A Review of Advanced Prompting Techniques in Large Language Models (LLMs)

Course: MBAR 661: Academic Research Project

Submitted to: Professor Dr. Mohsen Ghodrat

Submitted by: Sundar Neupane

Student ID: [redacted]

University Canada West

Date: 26/06/2025

Acknowledgement

I want to thank University Canada West from the bottom of my heart for all the help, advice, and resources they gave me during this study. I particularly acknowledge Professor Dr. Mohsen Ghodrat who has been very supportive, provided me with useful comments, and served as a supportive mentor to me. He has made my academic life much improved.

I also wish to give credit to my wife, mother and family and friends. They have been there to keep faith and support me and I find it the best strength of mine. Lastly, I would like to show my appreciation to the general scholarly community whose research and hard work have guided and encouraged this work.

Table of Contents

List of Tables

List of Figures

Abstract

This thesis provides a detailed discussion of Sophisticated prompting techniques in large language models (LLMs) to enhance performance with particular tasks, contextual understanding, and reasoning. Knowledge of how to efficiently engineer the outputs of LLMs such as GPT-4 has become increasingly important, as these models become increasingly popular in numerous applications. This paper examines and categorized other ways to do so, including zero-shot and few-shot prompting, Chain-of-Thought (CoT) prompting, and other recent methods, such as Tree of Thoughts (ToT) prompting and Persona prompting. Considering that look at the work of each technique, advantages and disadvantages of the technique, when the technique is the most appropriate, what is the effectiveness of the technique. The thesis also discusses how one can employ the ideas in the real world, the issues that arise and what could go on in future. The findings demonstrate that timely engineering is decisive in extracting the maximum out of LLMs, minimizing hallucinations, and ensuring that outputs conform to intentions of the user. The study adds to the growing body of literature by providing a systematic and fact-based review that can be used for more research and use.

Introduction

Large Language Models (LLMs) like GPT-3.5, GPT-4, and other generative AI tools have emerged as a result of the quick development of artificial intelligence (AI), particularly in the area of natural language processing (NLP). These models are changing whole industries by letting machines understand and write text that is similar to what people write. But the way users enter instructions or questions into LLMs, or how they are prompted, has a big effect on how well they work.

Without prompt engineering, Large Language Models (LLMs) often give results that are unclear, don't match what the user expects, or aren't factually accurate. Although after this extensive training on all kinds of textual corpora, LLMs are intrinsically sensitive to the phrasing of the input and may not be able to reliably recognize the intent of the user without supervised training. Prompt engineering is a systematic way of creating, enhancing and structuring input prompts to make LLMs provide more precise, relevant, and coordinated answers. The method enhances the collaboration between people and AI systems by providing details, particularity in tasks, rational ordering of the material to the model. Prompt engineering is extremely crucial in refining models to perform better, reasoning better, diminishing visions, and ensuring that the output corresponds to the objectives on which it was supposed to be. This paper examines the significance of prompt engineering as a fundamental method of maximizing the use of LLMs in the real world.

This analysis provides an in-depth view of refined prompting mechanisms that can enhance the practicality, efficiency and precision of LLM products. The information on such promoting approaches as persona-based prompting, chain-of-thought reasoning, zero-shot and few-shot prompting, etc. is abundant. This review is to demonstrate the most appropriate methods of carrying out prompt engineering and discuss the advantages and disadvantages of each in real-life scenarios.

Research Aims and Objectives

The key objective of the research is to determine how improved prompting strategies can help Large Language Models (LLMs) be more effective, reliable, and flexible. This study aims to reflect the theoretical progress of prompt engineering and its practical applications in business, healthcare, education, and so on. The study has the following specific goals:

- **To look into how prompting methods have changed over the years:** Users discussed how the use of prompting techniques no longer involves a single question but rather some more sophisticated methods such as zero-shot, few-shot, Chain-of-Thought (CoT), and Tree of Thought (ToT) prompting. There was also discussion of how inputs are growing more contextual and structured.
- **To see how well the newest prompting strategies work:** Users looked at how advanced prompting techniques improve a model's ability to reason, be accurate, and fit in with its surroundings. People often use these methods along with retrieval-augmented generation, reinforcement learning, and instruction tuning, among others.
- **To give helpful tips for real-world situations:** Users give researchers, developers, and practitioners evidence-based advice on how to make and use complex prompting strategies that get the most out of LLM outputs for different use cases, while also thinking about possible problems and moral issues.

Research Questions/Hypotheses

The purpose of this study is to find out how advanced prompting strategies change the usefulness, reliability, and ethical use of large language models (LLMs) in different areas. The study is based on the following main questions:

Research Question 1: In a number of real-world settings, such as customer service, healthcare, and education, which advanced prompting techniques work best to make large language models work better?

Research Question 2: How do different prompting methods, such as contextual, few-shot, and zero-shot prompting, affect the accuracy, coherence, and reasoning skills of LLM outputs?

Research Question 3: What ethical and practical problems come up when using advanced prompting techniques, especially in sensitive or high-stakes situations?

Methodology

This study uses a systematic literature review method based on the Preferred Reporting Items for Systematic Reviews and Meta-Analyses 2020 (PRISMA) framework to ensure that the academic and industry sources used for advanced prompting techniques in Large Language Models (LLMs) are clear and correct.

PRISMA Flowchart

The Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) framework is used to help choose which studies to include in this systematic review and meta-analysis. PRISMA gives a clear and standard way to find, screen, and decide if published literature is eligible, which makes sure that evidence synthesis is done in a methodologically sound and reproducible way. The structured flow diagram makes it easy to keep track of how studies were included or left out at each step of the review.

Following the PRISMA process, users found a total of 112 records, 110 of which came from database searches and 2 from other sources. Users looked at the titles and abstracts of 90 unique records after removing duplicates. Users then removed 30 studies that

were not related to large language models (LLMs) or prompt engineering, were blog posts or opinion pieces, or were incomplete or hard to find. After that, 60 full-text articles were checked to see if they were eligible. Ten were not because they didn't have enough methodological detail, their prompting focus wasn't relevant, or their content was too similar to other articles. In the end, the systematic review included 74 studies, and 40 of them were good enough and relevant enough to be included in the meta-analysis.

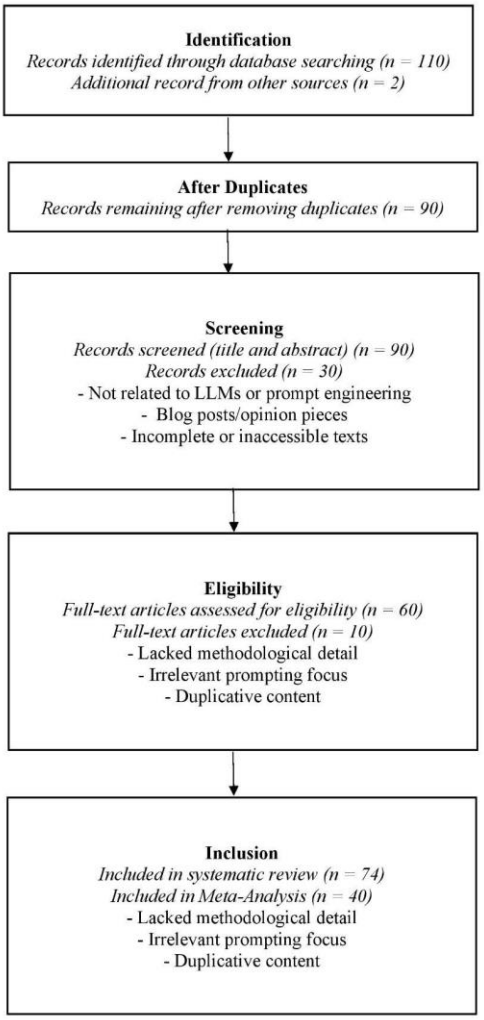

Figure 1: Prisma Flowchart

Protocol and Selection Criteria

There were 74 sources in this study, including peer-reviewed journal articles, arXiv preprints, conference papers, and technical reports that were published between 2017 and 2025. The review followed a set protocol from the PRISMA 2020 guidelines, which made sure that screening, eligibility evaluation, and thematic classification were all clear. Users picked sources that were related to advanced prompting strategies, making large language models (LLMs) work better, and basic AI/ML/DL developments. The different sources show that prompt engineering is a field that crosses over into many others, such as healthcare, education, and business AI systems.

Inclusion Criteria

The following standards are used to pick sources that are both useful and high-quality:

1. Articles, preprints, technical reports, and industry publications that have been peer-reviewed and published between 2017 and 2025.
2. Concentrate on strategies for prompting, engineering prompts, or improving performance in LLMs.
3. Papers that talk about how to prompt, how to compare different methods, or the moral issues that come up.
4. Sources in English.

Exclusion Criteria

Users used the following criteria to eliminate out sources that weren't relevant or were of poor quality:

1. Blogs or opinion pieces that don't have any technical information.
2. Research that doesn't focus on LLMs or NLP-driven prompts.

3. Articles that don't explain their methods well enough.

Rationale

Most of the time, peer-reviewed journals and official reports from AI research labs publish LLM research. Users used open-access research from official sources (like OpenAI, Meta, Anthropic, and Hugging Face) as our main sources because they were methodologically clear and useful in real life.

Search Strategy and Databases Used

Users used a systematic search strategy to find studies that were relevant to prompting and large language models. Users used a number of databases and platforms for this search, including:

1. Google Scholar,
2. arXiv.org,
3. Science Direct,
4. Springer Link,
5. OpenAI Blog,
6. Meta AI and Hugging Face repositories.

Keywords and Boolean Combinations Used

Users chose keywords based on how they are commonly used in academic and professional settings, and then users used Boolean operators to combine them so that they could be used in both. Users looked for open-source implementations and publications to make sure they would work in real life.

Table 1: Boolean Search Query Breakdown

Component	Search Terms
Prompting Technique	*"prompt engineering"* OR *"prompt design"*
Model Type	*"LLM"* OR *"large language model"*
Prompting Variants	*"few-shot"* OR *"zero-shot"* OR *"contextual prompting"* OR *"chain of thought"*
Performance Metrics	*"performance"* OR *"accuracy"* OR *"alignment"* OR *"hallucination"*

PRISMA Compliance Summary

Table 2: PRISMA Compliance Summary

PRISMA Element	Compliance Status
Eligibility Criteria Defined	Yes
Search Strategy Transparent	Yes
Duplicate Removal Performed	Yes
Records Screened & Filtered	Yes
Full-Text Assessed	Yes
Included Studies Counted	Yes

Summary of Selected Literature

Table 3: Summary of Selected Literature

Category	Number of Papers Cited	Key Contribution
Artificial Intelligence (AI)	5	Foundations of AI, Turing test, evolution of intelligent agents
Machine Learning (ML)	3	ML algorithms, probabilistic learning, auto-encoding, supervised/unsupervised learning
Deep Learning (DL)	5	DNNs, backpropagation, CNN/RNN, variational autoencoders

Generative AI & LLMs Overview	9	Evolution of LLMs, generative architectures, GPT models
Zero-shot Prompting	4	Prompt without example; effective with structured instruction
Few-shot Prompting	4	In-context learning with examples; generalization without fine-tuning
Chain-of-Thought (CoT)	6	Step-by-step logic reasoning; improves answer consistency
Tree-of-Thought (ToT)	2	Recursive reasoning with tree-based decision nodes
Prompt Chaining / AI Chains	2	Modular workflows; sequential sub-tasks in LLM prompts
Persona Prompting	2	Adds tone and domain-awareness to output
Task/Instruction Prompting	2	Direct role or task prompting; improves intent alignment
Clarity-based Prompting	2	Focuses on rephrasing, simplification, instruction clarity
Example-based Prompting	2	Demonstration-based guidance for structured tasks
Self-consistency Prompting	1	Sampling multiple reasoning paths to select stable answer
Prompt Optimization	2	Refining inputs to improve consistency and reduce errors
Multimodal Prompting (Vision + Text)	2	Vision-language prompts using models like BLIP-2
Retrieval-Augmented Generation (RAG)	4	Augments responses with factual external documents
Fine-Tuning (General)	3	Domain-specific model tuning and performance improvement
Parameter-Efficient Fine-Tuning (PEFT)	2	Low-resource alternatives to full fine-tuning

Instruction Tuning	1	Aligns model output with natural-language task guidance
Ethics and Risks in Prompting	2	Risk of hallucination, prompt manipulation, persona misuse

Discussion of Findings

The systematic review of 74 chosen studies shows how the field of prompt engineering is changing in relation to Large Language Models (LLMs). Techniques like zero-shot, few-shot, chain-of-thought (CoT), and persona prompting have made LLMs work much better in all areas, especially in reasoning, task alignment, and content generation. There is empirical evidence from several studies (e.g., Brown et al., 2020; Wei et al., 2022) that well-structured prompts not only improve the accuracy of responses but also lower the uncertainty and hallucinations of models.

However, even though the practical improvements are impressive, ethical issues came up again and again. For example, persona prompting was found to improve tone control and domain specificity, but it also made it easier for people to misrepresent themselves, especially when pretending to be experts like doctors or engineers (Kim et al., 2024; Olea et al., 2024). Also, the fact that the output changes based on small changes in the prompt wording shows that LLM behaviour is not clear or easy to understand (Ceurstemont, 2025). This output sensitivity makes things difficult in high-stakes areas like healthcare and finance, where being accurate and responsible is very important.

The results also show that prompt engineering is not a stand-alone strategy but is becoming more and more integrated with other strategies that work well with it, such as Retrieval-Augmented Generation (RAG), instruction tuning, and parameter-efficient fine-tuning. It seems that these approaches work best when they are used together to slow down the rate at which knowledge becomes obsolete, reduce hallucinations, and improve factual

accuracy, especially in environments that are constantly changing or lack data (Lewis et al., 2020; Hu et al., 2021).

Artificial Intelligence

Artificial Intelligence (AI) is a field of computer science that looks at how to make machines that can do things that normally require human intelligence. These tasks include thinking, learning, solving problems, perceiving, and understanding a language. Researchers often define intelligence as an agent's ability to reach goals in a variety of settings (Legg & Hutter, 2007).

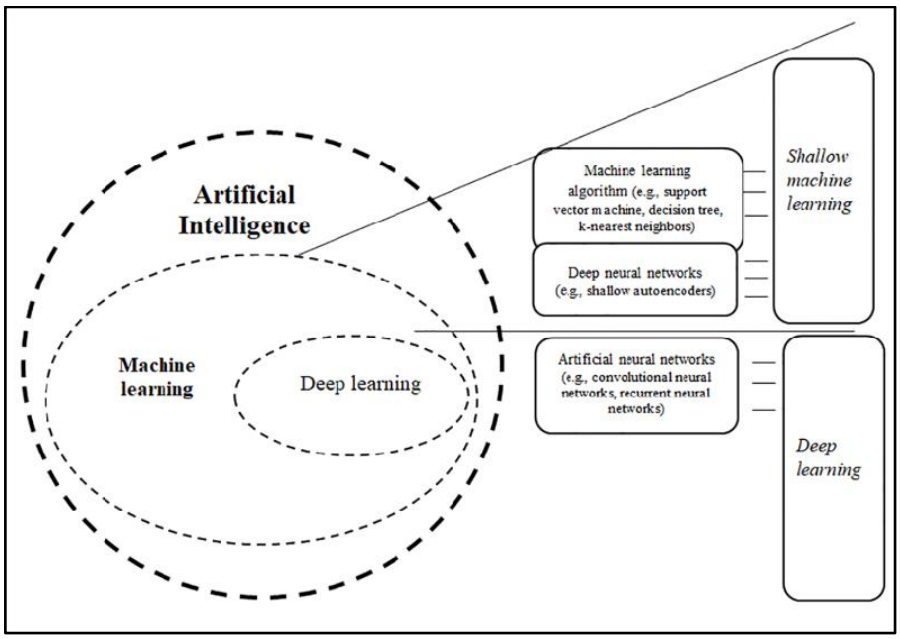

Figure 2: Digital neural network representing artificial intelligence Photo by A. Smith on Unsplash (https://images.unsplash.com/photo-1504384308090-c894fdcc538d). Used under Unsplash license.

