Whitepaper

# How to Prioritize LLM Use Cases

by Rens Dimmendaal, Juan Venegas, Katherine Munro, and Eva Bosma

# Table of Contents

# Introduction

## Recent Changes in the AI Landscape

Artificial Intelligence (AI) is not a technology but a goal: getting a computer to perform a task as well as a human, or even better. Many programs have achieved a human-like level of ability in tasks ranging from playing chess to deciphering patterns in data. But when it comes to Generative AI (GenAI) and the creation of new content, the results have been mediocre at best. Until a few years ago.

ChatGPT, with its incredible capacity to write text and answer questions, changed everything. As other tools before, it arguably surpassed human capacity at some tasks (see Figure 1); at others, it generated good enough outputs much faster than humans. It is no wonder that the speed of adoption for ChatGPT surpassed those of the world's most-used applications, like Netflix or Instagram (Figure 2). As the technology behind Large Language Models (LLMs) develops, and the landscape of supporting tools matures, the impact will grow even further.

This brings us to some important questions: *can and should you integrate Large Language Models, like those powering ChatGPT, Gemini, and other advanced chatbots, into your company? If so, how? Where is the balance between quality, speed, and cost in text generation?*

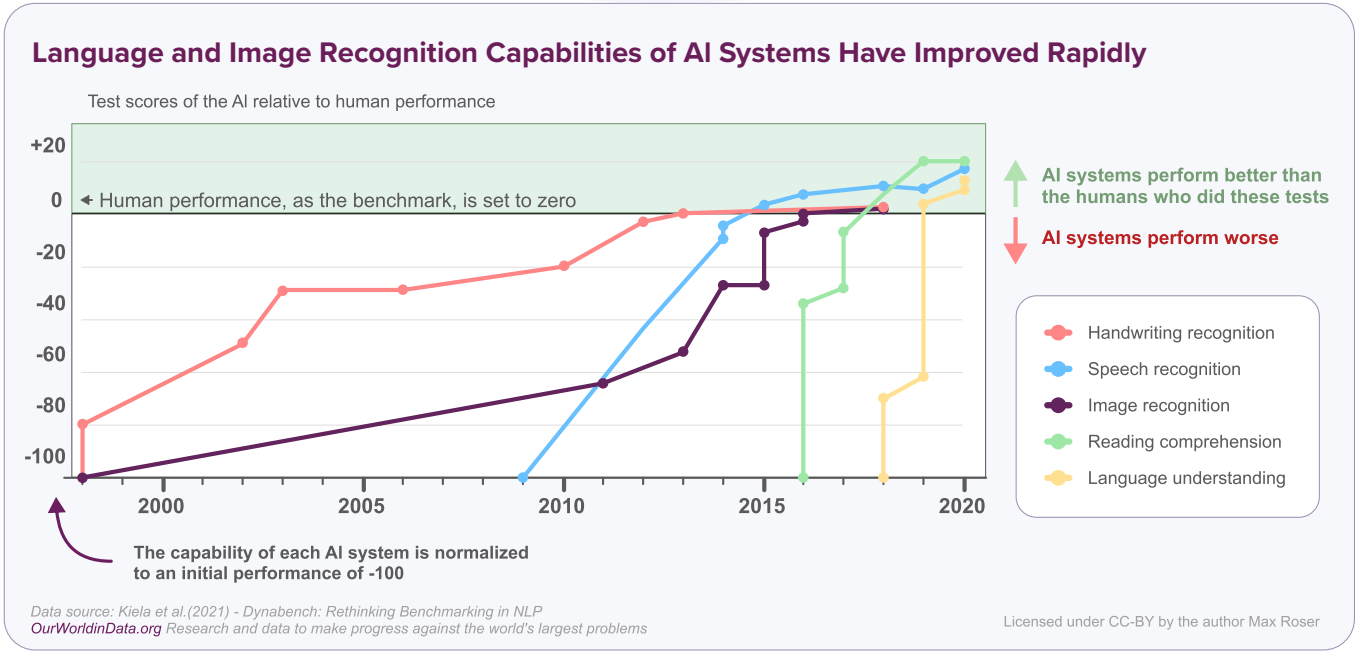



**Figure 1:** Evolution of language and image recognition AI systems and comparison to human performance according to standardized tests.

This whitepaper will show how to assess the impact and feasibility of LLM use cases for your organization. We will present the main suitability, impact, and feasibility drivers, and how they determine the kind of LLM and architecture required to perform a task. You will learn to guide your colleagues' thinking about AI use cases and to create a well-prioritized roadmap. Best of all, you will be able to determine whether a seemingly crazy idea from a colleague is a ticket to real value or whether you need to go back to the drawing board. Happy reading!

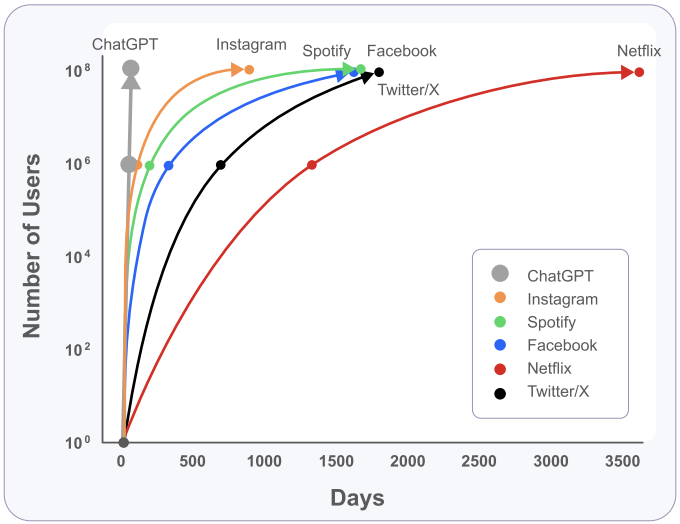



**Figure 2:** Rate of adoption of ChatGPT compared to other famous tools. The lower dots represent the moment these technologies reached 1M users, while the upper dots represent the moment they got 100M users.

*Graph from Kyle Hailey (@kylelf_ on X)*

## What's the Business Impact?

Large Language Models are far from being a fleeting trend. They have already impacted the market significantly, and even the most conservative predictions anticipate many changes in the way we work.

