



Typical Pitfalls and Key Principles of Generative AI in Technical Disciplines

Essential Insights for Harnessing AI Tools Like ChatGPT

Summary: Generative Artificial Intelligence software such as ChatGPT is increasingly being used as a versatile tool in technical areas. However, the “naive” use of this software may entail several pitfalls. These can be avoided if the basic principles of these AI algorithms are known. With this knowledge, users can not only achieve better results when applying the technology but also make more informed decisions about the strategic and practical use of generative AI. This article uses simple examples to illustrate the typical misunderstandings that can occur on the part of the human user as well as the fundamental limitations of so-called large AI language models. The principles underlying these misunderstandings and limitations are explained to provide an intuitive understanding of the current state of generative AI.

Target Audience: This article is intended for novice users of generative AI as well as those who have already used it but like to know more about the inner workings of Large Language Models. This is neither a prompt engineering tutorial nor are we going to dive deep into the various features of today's chat AI products. Instead, this article focuses on the core principles common to all of those.

Introduction

F.: “Write a short introduction to this article that very briefly addresses the motivation why systems engineers could also work with generative AI. Formulate it factually and concisely for technically trained readers.”

ChatGPT: “[...] Generative AI, as used in models such as ChatGPT, offers promising potential.

With the ability to process extensive text data and generate relevant information, this technology can help engineers to create documentation, analyse technical specifications and develop creative solutions to complex problems. The targeted use of generative AI could therefore enable significant efficiency gains and qualitative improvements in system development.”

This is an impressive example of what few outside the AI world would have expected just over two years ago. In 2018, the first versions of so-called large language models were still unable to generate more than a few coherent words before falling into meaningless strings of letters. Surprisingly, the leap in improvement was less due to further development of the core algorithms but more to the extreme size of the underlying neural networks and the huge amount of text used to train them. The introduction above was generated by ChatGPT 4o and is the result of more than 1.7 trillion ($1.7 \cdot 10^{12}$) neural parameters trained with approximately 5 trillion words (estimates).

Despite this power, using this tool needs to be learnt. And this is easier if you understand the basic functionality of ChatGPT and co. This article therefore highlights pitfalls that new AI users typically encounter and that are particularly relevant for technical tasks. To gain an intuitive understanding of how generative AI works, its core principles are explained. For a better understanding, the article is limited to working with chat prompts in ChatGPT, but the principles are the same for most other current large language models. In addition, a brief introduction of advanced application techniques and AI architectures is given.

First Steps

Let's follow the first steps of our colleague "H." using generative AI. H. is an experienced systems engineer who has in-depth knowledge of the *Systems Engineering Modeling Language* SysML. Recently a new version 2 of that language was released, facing H. with about 1,000 pages of specification to read. H. has heard about the possibilities offered by *Large Language Models* (LLMs) such as ChatGPT and wonders if it could help working through that specification.

The first place to go is the sleek ChatGPT website, which essentially consists of a single input field, where H. types in his first question:

H.: "Hello, I'm H., a systems engineer. Can you help me understand the new version 2 of the SysML systems modeling language?"

ChatGPT: "Absolutely, H.! SysML (Systems Modeling Language) version 2 is a major upgrade over SysML V1, designed to address some of the limitations of its predecessor and provide a more robust foundation for systems engineering. Can I tell you more about a specific topic?"

H. is pleasantly surprised by the AI's friendly tone and a lively dialogue ensues, in which H. talks less about those specifications than about himself, how he came to the task and what difficulties he is worried about. ChatGPT seems to be an attentive listener and asks helpful questions. Until H. realises that he has strayed somewhat from his actual question and ChatGPT is basically just answering in very general terms.

Pitfall #01: The ELIZA effect.

We tend to attribute human characteristics such as understanding, logical thinking, experience or empathy to a machine just because it can "talk".

Basic principle: This psychological effect was named after the ELIZA software introduced in 1966. It is because we have evolved among humans and that these are the only entities known to us that use language like we do. ELIZA was the first software you could - apparently - communicate with in English. It mimicked a psychotherapist, essentially recognising a few key words in the input and using them to formulate questions in the simplest possible way. The software contained no intelligence or knowledge base. Nevertheless, people could "converse" with the software for hours and attributed human characteristics to it. In some cases, even after the principle behind it was explained to them. This frustrated the developer of ELIZA, Joseph Weizenbaum, and he subsequently became a well-known critic of technology-focussed societies [1].