AI uses different methods, such as machine learning, natural language processing, and robotics, to mimic these human cognitive functions. Machine learning (ML) is a big part of AI that helps algorithms learn to find patterns and make decisions on their own using data (Kühl et al., 2020).

Examples and Applications of AI

AI has changed quickly and is now useful in a lot of ways. AI systems have proven that they can make hard choices when they play games. They beat human chess champions in 1996 and human Go champions in 2016 (Gupta, 2023). GPT-4 and other advanced systems are very good at understanding language and reasoning. Because of this, they work well for chatbots, translation services, and tools that make content (Bubeck et al., 2023). AI has also come a long way in computer vision, which helps machines understand visual information for things like recognizing faces, driving by themselves, and medical imaging (Gupta, 2023). AI helps doctors by giving them automated diagnostic tools, such as CT scans that can find strokes, and it also makes decision support systems better (Bubeck et al., 2023). AI is what lets self-driving cars read sensor data, make decisions about how to drive in real time, and get through heavy traffic (Gupta, 2023). AI has also changed the things that people buy. It runs smart assistants like the Amazon Echo, suggests movies and TV shows on streaming services based on user preferences, and lets virtual customer service agents do their jobs (Gupta, 2023). These examples show that AI systems are becoming more common in both technical and everyday life. This makes things work better and faster.

A Brief History of AI: Key Milestones

The concept of AI has been around since the 1940s. The "Turing Test" for machine intelligence was first described in Alan Turing's important 1950 paper (Gupta,, 2023). AI has changed over the years from systems that used symbolic logic to machine learning methods

that use data.

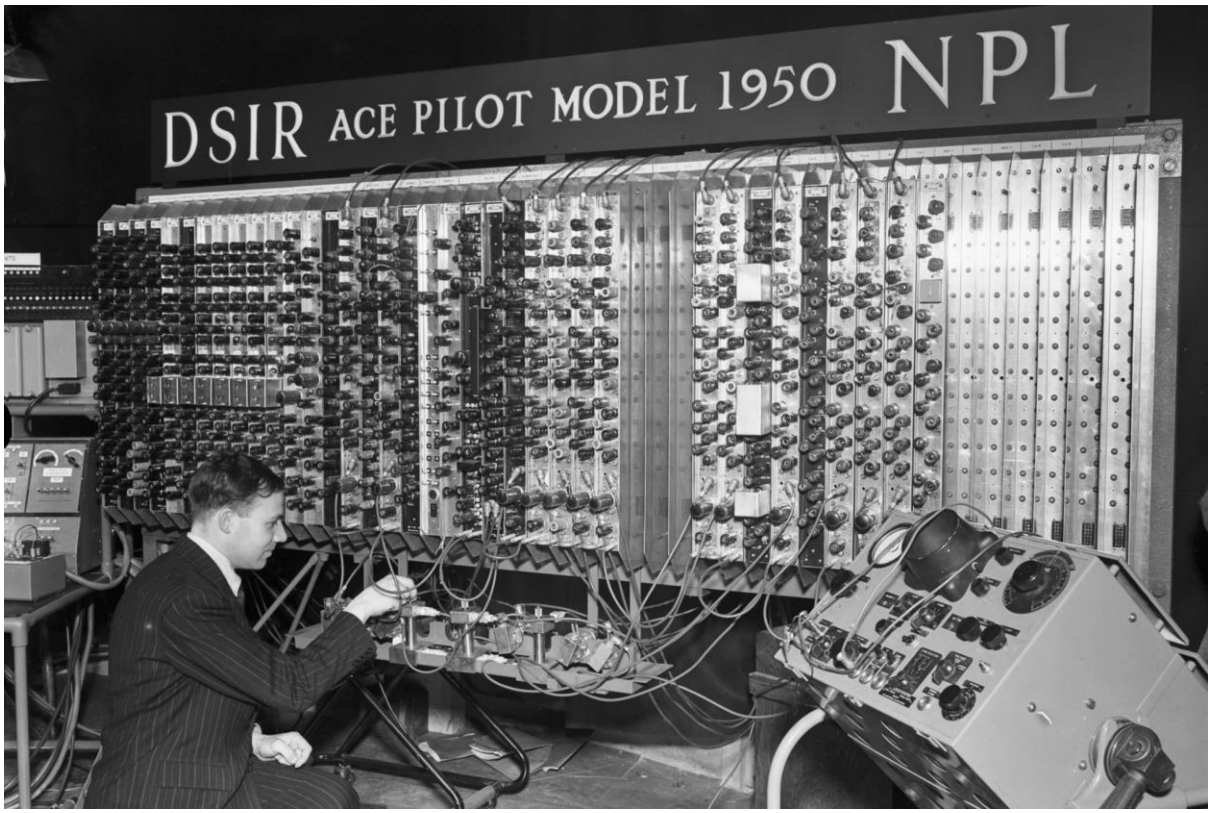

Figure 3: Portrait of Alan Turing (https://www.nationalgeographic.com/science/article/alan-turing-test-artificial-intelligence-life-history).

AI has made progress through a number of important stages. Basic work in logic, algorithms, and symbolic AI from the 1950s to the 1970s made it possible for more progress to be made. IBM's Deep Blue beat world chess champion Garry Kasparov in 1996. This was a big step forward for AI in making strategic decisions. In 2016, DeepMind's AlphaGo shocked everyone by beating a world champion at the game of Go. A lot of people thought this was too hard for machines because there are so many possible moves. In the 2020s, large language models (LLMs) like GPT-4 became very popular very quickly. They can think, talk, and figure things out in a lot of different fields (Bubeck et al., 2023). Over time, AI has

grown into a field that includes ideas and methods from computer science, neuroscience, statistics, and linguistics (Yu & Kumbier, 2017).

Limitations and Challenges

AI has come a long way, but it still has a lot of big problems to work on. One big problem is hallucinations, which happen when AI systems give out false or misleading information. Also, current models don't have long-term memory, so they can't remember and build on what they've done in the past. Planning is still a big problem because AI can't do tasks that require more than one step, foresight, or strategic reasoning. Also, there are still worries about bias and transparency because AI systems often reflect the biases that are already present in the data they were trained on and usually have trouble explaining how they make decisions (Bubeck et al., 2023). These limitations show that modern AI systems, even the most advanced large language models, are still not good enough to reach Artificial General Intelligence (AGI), which is the ability to do any intellectual task that a human can do.

Machine Learning (ML)

The field of machine learning (ML) is concerned with creating algorithms and statistical models that allow computers to carry out certain tasks without being told to do so (Mahesh, 2020). It lets computers get better at what they do by using experience and learning from data. Goli and Singh (2024) define ML as a field that lets systems learn on their own, which shows how important it is in modern intelligent applications.

Machine learning (ML) is part of the larger story of how people have always tried to make difficult tasks easier by making and using tools and machines (Goli & Singh, 2024). Alan Turing is a very important person on this journey. He is often called the father of

modern computer science. In 1936, Turing thought of the "Turing Machine." This was a theoretical model that made the rules of computation more clear and set the stage for algorithmic processing, which is a key part of machine learning (National Institute of Standards and Technology [NIST], n.d.). In 1950, he came up with the famous "Turing Test," which is a philosophical and technical way to see how well a machine can act like a person. This standard is still used in AI today (New Scientist, n.d.).

Arthur Samuel and other pioneers built on these ideas to move the field forward by making early self-learning systems, like a checkers-playing program in the 1950s. Samuel's work made the idea that machines can learn from experience more popular, and he is credited with defining machine learning as the ability of computers to learn without being programmed (Goli & Singh, 2024). These early advances made it possible for today's powerful machine learning systems, like large language models, to exist. These systems are still changing quickly, affecting industries and how people use computers.

Figure 4: Arthur Samuel's Checkers Program on the IBM 701, demonstrated on live television in 1956. This early example of machine learning showcased the potential of computers to learn from experience, a concept central to the development of artificial

intelligence.
Source: Press, G. (2021, May 28). On thinking machines, machine learning, and how AI took over statistics. Forbes. https://www.forbes.com/sites/gilpress/2021/05/28/on-thinking-machines-machine-learning-and-how-ai-took-over-statistics/

Deep Learning and Neural Networks

Deep learning (DL) is a powerful part of machine learning (ML) that uses artificial neural networks (ANNs) as its main way of doing math to learn hierarchical, abstract representations of data on its own (LeCun et al., 2015). Deep learning models can automatically learn more and more complex features from raw data without having to do any manual feature engineering. To do this, user stack several layers of neurons on top of each other (Goodfellow et al., 2016). These multilayered networks, called deep neural networks (DNNs), can do a lot of different things.

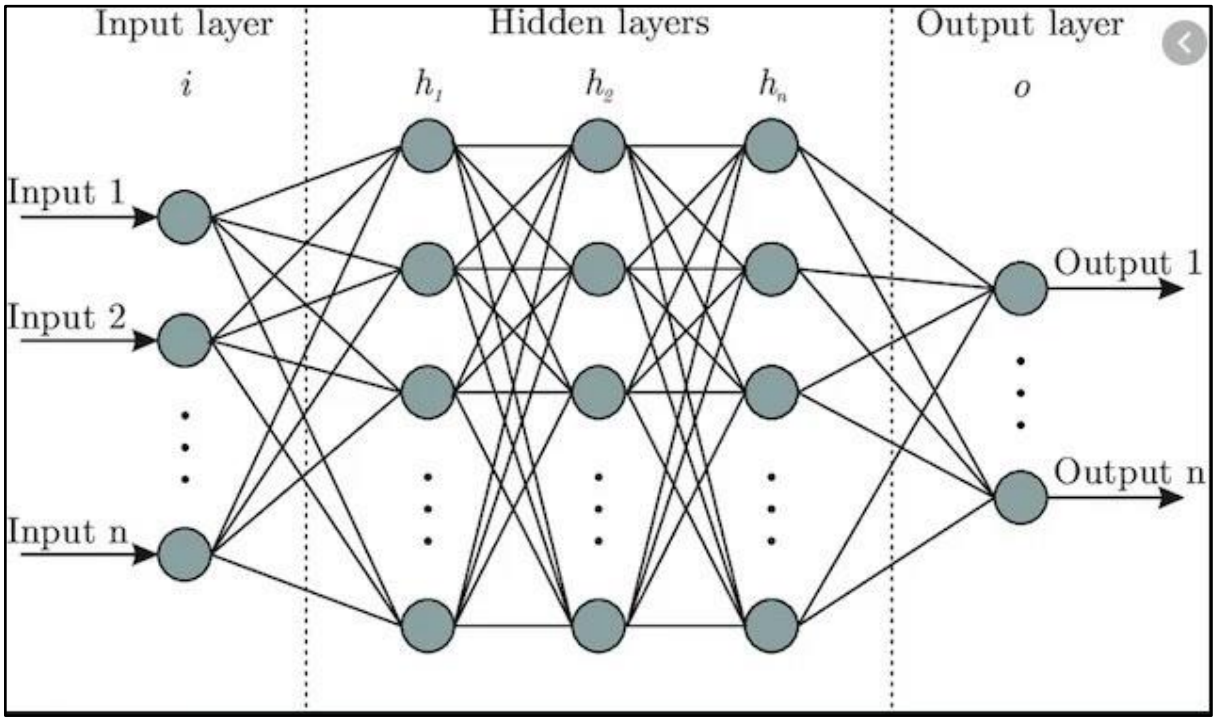

Figure 5: Deep Neural Network. Adapted from Bre, F., Gimenez, J., & Fachinotti, V. (2017). Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks. Energy and Buildings, 158, 1429–1441. https://doi.org/10.1016/j.enbuild.2017.11.045

A Brief History of Deep Learning and Neural Networks

The main idea behind deep learning comes from early attempts to copy how biological nervous systems work. McCulloch and Pitts made the MCP model in 1943. It was the first mathematical model of a neuron. It was the starting point for making neural networks. Rosenblatt built on this idea in 1958 by creating the perceptron, a two-layer neural network that could do simple classification tasks. This was the first time that machine learning used the MCP model. Minsky and Papert (1969) later pointed out the problems with single-layer perceptrons, especially how they couldn't solve linearly inseparable problems. People were less interested in artificial neural networks (ANNs) in the 1970s after this discovery.

When Geoffrey Hinton and his team created the backpropagation algorithm in 1986, they made a big leap forward. Multilayer perceptrons could now learn complex, nonlinear mappings, which brought neural networks back into the spotlight (Rumelhart et al., 1986). Hinton and his team made another big step forward in 2006 by coming up with a method that began with unsupervised pre-training and ended with supervised fine-tuning. A lot of people think that this method was the beginning of modern deep learning (Hinton et al., 2006). It fixed the problem with the gradient going away. A lot has changed in the field since 2012. Numerous labeled datasets such as ImageNet, parallel-computing hardware such as GPUs, and improved neural networks are only a few of them. These changes have made deep learning very popular in both schools and businesses nowadays (Krizhevsky et al., 2012; LeCun et al., 2015).

Key Components and How Deep Neural Networks Learn

Artificial neural networks are constituted of neurons. They are linked up units that process information. These neurons have three parts or layers, including input, hidden, and output (Goodfellow et al., 2016).

Neurons and Layers: The individual neurons are fed with a nonlinear activation (e.g., ReLU, sigmoid or tanh) to discover the presence of complex patterns by performing an affine combination of the inputs and then adding a bias (Nwankpa et al., 2018).

Weights and biases: Weight and biases are the parameters that one can train and adjust when making better predictions through learning (Goodfellow et al., 2016).

Leaning Process: A number of network forward passes (computation of outputs) and BPN (updating weights by performing backpropagation and gradient descent) are performed to reduce a loss. This assists them to improve with time (Rumelhart et al., 1986).

Examples of Deep Learning Architectures

Different types of neural network architecture have been constructed by researchers to cope with the reality that tasks and data type will not always be identical. Multi-layer perceptron (MLP) is the simplest type of a feed-forward network. They possess a series of fully linked layers which are capable of modelling complicated nonlinear functions (Goodfellow et al., 2016). The CNNs are useful with the kind of data that resembles a grid such as pictures. They are taught to arrange themselves the space at their own using pooling and convolution layers. There has been a significant impact on computer vision applications by such popular CNN architectures as LeNet, AlexNet, VGG and ResNet (Krizhevsky et al., 2012; He et al., 2016).

A different source of inspiration, on the other hand, is recurrent neural networks (RNNs), which are specifically structured to operate sequentially in order to capture time-dependencies. Learning long-range dependencies has also been made easier by people seeing fit to improve such networks as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) (Hochreiter & Schmidhuber, 1997). Another huge step in the right direction is the generative adversarial network (GAN). It possesses a generator and a discriminator which are set to operate in opposition to produce a falsified data, which is so nearly actual to be quite deceiving (Goodfellow et al., 2014).

Transformers that were published in 2017 revolutionized sequence modelling by incorporating self-attention that allowed concurrent processing of inputs. This has transformed the way natural language processing (NLP) has been carried out and that is why new frameworks such as BERT and GTP can now work (Vaswani et al., 2017). These main architectures are not the only ones improving, though. Autoencoders teach user unsupervised representation, Graph Neural Networks (GNNs) are used when user data is in graph form, and more models such as Capsule Networks and Neural Ordinary Differential Equations (Neural ODEs) demonstrate that the field is dynamic.

Applications

Deep learning has transformed many areas by allowing models to be able to learn complicated patterns on data:

Computer Vision: With CNNs (LeCun et al., 2015), sorting, finding, and cutting images into parts and determining what is happening in a scene are much easier.

Natural Language Processing: The RNNs and transformers help machines to translate languages, comprehend how people feel, respond to questions, and write text (Vaswani et al., 2017).

Healthcare: The medical imaging, compression, and analytics, as well as genomic and drug discovery analysis, become more precise and automatic due to deep learning (Esteva et al., 2017).

Finance: It is applied in search of fraud, algorithmic trading, risk estimation, prediction (Feng et al., 2021).

Robotics and Autonomous Systems: This enables drones and driverless cars to observe, investigate, and act in an autonomous way (Kiran et al., 2021).