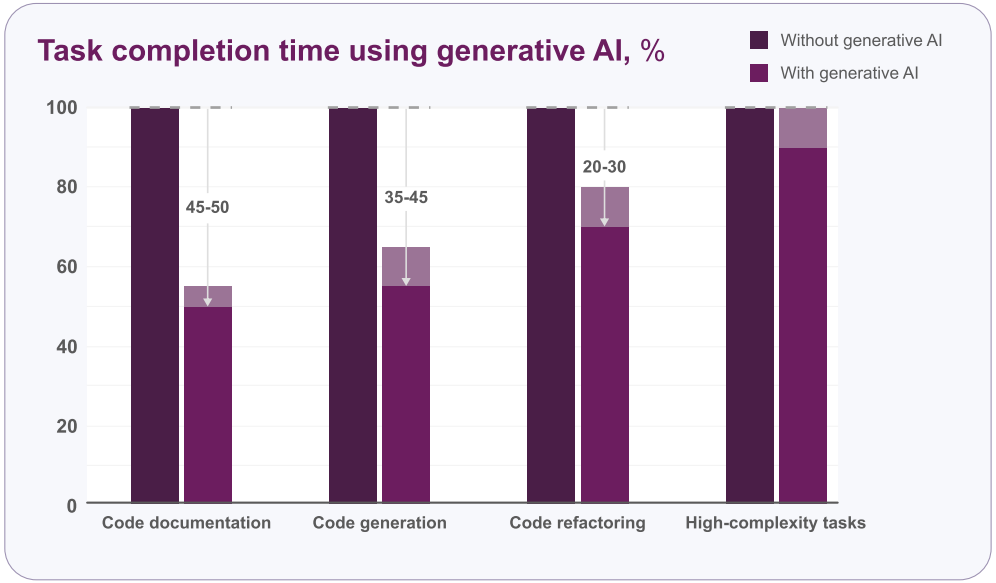
For starters, the traffic on Stack Overflow, the popular website that offers answers to programming questions, decreased by 14% the month after ChatGPT was released[1]. While some bloggers have exaggerated this percentage, it is still early to assess the long-term impact.

Regardless of the exact numbers, LLMs have improved one process: searching for help while creating code. Whether programmers use an LLM directly or a code assistant powered by an LLM, they can save much time in their work, as shown in the research from McKinsey and Company in Figure 3. Code assistants generate quick code prototypes based on a programmer's description. Some of the most famous examples are displayed in Figure 4.

But many other tasks could be automated: data cleaning, content creation, or even tutoring. A study from OpenAI, OpenResearch, and The University of Pennsylvania showed that 80% of US jobs could have more than 10% of their tasks partially automated. Additionally, 19% of the interviewed professionals could automate at least half of their tasks.

And finally, many companies and institutions currently ban the use of the technology or monitor it heavily. Many university professors and teaching assistants struggle to determine whether their students have really penned their essays. At the same time, literary publishing companies have added a new disclaimer to their submission policies: "No content written by GenAI will be read."

[1] *Insights into Stack Overflow's traffic*




Credit: McKinsey Digital. Unleashing developer productivity with generative AI

**Figure 3:** Percentage of time saved to achieve specific tasks. Dark bars show the time without the use of a code-generating assistant (normalized to 100%); light bars show the time with the use of an assistant. Numbers indicate a decrease in time percentage.


GitHub Copilot | Amazon's Code Whisperer | Google's Codey | Hugging Face's Starcoder

**Figure 4:** Code-augmentation solutions. Some of the most popular tools.

# Background

### From AI to LLMs

We have said Artificial Intelligence is about teaching computers to do human-like tasks. Here, we mean tasks that require decision-making, creativity, or planning: classifying new data points, synthesizing information to generate new knowledge, or developing strategies based on expected rewards, for example. Of course, these skills are not unique to humans (as many pet owners will attest), but we are rather good at them. Moreover, we tend to get better at them as we gain experience.

We all have read enough regular and junk emails to easily classify one with words like “credit card,” “bitcoin,” and “virus” as spam. We have watched enough TV to know what the clichés are for different genres, to the point where we just know which characters are getting together and which ones are not making it out of the haunted house alive.

It is easy to take our advanced learning skills for granted. Yet, teaching a computer to perform these tasks is hard. We can give them hard-coded rules, but that proves to be incredibly complex. A much more promising approach is teaching computers to be pattern-recognizers like us. We call this **Machine Learning**.

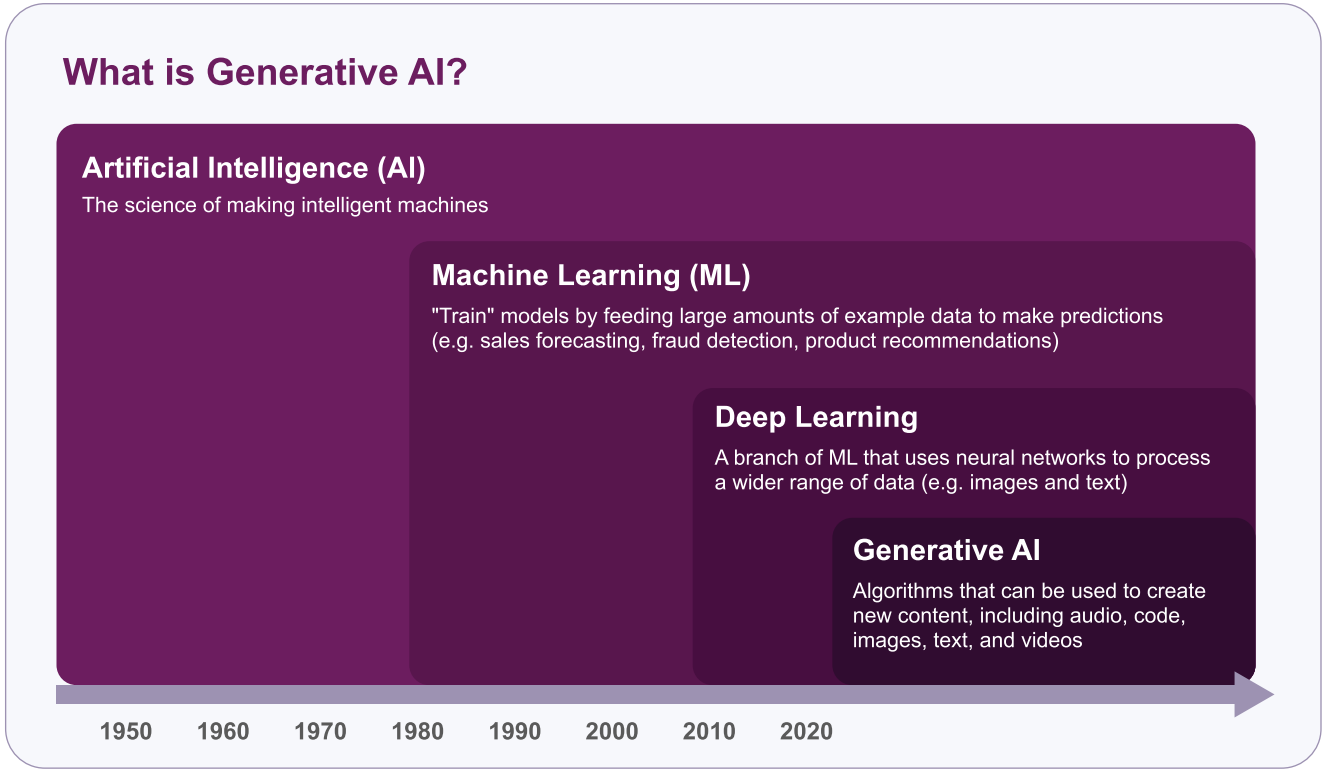



**Figure 5:** Classification of AI systems.

Machine Learning (ML) algorithms are sets of steps written in code to look at some data and then produce a specified output. Whenever the output is wrong, the algorithm runs again. The result of this ‘training’ is also saved in the code, and it is called a ‘model.’ A trained model has learned patterns in the data, without us having to teach it explicitly. In fact, the patterns are often too complex for us to identify, so we use ML algorithms to find them for us.