Tip: Today's language AIs are far more complex than ELIZA and we need to actively make ourselves aware that it is still just a machine (or rather a software). It may help if we refrain from using phrases that we would use in human conversations, at least in chats that serve a technical purpose. Examples are "Hello", "Could you please", "Thank you, that was great.". The machine has neither motivation nor ego and will generate the same results without complaint or grudge.

Next H. tries to get a summary of a *Risk Analysis and Assessment Modelling Language* (RAAML) that he is working on. He needs a list of the language's formally defined core concepts for an article he is writing.

H. "Briefly explain the most important RAAML Core Concepts and give the SysML/UML concepts each one maps to."

ChatGPT: "In RAAML, core concepts such as Risk, Hazard, Risk Assessment, Control Measure, Traceability, Risk Level, Likelihood, Severity, Risk Control, Mitigation, Verification, and Validation play crucial roles in safety and reliability analysis. Risk is mapped to a SysML Block, representing potential negative events that could impact the system. Hazards, linked to SysML Requirements and Constraints, are conditions that increase the likelihood of risks. [...]"

H. almost copies this text for his article but quickly realises that something is wrong here. The terms listed do represent conceptual aspects of risk analysis in the broadest sense. However, strange as it may seem, Risk is not a formal model element of the Risk Analysis and Assessment Modelling Language [2] and therefore can't be mapped to any SysML model element. Most of the other terms in ChatGPT's text are similarly incorrect.

Later in this article we will understand why language models may produce factually incorrect output. For now, let's look at why we tend to overlook these mistakes.

Pitfall #02: Fluency heuristic.

Just because it's well written doesn't mean it's correct.

Basic principle: Mental heuristics help us to understand the world even though we lack information. We make assumptions that help us come to quick conclusions that are often good enough to fulfil a task in everyday life. Our heuristic of fluency allows us to assume that a text that is easy to read and understand is also valuable and factually correct [3].

Tip: AI results must always be checked for factual correctness. However, it is sometimes not easy to assess which is less work: checking the results or doing all the work yourself without AI support.

How „It“ does (not) work

H. tries to find out why ChatGPT has chosen these wrong concept terms and whether it knows the correct ones, the most important of which is the *Situation*.

H.: "What about Situation?"

ChatGPT: "In RAAML, core concepts such as Risk, Hazard, Risk Assessment, Control Measure, Traceability, Risk Level, Likelihood, Severity, Risk Control, Mitigation, Verification, Validation, and Situation are crucial for safety and reliability analysis. Risk, mapped to a SysML Block, represents [...] Situation, mapped to State and Use Case in SysML, represents scenarios or conditions under which risks are assessed and managed. [...]"

H. is surprised that the answer is not specific to the model element *Situation*. Instead, ChatGPT repeats the same incorrect answer from before, supplemented by likewise incorrect information on Situation (underlined).

In another attempt, H. tries to explain to ChatGPT where it was wrong, hoping for a correction:

H.: "But this is wrong. Don't you see that Situation is a Core Concept model element and most others are not? Also your mappings are wrong."

ChatGPT: "In RAAML, the core concept "Situation" is a significant model element specified within the Core Library package. "Situation" is mapped to a SysML State and Use Case and represents scenarios or conditions under which risks are assessed and managed. Other essential RAAML core concepts include Risk, Hazard, [...]"

H. had not imagined working with the highly praised AI to be so frustrating.

Pitfall #03: LLMs don't understand. LLMs do not think.

LLMs are word predictors, not thinking machines.

Basic principle: What appears to be an understanding of meaning is actually "just" a sophisticated mathematical "trick" (or rather a whole series of very clever ideas). One of the central components of ChatGPT, for example, is the actual "word model" (Fig. 2). During training, it is assumed that words that frequently occur near each other in the training texts share the same context and therefore a similar meaning. The transformer component uses that word model

to find the most suitable next word for an input text (the so-called prompt). In this step, several levels of abstraction, i.e. different perspectives of a word's meaning, are considered. In fact, only a single word is generated at first. In the next step, the prompt plus that generated word is again used as input to find the next suitable word. This continues until an end criterion is reached and a text is output (Fig 1). [4]

That is actually (almost) everything. There is no simulation of thought processes, there is no explicit knowledge or fact base and no rules that explain how language works. It has been shown that the amazing abilities of LLMs correlate strongly with their size.