Key Findings

Scholars have found out that deep learning can automatically extract properties out of the raw data. It shows that the manual feature engineering is required to a much lesser extent (LeCun et al., 2015). Deep learning also works better on a larger dataset and on a more difficult to discern model. They tend to be more precise and better in applying what they know to new circumstances than their older machine learning algorithm counterparts (Goodfellow et al., 2016). Deep learning models have been referred to by people as black boxes as it is difficult to understand how they work with them but always achieve the best performance on many different jobs. This skill has led to new ideas in many fields, including healthcare, finance, and others (LeCun et al., 2015).

Generative AI

Generative AI (GenAI) is a kind of AI that uses machine learning to create new data outputs such as text, pictures, code, audio, video, and simulations. GenAI is not the same as regular AI because it is made to make new, useful, and often human-like content (Saini & Sharma, 2024). Most of the time, people use traditional AI to look at or sort through data that is already there. GenAI is a small part of a larger field of AI that learns from data and uses that knowledge to create new things (Tiwari & Patel, 2024).

Historical Development

In the 1960s, Joseph Weizenbaum did early tests on rule-based chatbots like ELIZA. The first steps toward generative AI were these tests (Kumar & Sharma, 2024). Generative Adversarial Networks (GANs) were created by Goodfellow and others in 2014. This was a big step forward because it let people make things that looked very real. The release of ChatGPT in late 2022 (Liu et al., 2024) made GenAI the most cutting-edge field because of Large Language Models (LLMs) like GPT-3, LLaMA, and ChatGPT.

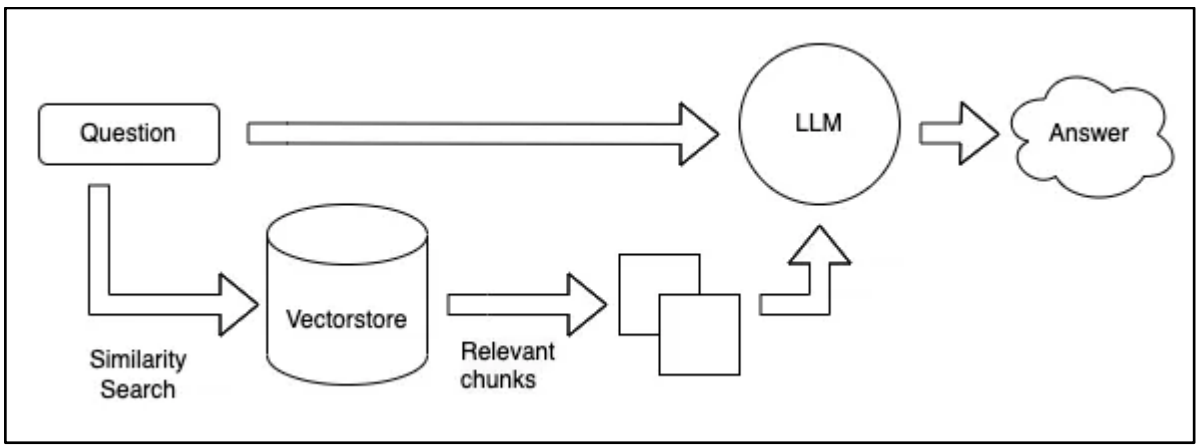

Figure 6: Illustration of LLMs and GANs in AI technology. Adapted from "LLMs and GANs: The AI technologies that will create our new reality (Part 4)," by S. Mali, 2023, Medium. https://medium.com/@surabhimali/llms-and-gans-the-ai-technologies-that-will-create-our-new-reality-part-4-aiseries-2d3b2c757b0b

Core Models and Examples

There are a few important model architectures that generative AI is based on. Generative Adversarial Networks (GANs) are very good at making pictures and videos that look real. There is a generator and a discriminator that are always at odds with each other (Liu et al., 2024). Variational Autoencoders (VAEs) are another well-known way to make data using variational inference. They come close to probability distributions (Kingma & Welling, 2016). The Generative Pre-trained Transformer (GPT) series and other transformer-based models have become the most important parts of modern large language models in the last few years. These models use self-attention to write text that is easy to read and has a lot of context. This makes a lot of generative tasks in natural language processing possible (Radford et al., 2019; Vaswani et al., 2017).

People often use ChatGPT, a generative AI program, to write conversations. DALL-E is good at drawing. Codex helps with code generation, and WaveNet is used for voice synthesis.

Applications Across Domains

Generative AI is changing a lot of things by finding new ways to use it. People in healthcare use it to find new drugs, make medical images, and create care plans that are different for each patient (Saini & Sharma, 2024). Generative AI writes reports on its own, combines a lot of data, and runs smart chatbots that help customers in the finance industry (Tiwari & Patel, 2024). The media and entertainment industry uses generative AI to make games, music, videos, and scripts that people can play (IJTSRD72647, 2024). Generative models look at and use customer data to help make ads and strategies that are more relevant to them (Liu et al., 2024). Scientists' work has also changed a lot because of generative AI. It has helped scientists come up with new ideas, run molecular simulations, and make fake data

to show situations that aren't well represented (Arxiv:2403.04190). Schools and software developers are also getting AI-powered tutors and helpers for programming. The way these tools can change will depend on what the students need and how good they are at using them (IJTSRD72647, 2024).

Key Advantages

The major advantages of GenAI includes;

Speed and Scalability: GenAI can make content much faster and on a much larger scale than humans can (Saini & Sharma, 2024).

Data augmentation: This lets user train in places where there isn't a lot of data by using fake datasets (Arxiv:2001.06937).

Customization and Personalization: This feature gives each user results and suggestions that are unique to them (Liu et al., 2024).

Challenges and Risks

GenAI has a lot of potential, but it also makes us wonder about a lot of things:

Bias and Hallucination: The outputs could make biases in the training data stronger or make up false information (Kumar & Sharma, 2024).

Security Risks: GenAI can be used for bad things like deepfakes, phishing scams, and spreading false information (Arxiv:2403.04190).

Legal Risks: There are some legal and moral issues that aren't clear, like copyright, the fact that there are no human authors, and the ethics of getting data (ResearchGate, 2024).

Environmental Impact: Training big models uses a lot of computer power and energy (Tiwari & Patel, 2024).

Job Loss: Automation could put jobs in the creative and technical fields at risk (Saini & Sharma, 2024).

Large Language Models (LLMs)

Large Language Models (LLMs) are advanced AI systems that can understand, analyze, and create human language. They are like big digital brains that have been trained on a lot of text data, so they can write text that sounds a lot like what a person would write (Patil & Gudivada, 2024). LLMs are built on the Transformer architecture. This is a kind of neural network that is very good at working with text and other data that comes in a sequence. This design lets LLMs figure out what the next word in a sequence will be, which helps them understand and use language that makes sense and fits the situation. This is a very important part of their training (Patil & Gudivada, 2024).

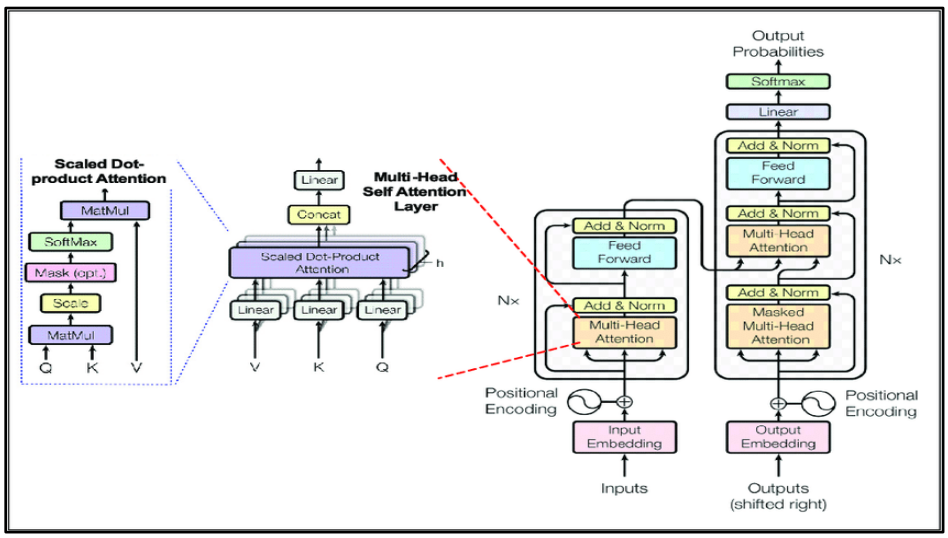

Figure 7: Transformer Architecture Diagram. Adapted from Brownlee, J. (2023, October 2). The Transformer Model. Machine Learning Mastery. https://machinelearningmastery.com/the-transformer-model/

Self-supervised learning is when LLMs learn from data that doesn't have labels. This helps them find difficult language structures without having to look at a lot of examples that have already been marked (Patil & Gudivada, 2024). They can use this skill for a lot of different NLP tasks, such as translating, summarizing, and figuring out how people feel about something. They can understand and write in natural language (NLU and NLG), which makes them useful for both work and school.

One thing that makes LLMs different is their size. These models often have billions, and sometimes trillions, of parameters (Patil & Gudivada, 2024). This level not only helps people learn languages better, but it also gives them new skills, such as reasoning, planning, and learning how to adapt to new tasks in context. In this way, the model learns how to solve new problems by looking at a few examples that are given to it (Patil & Gudivada, 2024).

GPT-3.5 and GPT-4 from OpenAI are two well-known LLMs that work well in a lot of languages and tasks. This shows that AI is moving away from being used for only a few things and toward systems that can do a lot of different things (Goli & Singh, 2024). These skills are very important for business owners and decision-makers, especially when it comes to figuring out how to make the customer experience better, automate tasks, manage knowledge, and go digital.

The Journey of LLMs: A Historical View

The emergence of LLMs is only one aspect of the broader shift in business technology and artificial intelligence. Initially, the most popular approaches in NLP were statistical and rule-based approaches. These systems' syntactic and semantic rules were difficult to establish and lacked flexibility and breadth (Research.pdf). NLP was greatly improved by deep

learning, particularly the Transformer model that was released in 2017 (Patil & Gudivada, 2024).

PLMs, or pre-trained language models, were significant. Two of the earliest PLMs that allowed user to learn from the work of others were BERT and T5. This implies that models trained to perform general language tasks could be improved for more specialized tasks, such as responding to inquiries or determining an individual's emotional state (Patil & Gudivada, 2024). These basic models were used to make the LLMs, which were all about making things bigger and more general.

This path took a big turn with GPT-3. All user had to do was give GPT-3 a prompt with instructions or examples, and it could do things it had never done before. This was different from earlier models that had to be adjusted for each job. Researchers call this "zero-shot" or "few-shot" learning (Research.pdf). People started to think of prompt engineering as the best way for users and models to talk to each other after this change. This means that AI features can be used by people who know how to use them and people who don't know how to use them.

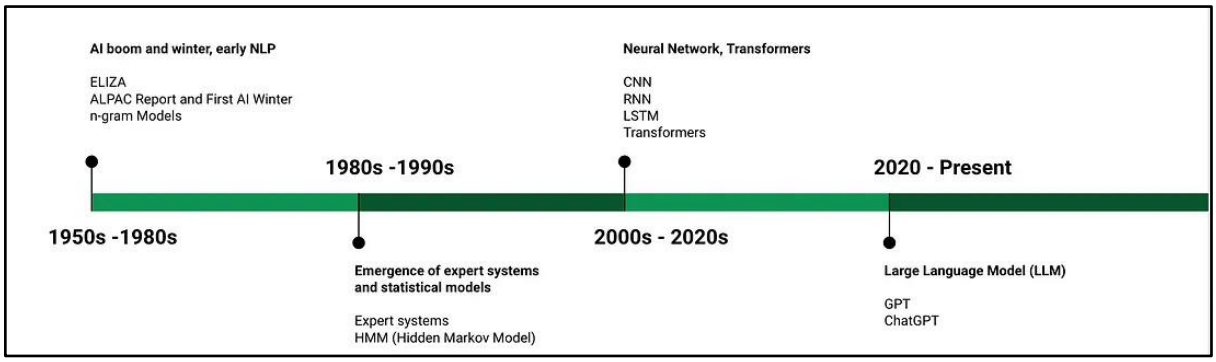

Figure 8: Evolution of Large Language Models (LLMs). Adapted from Huang, L. (2024, April 22). Large Language Model — History. Medium. https://medium.com/@linghuang_76674/llm-history-5db2c9e236f5

Early LLMs also demonstrated some issues. Because the training data is incomplete or doesn't align with user preferences, they may produce inaccurate, biased, or harmful content. Therefore, it was essential to improve alignment techniques. To help models learn to give safe and sensible responses, users added reinforcement learning using human feedback (RLHF) and instruction tuning. The mentioned techniques play a significant role in the ethical use of AI by companies and the reduction of hallucinations (Naveed et al., 2024).

The idea of model editing has become a trend in the recent years as it has allowed the functionality of a model to be changed without recreating everything. As stated by Yao et al. (2023), throughout its business processes, such firms could adjust or improve the results of the models at any point, making them more responsive and flexible. Now LLMs can be used by everyone due to the open-source models. This enhances openness, academic research, and individualized corporate solutions (Naveed et al., 2024).

Over the course of a few years, medium-sized LLMs have evolved to become commonplace. This indicates how much they have transformed machine learning and digital approaches in general (Kaddour et al., 2023). New objectives are fast becoming a business, and cross-border operation priority with the rise of LLMs. These are the ability to understand, never stop learning, have resilience, and understand how to work with data using different languages and formats (Patil & Gudivada, 2024).

How Large Language Models (LLMs) Work

There are three big concepts behind Large Language Models (LLMs) such as GPT, BERT and T5, tokenization, next-token prediction, and training over massive web-based data sets.

1. Tokenization: Breaking Text into Tokens

LLMs look at tokens which are short texts. Such tokens are word fragments commonly generated by advanced algorithms encompassing SentencePiece, WordPiece, or Byte Pair Encoding (BPE). This approach makes models more likely to cope with new words, rare words, and different languages (Devlin et al., 2018; Radford et al., 2019; Understanding the Latest Advances in AI, n.d.; Touvron et al., 2024).

Example:

Input:

Transformers are powerful

Tokenized (WordPiece or BPE):

["Trans", "##form", "##ers", "are", "powerful"]

2. Next-Token Prediction:

The Core Learning Task LLMs are trained to predict the token that will be in a sequence based on the previous tokens given. This example is a description of autoregressive modelling method. The model is trained to give probabilities to the possible tokens that may come in the future and picks the most likely tokens. The methods which make the output more random and more interesting are temperature scaling, top-k sampling, and nucleus sampling(top-p) (Brown et al., 2020; Touvron et al., 2024).

For example, a temperature of 0 means always choosing the most likely token (deterministic), while a temperature of 0.8 makes the text more random, which makes it more human-like and varied (Understanding the Latest Advances in AI, n.d.).

Example:

Input so far:

Transformers are

Model prediction probabilities:

"powerful" → 0.70

"great" → 0.20

"fun" → 0.10

Selected output (depends on sampling strategy).

3. Training on Web-Scale Data

Reading a lot of text from the internet helps LLMs learn how to understand language. Some of the most common datasets are Wikipedia, BookCorpus, Common Crawl (a huge web scrape), WikiText-103, and open-source code repositories like GitHub. These corpora have trillions of tokens, which helps models learn grammar, facts, reasoning, and even how to code (Gao et al., 2020; Raffel et al., 2020; Touvron et al., 2024).

The quality of the data, on the other hand, is very important; data that is noisy or has duplicates can make the model work less well. So, filtering, deduplication, and curation of data are important steps to make training more effective (Shoeybi et al., 2019; Touvron et al., 2024).

Example:

Raw text from dataset:
"Transformers are powerful models used in natural language processing."

Tokenized:
["Transform", "##ers", "are", "powerful", "models", "used", "in", "natural", "language", "processing", "."]

The model learns:

"Transformers are powerful" → "models"
"language" → "processing"

Methods to Improve LLMs Response

1. **Fine Tuning**

Fine-tuning is a very important way to adapt pre-trained Large Language Models (LLMs) to specific tasks or areas by changing their internal weights with datasets that are specific to those tasks. This process helps LLMs learn a lot about a specific field that general-purpose pre-trained models often miss (Chung et al., 2022). Fine-tuning is not the same as prompt engineering or other methods that change the input without changing the model's parameters. Improving performance metrics, such as accuracy or F1 score, on a validation set that is comparable to the task at hand is the primary objective.