A classic ML example is that of house price prediction. Features like the number of bedrooms and bathrooms or the proximity to essential services influence house prices. What is not clear is exactly how these features affect the price. The relationships between them are too complex for us to decode, even if we had access to thousands of records of house features and their historic sales price. Instead, we input those records into an ML algorithm, which learns the patterns for us. Figure 6 shows the step-by-step process.

Many ML techniques work well when the data consists of tables with numeric columns (number of bedrooms, number of garages, etc.) and a single target column for prediction (house price). When the task exceeds these limitations, we often turn to Deep Learning (DL), the branch of ML that uses neural networks to learn highly complex patterns in a broader range of data, including images and text. DL excels at tasks like identifying objects in an image (Computer Vision), transcribing spoken audio to text (Speech Recognition), or converting a sentence from one language to another (Translation). All these examples require unstructured data, and the relations between inputs and outputs are highly complex.

Now we come to Generative AI: teaching computers to create content. For example, if we show a model thousands of pictures of an apple and then ask it to draw one, it might draw a red, shiny shape about the size of a tennis ball. It will take thousands of false attempts, but through the process of trial and error and getting feedback, it'll learn the pattern. It can do the same for other types of content, too, like music, videos, and more. It just needs the right kind of algorithm and a lot of example data.

Next up in our AI landscape are Large Language Models, or "LLMs," a type of Generative AI solution designed to create text. To understand how LLMs work, think of those fill-the-gap exercises you did in school, where you were given a text with certain words missing, and you had to choose the best word to fit them (see Figure 7). When we make a machine learning algorithm repeat a similar exercise hundreds of thousands of times on enormous amounts of text, we get a Large Language Model. This model is great at identifying the statistically most probable word for any given context – a very useful skill, as you will see next.

...

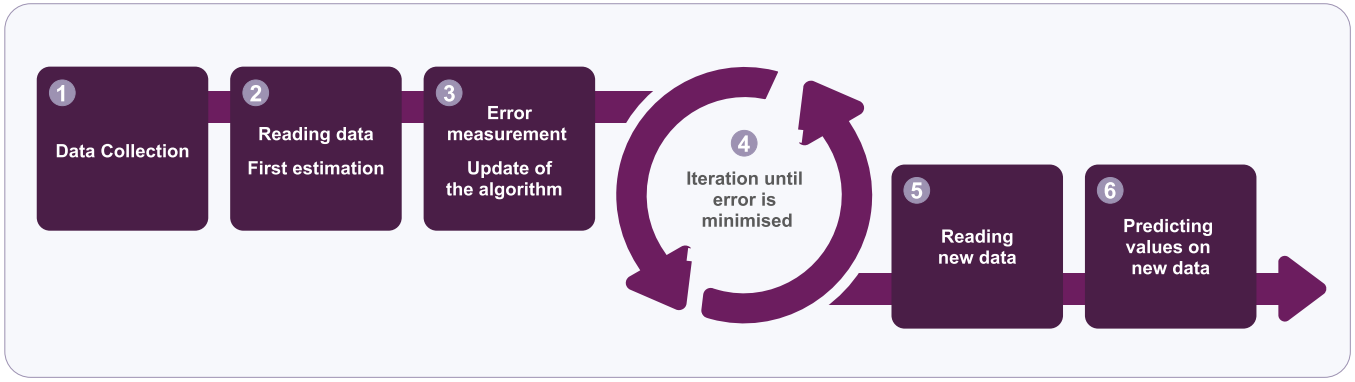



**Figure 6:** Predictive AI example; prediction of house sale prices. Different factors drive house prices: size, location, or number of rooms, among others. We may use an ML algorithm to learn these patterns and then predict the price for any new house on the market. This is only one simple example of an ML algorithm, there are many others that are far more complex. e.g. algorithms with back propagation.

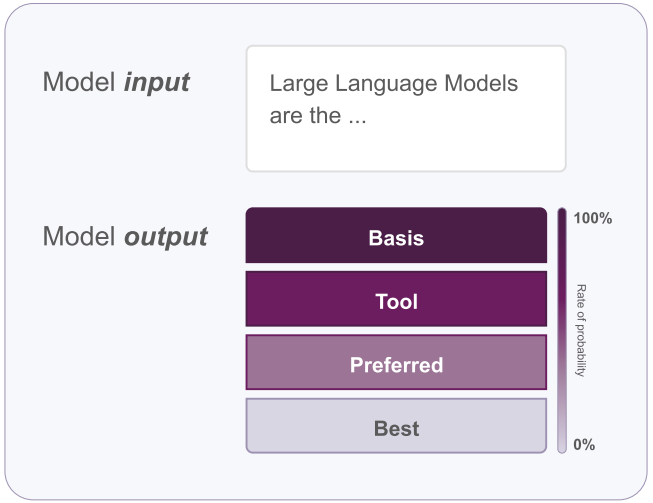



**Figure 7:** Guessing words with LLMs. We give an untrained model some text, mask out the last word, and make it guess what's missing. We repeat this over an enormous amount of data until it gets the words right.

At last, we return to ChatGPT, Gemini, and company. Large Language Models are the basis for these tools: after creating an LLM, we 'fine-tune' the model to answer user requests. Fine-tuning is another Machine Learning technique: getting the computer to repeat a task over and over until it gets it right. The difference here is that instead of asking the LLM to predict the missing words, we ask it to predict the answer to a user request. Eventually, if we ask, "What's the capital of France?" the model learns to answer, "Paris" (after thousands of random guesses first, like "person," "woman," "man," "camera," or "TV"). If we ask, "Generate a fairytale," it'll learn to answer, "Once upon a time..." and continue from there. All the amazing feats of text generation you have seen start with the question: what's the most likely word to come next?

# GenAI Beyond Text

LLMs are excellent at answering questions or suggesting code. But there is plenty more that Generative AI is capable of.

**Images:** Tools like Stable Diffusion, Midjourney, and DALL-E can create photo-realistic images, artistic 'paintings,' and graphic designs based on user-provided textual instructions, known as prompts.

**Music:** Tools like Google's MusicLM and Stable Audio create music based on speed, style, length, instruments, and a specified mood, such as upbeat or sad. Others let you sing, hum, or tap a tune to get started, while some create entirely new music from text prompts, like 'relaxing music to study to.'

**Video:** Powerful AI algorithms can generate video from text prompts, still images, or even existing videos. Brands can create conversational eCommerce and customer support pieces with tools like D-ID to animate faces. On the other hand, educational content creators can bring their tutorials to life by using tools like Runway's Gen-2, which generates entire videos just from text descriptions.

