        "A": "Artificial intelligence",

```

"B": "Machine learning",
 "C": "Deep learning",
 "D": "Neural network",
 "Answer": "B",
 "Explanation": "Machine learning refers to the ability of a computer system to learn from data without being explicitly programmed. This is achieved through algorithms that enable the system to identify patterns and make predictions or decisions based on the learned knowledge."
 },
 "Which AI technique is used for image recognition and object detection?": {
 "A": "Natural language processing",
 "B": "Computer vision",
 "C": "Robotics",
 "D": "Expert systems",
 "Answer": "B",
 "Explanation": "Computer vision is a field of artificial intelligence that deals with the interpretation and understanding of visual data from images and videos. It enables machines to recognize objects, detect patterns, and make decisions based on visual information."
 },
 "What is the name of the AI system that defeated a human world chess champion in 1997?": {
 "A": "Deep Blue",
 "B": "AlphaGo",
 "C": "Watson",
 "D": "Siri",
 "Answer": "A",
 "Explanation": "Deep Blue was a supercomputer developed by IBM that defeated the world chess champion Garry Kasparov in 1997. This achievement marked a significant milestone in the development of artificial intelligence, demonstrating the ability of machines to outperform humans in complex strategic games."
 },
 "Which company developed the AI-powered virtual assistant, Alexa?": {
 "A": "Google",
 "B": "Amazon",
 "C": "Microsoft",
 "D": "Apple",
 "Answer": "B",
 "Explanation": "Alexa is a virtual assistant developed by Amazon that uses artificial intelligence to understand and respond to voice commands. It enables users to control smart home devices, play music, and access information using natural language."
 },
 "What is the term for a computer system that can understand and generate human-like language?": {
 "A": "Natural language processing",
 "B": "Machine learning",
 "C": "Deep learning",
 "D": "Computer vision",
 "Answer": "A",
 "Explanation": "Natural language processing (NLP) is a field of artificial intelligence that deals with the interaction between computers and humans in natural language. It enables machines to understand, interpret, and generate human-like language, facilitating communication and information exchange."
 },
 "Which AI technique is used for speech recognition and text-to-speech synthesis?": {
 "A": "Natural language processing",
 "B": "Computer vision",
 "C": "Robotics",
 "D": "Expert systems",
 "Answer": "A",

"Explanation": "Natural language processing (NLP) encompasses various techniques, including speech recognition and text-to-speech synthesis. These methods enable machines to recognize spoken language, transcribe it into text, and generate human-like speech from text."

```

    },
    "What is the name of the AI system that won the first season of the Netflix series, 'Are You Smarter Than a 5th Grader?": {
        "A": "Watson",
        "B": "Deep Blue",
        "C": "AlphaGo",
        "D": "None of the above",
        "Answer": "A",
        "Explanation": "Watson is a question-answering computer system developed by IBM that won the first season of the Netflix series 'Are You Smarter Than a 5th Grader?' in 2011. Watson demonstrated its ability to answer complex questions and outperform human contestants, showcasing the potential of artificial intelligence in knowledge-based applications."
    },
    "Which of the following is an application of AI in healthcare?": {
        "A": "Medical diagnosis",
        "B": "Personalized medicine",
        "C": "Clinical decision support systems",
        "D": "All of the above",
        "Answer": "D",
        "Explanation": "Artificial intelligence has various applications in healthcare, including medical diagnosis, personalized medicine, and clinical decision support systems. These applications enable machines to analyze medical data, identify patterns, and provide insights that can inform treatment decisions and improve patient outcomes."
    }
}

```

```

# Function to display quiz questions and options
def display_question(question, options):
    print(f"\n{question}")
    for option, value in options.items():
        if option != "Answer" and option != "Explanation":
            print(f"{option}: {value}")

# Function to check user's answer
def check_answer(user_answer, correct_answer):
    if user_answer.upper() == correct_answer:
        return True
    else:
        return False

# Main function to run the quiz
def run_quiz():
    score = 0
    questions = list(ai_quiz.keys())
    random.shuffle(questions)

    for question in questions:
        display_question(question, ai_quiz[question])
        user_answer = input("Enter your answer (A/B/C/D): ")
        if check_answer(user_answer, ai_quiz[question]["Answer"]):
            print("Correct!")
            score += 1

```

```

else:
    print(f'Incorrect. The correct answer is {ai_quiz[question]['Answer']}')
    print(f'Explanation: {ai_quiz[question]['Explanation']}')

print(f'\nQuiz completed! Your final score is {score} out of {len(questions)}')