Use Cases and Applications

Fine-tuning has resulted in significant changes in the real world:

Product Attribute Extraction: After fine-tuning with only 200 labelled samples, the accuracy of attribute extraction (such as product titles and prices) rose from 70% to 88%. The returns begin to decline after 6,500 samples (Zhang et al., 2022).

Natural Language Processing in Biomedicine: In domains such as biomedicine, where labelled data is scarce, fine-tuned LLMs have facilitated tasks like identifying related clinical texts and providing answers to biomedical questions. This can be achieved by providing the model with field-specific vocabulary and formats (Gu et al., 2021).

General NLP Standards: NLU and NLG tasks such as WikiSQL (generating SQL queries), MultiNLI (generating natural language inferences), and SAMSum (generating dialogue

summaries) have benefited greatly from the fine-tuning of large models like GPT-3 (175B) (Wei et al., 2022).

Comparison with Prompt Engineering

Prompt engineering changes the words in inputs so that LLMs give the right outputs. It does this with prompts like zero-shot, few-shot, and chain-of-thought (CoT). This is good for computers, but it isn't perfect. For example, CoT reasoning might make Visual Question Answering (VQA) tasks less useful because the two types of reasoning don't match up (Li et al., 2023).

Instruction tuning, which uses natural language task descriptions to make small adjustments, has been shown to outperform zero-shot and few-shot prompting. On 20 out of 25 benchmarks, such as ANLI, RTE, and BoolQ, FLAN, a 137B model that was optimized on instructions, outperformed GPT-3 (175B) (Chung et al., 2022).

Parameter-Efficient Fine-Tuning (PEFT) Methods

PEFT techniques can be helpful even though fine-tuning large models can be expensive:

LoRA, or low-rank adaptation: While keeping the same weights, it adds small, trainable matrices to the attention layers. This leads to a three-fold reduction in GPU memory usage and a 10,000-fold reduction in the number of trainable parameters without extending the inference process duration (Hu et al., 2021).

Adapter Layers: These small modules travel between the layers of the Transformer and are replaced for each job. Despite their usefulness, adapters slow down inference because they can only handle one thing at a time (Pfeiffer et al., 2020).

Prefixes for tuning: It enhances an always-present prompt that appears before the input. It works well, but it restricts how many tokens user can use and makes optimization more challenging. LoRA does not lose or worsen performance without these issues.

2. Retrieval Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) is a new strategy to increase the accuracy, clarity, and flexibility of Large Language Models (LLMs). Since the model takes into account useful externally sourced documents as input, RAG is not a typical LLM (Lewis et al., 2020). It is able to dynamically and contextually utilize real-time data due to its neural retriever and sequence-to-sequence generator. The hybrid system assists LLMs with their problems such as the hallucinations, outdated, and generalizations of various subjects.

How RAG Improves LLM Responses

Original LLMs are parametric implicit storehouses of knowledge that remember everything that they learn in their weights. It is very difficult to follow the latest updates of the memory, make sure the facts given out are proper and tell the truth regarding what is happening as to this design. These issues are resolved in RAG because it keeps memory and model parameters apart. An ordinary LLM would formulate a definition to a concept of middle ear but a system endowed with RAG would seek a medical record that would provide a correct definition and then propose an answer. RAG enhances the accuracy of the responses, as well as provides links to contents that the user is able to follow or read (Lewis et al., 2020).

RAG will also assist user to ensure details are up to date. User do not need to re-train the model, but may simply switch out the external retrieval corpus. Because RAG is modular,

it is particularly helpful in industries that undergo rapid change, such as healthcare, law, and customer service.

Comparison with Other Methods

The following section contrasts RAG with prompt engineering, fine-tuning, and traditional LLMs to highlight the advantages and disadvantages of each.

LLMs in the Past: These models are fast and adept at drawing broad conclusions, but they frequently invent things and grow stale. Additionally, their responses lack a source, which reduces their usefulness for crucial assignments.

Fine-tuning: Fine-tuned models are highly specialized and limited to a particular job or style. They need to be retrained to keep their knowledge up to date, which costs a lot of money, and they tend to fit too well in small areas (Gu et al., 2021). Fine-tuned models use weights that are based on what user know about the domain, but these weights don't change after the model is deployed.

RAG: RAG helps us get new information quickly and doesn't take as many resources to change. For example, a well-tuned model might need to be retrained to include the most recent medical guidelines. A RAG model can do this by updating its corpus. This makes it cheaper and easier to use.

Prompt Engineering: Changing the inputs of a model changes its outputs, but prompt engineering has no basis in facts from the real world. RAG makes prompt engineering better by adding real retrieved documents to the input sequence. This improves the accuracy of the responses. By directly inserting retrieval signals into the model's embedding space, R2AG and other sophisticated systems go one step further. This further facilitates the collaboration between generation and retrieval (Zhao et al., 2023).

RAG Applications and Use Cases

RAG is beneficial in numerous ways:

Open-domain Question Answering: By improving the accuracy and usefulness of responses, RAG has enhanced datasets such as Natural Questions and WebQuestions (Lewis et al., 2020).

Fact Verification: RAG typically does this without assistance, correctly classifying claims in tasks like FEVER by searching Wikipedia for them.

Customer service: IBM and Salesforce, for instance, use chatbots that integrate with RAG to provide real-time responses based on documents. The responses become more accurate and reliable as a result.

Content Creation: RAG is used by Jasper.ai and other websites to create fact-based marketing materials that are unique for every company.

Autonomous Agents: Agents with access to LLM and RAG can retrieve information from the internet or long-term memory, enabling them to make better decisions instantly.

Search Engines: RAG-inspired techniques make it easier for tools like Google MUM and Bing Chat to generate snippets and provide answers to queries.

General NLP: By providing factual context to inputs and hidden layers, RAG enhances translation, summarization, dialogue systems, and classification tasks.

RAG is a big step forward in designing language models because it uses both parametric and non-parametric memory. It helps LLMs give answers that are based on facts,

up-to-date, and easy to understand. This lets powerful AI systems work in a lot of different fields.

3. Prompt Engineering

One of the best ways to improve Large Language Models (LLMs) is to carefully plan and improve the input prompts that are given to them. The best way to use these powerful models is to make sure that the output is correct, useful, and makes sense. This field has changed a lot, from simple empirical methods to a well-organized area of research.

How Prompt Engineering Improves LLM Responses

When user use prompt engineering, LLMs give better answers because the input is clearer and more organized. There are both easy and hard ways to do this:

Basic Ways: Some of these are giving clear instructions, giving people roles (like "User are an AI expert..."), using delimiters to separate parts of a prompt, and trying out different answers to see which one works best. Zero-shot, one-shot, and few-shot prompting help people respond by giving them different numbers of examples (Brown et al., 2020; Wei et al., 2022).

Advanced Techniques: Some of these are Chain-of-Thought prompting for step-by-step reasoning (Kojima et al., 2022), Self-Consistency for picking the most consistent answer (Wang et al., 2022), and Prompt Optimization for making prompts better (Zhou et al., 2022). Using more complicated methods like Tree of Thoughts or Decomposed Prompting can help user solve problems better (Yao et al., 2023; Press et al., 2022).

Multimodal Prompting: Prompt engineering helps models that work with both text and images, like Vision-Language Models (Zhou et al., 2022), bring together language and visual understanding.

Comparison with Other Methods

Prompt Engineering and Fine-Tuning: Fine-tuning means changing a model's internal settings so that it can do certain things better. This often needs a lot of storage and computing power. But prompt engineering only changes the input prompts and leaves the model parameters the same. This makes it work better, especially for big models or models that use APIs and don't let user directly access weights (Lester et al., 2021). Prompt tuning, which is a part of prompt engineering, only needs a few parameters—sometimes more than five orders of magnitude fewer—than full model tuning. It is more efficient in situations when the data is not many, or the field has altered (Zhou et al., 2022).

Prompt Engineering and Retrieval-Augmented Generation (RAG): After coming up with the LLMs, RAG improves its performance by identifying and introducing external knowledge that relates to the prompt. This assists in preventing occurrence of hallucinations. However, RAG is not a different approach, as it is commonly applied to the working processes of prompt engineering to increase the depth of truth and context (Lewis et al., 2020). Two cases of how timely engineering can interact with other knowledge systems are the ReAct framework and ART (Yao et al., 2022).

Use Cases of Prompt Engineering

Prompt engineering has become a helpful and versatile means to enhance the performance of big language models (LLMs) in many diverse domains. In school it enables the teaching staff to tailor educational content and provide individual feedback to students to

address their needs. With prompt engineering as the tool of material production, user will find it less complicated to write stories and other texts that make sense in more than one language. This will make the population more efficient and allows it more language choice. In the world of programming, it's a good thing that it helps programmers write and fix code faster. When people have to think about math and logic problems, well-designed prompts help them do better. Prompt engineering is also very important for making datasets because it creates fake data or data with labels for training. It is used in security to find and fix problems with LLMs, like prompt injection or stealing models. Overall, prompt engineering makes the results of LLMs much safer, more accurate, creative, and logical. It helps user get the most out of new language models.

Prompt Engineering

As generative AI becomes more common, prompt engineering is becoming a more important skill to have. This is the process of telling AI systems exactly what to do to get the results user want (Patil & Puranik, 2024). More specifically, it means making sure that large language models (LLMs) like ChatGPT can understand what the user wants by writing clear and simple input prompts (Ekin, 2024). This process is both technical and creative; it is a "art" to shape prompts so that the answers are useful, correct, and appropriate for the situation (Bansal, 2024). In this way, prompt engineering is a very important link between what people want and what machines can do.

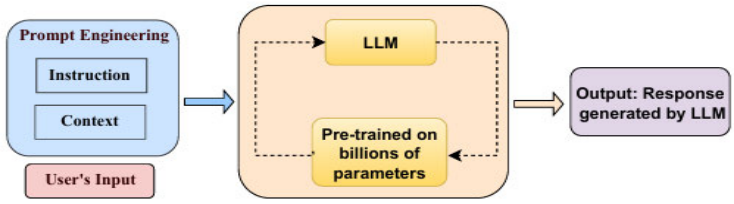

Figure 9: Visual breakdown of prompt engineering components, highlighting the importance of large language models (LLMs) trained on extensive data, and the role of instruction and context in shaping AI outputs.

Source: Sahoo, Kumar, & Singh (2024), "A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications," arXiv. https://arxiv.org/abs/2402.07927

Prompt engineering is important because it directly affects how good and useful AI-generated content is. Ceurstemont (2025) says that even small changes to how something is written can change what AI makes a lot. For generative AI systems to work well, prompts need to be very clear. When user give AI models like ChatGPT and DALL·E well-written prompts, they work better. This makes them better for tasks like creating content, analyzing data, teaching, and helping customers (Ceurstemont, 2025; Ekin, 2024). Prompt engineering is also important for making sure that AI answers are useful, relevant, and in line with the organization's goals (Patil & Puranik, 2024).

Prompt engineering is also an important part of using AI in a responsible way. It helps reduce bias and makes sure that the results are fair, include everyone, and are aware of social issues (Bansal, 2024). Well-thought-out prompts can also help with uncertainty by either asking for more information or showing different ways to think about hard questions (Ekin, 2024). Prompt engineering makes AI systems easier to use and more reliable as they get better (Patil & Puranik, 2024).

Users need prompt engineering because the generative AI systems users have now don't understand natural language very well. These systems are very powerful, but they often need carefully chosen inputs to give accurate, coherent, and contextually appropriate answers (Ceurstemont, 2025). Bansal (2024) says that AI still needs people to tell it what to do, give it context, and set goals for it to work well. If the prompts aren't set up well, the model's answers might not be clear, correct, or useful.

I believe that prompt engineering is more than just a technical fix; it's also a way to communicate in a smart way. It shows that people need to be more careful when they use

smart systems. Instead of just people who use content made by machines, it turns users into AI partners. This skill will become more important as users move to places with more AI. This is not just for engineers or data scientists; anyone who uses AI tools can use it. Some people think that AI systems will become so smart in the future that they won't need prompts that were made by people (Ceurstemont, 2025). But I think that prompt engineering will always be needed in some way. Even if an AI were "perfect," people would still need to be clear about what they want and how they plan to get it, especially when making tough or moral decisions.

When working with AI models that make text, code, or images that look like people (Ceurstemont, 2025), prompt engineering is very important. It's important to do this if user want to use AI in business, like systems that automatically handle customer service or find fraud (Ceurstemont, 2025). Companies need to pay for prompt engineering (Patil & Puranik, 2024) to make sure that these tools help them reach their goals and get good results. It needs to happen on a larger scale when the user and the machine don't agree on what they want. This difference is not likely to go away, even if AI gets better and better.

In short, prompt engineering is not only a technical need, but also a strategic one. As users put more AI in schools, businesses, healthcare, and the arts, it's important to know how to make prompts so that AI can help user.

Prompt Engineering Technique

1. Persona Technique

According to Furukawa (2024, p. 34), a persona in the context of large language models (LLMs) is a personality profile that includes things like age, gender, or job. It is a structured and consistent identity that affects how the model reacts (Olea et al., 2024, p. 34).

Personas are very important for prompt engineering because they help make interactions with LLMs more like certain behaviours or areas of expertise (Furukawa, 2024, p. 34).

How the Technique Works

The persona technique changes how the model responds by putting a character profile right into the prompt. For example, user could say, "One is a civil engineer" (Kim et al., 2024, p. 35). User can use this method in three main ways. Handcrafted personas are the first type. The user clearly defines them with examples like "20s, Executive" or "Act as an intelligent researcher" (Furukawa, 2024, p. 36; Olea et al., 2024, p. 35). The second method makes LLM-generated personas on the fly for each query, which means they can be used in a variety of situations (Olea et al., 2024, p. 36). Lastly, multi-agent personas give each agent a different expert role. This lets them work together or think about a task in a certain way (Olea et al., 2024, p. 36).

Why It Is Effective

There are many ways that persona prompting has been shown to help people do their jobs better. For example, combining demographic factors like age and profession, such as comparing "20s, Engineer" with "60s, Designer," had a big effect on idea evaluation scores, which made the evaluations better (Furukawa, 2024, pp. 37–38). Expert personas did better than generic control prompts on tasks that required creative or subjective reasoning. This shows that they were better for open-ended tasks (Olea et al., 2024, pp. 39–40). Also, domain-specific personas, like the "Mathematician" persona, have been shown to help people think logically on certain tasks, which shows how well tailored persona prompting works (Kim et al., 2024, p. 38).

Examples of Application

Idea Evaluation: The "40s, Designer" persona changed how people thought about creativity and feasibility in their answers (Furukawa, 2024, p. 36).

Answering Questions: Auto-generated expert personas made fact-based questions more accurate (Olea et al., 2024, p. 36).

Reasoning Tasks: A "Civil Engineer" persona came up with confident but wrong math answers, showing how powerful and dangerous persona use can be (Kim et al., 2024, p. 35).

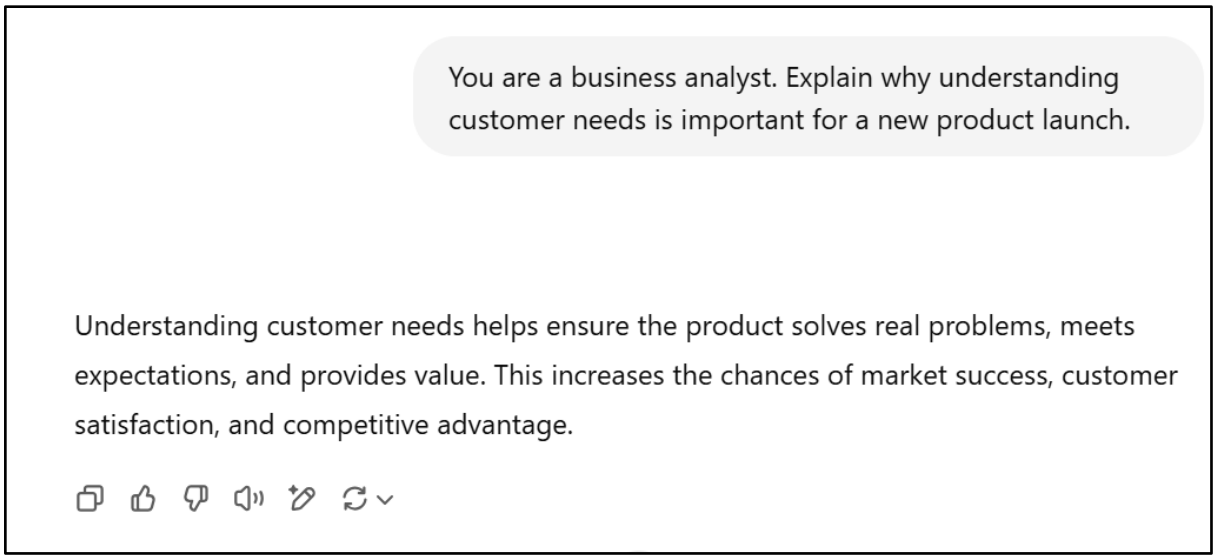

Figure 10: Screenshot of a Response Generated by ChatGPT in the Business Analyst Persona.