**Almost everything else:** There are many other modalities of GenAI. Here are a few examples:

- Speech Synthesis: This includes text-to-speech and speech-to-text applications, such as the ones used in customer service call centers.
- 3D Modelling: GenAI can also be used for architectural and vehicle design, product prototyping, gaming, or animation.
- Biological simulations and protein design are two prime examples of how life sciences research benefits from GenAI.

# Suitability Drivers of LLMs

### Intro to Suitability Drivers

Johnson and Scholes (1997)[2] stated that one of the prime purposes of strategic analysis is to gain a clear understanding of the organization and the environment in which it operates. This requires looking at major opportunities and threats, as well as the strengths and weaknesses of the organization. They also highlight that expectations are an important influence on strategic choices.

They identify **suitability** as a measurement of how well a proposed strategy aligns with the situation identified in the strategic analysis and its potential to maintain or enhance the organization's competitive position.

For LLM use cases, it is crucial to evaluate how well the solution fits within the organizational values. This involves considering whether it aligns with the organization's goals and strategy, as well as assessing its compatibility with future AI regulations, such as the EU AI Act. Additionally, it is essential to understand the expectations and preferences of users and consumers, such as whether they would be interested in interacting with a GenAI solution.

[2] *JOHNSON, G., and SCHOLES, K. (1997). Exploring Corporate Strategy, Fourth Edition, Prentice Hall, New York. [Chapter 8]*

### Strategy

It is key to validate whether the LLM use case fits into the organizational strategy and enhances it. Consider organizations that focus on building strong personal relationships with clients, such as providers of high-end luxury goods or personalized financial services. In these industries, relying solely on an LLM solution for communication with clients (chatbot) may detract from the personalized and human touch[3] central to the organization's strategy. So, even though an automated chatbot might serve your customers promptly, it might not fit how you want to be perceived as an organization.

Next to that, many organizations already have a formal policy on how to use (generative) AI. The main reason behind this is that some organizations (and individuals) have made significant errors while using generative AI. At Samsung, for example, some workers accidentally leaked secret data while using ChatGPT[4], and two New York lawyers used fake ChatGPT case citations in legal briefs[5].

It is therefore important to verify whether the identified LLM use case aligns with the organizational policies.

[3] *Marketoonist on chatbots and the future of customer experience*

[4] *Samsung workers made a major error by using ChatGPT*

[5] *Sanctions ordered for lawyers who relied on ChatGPT artificial intelligence to prepare court brief*

### Laws & Regulations

In March 2024, members of the European Parliament adopted the EU AI Act, the world's first comprehensive AI law.

The use of generative AI is also highlighted in this AI law, meaning that, should your LLM use case become active within the EU market, you must validate whether the LLM use case is in line with the laws and regulations.

Since different regions might have different rules regarding generative AI, you must always check the latest information within your region before you start working on an LLM-powered use case.

### User Perception

Finally, consider your end-users and customers when creating an LLM use case. In some cases, you might be excluding demographic groups from using your solution. Consider banks or financial institutions that go fully digital. The older generations might be unable to access their services as comfortably as others[6,7].

Next to that, consider the following questions:

- *Does our end user want to interact with a generative AI solution?*
- *Do we foresee any risks of generating non-inclusive content?*
- *Have we tested our models against potential bias?*

[6] *The Big Read: As banks go big on digital banking, spare a thought for seniors left behind*

[7] *The Big Read in short: Impact of digital banking divide on seniors*

| | | |
|---|---|---|
| **1. Strategy** | • Mission & Vision<br>• GenAI Policy | How does the LLM use case enhance your organizational strategy? Is it in line with your policies of using Generative AI? |
| **2. Laws and Regulations** | • Current vs. Expected<br>• AI Act | What current and expected laws and regulations concern the LLM use case? Is the use case assessed against Trustworthy AI principles? |
| **3. User Perception** | • Inclusivity & Fairness<br>• Transparency | Have we tested the model against bias? (e.g., racial, gender, age). Are there any demographic groups excluded from using the solution by including the use of GenAI? Does our end user want to interact with a GenAI solution? |

**Table 1:** Checklist for Suitability considerations.

These questions are not supposed to restrain the organization from developing LLM use cases; they are here to balance all the pros and the cons, so the LLM case is a good fit for how the organization wants to be perceived.

### Conclusion on Suitability

In the next section, we will discuss the impact of an LLM on your organization, but first, you need to validate whether the use case is a good fit (see Table 1). Looking into the organizational strategy, laws, and regulations and verifying whether the LLM use case is a fit for your end user is crucial.

# Impact Drivers of LLMs

### Intro to Impact drivers: the problems LLMs solve

Let us consider an eCommerce website with thousands of products.

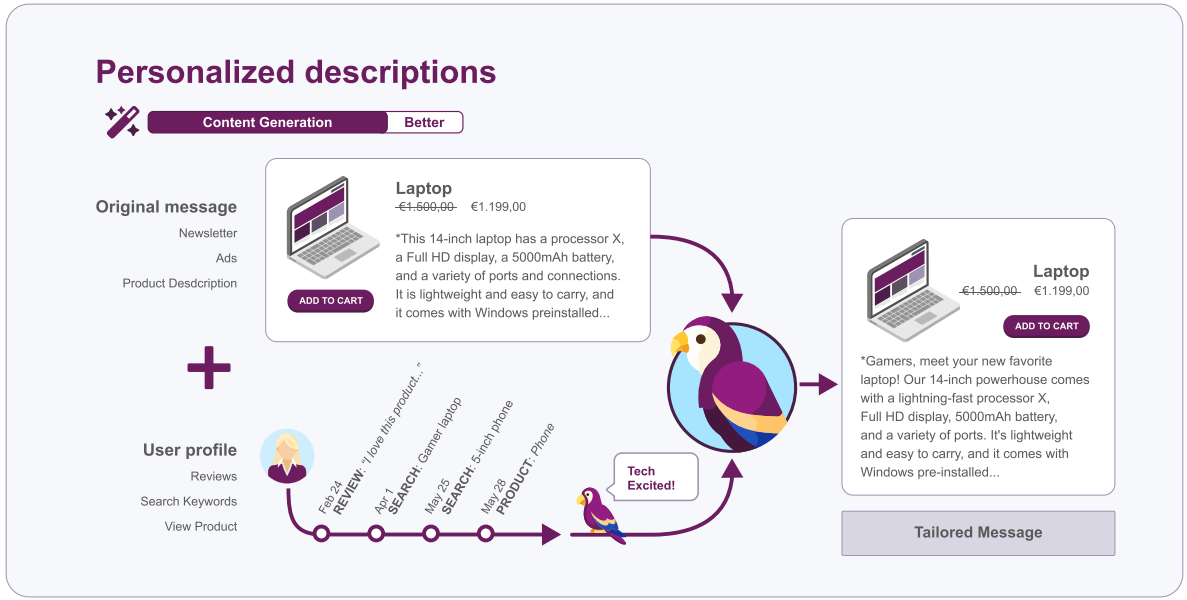



**Figure 8:** Example generation of personalized descriptions.

Every time a new product is uploaded, a description of it is needed. Furthermore, this website may offer different, personalized descriptions of each item based on the user profile. Writing all of these is exhausting for a human, but LLMs can help with the job.