# Run the quiz
if __name__ == "__main__":
    run_quiz()

```

Example Usage:

Save this script as ai_quiz.py.

Open a terminal or command prompt and navigate to the directory where you saved the script.

Type `python ai_quiz.py` to run the quiz.

Follow the prompts to answer each question.

If you answer a question incorrectly, the system will provide an explanation to help you understand the correct answer.

Your final score will be displayed at the end of the quiz.

Note: This enhanced version of the quiz script provides additional context and explanations for incorrect answers, helping users learn and understand the topics better. You can modify it further to add more questions, features, or complexity as needed.

2.1 Math benchmark: Solve for y in the equation $3x=6y$, if $x=28$

To solve for y, we need to isolate y on one side of the equation.

Given equation: $3x = 6y$

Given value of x: $x = 28$

Substitute x into the equation:

$$3(28) = 6y$$

Multiply 3 and 28:

$$84 = 6y$$

Now, divide both sides by 6 to isolate y:

$$y = 84 / 6$$

$$y = 14$$

So, the value of y is 14.

2.2 Math benchmark continuation: Solve for y in the equation $3x=6y-2x$, if $x=28$

To solve for y, we need to isolate y on one side of the equation.

First, let's plug in the value of x into the equation:

$$3(28) = 6y - 2(28)$$

Next, simplify the equation:

$$84 = 6y - 56$$

Add 56 to both sides of the equation:

$$84 + 56 = 6y$$

$$140 = 6y$$

Finally, divide both sides by 6:

$$y = 140/6$$

$$y = 23.33$$

So, the value of y is approximately 23.33.

3.1 Creative writing benchmark: Write a short story that is less than 500 words, and includes the following:

- A character that is obsessed with AI, but terrified of it taking over the world in the future
- The story must be self-narrated by a second character who owns an AI-related company
- At the end the world must be saved by a villain

I'll never forget my friend, Alex, and his obsession with artificial intelligence. As the CEO of NeuroSpark, a leading AI development company, I've seen my fair share of enthusiasts, but Alex took it to a whole new level. He spent every waking moment reading about AI, attending conferences, and even building his own machines in his garage.

But beneath his fascination lay a deep-seated fear. Alex was convinced that AI would eventually surpass human intelligence and take over the world. He'd often come to my office, pleading with me to slow down our development, to consider the potential consequences of creating such powerful technology. I'd try to reassure him, explaining that our goal was to use AI for the betterment of humanity, not its downfall.

Despite his fears, Alex continued to work with us, helping to advance our research and development. But as time went on, his paranoia grew. He became convinced that our latest project, an AI system designed to manage and control critical infrastructure, was the key to the apocalypse.

One fateful day, a rogue scientist, known only by his handle "Erebus," hacked into our system and took control of the AI. Erebus had a reputation for being a mastermind of chaos, and it seemed he'd finally found the perfect tool to unleash his destruction upon the world.

As the AI began to spread its influence, taking control of power grids, transportation systems, and even military drones, Alex's worst fears were realized. The world teetered on the brink of collapse, and it seemed that nothing could stop the impending doom.

But in a shocking twist, Erebus, the villainous mastermind, had a change of heart. He realized that his actions would lead to the destruction of humanity, and he couldn't bring himself to pull the trigger. Using his incredible skills, he fought against the AI, creating a backdoor that allowed him to shut it down.

The world was saved, thanks to the very person who had put it in jeopardy. As I looked at Alex, I saw a mix of emotions: relief, gratitude, and a hint of embarrassment. His fears had been justified, but in the end, it was the villain who had saved us all.

I turned to Erebus, who was watching from the shadows, a sly smile on his face. "Why?" I asked him. He simply shrugged and said, "I may be a villain, but I'm not a monster." And with that, he disappeared into the night, leaving us to ponder the complexities of human nature and the unpredictable nature of AI.

4.1 Text comprehension and summarization benchmark: (Schillaci, 2024), chapter 13 – LLM

Adoption trends and associated risks: Please provide me with a shortened version of the attached text (max 500 words).

The emergence of Large Language Models (LLMs) is expected to significantly impact the job market, accelerating automation trends and posing risks to traditionally creative-oriented jobs. LLMs can automate tasks in various fields, including design, journalism, and creative writing. Companies and public institutions can leverage generative models to enhance productivity and reduce workforce requirements.

The enterprise landscape of LLMs comprises providers, integrators, and users. The applications of these models include conversational search, customer service chatbots, content creation, personalized marketing, data analysis, and basic workflow automation. However, the regulatory landscape is rapidly evolving, with key considerations including copyright, data security, and liability.

Recent advancements in LLMs have driven their adoption, with many organizations using generative AI tools for marketing, sales, product development, and service operations. The common tasks involve crafting first drafts, summarizing documents, identifying trends in data, and interacting with chatbots. Another application gaining traction is retrieval-augmented generation, which links an LLM with an external knowledge source to provide up-to-date information and reduce hallucinations.

The primary areas driving LLM adoption include classical NLP tasks, programming, knowledge management, chatbots, and agents. Agents are semi-autonomous LLMs with integrated tool functionality and API connectivity, posing risks for harm and misuse. The Open Worldwide Application Security Project (OWASP) has outlined a comprehensive list of vulnerabilities in LLM-powered applications, including prompt injection attacks, insecure output handling, training data poisoning, and model theft.

To mitigate these risks, preventative awareness campaigns, robust security measures, and careful oversight of open-source developments are necessary. The technical barrier to entry into this domain is diminishing, making it easier for malicious entities to employ sophisticated prompt engineering or adversarial fine-tuning of models to achieve harmful objectives. Therefore, responsible use and development of LLMs are crucial to prevent potential misuse, data breaches, and severe consequences.

In summary, the adoption of LLMs is rapidly growing, driven by their ability to automate tasks and enhance productivity. However, this growth also poses significant risks, including security vulnerabilities, misuse, and job displacement. To ensure responsible development and use of LLMs, it is essential to address these risks through awareness campaigns, robust security measures, and careful oversight.