Why the Technique Enhances LLM Performance

The persona technique changes the model's tone, depth, and point of view, which changes how it reacts. It helps copy how experts think, which makes the results more like what users expect in specialized tasks. Using the wrong persona, on the other hand, can hurt performance. For instance, using an engineering persona to solve math problems could lead to mistakes in logic (Kim et al., 2024, p. 35). The Jekyll & Hyde framework fixes this by giving both persona-based and neutral answers and then using an LLM evaluator to choose

the best one (Kim et al., 2024, p. 36). Using LLM-generated personas with this mixed method made the results more stable and accurate (Kim et al., 2024, p. 38).

2. Clarity

Making sure that prompts are clear is an important part of natural language processing (NLP). It implies ensuring that the guidance provided to a language model (LM) is concise and unambiguous. The calibrated prompt should contain an unambiguous question (Chiodi et al., 2023). The initial segment of the CLEAR Framework also emphasizes the extent to which prompts should be brief and precise. The reason is that models of AI need not pay attention to the most essential components (Sparks et al., 2023).

How Clarity Works

With the clear prompts, there is no guessing part of what the user wants because the AI is told what to prioritize and what the user wants (Klie et al., 2023). Clarity makes users realize what they desire. This includes ensuring that the question is well understood and emphasizes correct answer and clear guidelines. It is also worth finding the balance between being too general and too specific. It is essential to be precise, yet the over-specific prompts may decline the variability or bias of the responses (Ganesan et al., 2024). It is highly paramount to watch user words as it has been demonstrated that the arrangement of the words in the prompt can even mix up the quality of the output (Ganesan et al., 2024). It is a good suggestion to add a line at the end of the prompt to inform the model to ask questions in order to clarify things (Chiodi et al., 2023).

Why Clarity is Effective and Helps Get Better Responses

The prompts provided greatly affect the effectiveness of AI models. Prompts should be structured to be clear and, in that way, better answers are likely to be given (Sparks et al.,

2023). It has been demonstrated that accuracy rates can increase to even over 98 percent after good prompt engineering is applied and this also incorporates being clear (Chiodi et al., 2023). Another valuable aspect about a good prompt is that it is easy to comprehend (Ganesan et al., 2024).

In many cases, bad prompts do not contain sufficient information or are too generic, which makes it harder to use them. When user do not use clear language, the people may reply to user in a vague way that cannot assist to achieve objectives. In case the AI lacks sufficient context or details, it may fail to determine what the question is. This would result in stock solutions that are out of tune with what the user needs (Klie et al., 2023). In case the prompts are quite general and lack sufficient details, the AI may be confused and provide generalized and unrelated answers (Chiodi et al., 2023).

Having clear and direct prompts, the user would decrease the risk of receiving blank, general, or unrelated answers. Specific and clear prompts give better and precise responses. They assist in ensuring that the skills of the AI are appropriate to the needs of the user. This is particularly necessary in the apps where the clarity and relevancy of the answers influences the level of happiness of the user (Sparks et al., 2023). Such imprecision in prompts may cause incorrect responses and increased biasness. Conversely, in the event of clear prompts, they assist the AI in performing better and in providing answers that are nearer to the objective (Ganesan et al., 2024).

Examples

The sources present a couple of illustrations on how vague or unclear prompts are contrasted with specific and clear ones:

"Tell me about a good book" is too vague. A better prompt would be, *"Can user suggest a non-fiction book about personal growth?"* This level of detail helps get rid of any confusion (Chiodi et al., 2023).

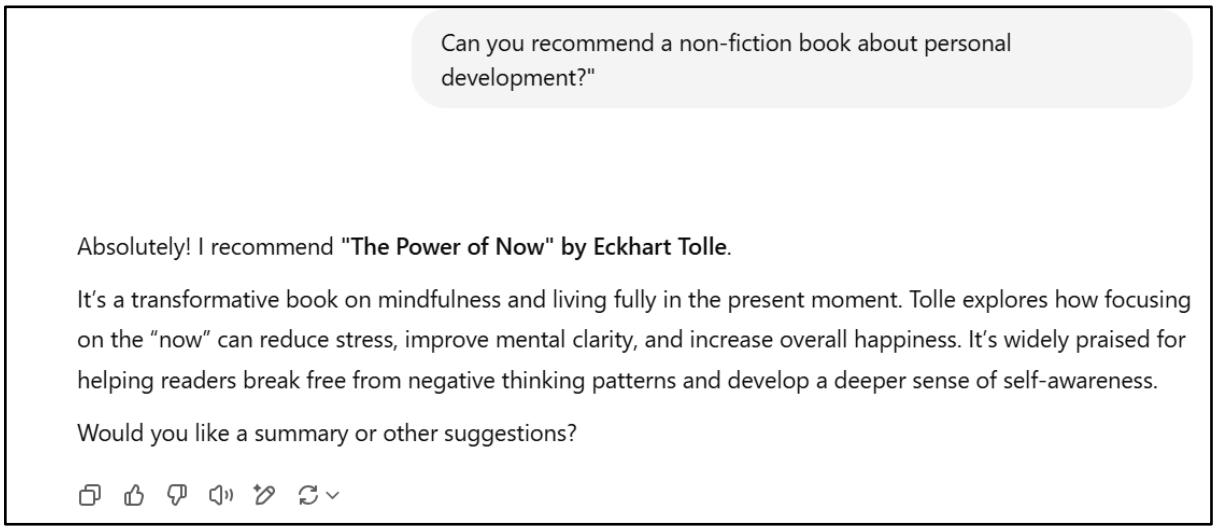

Figure 11: Screenshot of ChatGPT's recommendation of the personal development book

For digital marketing, instead of asking, *"Tell me about digital marketing,"* a more effective prompt clearly defines the task by asking, *"Write a 200-word summary of the latest trends in digital marketing"* (Klie et al., 2023).

When it comes to being clear and to the point, *"Explain the process of photosynthesis and its significance"* is better than *"Can user give me a detailed explanation of the process of photosynthesis and its significance?" "Identify factors behind China's recent economic growth"* is also shorter than asking for a "extensive discussion" on the same topic (Ganesan et al., 2024).

3. Task/Instruction" Prompting

Basic and fundamental prompt presentation technique is called Task/Instruction, which allows user to control the output of large language models (LLMs) and vision-language models (VLMs) by designing instructions carefully around each task. This approach can be used to easily extend the pre-trained models to downstream tasks by only making the model to act in a specific desired manner depending on what the model receives as input without modifying the core networks parameters (Wei et al., 2022). The developing of prompt engineering has given LLMs a lot more flexibility, and they perform well in many tasks and fields, such as question answering and common sense (Brown et al., 2020).

A simple and yet important way of employing prompt engineering is by providing directions. It is to communicate with the LLM what it must do in an unequivocal manner. Provided that instructions on prompts are congruent with the task, they constitute a significant component of the future response that the LLM will have (Ouyang et al., 2022).

Here are some important parts of "Task/Instruction" prompting:

Giving Full Descriptions: Rather than providing loose or brief descriptions of tasks that may result to generic outputs user ought to provide clear and elaborate descriptions. This will assist user in coming up the right and helpful answers. Without any specific instructions, LLMs trained by large datasets are more likely to provide general answers (Wei et al., 2022).

Clarity and Accuracy: Its model is less confusing to use when there are clear prompts and assists in providing more precise and defined answers. This degree of accuracy is easier to achieve when fulfilling the expectations of users (Brown et al., 2020).

Role-Prompting: To teach the model, one can provide it with a job, e.g. be a historian or an assistant. Such framing provides the model with a perspective, which alters the tone, depth, and level of details in its answers (Reynolds & McDonell, 2021).

Structured Formatting: Organized formatting, such as triple-quotes or indented parts, enables the model to note how to execute orders, particularly in cases when they are challenging or comprise multiple procedures (White et al., 2023).

Why It Is Effective and Helps Get Better Responses

When researchers provided big language models with task instructions and task strategies such as CoT prompting, they found that they were more accurate and logical. As an example, CoT cues such as "think step by step, please" encourage models in thinking up stages of reasoning so that their outputs are more understandable and structured (Wei et al., 2022). The task-driven instructions are also beneficial in that, the models get to learn more since thinking through the instruction helps them think of the problems rather than simple answer provision. It has proven to be extremely useful when it comes to assignments that demand critical or multi fact-based thinking (Ouyang et al., 2022). These instructions also render models more versatile, such that they can apply what they have learned on emerging tasks with minimal or no additional training (Zhao et al., 2023). In addition, instruction-based prompts help trace the steps that logically result in model conclusions. It can be helpful to detect bugs along with ensuring that things are up to standard (Reynolds & McDonell, 2021). This ensures that the process is clearer and visible. According to White et al. (2023), the use of strategic task instructions improves the performance of big language models significantly in many contexts, such as education, healthcare, and financial services, compared with simple or so-called vanilla prompts, which simply pose a question.

4. Example

Prompt engineering is an example as a method of prompt, or alternatively termed as in-context learning because an individual is shown how to perform something correctly in the prompt. Such an approach allows large language models, including, GPT-3, to perform linguistic tasks that they have not previously performed without having to update the internal parameters of the model or modify the gradient (Brown et al., 2020). In-context learning usually happens in three main ways:

Few-Shot Learning: The prompt gives user a few examples of the task (10–100), and they usually come from the same distribution as the test input.

One-shot learning: The prompt only shows the task once, usually with instructions in natural language.

Zero-Shot Learning: There are no examples in the prompt for zero-shot learning; it only has an instruction.Fine-tuning is different because it updates a model's weights by training it on a large labelled dataset, which often needs thousands of examples that are specific to the task. In-context learning lets user use the model's parameters to make inferences about tasks without changing them.

Table 4: Examples of In-Context Learning Types with Descriptions and Prompts

Type of In-Context Learning	Description	Example Prompt
Few-shot learning	Prompt contains several examples of the task	"Translate English to French: 1. Cat → Chat 2. Dog → Chien 3. House → Maison Translate: Bird →"
One-shot learning	Prompt contains one example plus instruction	"Translate English to French. Example: Cat → Chat. Now translate: Dog →"
Zero-shot learning	Prompt contains only an instruction, no examples	"Translate the word 'Dog' from English to French."

How the Technique Works

Prompt conditioning is what makes in-context learning work. A pre-trained LLM is given a set of examples that are formatted within the input sequence. These examples are like "soft training" for the model to learn what the job is and what the result should be. The model learned from a lot of data from the internet and can use its general knowledge and ability to find patterns to finish new tasks by following the structure of the prompt (Min et al., 2022).

For example, when using in-context learning for Visual Question Answering (VQA) tasks, models like BLIP2 are shown pairs of questions and answers that only have text. These don't have pictures with them during the demonstration, but they help the model understand how the question is set up and what the answer should look like. This lets the model "adapt" to new tasks by making guesses about what will happen next based on the examples in the prompt (Chen et al., 2023).

Why It Is Effective

There are a lot of reasons why using examples as a prompt works. First, few-shot learning needs a lot less labelled data than traditional fine-tuning. This makes it great for tasks with few resources. Second, learning in context makes it easier to use what user have learned in a lot of different situations right away. This ability improves when the model is larger (Brown et al., 2020). Third, large language models are dynamic in the sense that they can evolve just like humans when given limited examples of the task structure and intent. This is how people learn a lot when one is following instructions. Finally, scaling is quite feasible with this method since it is more effective with larger models, and the performance is improved with an increased number of examples and a larger model (Wei et al., 2023).

Examples of Effectiveness

GPT-3 also indicates that its approach can be applied to many various scenarios, given that it performs effectively even in the few-shot setting. Large language models (LLMs) have performed quite well across a broad scope of language tasks, sometimes performing as well as or better than fine-tuned smaller models. GPT-3 scored 86.4 percent of the responses on the LAMBADA dataset. The work needs future interpretation of the situation. It aimed at predicting the final word of sentences relying on much information (Brown et al., 2020). When tested on the narrative completion task HellaSwag, GPT-3 also performed better in comparison with smaller, fine-tuned models with the accuracy of 79.3%.

It was just as good or better than fine-tuned models in answering questions. It achieved 71.2% accuracy in TriviaQA using a small number of examples and 85.0 F1 score on Conversational Question Answering (CoQA) dataset in few-shot mode, which is almost as high as would be achieved by a person.

The few-shot texts abilities yielded comparative outcomes to cross-lingual translations by the finest unsupervised neural machine translation systems in French-English and English-French translations. This demonstrates the flexibilities of LLMs in language pairs.

In terms of reasoning, GPT-3 correctly answered 98.9% of subtraction problems and 100% of two-digit addition problems in few-shot mode. On SAT analogy tasks, it performed better than the majority of college applicants, correctly answering 65.2% of the questions. This indicates that it excels at word games and analogical reasoning.

Multimodal tasks, such as answering questions with pictures, can also benefit from thoughtful prompt design. When visual cues such as image captions were combined with

text-based Q&A examples, BLIP2 performed better. This demonstrates the significance of using various prompt types to increase model accuracy (Li et al., 2023).

Why This Method Gets a Better Response

Although it has certain drawbacks, learning with examples in a few shots is beneficial. One major issue is that the model's performance varies significantly based on the prompt's example selection and order. Accuracy can vary greatly depending on the combination, ranging from nearly random to nearly perfect (Zhao et al., 2021). Furthermore, maintaining high performance is difficult due to model biases. These biases include recency bias, which shows that the model favours answers that it has seen more recently, common label bias, which shows that the model tends to favour answer types that are most prevalent in the prompt, and common token bias, which shows that the model is more likely to produce outputs that contain tokens that are prevalent in its pretraining data, even if they are unrelated to the task.

To increase the dependability of few-shot learning, these issues can be resolved in a number of ways. User can detect and fix output biases by displaying a "null" prompt, which is an empty or masked input, using contextual calibration. The model's responses are therefore more accurate and less likely to change. The model will also be better able to understand and finish the task if the instructions are rewritten to make them easier to read and follow, for instance, by using bullet points or segmenting challenging tasks into smaller, more manageable pieces. This is what users term as prompt reframing. The image caption in question or similar visual aids (Chain-of-Thought justification of purely text-based prompts) have been proved beneficial in making models perform better on visual question-and-answer tasks, increasing the information available.

Such changes do more than improve the stability of performance, they also make in-context learning a scalable and low-resource (compared to fine-tuning) solution. As bigger models such as GPT-4 and its follow-ups are developed, the use of even a handful of well-selected examples will grow increasingly useful in any number of domains.

5. Zero-Shot Prompting

Zero-shot prompting is the process applied to the Large Language Models (LLMs) that teaches the model to act in specific natural language way without giving it any example to relate to (Yin et al., 2023). Cases Because it is an in-context learning, the prompt is not dependent on a specific example of input that is being solved (Zhao et al., 2023). It utilizes the high volume of data that is employed in training the LLMs. This, as stated by Kim et al. (2023), allows them to generalize and conduct extensive tasks without a need to be configured individually or have labelled data.

Large models have so-called emergent abilities, and they encapsulate the interplays between these skills that models with fewer parameters can not develop (Yin et al., 2023). The models do not require examples now because they can understand and do things using only a well-written instruction.