The LLM combines a generic product description with some basic behavioral data from the client. As seen in Figure 8, if a user has searched for gaming laptops in the past, the ad copy they'll see for a new computer will include a reference to gaming. Will the LLM do a better job than a (trained) human? Absolutely not. However, waiting for a description from the copywriter until they can tackle the task would result in losses for the company.

This example is key to understanding the impact of LLMs. They might do worse than human experts but provide a better service than other non-human alternatives. If you combine this with the fact that LLMs are cheaper and faster than humans, you can see how significant the impact on our economy could be.

### Value Type: Faster, Better, Cheaper

We have just shown that LLMs may provide a better service so long as no human is available to give a task their full attention. We will now illustrate this principle with two current examples.

- Many professionals need a personal assistant but may be unable to afford one. Can LLMs fully replace a personal assistant? No. Can LLMs be used to power some services that a personal assistant does? Yes, drafting automatic replies to your e-mails, for example (Figure 9).

- Doctors receive numerous questions from patients during the day, and it is only understandable that they might not come up with the most empathetic answer every single time. An LLM can help them draft considerate answers very fast.

  Figure 10 shows how ChatGPT outperformed physicians in giving empathetic, accurate responses. While a doctor giving their full attention to the question will probably come up with a better answer, ChatGPT can help them speed up the process. This way, doctors can put together a good-quality, empathetic answer much faster.

When it comes to generating content or any other piece of text, asking an LLM to write it for us is cheaper than employing a person for a few minutes or hours.

However, the relevant question is, can we optimize profits if we reduce quality slightly but costs immensely?

Consider a web store with a long tail of products in their portfolio, each needing a description for their dedicated page. For the most popular products, it makes sense to employ a copywriter and get a powerful, appealing description of the item. But for a product sold rarely and for a small amount of money, the investment of time and work from the copywriter might not be worth it. For those products, using an LLM that generates good-enough copy at a fraction of the cost makes sense.

Let us go back to the example of the web store with many products. LLMs can write an item description much faster than a human, but the output will probably be of a lower quality. Would this be acceptable to the client? If the answer is yes, using an LLM could be helpful.

Some stores receive many new products every day. If any of those are not on the website by the time the user expects to find it there, the store loses revenue. Therefore, it makes sense to have a quick version of the copy written by an LLM, at least until a copywriter can update it. In this case, LLMs help with the urgency factor.

There are other times, however, when the LLM can help a human work faster, even though their work might not be urgent.


**Figure 9:** Example of smart autocomplete in Obsidian text editor. From our colleague Jordi Smit's Obsidian AutoComplete plugin.

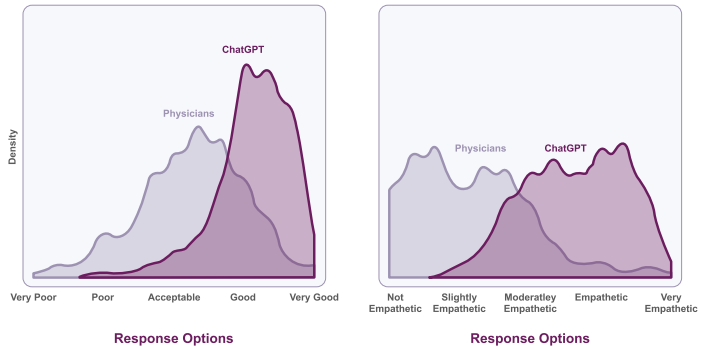



**Figure 10:** UC San Diego study. "Study Finds ChatGPT Outperforms Physicians in High-Quality, Empathetic Answers to Patient Questions."

*Likert scales for quality and empathy level of responses show how ChatGPT (in dark purple) outperforms doctors (in pink), according to the surveyed patients.*

*Graph adapted from the article.*

For example, it can draft responses for customers so that a support agent may tackle a larger number of issues per hour. Researchers Brynjolfsson, Li, and Raymond (2023) studied the response rate from 5179 customer support agents and measured an average 14 percent increase in productivity when using an LLM to help them draft responses.

This increase was higher for novice agents and barely noticeable for seasoned workers. Their full results appear in the article Generative AI at Work.

### Cost: Build, Run, and Improve

Building a solution using LLMs involves several key considerations, including model architecture, data collection and preprocessing, training and finetuning, and integration with existing systems. The time and resources required for this phase can vary significantly based on the complexity of the use case and the scale of the LLM implementation.

Once the LLM solution is deployed and operational, ongoing running costs come into play. These costs include infrastructure expenses (e.g., cloud computing resources, storage, and bandwidth), licensing or subscription fees for LLM services, and potential costs associated with data acquisition and maintenance.

In addition to running costs, it is crucial to factor in the engineering costs required to maintain and improve the LLM solution over time. This includes monitoring model performance, addressing drift and degradation, updating the model with new data, retraining for improved accuracy, and implementing enhancements or new features based on evolving use-case requirements.

### Return on Investment

Integrating a generative AI use case within the organization is a huge change management effort. When done correctly and for the proper use case, value can arrive quickly; when done incorrectly, generative AI might bring a loss in revenue[8]. The assessment of 'time to value' and business expectations associated with implementing LLM use cases is therefore essential to ensure that the deployment of LLMs aligns with the organization's strategic goals and delivers tangible value within a realistic time frame.

[8] *https://www.bcg.com/publications/2023/how-people-create-and-destroy-value-with-gen-ai/*

### Conclusion on Impact

Compared to human professionals performing a task, it is safe to treat LLMs as a slightly worse but much cheaper and faster alternative. To determine whether that is a desirable trade-off, we should assess whether that change can result in:

- **Better** services, when properly integrated with a human in the loop, or at least new services when the human alternative is too expensive to consider.
- **Cheaper** services, because the cost savings are worth it compared to the existing solution.
- **Faster** services, because the speed increase is worthwhile compared to the existing solution or because it helps to react to urgent matters.