Recently so-called "reasoning models" have been added to the increasing range of language AI products. They try to mimic what is called a "chain of thoughts" and they can solve more complex problems step-by-step and by trying out alternatives. It is important to understand, however, that this capability is achieved by internally running LLMs many times (Fig. 2, bottom). This results in higher resource consumptions and longer response times than standard models. Also, everything said here about LLMs is still applicable to reasoning models.

Tip: Computer scientist Katharina Zweig has found a good metaphor for LLMs: "ChatGPT is autocorrect on steroids" [5].

H. now suspects that the incorrect answers were because the relevant specification had not been included in the training data. He has heard that ChatGPT can also be taught new things, for

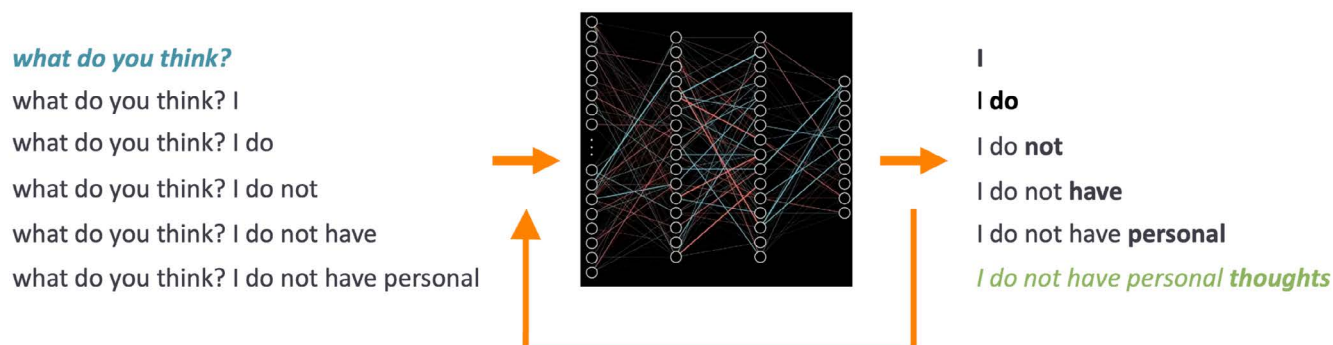


Figure 1. LLM's responses are generated word by word using the same process

example, by uploading a PDF file. Indeed, the correct Core Concept model elements are now listed. On this basis, H. can generate a simple example model that did not look bad at all.

The next day, H. opens a new ChatGPT window to continue working on that model. Unfortunately, he now got similarly incorrect results, just like at the beginning of the previous day.

Pitfall #04: LLMs do not learn during a chat. LLMs do not learn from past chats.

The learning of language and context happens only in a separate, one-off training phase.

Basic principle: Training an LLM is a separate, very resource-intensive phase that can take months (Fig. 2, left). A window is moved word by word over the entire training texts (Fig. 3). With each step, the neural networks involved learn to predict the next word. In the process, relationships between all words within the current window are calculated (*attention* concept [6]). This area is therefore referred to as the context or context window. Instead of words, LLMs actually use word components, so-called

tokens [4], but this is of little importance for a basic understanding.

The language model can only be used after the training phase. It is then no longer possible to add or remove data. During use, LLMs take the prompt as a context to generate the following words based on their training data. For each new prompt, this and all the previous text of *the current chat* is used as if it were all a single new prompt. There is no explicit “mental model” of the previous conversation and no algorithms that mimic a step-by-step thought process.

We have seen that new information can indeed be provided within prompts or by uploading files. However, this just further restricts the context of the query and thus reduces the search space of the LLM. This may lead to better answers because it guides the LLM's focus. Even though the actual LLM is not changed, it is sometimes referred to as learning, e.g. with the *in-context learning* method, where examples are provided in the prompt [4].