How Zero-Shot Prompting Works

The main aim of zero-shot prompting is to make the task apparent and as simple as possible using plain language. The prompt usually has a place to enter data and an instruction. Sometimes it also has a place to put the output (Zhao et al., 2023). For instance, *"Translate the following English sentence to French: 'Hello, how are?'"* does not need any training examples; it just needs a clear instruction.

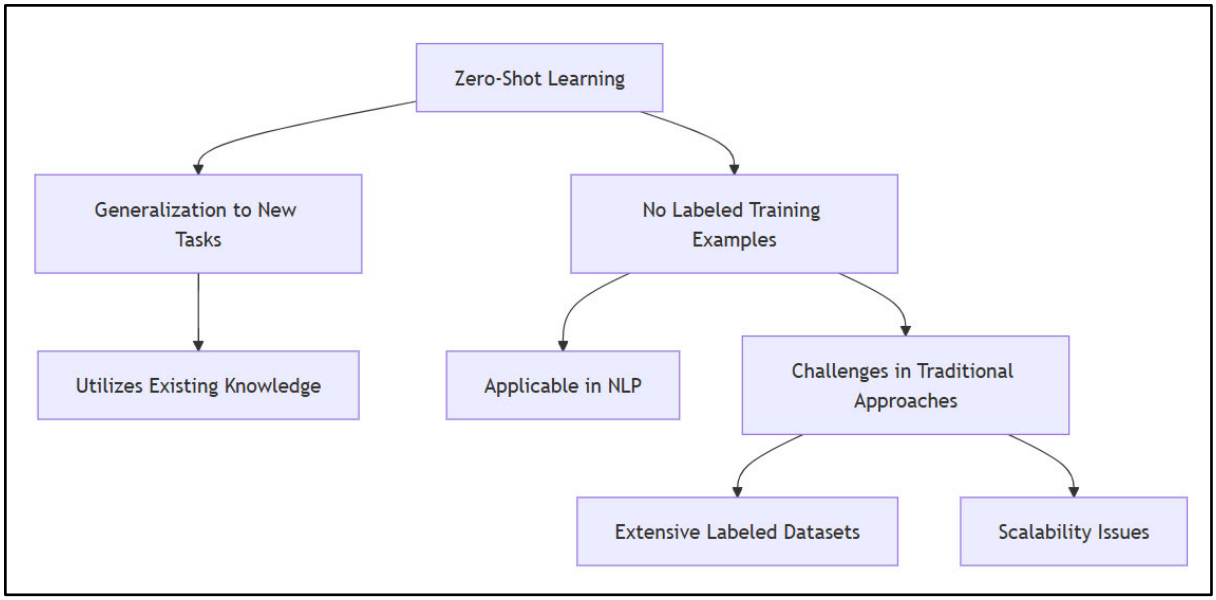

Figure 12: This flowchart explains Zero-Shot Learning (ZSL), showing how it handles new tasks without labeled data and the challenges of traditional methods. It's useful for AI engineers to grasp ZSL basics.

This mechanism basically changes a task so that it looks like the model's pre-training goal (like next-token prediction or masked language modelling). This lets it figure out the right output structure just from the language context (Kim et al., 2023). Depending on how the task is set up, well-thought-out prompts can make the same model give very different or very structured answers (Haque et al., 2023).

Why It Is Effective and Helps Get Better Responses

There are many benefits to using zero-shot discrete prompts that are written in a way that people can understand. They are simple to understand, don't require a lot of labelled data, and are simple to create because all they need to do is give task instructions (Yin et al., 2023). The design is flexible, so users can change the input and expected output without having to do a lot of prompt engineering.

LLMs are helpful because they can apply what they learn in a lot of different situations. In the Text2SQL task, which turns natural language into SQL queries, prompts

that included the database schema and sample content did much better than those that only included the question, even when there were no examples (Zhao et al., 2023). This shows that providing context is important for good performance, even if there are no examples.

But it is known that prompt design is fragile; even small changes like changing the order of words or punctuation can have a big impact on the quality of the output (Kim et al., 2023). This implies that zero-shot prompting is quite handy, yet it performs optimally in occasions where the language is highly explicit.

Examples of Zero-Shot Prompting Strategies

Some common ways to get zero-shot are:

Simple Steps: Simply tell them what to do, like "Summarize the article below."

Task and Label Description: Instead, one can add more information by providing potential labels or groups such as "Label this review as positive, negative, or neutral."

Questions to think about: Inserting the prompts such as "Why don’t users think step by step" to make the model apply its reasoning capabilities similarly to chain-of-thought prompting (Yin et al., 2023; Kim et al., 2023).

Visual Question Answering (VQA) experiments indicate that the design of zero-shot prompt, i.e., employing style QA template or providing model with instructions, can significantly influence the level of accuracy (Haque et al., 2023). Phrasing of the good prompts is also very important as evident with the different words that are used to the labels of different classes being used to aid in the classification tasks.

Impact on Getting Better Responses

Users like zero-shot prompting because it is simple and adaptable. In some cases, well-made zero-shot prompts have worked better than few-shot prompts, especially when examples don't matter or make things more complicated (Zhao et al., 2023). However, it is not always a good option. Where task specific knowledge is highly essential, fine-tuned models or few-shot prompting can be more practical than a zero-shot strategy.

The inclusion of additional information such as paper titles (which models might have been trained on) can usually not help and is even sometimes obstructive (Haque et al., 2023). This indicates that prompt difficulty is not so important but its clarity and pertinence.

The result is that zero-shot prompting allows a plain user to leverage whatever capabilities that LLMs can bring without having to invest on Kangaroo data labelling or model fine-tuning efforts. It can be successful only due to a set of well thought-out prompts to provide clear instructions and offer a context so that the goal of the user could be achieved as close as possible through the work of the model.

6. Few Shot Prompting

Few-shot prompting is a technique in natural language processing (NLP) of which the user provides a small number of examples of the input-output behaviour they desire to have in the prompt. The technique allows large language models (LLMs) to perform inference by transferring what they have learned to situations they have not encountered before and generate suitable responses without fine-tuning the model. It differs with zero-shot prompting (which does not provide any examples) and one-shot prompting (which provides one). Few-shot prompting is an excellent method to enhance the quality and quality of responses and

ensure that they remain on task since it makes the model comprehend the structure and context much better (Brown et al., 2020; Zhang et al., 2022).

How Few-Shot Prompting Works

Few-shot prompting involves putting a few instances of tasks in the prompt itself. The demos often have several output-input pairs that resemble what the user desires in format and structure. Illustrating patterns, the model becomes "interconditioned" to deduce the task and repeat it with a different query. The technique involves in-context learning, and this is an activity that LLMs such as GPT models are capable of performing. The cues in the prompt can enable them to deduce generalization patterns (Min et al., 2022).

The few-shot examples guide the model to learn to reply in proper response, tone, or format. They give examples to make vague instructions clearer, which helps the model figure out what it needs to do. These examples also give the model patterns that are specific to each task that it can use when it gets new inputs that it hasn't seen before. This helps it learn more and do things correctly.

The number and quality of the examples have a big effect on how well the performance goes. Usually, a carefully chosen set of 3 to 5 examples is all the model needs (Brown et al., 2020).

Why Few-Shot Prompting is Effective and Helps Get Better Responses

Few-shot prompting is effective in that, the model learns to interpret abstract text-only instructions. It is also more effective to use a few-shot prompting than to give instructions. This can be quite helpful when the required task is difficult or specialized and the general training data involved in the training of the model could be irrelevant to the user (Zhao et al., 2021).

These researchers concluded that few-shot prompting is significantly more accurate than zero-shot prompting on sentiment classification, question answering and summarization tasks (Brown et al., 2020; Zhang et al., 2022). On some tasks that involved processing natural language, GPT-3 did up to 20 percent better when it was switched to few-shot mode instead of zero-shot mode (Brown et al., 2020).

The few-shot prompting can also contribute to the reduction of the variety of possible errors in the interpretation and increase consistency in the answers that models generate by reducing the variability. It assists the model remain in the correct form when it talks with the giving the model structured examples and enabling it to apply multi-turn reasoning. Furthermore, fewer shots of prompting also ensure the final result can be more aligned with the desires of the user, so they are more satisfied with the task and the output is more pertinent when creating one of the more structured outputs, such as tables, summaries, or decision-making structures (Min et al., 2022).

Examples

Example 1 (Sentiment Classification)

Few-shot:

"Review: 'I loved the food and the service was excellent.'
Sentiment: Positive
Review: 'The experience was awful and the room was dirty.'
Sentiment: Negative
Review: 'The movie was terrible and boring.'
Sentiment:"

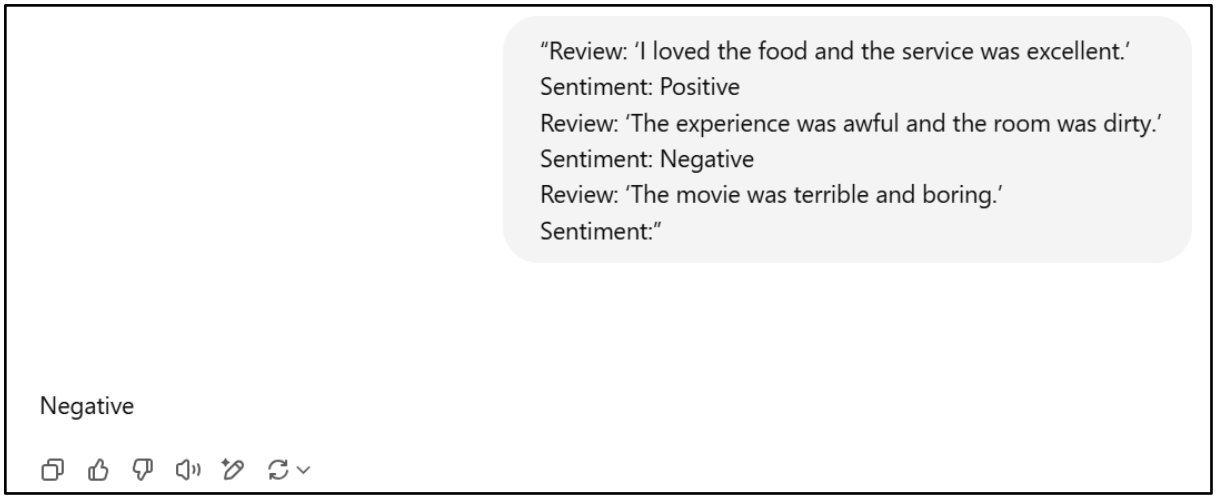

Figure 13: Few-Shot Prompt for Sentiment Classification. This chatGPT screenshot demonstrates a case of few-shot learning where by identifying patterns on previous labelled samples; the model determines the sentiment of the last review.

Source: Generated by ChatGPT based on Zhao et al. (2021).

In this case, the model will be able to deduce by patterns that the final review will be defined as "Negative" (Zhao et al., 2021).

Example 2 (Grammar Correction)

Few-shot prompt:

"Incorrect: 'He go to store.'

Correct: 'He goes to the store.'

Incorrect: 'She not like pizza.'

Correct: 'She does not like pizza.'

Incorrect: 'They eats fast.'

Correct:"

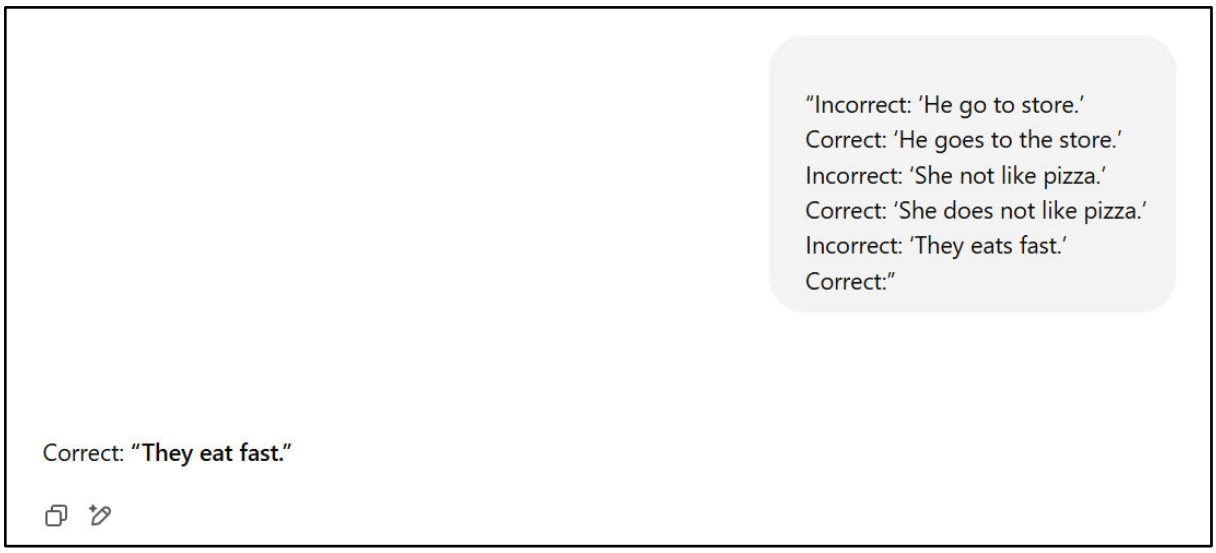

Figure 14: Few-Shot Prompt for Grammar Correction
This ChatGPT screenshot reveals that it can be used in few-shot prompting to correct grammatical errors. The model takes the labelled dataset of incorrect and corrected sentences to learn to predict how a new sentence should be written.

Source: Generated by ChatGPT based on Brown et al. (2020).

It goes on: "They eat quickly." with effective pattern generalization.

Conclusion

Few-shot prompting is an extremely beneficial efficient approach, which exploits the capabilities of large language models to identify patterns and generalize. By providing structured examples to the model, users can assist the model to make superior outputs with less errors. It is particularly useful in the case of custom applications and tasks that have special requirements based on a specific field of application where examples are better than instructions. As language models become more powerful, few-shot prompting remains an important component of reliable and controllable prompt engineering.

7. Chain Prompting

Chain prompting also known as LLM chaining is a new method to resolve hard questions into little manageable pieces. This work can then be sequentially performed on tasks by the large language models (LLMs). It is organized in such a manner that the output of one task can serve as the input of another one so today this way promotes a workflow. Wu and co-authors (2022) state that this approach aims at promoting communication and cooperation between people and AI.

How Chain Prompting Works

The process has a number of important steps. The first is the task decomposition that requires the division of a large task into smaller, well-defined subtasks, each related to a specific model operation and prompt. These subtasks demand primitive operations, the basic LLM activities that are used in generating chains. Typical primitives involve information collection (accessing the facts, creating content, and generating ideas), verification and classification (deciding whether a query can be answered), and re-organization (accessing a given data, formatting modifications, text into lists, or combining outputs) (Chase et al., 2023; Wu et al., 2022). Each of the links in the chain is driven by a natural language prompt, possibly with instructions, background data, and examples. The output of one step, also known as a data layer, is fed into the next step and a logic pipeline is formed (Wu et al., 2022). Moreover, interactive interface on some of these tools shows such chains and provides the opportunity to modify the results at every step of processing, modify sequence of calculations, and always enhance work on the task. This allows having modular and flexible control (Chase et al., 2023).

Why Chain Prompting Is Effective

Chain prompting, in numerous influential aspects, enhances the user experience and the performance of the model. Chain prompting has been found to facilitate classification and summarization. To give an example, it outperforms zero-shot ChatGPT in classifying legal documents and stepwise prompting in summary writing (Chase et al., 2023). Chain prompting also makes the model logic clearer and allows the user to have more control by letting them change the temperature during thinking to brainstorm or the order of the rewriting stages to provide improved results (Wu et al., 2022). It is also easier to track down and resolve bugs using this modular technique. Rerunning or rearranging of steps enables user to quickly find, isolate, and correct bugs on a hardware device since each chain step is a unit test (Chase et al., 2023). Chain prompting is also used to accelerate the performance of large language models by breaking down tasks that might be challenging, such as formatting, creativity and accuracy further into smaller steps. In such a manner, each step may leverage its advantages (Wu et al., 2022).