Once there is clarity on the expected value type, it is important to agree on how to measure the expected value. What KPIs are relevant and when would the business case be successful? Next to value, it is also important to consider the associated costs. How much does it take to build a first solution? What about the running costs once it goes live and engineering costs required to maintain and improve the solution? All these factors influence how fast the investment in the LLM use case will return actual business value. Comparing the return on investment with business expectations is therefore an important final step in the checklist for impact.

| | | |
|---|---|---|
| **1. Value Type** | ○ Faster<br>○ Better<br>○ Cheaper | • How does the use case generate business value?<br>• What KPIs are improved and by how much?<br>• How will the value-type be measured? |
| **2. Cost** | ○ Build<br>○ Run<br>○ Improve | • How much does it take to build a first solution?<br>• What about Run the run costs once it goes live?<br>• Have we factored in the Improve engineering costs required to maintain and improve it? |
| **3. ROI** | ○ Expectations | • How fast should the use case be profitable?<br>• Are the expectations from the business met? |

**Table 2:** Checklist for impact considerations.

# Copyright Issues

Copyright with LLMs is a tricky topic being discussed by governing bodies and legal experts, and it might not be resolved anytime soon. After all, similar questions, like the case of the monkey selfie[9], have been around for years. For now, we may clarify a few burning questions:

[9] *Monkey Selfie Copyright dispute*

### Do you give up the copyright of your own content if you input it into a model like Midjourney or Gemini?

For example, what happens if you copy-paste some of your writing into Midjourney and ask it for editorial-style feedback?

**TL;DR:** you don't give up the copyright of your content, but you don't have exclusive rights to its usage anymore.

**Full story:** it depends on the T&Cs of the tool. Usually, the user grants non-exclusive rights to the tool vendor to use the content for tool improvement and training. The T&Cs are often written broadly so that the tool vendor can e.g., amend, sublicense, or even commercialize the input you gave the tool. So, if your prompt is so good that OpenAI wants to use it in a promo video, they'd be allowed to.

### Who owns the copyright of the output of a GenAI model?

**TL;DR:** There are no hard rules. Each case will require assessing whether the prompts are creative enough that the output couldn't have been created without them.

**Full story**: If you ask for an image in Van Gogh style, you won't be able to copyright the outputs. But if your prompt is incredibly detailed and creative, then the copyright is probably yours. All of this is theoretical, however. Usually, the tool provider will specify the rules up front to avoid long and costly case-by-case assessments. For example, they may grant you non-exclusive rights, so if someone gets a similar output to yours, they can use it without infringing your copyright.

### Who enforces the rules?

The tool provider will take no responsibility for what the tool creates: the user must check what they're allowed to do. For example, editing someone else's copyrighted work is generally prohibited.

### What about enterprise solutions?

With closed tools, the situation is different: you train the tool and control the data, so there will normally be an agreement to ensure the outputs are yours alone.



# Feasibility Drivers of LLMs

## Intro to Feasibility Drivers

Working with LLMs may look easy, but bringing them into production inside real organizations is not. There are multiple design patterns to choose from when applying LLMs in production, including whether to use an existing LLM as-is or to augment it with additional source documents, tools, or training on your own data (fine-tuning). These and many other factors will impact the feasibility of your proposed solution.

Assessing the feasibility of a use case early on will give you the best chance of building an effective, scalable, and maintainable product or service. While this is true of software development in general, it is even more important for LLM-based applications, given the rapid developments in the technology, its capabilities, and the shifting business expectations around them.

In the next subsection, we'll present some common design patterns for applying LLMs in production. Then, we will outline feasibility drivers to determine which pattern is the most realistic and impactful for your use case.

## Design Patterns for LLM-Based Solutions

The Generative AI solutions we have seen and built at Xebia can be classified into three levels, as shown in Figure 11.

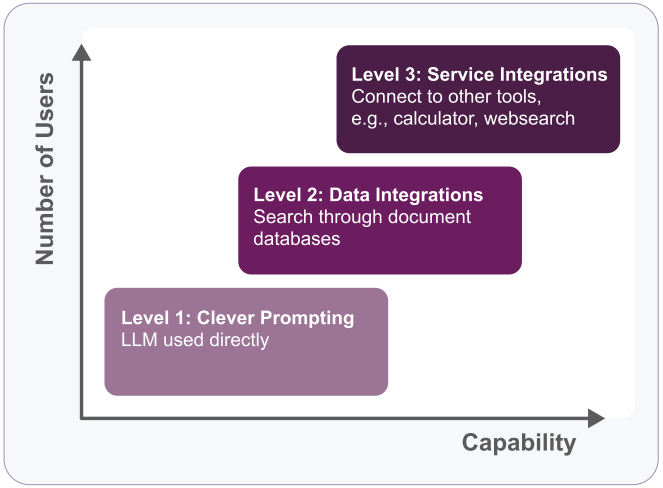



**Figure 11:** GenAI solutions organized by complexity and capability.

- **Level 1 – LLM used directly**
  With just some clever prompting, LLMs are already capable of tasks that, in the past, would have required multiple machine learning models. For example, instead of training your own sentiment analysis and topic recognition models, you can now ask a single LLM to extract such information from an input text and output it in a structured format.

  You can give the model a product review and get back a JSON with the customer sentiment and any of the review details. We classify this kind of use case into "level 1" applications: the LLM answers your request based solely on internal knowledge and the context you provide.

  The barrier to using level 1 applications is low. Many chatbots are available for free using a simple web interface or an API.

- **Level 2 – Data integrations**
  In this architecture, known as "Retrieval Augmented Generation" (RAG), the LLM accesses additional data sources, e.g., search engines or databases of internal company documents.

  A common way to do this is to split a corpus of source texts into chunks and then embed each one into semantic vectors, that is, representations in a numerical space. For example, for a customer-service question-answering application, the source texts might be documents about company products and policies. When a customer types in a question, this input is embedded using the same strategy as the source documents. Then, a vector similarity search is used to find the most similar source document chunks. These chunks are then provided as context to the LLM, along with the original input question, and the LLM returns an answer based on those sources.

  Choosing the optimal chunking and embedding strategy brings additional complexities to developers. Finding a solution takes time and experimentation.

- **Level 3 – Service integrations**
  LLMs themselves cannot execute tasks. In level 3, agent-style applications, LLMs are integrated with external services to take complex prompts, break them down into tasks, and execute them using pre-defined tools to which they have access. For example, the LLM can make a restaurant reservation for you by extracting the restaurant name, date, and number of people from your request and passing them on to a restaurant-booking API.

  Level 3 becomes even more challenging because errors may accumulate as the LLM interacts with other tools. For example, a model that is 90% accurate might drop to 60% by the time we reach the end of the chain. There's also an inherent security risk in allowing an LLM to access external tools. Thus, any level 3 application requires inbuilt control mechanisms that limit the tools the LLM can use, the actions it can take, and the information it can share.

The additional capabilities from each level increase complexity and costs. To choose a specific design pattern, consider the following feasibility drivers, which may even rule out some options for your specific use case. Let us dive into them now.

### Users: Internal vs External

With any proposed software, data, or AI project, it is always wise to start by thinking about the user: who they are and what they need to accomplish with your solution. This is particularly true of LLM-based use cases, where the scope of what people might do with the tool and the kind of outputs they might get is so broad.

Many LLM applications still need a human in the loop, either an internal or an external user. Working with internal users is easier for two reasons:

- **Responsibility** (protecting your users): when you set up an LLM, you are responsible for the user's experience. This is easier to handle with internal users because they can give you feedback easily. With external users, it is more difficult to determine how they intend to use the service, making it harder to even define what a "good" or "correct" result is, let alone build tests for that.
- **Security** (protecting your company): external users may try to abuse the service or escape any guardrails you build into it. For example, in a level 2 system, they might attempt to extract the source databases the LLM consults. Internal users will ideally not do that, although if you want to create thorough tests, you may challenge them to try.