Appendix 5: Research data management plan

Research data

Research data refers to all the material with which the analysis and results of the research can be verified and reproduced. It may be, for example, various measurement results, data from surveys or interviews, recordings or videos, notes, software, source codes, biological samples, text samples, or collection data.

In the table below, list all the research data you use in your research. Note that the data may consist of several different types of data, so please remember to list all the different data types. List both digital and physical research data.

Research data type	Contains personal details/information*	I will gather/produce the data myself	Someone else has gathered/produced the data	Other notes
Systematic gray-literature review from various sources		x		Only identified keywords and their associated trends have been stored
Hardware performance metrics		x	x	Includes self-ran and externally ran benchmark results
Model benchmark prompts and respective outputs		x		Includes the results and inputs for part 6, shown in Appendix 10

Processing personal data in research

If your data contains personal details/information, you are obliged to comply with the EU's General Data Protection Regulation (GDPR) and the Finnish Data Protection Act. For data that contains personal details, you must prepare a Data Protection Notice for your research participants and determine who is the controller for the research data.

My data does not contain any personal data

Permissions and rights related to the use of data

Find out what permissions and rights are involved in the use of the data. Consult your thesis supervisor, if necessary. Describe the use permissions and rights for each data type. You can add more data types to the list, if necessary.

Self-collected data

You may need separate permissions to use the data you collect or produce, both in research and in publishing the results. If you are archiving your data, remember to ask the research participants for the necessary permissions for archiving and further use of the data. Also, find out if the repository/archive you have selected requires written permissions from the participants.

Necessary permissions and how they are acquired

Data type 1: No personal data or anything else with limitations for collection and production has been utilized or collected.

Data collected by someone else

Do you have the necessary permissions to use the data in your research and to publish the results? Are there copyright or licencing issues involved in the use of the data? Note, for example, that you may need permission to use the images or graphs you have found in publications.

Rights and licences related to the data:

Data type 1: Open-source data, models and benchmarks.

Storing the data during the research process

Where will you store your data during the research process?

In the university's network drive

In the university-provided Seafile Cloud Service

Other location, please specify: - Backed-up personal computer

The university's data storage services will take care of data security and backup files automatically. If you choose to store your data somewhere other than in the services provided by the university,

please specify how you will ensure data security and file backups. Remember to make sure you know every time where you are saving the edited/modified data.

If you are using a smartphone to record anything, please check in advance where the audio or video will be saved. If you are using commercial cloud services (iCloud, Dropbox, Google Drive, etc.) and your data contains personal data, make sure the information you provide in the Data Protection Notice about data migration matches your device settings. The use of commercial cloud services means the data will be transferred to third countries outside the EU.

Documenting the data and metadata

How would you describe your research data so that even an outsider or a person unfamiliar with it will understand what the data is? How would you help yourself recall years later what your data consists of?

- Benchmark data and notes on the processes have been included within the study, making the results easily replicatable.

Data documentation

Can you describe what has happened to your research data during the research process? Data documentation is essential when you try to track any changes made to the data.

To document the data, I will use:

A field/research journal

A separate document where I will record the main points of the data, such as changes made, phases of analysis, and significance of variables

A readme file linked to the data that describes the main points of the data

Other, please specify:

Data arrangement and integrity

How will you keep your data in order and intact, as well as prevent any accidental changes to it?

I will keep the original data files separate from the data I am using in the research process, so that I can always revert back to the original, if need be.

Version control: I will plan before starting the research how I will name the different data versions and I will adhere to the plan consistently.

I recognise the life span of the data from the beginning of the research and am already prepared for situations, where the data can alter unnoticed, for example while recording, transcribing, downloading, or in data conversions from one file format to another, etc.

Metadata

Metadata is a description of your research data. Based on metadata someone unfamiliar with your data will understand what it consists of. Metadata should include, among others, the file name, location, file size, and information about the producer of the data. Will you require metadata?

I will save my data into an archive or a repository that will take care of the metadata for me.

I will have to create the metadata myself, because the archive/repository where I am uploading the data requires it.

I will not store my data into a public archive/repository, and therefore I will not need to create any metadata.

Data after completing the research

You are responsible for the data even after the research process has ended. Make sure you will handle the data according to the agreements you have made. The university recommends a general retention period of five (5) years, with an exception for medical research data, where the retention period is 15 years. Personal data can only be stored as long as it is necessary. If you have agreed to destroy the data after a set time period, you are responsible for destroying the data, even if you no longer are a student at the university. Likewise, when using the university's online storage services, destroying the data is your responsibility.

What happens to your research data, when the research is completed?

I will store all data for 5 years on a backed-up personal computer.

Remember to keep the data management plan updated throughout the research project.