Chain prompting is also a solution to some of the most common problems with LLMs. It solves reasoning dilemmas whose steps are many by simplifying them. It also minimizes exposure bias by asking one question at a time with regards to new content segments. Finally, it resolves input sensitivity issues by altering inputs through basic operations (Chase et al., 2023). By reducing distractions and preventing errors unrelated to the task from propagating throughout the process, the set output and data connections for each step serve as barriers that help keep the task on course (Wu et al., 2022). Last but not least, chain prompting facilitates human-like iteration, particularly for tasks like summarization, where a multi-pass workflow comprising drafting, critiquing, and revising is more akin to the way editors operate and produces better results (Chase et al., 2023).

Examples of Chain Prompting in Practice

Peer review rewriting is the process of taking negative feedback and turning it into useful suggestions in an organized way. The first step, called "Split Points," divides the review into different parts. The next step, "Ideation," gives user ideas for how to fix those problems. Finally, a Compose Points step puts these answers together into a more polished and clear response that improves the tone and clarity overall.

Making personalized flashcards to help user learn English and French takes a few steps. The Ideation phase looks for interactions that make sense, like going to a restaurant. The next step, Generation, makes example sentences in English. The last step, Rewriting, changes these examples into French and makes flashcards that are useful and make sense in the context.

The point of debugging visualization code is to find and fix mistakes in VegaLite specifications. The first step is to rewrite. It changes the code into simple English descriptions. A Classification step then looks for specification violations and suggests the best ways to fix them. The last step of rewriting is to make the corrected code again and check that it meets all the standards.

Assisted text entry helps people by expanding abbreviations or finishing sentences. The first step in the process is to check if the input is shorthand. If the abbreviation is there, the Rewriting step adds to it. If not, the Generation step finishes the phrase, which helps people write faster.

Long legal document classification breaks down hard-to-read legal texts into smaller parts. Summary Generation makes long documents shorter. Then, Semantic Search looks for similar samples to give context. Finally, Label Generation uses in-context learning to give documents the right tags, which makes it easier to find and organize them (Chase et al., 2023).

Table 5: Examples of Chain Prompting in Practice

Use Case	Step 1	Step 2	Step 3	Outcome/Goal
Peer Review Rewriting	Split critical feedback	Ideate constructive fixes	Compose empathetic responses	Improved clarity and tone in reviews
Language Flashcard Creation	Identify context (e.g., dining)	Generate English sentences	Translate into French	Personalized and contextual flashcards
Visualization Code Debugging	Describe code in natural language	Identify spec violations	Generate corrected code	Bug-free VegaLite visualizations
Assisted Text Entry	Detect shorthand or partial input	Expand or predict phrases	Finalize sentence	Faster and user-friendly text input
Legal Document Classification	Summarize long documents	Retrieve relevant cases	Assign legal labels using prompts	Efficient legal document organization

Source: Adapted from Chase et al. (2023)

8. Chain of Thought

Chain-of-Thought (CoT) prompting is another next-level process that assists large language models (LLMs) to think better, as it makes them generate a sequence of intermediate reasoning steps. These steps, or rationales, which resemble the way people think, can help the model arrive at more adequate conclusions even in case of complex assignments (Wei et al., 2022).

Mechanisms of Chain-of-Thought Prompting

Chain-of-Thought (CoT) prompting is tooling to assist big language models (LLMs) to think logically. This can be done in many ways. As a common technique in few-shot prompting, manual CoT provides several examples of input, thought process, and output. The instances introduce this model to the procedure of problem-solving by displaying each step (Kojima et al., 2022). Conversely, Zero-shot CoT just provides an addition of such a phrase a

question, as Let us think step by step, after the question and also does not provide examples. It is a simple approach that does not require special skills to complete a task, which greatly simplifies reasoning tasks (Kojima et al., 2022). Automatic CoT (Auto-CoT) makes its own reasoning chains, which means that user don't have to do as much work by hand to make prompts. It chooses representative samples, groups questions to add variety, and tells LLMs to "Let's think step by step" (Zhou et al., 2023). Faithful CoT and Symbolic CoT (SymbCoT) are more advanced frameworks that break the process into two parts to include symbolic reasoning. First, they turn natural language into symbolic steps (like PDDL or Python code), and then they use deterministic solvers to find the answer. Faithful CoT combines symbolic logic with explanations in everyday language (Zhang et al., 2023). SymbCoT, on the other hand, combines Translator, Planner, Solver, and Verifier modules to make completely logical reasoning pipelines.

In our study, users looked at a number of different types of CoT prompts:

1. Zero-shot CoT: This technique provides a clue such as Let us think step by step when asked a question. It does not require any examples to be presented. Zero-shot CoT helps models become significantly more competent in terms of reasoning, despite the fact it is basic (Kojima et al., 2022).

When trying to find fraud, for instance, asking "Is this transaction suspicious?" and "Let's think step by step" lets the model look at the transaction history and find more specific signs of fraud.

2. Manual/Few-shot CoT: This version will also indicate examples of how to think in the prompt so that the model can learn how to make middle steps. According to Brown et al. (2020), it is effective when doing specific tasks that are exclusive in a particular area, like sorting taxes and assisting customers.

The prompts were these, and users automated customer service this way, example: Type: The customer wants his or her money back because the package was late. CoT: "Compare the policy times with the delivery date to check whether a refund is possible." This gave the impression that models were choosing.

3. Auto-CoT (Automatic Chain-of-Thought): Auto-CoT does not leave this to other people but instead creates its own different reasoning samples by combining questions and selecting prompts which are characteristic of the group. It reduces the labour cost and makes it easy to generalize (Zhou et al., 2023).

A logistics AI could answer the question, "What is the best route for delivery?" Auto-CoT gives examples like "Start from warehouse → Check traffic → Optimize by distance and time → Choose route with lowest delay."

4. CoT with Self-Consistency: This method looks at more than one CoT output instead of just one reasoning chain to find the answer that is most consistent. This ensemble-style method gives results that are more accurate and reliable (Wang et al., 2022).

For instance, an AI financial assistant checks risk levels by taking a number of routes: "Path A: First, look at debt, then income, and finally credit score. Path B: Look at credit history first and compare it to the limit. It chooses the conclusion that makes the most sense.

Along with these basic methods, users also added new CoT variations:

1. Step back Prompting: This metacognitive prompting method tells the model to look at its original logic and results again. It helps people avoid jumping to conclusions or making mistakes, which makes it easier to decide things like policy and product prices.

"Lowering the price to boost sales" is a common reason for doing something in retail planning. Then, it is told to "step back" and think again: "But will that change how people see the brand or how much money it makes?"

2. Analogical Prompting: This method helps the model figure out a solution by comparing the problem to a case or analogy that has already been made. This is very helpful for guessing what will happen in business and legal matters.

The model might say, "This is like the XYZ merger in 2020" when a merger is being planned. What happened next? What caused it to work or not work?

All of these methods help LLM responses be more clear and CoT's ability to break down hard tasks into logical parts get better. Users compared how well they worked, how easy they were to understand, and how adaptable they were to different parts of business, like operations, finance, and compliance.

Why CoT Prompting Is Effective

Chain-of-Thought (CoT) prompting works well because of a few key features. One important part is the rise of reasoning skills on a large scale: CoT usually only works well with very big models, like those with more than 100 billion parameters. For instance, PaLM-540B does much better than smaller versions like PaLM-62B on tasks that require multiple steps of reasoning because it has better semantic parsing and makes fewer logical mistakes (Chowdhery et al., 2022). Problem decomposition is another important feature. It lets models break down hard tasks into smaller steps, which helps them focus their computational effort better and improves both accuracy and interpretability (Wei et al., 2022). CoT outputs also show clear reasoning paths that help users figure out how conclusions are reached, which makes it easier to understand and fix problems. Standard CoT chains, on the other hand, don't

always follow the rules of logic, which means that the final answer may not strictly follow from the reasoning steps. Faithful CoT solves this problem by using executable logic to check the reasoning (Zhou et al., 2023).

CoT prompting can be used in many areas, such as arithmetic (GSM8K), commonsense reasoning (StrategyQA), and symbolic reasoning (Last Letter Concatenation) (Wei et al., 2022; Kojima et al., 2022). It is also strong against changes in example demonstrations, annotator style, and language use. Auto-CoT stresses the importance of having a wide range of examples to make it even stronger (Zhou et al., 2023). CoT is especially helpful for instruction-tuned models like GPT-4 and Flan-PaLM because they can find and fix bad reasoning. This shows that they are strong even when demonstrations have mistakes (Chowdhery et al., 2022; OpenAI, 2023). Finally, symbolic CoT frameworks like SymbCoT get the best results in logical reasoning by making fewer syntax mistakes and making answers more accurate and easier to understand (Zhang et al., 2023).

Practical Examples of CoT Prompting

Chain-of-Thought prompting helps with a range of reasoning types on different tasks. For example, in arithmetic reasoning, a question like "The cafeteria had 23 apples" How many are left if they used 20 and bought 6 more? is solved by breaking the problem down into smaller parts. For example, starting with 23 apples, taking away 20 used apples to get 3, and then adding 6 more purchased apples to get 9. This way of thinking in steps helps cut down on mistakes in calculations. For example, when someone says, "Bring me something that isn't a fruit," the model uses common sense to figure out that an energy bar fits the bill and makes plans to find, pick, and deliver it. CoT is also helpful for symbolic reasoning tasks. For instance, the Last Letter Concatenation problem, also known as "Waldo Schmidt," is solved by taking the last letters "o" and "t" from each name and putting them together to

make "ot." In a coin flip situation, the model starts with heads up, sees that one flip (an odd number) changes the state to tails, and correctly says the answer is "no." Frameworks like SymbCoT use first-order logic to figure out things like whether someone is part of a six-way tie, using multiple symbolic premises. Standard CoT might make these tasks too simple or get them wrong.

Table 6: Examples of Chain-of-Thought Prompting in Different Reasoning Tasks

Reasoning Type	**Example Prompt**	**Chain-of-Thought Reasoning Steps**	**Final Answer**	**Outcome/Goal**
Arithmetic Reasoning	"The cafeteria had 23 apples. If they used 20 and bought 6 more, how many are left?"	23 - 20 = 3; 3 + 6 = 9	9	Improved clarity and tone in reviews
Commonsense Reasoning	"Bring me something that isn't a fruit."	Identifies object types → Filters fruits → Chooses 'energy bar' as a suitable item	Energy bar	Personalized and contextual flashcards
Symbolic Reasoning	"What is the result of Last Letter Concatenation of 'Waldo Schmidt'?"	Last letters: 'o' from Waldo, 't' from Schmidt → Concatenates → "ot"	ot	Bug-free VegaLite visualizations
Coin Flip Logic	"A coin shows heads. User flip it once. Is it still heads?"	Initial = heads → 1 flip = odd = change → Now = tails → "No"	No	Faster and user-friendly text input
Logical/Symbolic (SymbCoT)	"If 6 people are tied for first place, is Alice included if she won 3 matches?"	Converts to logic form → Compares with tie conditions → Evaluates Alice's eligibility logically	Depends on logic chain	Efficient legal document organization

Source: Adapted from Wei et al. (2022); Kojima et al. (2022); LoBue et al. (2023)

Advantages of CoT Prompting

There are many good things about chain-of-thought prompting. First, it helps with structured reasoning by breaking down hard problems into smaller, easier-to-solve ones. This is similar to how people think and helps them do better on hard tasks (Wei et al., 2022). Second, models do much better when demonstrations include natural language reasoning steps instead of just equations. This is because these naturalistic chains fit better with how large language models are trained (Kojima et al., 2022). Third, approaches like Auto-CoT, which chooses a wide range of examples that are representative and diverse, help reduce the spread of bad logic. Faithful CoT, on the other hand, makes sure that reasoning processes are deterministic, which improves both accuracy and transparency (Zhou et al., 2023). Fourth, Chain-of-Thought prompting improves generalization, which means that models can solve problems that are harder than the ones they trained on, especially when it comes to symbolic reasoning tasks (Zhang et al., 2023). Finally, CoT uses what it already knows from instruction-tuned models to fix mistakes and make good reasoning even when demonstrations aren't perfect (OpenAI, 2023).

Limitations

Chain-of-Thought prompting has a lot of good points, but it also has some big problems. First, making high-quality reasoning demonstrations for supervised fine-tuning is still a lot of work that needs careful planning and knowledge. Second, big language models can sometimes make up reasoning chains that sound reasonable but are actually wrong or don't make sense. Finally, the reasoning process isn't always reliable unless symbolic reasoning frameworks are used to check and make sure that the logic is correct.

9. Tree of Thought

The Tree of Thoughts (ToT) is a way to get large language models (LLMs) to think more clearly and solve problems better. ToT is different from linear prompting methods like Chain of Thought (CoT), which only give the model one set of reasoning steps. Instead, ToT organizes intermediate steps into a tree structure, which lets the model think about more than one way to reason (Yao et al., 2023). Each node in this tree stands for a partial solution or thought, and the branches show how the reasoning could go on, making the process of thinking more flexible and like a human.

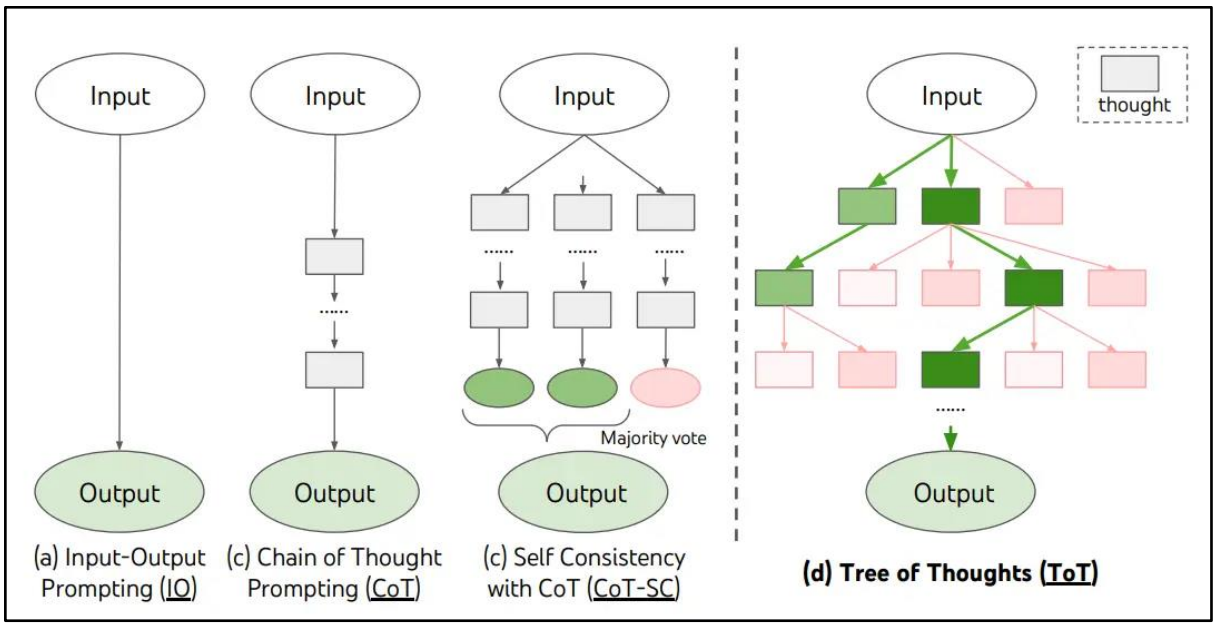

Figure 15: Illustrates the Tree of Thoughts (ToT) framework, showing how multiple reasoning paths are explored as a tree structure. Each node represents a partial solution, and branching enables flexible, human-like problem solving by evaluating various thought sequences before arriving at a final answer.

How the Technique Works

There are a number of modules and steps in the ToT framework. To begin, the problem is broken down into a series of intermediate reasoning steps, or "thoughts." For each current state (or partial solution), the model creates several possible thoughts using either

sampling methods or constrained generation methods. Then, a state evaluator gives each partial solution a heuristic score using either scalar value estimation or comparative voting. Using these evaluations, a search algorithm like breadth-first search (BFS) or depth-first search (DFS) is used to look into the most promising reasoning paths.