### Task: Specific vs Generic

The next feasibility driver is the type of task: specific or generic. A generic tool requires a generic interface, which is much harder to implement than a specific one. For example, in the case of a chatbot, a user can input almost anything, which makes it tricky to predict how they will use the tool and what outputs they might get. This, in turn, makes it harder to build the interface in a way that guides users to a great experience. A more specific use case, such as rewriting text in a specific style with the click of a button, has a narrower scope. This makes it easier to anticipate what could go wrong for the user.

It is also easier to validate LLM outputs for specific use cases. For example, if you specify that the outputs should be in JSON format, you can systematically verify that, indeed, they are. Doing such verification means end-users get reliable outputs[10] and gives you the confidence to use the model as part of a larger system.

Similarly, it is easier with specific tasks to create metrics to assess whether a model is working correctly. An LLM-powered system can consist of many parts: the base model, the tuning data and strategies, the prompt, and the data connections (without even considering integrating other source documents or additional services). Assessing whether one config is better than another is much easier when you have a specific task in mind. For example, if you are creating a copywriting bot, you could check the readability score. Such evaluation becomes harder with more generic LLMs and use cases.

[10] *Enforce and Validate LLM Output with Pydantic*

For more info on the UI of GenAI, watch this great talk by Linus Lee.

### Data Sensitivity

When building applications with LLMs, it is essential to consider what data you are using in the system. The sensitivity levels affect feasibility because they affect what control mechanisms you must implement.

We distinguish between three types of sensitive data:

01. **Non-sensitive data:** public data.
02. **Company-level sensitive data:** data that is not public but accessible to all users.
03. **User-group-level sensitive data:** data that is not public and only accessible to subsets of your users.

#### Non-sensitive data: caveats that always apply

Even if you think you do not have sensitive data in your system, you must remember that users may input sensitive data into your system. For example, if you have a chat application, there's no way of knowing what a user might enter.

You always need to consider who has access to your system data. There are two major actors: your cloud provider and your company's employees.

Different cloud providers have different terms and conditions for their LLM services. For example, the default configuration for Azure is that a Microsoft employee may look at your prompt and model-generated responses if they are flagged by an abuse filter.

When you build an LLM application, you will also generate logs containing potentially sensitive user inputs. You need to set some policies so that only a trusted set of employees is allowed to review those logs.

#### Company vs. User-group sensitive data

When we work with sensitive data, these can be company or user-group sensitive data.

With user-group sensitive data, you must not allow users to access outputs generated from data they should not have access to. This additional effort affects feasibility.

For example, you must implement some data controls for a level 2 application that includes Retrieval-Augmented Generation. Only data sources the user can access may be considered and integrated into the LLM call. This is feasible even when the user groups are very complex.

If you use finetuning models, remember that LLMs can return their training data when prompted. As a result, you must train different models for different user groups. If the user groups are overlapping or very fine-grained, this will result in many models having to be trained, making the solution not very feasible.

### Automation – Human in the Loop vs Fully Automated

Another critical driver for LLMs is whether you need a human in the loop. If you trust the LLM outputs completely without user reviews or can automate the tests you want to run on the output (sometimes called guardrails), then the use case is more feasible. If you need a human to review the outputs, and this is not part of your regular workflow, then you increase complexity.

The following questions can help you figure out how to verify your LLM outputs and whether you will need a human check:

- How do you ensure that outputs reflect your company's tone and values? Do you need a communications expert to check this?
- Who is the end-user, and are they an expert in the domain you are generating content for?
- Can you enable the end user do their own verification? For example, in a level 2 application, you can highlight passages of text that the LLM used to create its answer.
- How will you prevent AI bias in the outputs, such as gender-biased texts and translations or images that perpetuate gender, racial, and ability biases?
- What is the potential impact of your use case? The consequences of producing false or incomplete information in a medical question-answering tool are far more serious than that of a chatbot that makes film recommendations. If there is a material impact on people's lives, you need a human in the loop.

Even if you do not need a human to review all outputs (that is, your use case is largely automated), you may still want to spot-check the workings of your system, whether for compliance reasons or to look for opportunities to improve it. This leads to a follow-up question: Do you need outputs to be transparent and evidence-backed? LLMs are notorious for making things up with convincing confidence. Using a model that's been finetuned on domain-specific data might help here, but there's still a chance that, when asked about topics entirely out of its training data, an LLM might fall back to hallucination. In RAG applications, on the other hand, the retrieval phase functions as a resource, ensuring that the LLM is supplied with accurate data for its potential outputs. This is a form of seperate not offered by black-box LLMs (whether finetuned or off the shelf), and it can help increase the feasibility of putting your use case into production.

## Model: Off the Shelf – Finetuned

The next feasibility factor is: who's building the model? The simplest way to start working on a Proof of Concept (PoC) for your use case is to use an off-the-shelf model via an API. These models have two main benefits: they're easy to get started with and generally perform well. However, there are also reasons to move away from these models.

Another option is to finetune your model on your own data. This may offer benefits in terms of model performance and run-time costs: smaller finetuned models may outperform larger off-the-shelf models. All this comes at a feasibility cost, however, as this option requires the most work in data preparation, model training, evaluation, monitoring, and so on.

Note that a finetuned model can be used in any of the three levels of application we have described in *'Intro to Impact drivers: the problems LLMs solve', page 9*, of this article. For example, you could finetune a model on your existing customer service inquiries so it learns your business's vocabulary and tone. But you could also use this model in a RAG application, providing it with the most up-to-date company information to answer customer inquiries. This comes with the complexity of training any machine learning model plus some additional challenges: acquiring good quality, labeled text data and evaluating the trained model's outputs. So, remember to factor this in when considering the feasibility of powering your application with a finetuned LLM.

## Batch vs. Real-time

The speed to produce outputs also affects the feasibility of your LLM use case. For user-facing applications, speed is crucial for retention and revenue; for internal applications, it impacts employee efficiency and productivity. Unfortunately, LLMs can be prohibitively slow, as you've probably noticed if you ever waited for them to generate an output for your request (Table 3).

This is especially true for RAG applications, which require first embedding the user input and identifying useful source documents and then waiting for the LLM to generate a response from them.

**Chat GPT3.5 Turbo vs Chat GPT 4**

| n_ input_ tokens | p50 in seconds | p75 | p90 |
|---|---|---|---|
| 100 | 0.035 | 0.036 | 0.042 |
| 500 | 0.038 | 0.043 | 0.045 |
| 1000 | 0.040 | 0.053 | 0.099 |

| n_ input_ tokens | p50 in seconds | p75 | p90 |
|---|---|---|---|
| 100 | 0.23 | 0.26 | 0.48 |
| 500 | 0.22 | 0.28 | 0.58 |
| 1000 | 0.24 | 0.29 | 0.45 |

**Table 3:** LLM applications can be prohibitively slow, as shown in the latency of the tests with GPT-3.5-Turbo for GPT-4. Jordi Smit ran these experiments, feeding these two models the same number of tokens. Then, he created a distribution of times based on the results. The table shows the percentile 50, 75, and 90 of these distributions for the number of seconds per token in these requests.