It should be noted that the entire context of a chat is limited in length. For ChatGPT 4, this is a maximum of 8K (8192) tokens, which corresponds to approx. 6,000 words for

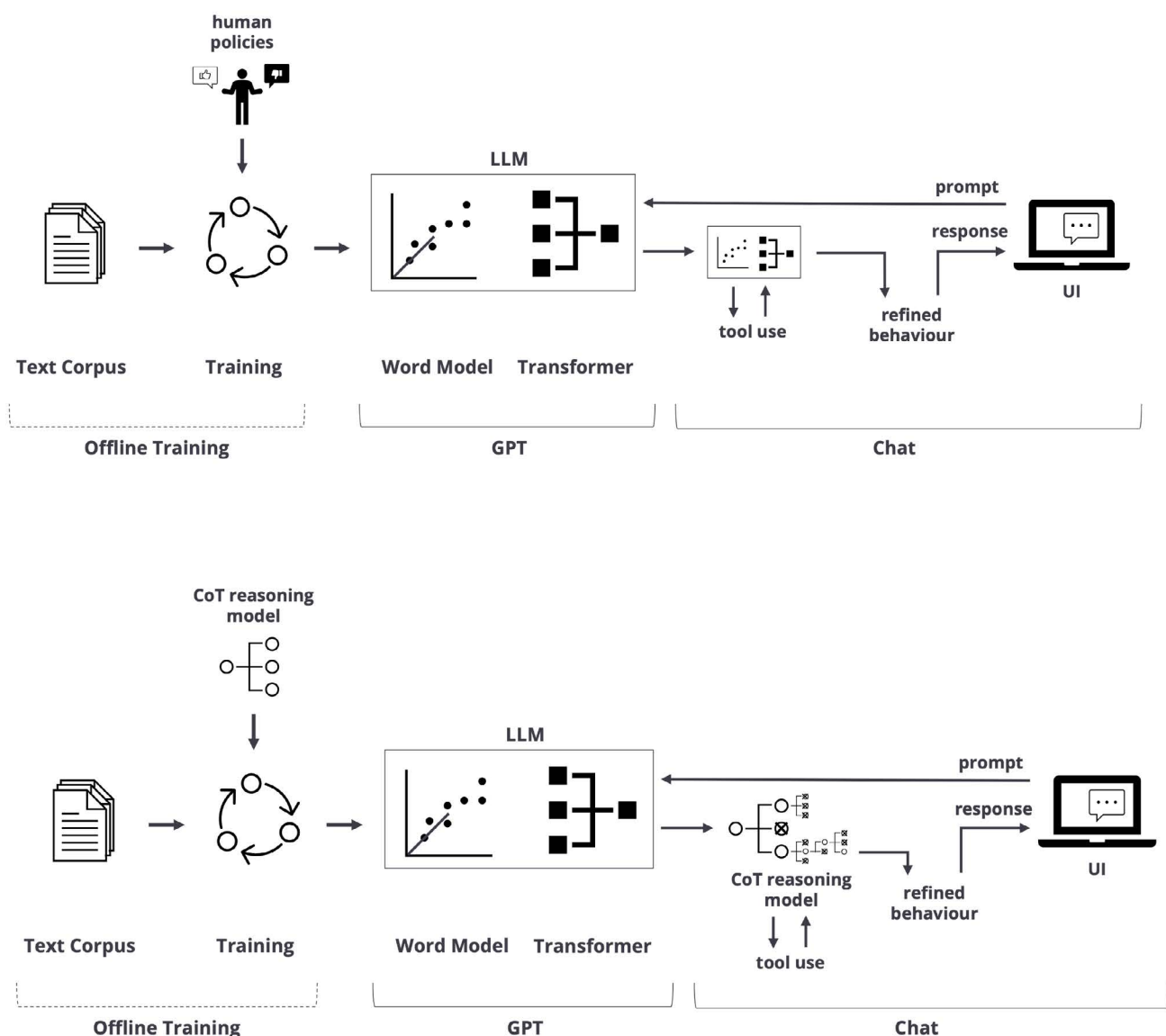


Figure 2. Rough architecture of ChatGPT with tool use (top) and reasoning model (bottom)

English and 4,000 – 5,000 words for German. The RAAML specification alone translates to approx. 41,000 tokens. A possibly very large part of these inputs can therefore not be used by this LLM without further measures (ChatGPT 4o can already process 128K tokens). In addition, there are observations that LLMs process information in long contexts differently

depending on their position in the prompt and may take less account of data in the middle section in particular [7, 8].