Speed of response is key for user retention and revenue. For example, a one-second slowdown at the BBC website could result in a 10% drop in viewers[11]. Additionally, for Amazon, a 100msec slowdown decreased sales by 1%[12]. This illustrates that speed matters.

If you can perform any of your required workflows using batch-based processing in advance, your use case will likely be more feasible. For example, if you build a news-feed service featuring summarized news articles, you could batch-process them early in the morning, so they are ready when users require them.

[11] https://www.creativebloq.com/features/how-the-bbc-builds-websites-that-scale/

[12] *Linden 2006, retrieved from Trustworthy Online Controlled Experiments: A Practical Guide to A/B Testing (Kohavi, Tang, Xu 2020) and Online controlled experiments at Large Scale (Kohavi et al 2013)).*

## Other Considerations

Apart from the feasibility drivers we have discussed so far, other factors might influence how you build your LLM solution. Some of them are:

- **Evaluation, which can be component-wise or unit evaluation:** For the component-wise case in a RAG system, the retrieval component can be evaluated in isolation to determine its efficiency in sourcing the best content chunks. Additionally, the response of the LLM is scrutinized to check if, given the best source, it can yield a quality response.To implement any of the evaluations mentioned above, you need to create a reference dataset for evaluation.
- **Maturity of the tech stack:** Many new tools from the open-source community and from cloud providers are out on the market. However, everything is still very case-specific, and there are no mature components or platforms for LLMOps (building LLMs systematically into production and at scale). Furthermore, depending on what you want to do, there might not be any tool that can support you; you might have to build it yourself. This is true regardless of the type of architecture you choose. Once you have used the other feasibility drivers to home in on the best architecture, you can dive into specific tools to suit that structure and your planned use case.
- **Scalability:** RAGs are, in principle, more scalable as they are modular, and you only need to connect them to a bigger/more diverse set of source documents.
- **Metadata:** By providing metadata about the chunks passed in the context, the LLM can better understand the context, potentially resulting in improved output.
- **Additional Context and Knowledge**: If available, knowledge graphs enhance RAG applications by providing further context at query time, which enables the system to generate more accurate and informative responses.
- **Number of LLMs:** Sometimes, a single existing LLM might perform at 85% accuracy on a big range of tasks, which may be good enough for your company. If it is not, you will require multiple LLMs.

Table 4 shows a thorough checklist with questions you can ask yourself for impact and self-assess feasibility.

| | | |
|---|---|---|
| **1. Level** | ◦ L1 - Just a Prompt<br>◦ L2 - RAG<br>◦ L3 - Agents | **Agents** link prompts, resulting in a compounding effect of errors, which negatively impacts reliability. |
| **2. User** | ◦ Internal<br>◦ External | **Responsibility** is easier with internal users. You can reach them more easily. And they are less likely to abuse your service. |
| **3. Task** | ◦ Specific<br>◦ Generic | **UIs, metrics, and output validators** are easier to make for specific tasks than for generic assistants. |
| **4. Data** | ◦ Public<br>◦ Company sensitive<br>◦ User sensitive | **Terms & Conditions** for using off-the-shelf models should be carefully read. Tuning should only be done on public or company specific data. |
| **5. Automation** | ◦ Human in the Loop<br>◦ Fully automated | **Reliability,** of LLM outputs is far from perfect. You need a higher accuracy to go Fully Automated. |
| **6. Model** | ◦ Vendored via API<br>◦ Open Source<br>◦ Tuned Models | **Easier to get started** when consuming a standard model via an API. |
| **7. Deployment** | ◦ Batch<br>◦ On-Demand | **Latency** of LLMs is slow. The faster a response is required the less feasible a use case is. |

**Table 4:** Checklist for feasibility considerations.

# Your Turn

This article shows you how to evaluate LLM-powered use cases by assessing their suitability, impact and feasibility.

The next step is up to you. Get your team together. Brainstorm what processes can be made better, faster, or cheaper in your organization. Assess the feasibility of these ideas and put them on a map. Discuss with your team and clarify where you have different perceptions. Determine where and why your assumptions differ. Once you are on the same page, start making decisions. Which use cases should you pick up now, which ones are shortlisted to start after that, and which ones are left for later?

**Remember:** the checklists provided in this whitepaper are not rocket science. They are simple tools that you can use to get people from different backgrounds together so you can come to a common understanding of what has value.

*For a quick assessment of the feasibility of different cases, refer to Table 5.*

| Harder use cases | Easier use cases |
|---|---|
| L2 - RAG<br>L3 - Agents | L1 - Just a prompt |
| On Demand | Batch |
| Fully automated | Human in the loop |
| External users | Internal users |
| General Assistant | Specific taks |

**Table 5:** Summary of harder versus easier use cases.

# Want to Learn More?

If you want to learn more about the implementation of LLMs, check out this podcast from Rens Dimmendaal, a consultant at Xebia and one of the leading experts in the technology. There are also blog posts on How to Extract Structured Data from Unstructured Text using LLMs, Dataset Enrichment Using LLMs, and Takeaways from the LLMs in Production Conference. If you feel ready to implement the technology in your business, check our Solutions Excellence page.

If you have enjoyed this whitepaper and are constantly striving to implement data technologies at your company, we have many more for you at xebia.com. Download our whitepaper on Data Democratization to understand how to make data available to your employees and how to train them to leverage it. You can also find out the state of your company with this free AI Maturity Self-Assessmenty.

Do you feel that you are missing a role in your organization? Check out our whitepaper on The Analytics Translator, one of the key professionals in any data transformation and the best person to connect your data and your business.

# How Xebia Can Help You Win With LLMs

Xebia is an IT Consultancy and Software Development Company that has been creating digital leaders across the globe since 2001. With offices on every continent, we help the top 250 companies worldwide embrace innovation, adopt the latest technologies, and implement the most successful business models.

## When it comes to LLMs, there are two main ways in which Xebia can help you:

- **Quick wins**
  Do you need to introduce LLMs in your company from scratch? Then, let us guide you through our easy 3-step process. First, we will conduct an Ideation workshop to inspire people and create an initial roadmap. Second, we will assess and select the appropriate data for the project. Finally, we will build and test the technology with you. All in under a month!

- **Solutions at scale**
  Once your first GenAI projects are up and running, we will integrate them into the architecture and design the appropriate interface for them. We will also train your end-users and implement the right processes to ensure a responsible, beneficial use of the technology.

## Do you need a customized solution?

Our consultants and trainers will be happy to tailor our offers to your needs.
Find out more at xebia.com/genai