Tip: Always start a new chat for new questions. This can also be helpful if the LLM has “lost its way”. Instructions to ChatGPT that should be used in every new chat can be entered in the settings. Be aware of the recently introduced “memory” feature of ChatGPT, where information from a chat is transferred to other chats. This feature can be switched off permanently or just for the new chat.

H. slowly realises that one needs to learn how to communicate with the machine so that it gives the expected answers. However, he is surprised that he repeatedly received incorrect answers before uploading the PDF file, even though the LLM apparently did not know or could not find any current texts related to his query.

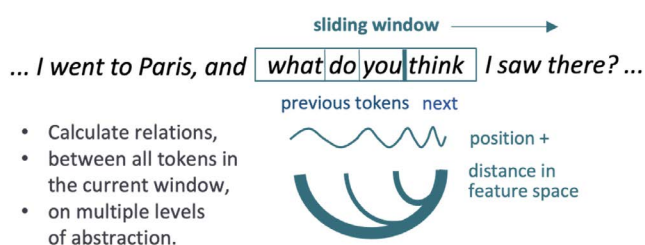


Figure 3. Context window during training with simplified attention concept

Pitfall #05: It doesn't say "I don't know".

There is always some kind of answer.

Basic principle: In general, neural networks always provide an output [9]. Whether and how this is communicated to the user depends on subsequent stages of the software. This may be a relatively complex step that is carried out before

the output of the transformer reaches the user (Fig. 2: “Chat”). Als like ChatGPT for example are tuned to be a good conversational partner and this obviously includes keeping the conversation going, instead of just saying “I don't know.”.

Retrieving Data

Meanwhile H. managed to get a correct answer based on uploaded specification documents. On H.'s question „How is the model element Part defined in SysML version 2?“, ChatGPT gives a correct but rather vague reply, so H. wants to look up the exact sections the language model had used:

H.: „Where in that specification document did you get the information for the Part definition from?“

ChatGPT: „The information about Part Definitions and Part Usages in SysML v2 is derived from an understanding of the enhancements and improvements introduced in SysML v2, as the language builds upon and refines concepts from SysML v1. [...] According to the official SysML v2 documentation and related resources, here's how parts are defined: [...]“

These general references (underlined) are not satisfactory, so H. insists:

H.: „What's the exact position in those files for your Part definition explanation?“

Interestingly, instead of giving an answer right away, ChatGPT starts to generate python code, which it then executes in the background. The code downloads specification files from the internet, extracts text from those PDFs and does a literal search for the word “Part”.

What happened here is, that H.'s last prompt exceeded the capabilities of Large Language Models. More precise, ChatGPT recognized that this type of prompt is better processed using a simple piece of software first. Downloading files, searching for exact content, and other tasks such as numerical calculations or analyses, is not what a neural network-based system is built for. Therefore, ChatGPT has the added capability of using tools that are not part of the LLM (Fig. 2, "tool use").

Pitfall #06: It's not a database. It is not a search engine.

LLMs cannot reference the exact source data used to create an output because that data does not exist in the model.

Basic principle: Neural networks do not store raw training data. They derive (usually unknown) features from this data and store them in a self-organised manner, distributed and superimposed in connections between the neurons [9]. Therefore language-based, generative AI applications such as ChatGPT can refer to uploaded files or files found on the Internet, but generally not to *specific locations* within these files. An AI application can attempt to solve this task "classically" by using external tools.

At this point, it is important to note the very different meaning of the term „model“ in technical disciplines such as model-based systems and software engineering versus its meaning in artificial intelligence. Engineered models are precise representations of systems which are usually just as engineering-like and mechanistic, also developed through systematic

thinking of experts. An AI model is the opposite in every respect.

In the field of AI, the term neural network should be used rather than model. The term "model" here comes from a view of what a *neural network* does mathematically: it approximates a complex target function [9]. However, this view obscures the fundamental character of neural networks, which were created by evolution to handle the vagueness and contradictions of the real world and *quickly* provide an answer that is *good enough* for the task at hand.

Neural networks are self-organised, learning only by "looking" at data, but they do not store or memorize any of this data in its raw form. This principle enables one of their most outstanding abilities, which is to handle vague or incomplete input data by generalising from the data they were trained on. They are able to recognise similarity in input data they have never seen before. However, this very ability to generalise means their outputs cannot be guaranteed to be 100% correct or certain.

The examples in this article were created with ChatGPT 4o in June 2024. Why is this information relevant?

Pitfall #07: Consistency between different LLMs.

Models differ, models change, answers change.

Basic principle: There are currently several large closed-source language models (e.g. GPT by OpenAI, *Gemini* by google, *Claude* by Anthropic) as well as a vast variety of open-source models (e.g. from Mistral, Aleph Alpha, Stable Diffusion

and many more). Some of these differ greatly in terms of their size, training data and internal architecture and “maths”. These differences may be just as great between successive versions of the same model as between completely different models. It can also happen that providers such as OpenAI make changes to already available chat tools without explicitly communicating the changes. This means that behaviour can change over time, even within the same public version number.

Outlook

The article showed why purely intuitive work with AI language models via the manufacturer’s chat interface is generally not enough. It is advisable to learn the basics of *prompt engineering* techniques, for example. In the next step, there are various approaches to building useful assistance solutions that contain an LLM (or several combined) as one of several software components and can also contain large amounts of your own data (including *retrieval augmented generation*, RAG and agent-based AI solutions). This can go so far as to refine pre-trained LLMs with your own data or even train your own LLMs completely. The effort required to implement these approaches increases exponentially.

In professional use, especially outside of technical application areas, further pitfalls and key points of the application of generative AI are relevant, which this article could not address. These include bias, security, scalability, costs, user dependency, explainability and determinism.

Acknowledgements

Special thanks to Hendrik Dahmke, Sebastian Diekhoff, André Brückmann for their curiosity, domain-specific examples and outside-in viewpoints. The author also wishes to thank Dr. Michael Sturm for our stimulating discussions on the theoretical foundations of AI and possible use cases in our technical field of work.

If you are interested in more details on how Artificial Neural Networks and Transformer Models work, the author highly recommends the excellent video series of Grant Sanderson aka *3Blue1Brown* on YouTube.

References

1. Joseph Weizenbaum, Die Macht der Computer und die Ohnmacht der Vernunft. Frankfurt: Suhrkamp, 2000. (first edition: 1978)
2. Risk Analysis and Assessment Modeling Language (RAAML) Libraries and Profiles, Version 1.0, OMG Document Number: formal/22-12-06, 2022. [Online]. Available at: <http://www.omg.org/spec/RAAML/>
3. R. Hertwig, S. Herzog, L. Schooler, T. Reimer, „Fluency Heuristic: A Model of How the Mind Exploits a By-Product of Information Retrieval“, Journal of Experimental Psychology Learning Memory and Cognition, Jg. 34, S. 1191–1206, 2008, doi: <http://dx.doi.org/10.1037/a0013025>
4. Y. Liu, H. He, T. Han, X. Zhang, M. Liu, J. Tian, Y. Zhang, et al., „Understanding LLMs: A Comprehensive Overview from Training to Inference“, 2017, doi: <https://doi.org/10.48550/arXiv.2401.02038>
5. Katharina Zweig, „ChatGPT - Perspektive der Informatik“, in „Dammbruch der KI? Was vom Hype um ChatGPT bleibt“, Stuttgart, 8. May 2023, Available at: <https://youtu.be/gUSPY-Hdu20?si=JRbGhY5iKYluRb71&t=1165>
6. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, „Attention Is All You Need“, 2017, doi: <https://doi.org/10.48550/arXiv.1706.03762>
7. N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, P. Liang, „Lost in the Middle: How Language Models Use Long Contexts“, Transactions of the Association for Computational Linguistics 2023, Vol. 12, S. 157–173, 2023, doi: https://doi.org/10.1162/tacl_a_00638
8. S. An, Z. Ma, Z. Lin, N. Zheng, J.-G. Lou, „Make Your LLM Fully Utilize the Context“, 2024, doi: <https://doi.org/10.48550/arXiv.2404.16811>
9. I. Goodfellow, Y. Bengio, A. Courville, „Deep Learning“, MIT Press, 2016
10. Neural Network image in Fig. 1 by <https://www.youtube.com/@3blue1brown>

Contact

If you have any questions, please do not hesitate to contact us via **sales@in-tech.com**.

in-tech contact:

Dr. Frank Höwing
Technical Head Data Analytics & AI
E-Mail: frank.hoewing@in-tech.com