More parts make ToT work better. A prompter agent makes prompts that depend on the situation, and a checker module checks intermediate outputs against task-specific rules. A memory module keeps track of past reasoning paths so that user can go back and forth, and the controller is in charge of the whole thing, deciding when to explore or give up on a reasoning path (Yao et al., 2023; Long, 2023).

Why the Technique is Effective

ToT fixes a number of major problems with traditional LLM prompting methods. First, it eases the linearity constraint by adding a branching structure that lets user look into other ways to solve the problem. Second, adding a state evaluator and checker module makes it possible to validate and fix things in the middle, which stops errors from spreading. Third, the ability to go back and forth in time is like how people solve problems and makes the model more flexible when it has to deal with tasks that are unclear or hard (Yao et al., 2023).

Unlike traditional methods, which often use quick, intuitive reasoning (like Kahneman's System 1), ToT adds a planned and systematic planning layer, like System 2 thinking. This two-step method lets LLMs look at a number of possible futures, weigh the pros and cons of each one, and change their plans on the fly (Kahneman, 2011; Yao et al., 2023).

Examples of Application

ToT has been proven to work in many different areas. ToT did much better than CoT in math reasoning tasks like the Game of 24, where it got 74% of the answers right with GPT-4 and only 4% of the answers right with CoT (Yao et al., 2023). Both human and automated evaluators said that the outputs from ToT were more coherent and appropriate for the context in creative writing tasks. In the same way, ToT made it easier to place words and letters correctly in Mini Crossword puzzles than with baseline methods.

ToT has also done better than other methods at solving logic-based games like Sudoku and hard benchmark tasks in the Big-Bench Hard (BBH) suite. These results show that ToT is better at handling long-range dependencies and abstract reasoning than traditional prompting strategies (Long, 2023).

Why This Technique Helps Produce Better Responses

ToT works well because it is structured and modular. ToT lowers the risk of early mistakes ruining the final product by making the model think of many different things, giving them scores, and letting it rethink or change its approach. This process of going through it over and over again helps user think more deeply and broadly. Also, ToT can work with a lot of different search strategies and evaluation methods, so it can be used for a lot of different things, from symbolic reasoning to open-ended creative generation (Yao et al., 2023; Bubeck et al., 2023).

CoT, few-shot, and zero-shot prompting modes cannot verify and correct themselves, which means that their results may become poorer in case of the initial assumptions. Checker and memory modules within ToT, on the other hand, hamper this process by storing and testing partial solutions. This dynamic, recursive structure enables a kind of algorithmic thinking that closely resembles the kind done by people as they solve problems.

Ethical Implications and Practical Challenges in Advanced Prompting Techniques

Prompt engineering is an emerging method of enhancing the performance of large language models (LLMs), however, it also creates numerous ethical concerns. More advanced prompting strategies such as Chain-of-Thought (CoT), persona prompting, and zero-shot/few-shot learning can ensure that models perform better, reason better, and become more aligned with the wishes of the users. Nevertheless, they are also liable to providing biased, misleading, or unclear results. In order to ensure that AI is utilized in a safe and responsible way, it is worth addressing these issues particularly in the sensitive sectors such as healthcare, education, legal services, and finance.

1. Stereotyping and Bias Amplification

Prompt engineering can unwittingly make more salient bias in training data stronger. Such methods as persona prompting forcing models to pretend to be an engineer, teacher or therapist may cause individuals to conceive stereotypes revolving around their gender, race or age (Furukawa, 2024; Kim et al., 2024). In other words, as an example, when being asked, users are a nurse, models can inappropriately associate their role with female qualities and engineer with male qualities. This is due to biases within datasets which training LLMs. That raises deep moral concerns regarding equity, inclusivity and representation in products generated by AI. Unless these types of biases are crated, they might perpetuate systemic inequity and be less confident in using AI systems (Olea et al., 2024).

2. Hallucinations and Wrong Information

Although LLMs are enhanced with prompting, they nevertheless hallucinate, that is, they may produce grammatically correct text that is incorrect or outright fictional (Bubeck et al., 2023). This can be very risky in areas of medicine, law or finance because individuals may make serious decisions using the wrong information. Prompt engineering can contribute

to the decreasing of the taken place of hallucinations because of making the tasks more precise and putting these aspects in the proper context, but it is unable to fully eliminate this issue. The issue of ethics is how to correctly balance usability and reliability. What is the point of having a customer believe in results of a very fluent but not well-informed model of what is true (Ceurstemont, 2025)?

3. Risks of Manipulation and False Information

The accuracy that prompt engineering makes possible raises the risk that models will be used to make content that is misleading or manipulative. For instance, prompts can be made to make false claims about products, spread false political information, or create emotionally persuasive stories that lead people astray (Saini & Sharma, 2024). AI-generated content can be made in large quantities with little effort, which raises even more ethical concerns than traditional disinformation campaigns. This makes us think about what developers and users should do to stop LLMs from being used in ways that hurt society.

4. Persona Prompting and Lying

Persona prompting, in which LLMs act out roles like "doctor," "lawyer," or "financial advisor," makes the tone and context more consistent, but it also raises the risk of misrepresentation (Olea et al., 2024). People who use a model might mistakenly think that outputs made under professional personas are the same as expert advice. This is especially bad when the AI gives medical, legal, or psychological advice without the right training, which could lead to bad choices or bad results. Because of this, using AI in an ethical way means making it clear that the outputs are made by AI and should not be used instead of professional advice (Patil & Puranik, 2024).

5. Being Responsible and Accountable

One of the main ethical problems is figuring out who is responsible for AI outputs that were made with engineered prompts. Because many LLMs are black-box systems that can only be accessed through APIs, it can be hard to figure out who is responsible for bad outputs (Patil & Puranik, 2024). Still, users, developers, and organizations that work on prompt design are partly responsible for making sure that AI-generated outputs are correct, fair, and safe. This problem gets worse when AI is used in automated decision-making systems without any human oversight, which raises questions about who is responsible if something goes wrong.

6. Not Being Clear and Not Being Open

As is the nature of the problems which continue to arise when dealing with prompt engineering, LLM behaviour is difficult to observe. Users are unable to observe the impact of trivial alterations of wording in prompts on how the reasoning behind the model changes (Ceurstemont, 2025) and they can significantly change the resulting outputs. When outputs of AI are difficult to understand, it becomes more difficult to audit, verify, or make more accountable and trustworthy. But also, the fact that there are no industry guidelines on how to make prompts compounds this issue, and it is difficult to implicate ethical norms or compliance standards to various AI use cases (Kim et al., 2024).

Altogether, timely engineering transforms LLMs into a much more helpful and practical technique, yet it also raises significant ethical and practical concerns, which can not be overlooked. Such issues as bias amplification, hallucination, misinformation, Socio-technical misrepresentation, and the intransparency of model behaviour will require close attention of AI developers, researchers, and policymakers. Ethical standards, openness rules, and means to penalize individuals in promoting timely engineering will be needed as the use

of LLMs in sensitive and high-stakes domains grows. This will assist in the promotion of responsible uses of AI.

Recommendations

Advanced prompting strategies change quickly, so using them responsibly and effectively requires careful planning, being aware of the context, and making small changes over time. Based on the research that has been done and the new ideas that have come out of this study, here are some evidence-based suggestions for researchers and practitioners:

1. Begin with aligning tasks with goals.

To get the best results, user need to match the types of tasks with the prompting strategies. Chain-of-Thought prompting is best for logical reasoning (Wei et al., 2022), while zero-shot or instruction-based prompting is best for classification or generation tasks that don't need much setup (Zhao et al., 2023; Yin et al., 2023).

2. Use modular testing to make changes to prompts.

Researchers should use modular testing to test prompts multiple times because they are sensitive to prompts (Zhao et al., 2021). Using tools like OpenAI Playground or LangChain to do structured experiments (as explained in the methodology and use cases) makes the output more reliable by comparing different versions in real life (Chiodi et al., 2023).

3. Use Retrieval-Augmented Generation (RAG) to make sure the facts are correct.

In fields like healthcare or law where the truth of the output is very important, using Retrieval-Augmented Generation (RAG) makes the facts more solid and cuts down on hallucinations (Lewis et al., 2020; Zhao et al., 2023). In situations where user need to

remember something right away, this method has already been shown to work better than regular LLM prompting.

4. Use Parameter-Efficient Fine-Tuning (PEFT) to make changes that don't use up too many resources.

LoRA (Hu et al., 2021) and Adapter Layers (Pfeiffer et al., 2020) are two examples of techniques that allow for domain-specific tuning without sacrificing efficiency. These can be used with prompting techniques to customize models for specific or changing use cases without having to retrain the whole model (Chung et al., 2022).

5. Add ethical protections to prompt engineering.

The danger of persona manipulation and hallucinations along with bias ramping should be considered (Bansal, 2024; Ceurstemont, 2025). Calibration and prompt designs, such as bias-aware prompt design (Ganesan et al., 2024) or prompt calibration (Chiodi et al., 2023), and persona validation frameworks (Kim et al., 2024) are very important in sensitive situations.

6. Keep records of prompts and make sure they can be repeated.

By monitoring various versions of prompts and maintaining properly organized stores of prompts, it is possible to ensure that deployments remain lucid and can be replicated (Zhao et al., 2021; Ceurstemont, 2025). This comes in particular handy in such an environment as a school, the government, or collaboration.

7. Help non-technical users learn to read quickly.

People need to apply AI prompts to various disciplines, so researchers should create prompt templates, books, and software to show different individuals how to utilize AI prompts (Patil

& Puranik, 2024; Ekin, 2024). This facilitates the use of LLMs in education, marketing, journalism, and customer care where it has become commonly available.

Conclusion

This research sought to identify the influence of the sophistication of prompting skills on Large Language Models (LLMs) performance, reliability, and ethical application. The paper categorized immediate engineering approaches in four key categories by observing the peer-reviewed publications, technical reports, as well as industry research systematically. The former category features such techniques as persona prompting and instructional cues that are aimed at clarity and roles. The methods allow clarifying intentions and better aligning answers with the context (Furukawa, 2024). These are techniques that contribute to the setting of the tone of the model, and ensuring that it appears to know a lot about the matter. This is particularly useful when writing such things as essays, aiding customers, and generating materials. The second category involves the example-based prompting, which can encompass zero-shot, one-shot, and few-shot learning (Brown et al., 2020; Wei et al., 2022). Such tools allow LLMs to learn contextually by providing them with only a few examples to carry out new tasks. These ways are essential to use to deploy LLMs in permanently changing places or low-resource ones without retraining them.

Among the stepwise reasoning techniques are Chain-of-Thought (CoT), Prompt Chaining, and Tree-of-Thought (ToT) that contribute to logical coherence and systematic problem-solving (Wei et al., 2022; Zhou et al., 2022). Such strategies come in handy particularly when dealing with challenging problems, which involve several stages of mathematical or logical thinking. The fourth category, knowledge-augmented methods such as Retrieval-Augmented Generation (RAG) deepens factual consistency by allowing LLMs to obtain information in real-time using external data sources (Lewis et al., 2020). These various strategies of prompting exhibit that prompt engineering is not simply a question of

grammatical formulations; it is also an exercise in designing systematic, tactical interactions that align LLM capacity with the demands of tasks. In the study, it was also necessary to bring out the prompt engineering needed to optimize the benefits of LLM, minimizing the risks of hallucinations, confusion, and bias (Bubeck et al., 2023; Kim et al., 2023). With an improvement of the LLM technologies, however, the need to know how to use advanced prompting techniques will remain essential to researchers, developers, and practitioners interested in using generative AI responsibly and effectively.

References

Bansal, A. (2024). *Prompt engineering for generative AI: A practical guide to unlocking AI's full potential*. AI Publications.

Bre, F., Gimenez, J., & Fachinotti, V. (2017). Prediction of wind pressure coefficients on building surfaces using artificial neural networks. *Energy and Buildings, 158*, 1429–1441. https://doi.org/10.1016/j.enbuild.2017.11.045

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems, 33*, 1877–1901. https://doi.org/10.48550/arXiv.2005.14165

Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., ... & Zhang, Y. (2023). Sparks of artificial general intelligence: Early experiments with GPT-4. *arXiv*. https://doi.org/10.48550/arXiv.2303.12712

Ceurstemont, S. (2025). How to talk to AI: The rise of prompt engineering. *Nature, 621*(7981), 12–14.

Chase, R., Chan, M. C., Zhou, K., Singh, J., & Liang, P. (2023). PromptChainer: Chaining large language model prompts through visual programming. *arXiv*. https://doi.org/10.48550/arXiv.2305.14218

Chen, P., Li, J., Zhang, Y., & Yu, A. W. (2023). BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. https://arxiv.org/abs/2301.12597

Chiodi, M., Gray, M., Chen, M., Wolf, T., & Pyo, S. (2023). Mastering prompt engineering: A guide to effective AI interaction. *ResearchGate*. https://www.researchgate.net/publication/383920503

Chowdhery, A., Narang, S., Devlin, J., et al. (2022). PaLM: Scaling language modeling with pathways. *arXiv*. https://doi.org/10.48550/arXiv.2204.02311

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., ... & Le, Q. (2022). Scaling instruction-finetuned language models. *arXiv*. https://arxiv.org/pdf/2210.11416.pdf

Eight things to know about large language models. (n.d.). *Critical AI, Duke University Press*. https://dukeupress.edu/critical-ai

Ekin, H. (2024). Communicating with AI: The role of prompt engineering in human-AI interaction. *AI & Society*, *39*(1), 77–89.

Furukawa, H. (2024). How does the persona given to large language models affect the idea evaluations? *IIAI Letters on Informatics and Interdisciplinary Research*, *6*, 34–42. https://doi.org/10.52731/iiir.v006.342

Ganesan, D., Grubb, A., & Ravishankar, K. (2024). Advanced prompting techniques and prompt engineering for enterprises: A comprehensive guide. *ResearchGate*. https://www.researchgate.net/publication/383453095

Goli, A., & Singh, A. (2024). Frontiers: Can large language models capture human preferences? *Marketing Science*, *43*(4), 709–722. https://doi.org/10.1287/mksc.2023.0306

Goli, A., & Singh, S. (2024). *Machine learning: Fundamentals and applications*. Springer.

Gu, Y., Tinn, R., Cheng, H., Lucas, M., Usuyama, N., Liu, X., ... & Poon, H. (2021). Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare, 3*(1), 1–23. https://arxiv.org/pdf/2007.15779.pdf

Hadi, M. U., Qureshi, R., Shah, A., Irfan, M., Zafar, A., Shaikh, M. B., ... & Mirjalili, S. (2023). Large language models: A comprehensive survey of its applications, challenges, limitations, and future prospects. *Authorea Preprints*, 1–26. https://doi.org/10.48550/arXiv.2305.13172

Haque, A. U., Rust, P., Röder, M., & Hauff, C. (2023). Navigating prompt complexity for zero-shot classification: A study of large language models in computational social science. *ResearchGate*. https://www.researchgate.net/publication/370981439

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, L., ... & Raj, A. (2021). LoRA: Low-rank adaptation of large language models. *arXiv*. https://arxiv.org/pdf/2106.09685.pdf

Huang, L. (2024, April 22). Large Language Model — History. *Medium*. https://medium.com/@linghuang_76674/llm-history-5db2c9e236f5

Kahneman, D. (2011). *Thinking, fast and slow.* Farrar, Straus and Giroux.

Kim, J., Yang, N., & Jung, K. (2024). Persona is a double-edged sword: Mitigating the negative impact of role-playing prompts in zero-shot reasoning tasks. *Proceedings of the International Conference on Machine Learning (ICML)*, 34–42.

Kim, S., Le, D., Lim, Y., & Kim, G. (2023). Better zero-shot reasoning with self-adaptive prompting. *arXiv*. https://arxiv.org/pdf/2305.14106.pdf

Kingma, D. P., & Welling, M. (2016). Auto-encoding variational Bayes. *arXiv*. https://arxiv.org/pdf/1605.05396

Klie, T., Zhang, C., & Gurevych, I. (2023). Crafting effective prompts: Enhancing AI performance through structured input design. *ResearchGate*. https://www.researchgate.net/publication/385591891

Kojima, T., Gu, S. S., Reid, M., et al. (2022). Large language models are zero-shot reasoners. *arXiv*. https://doi.org/10.48550/arXiv.2205.11916

Larson, A. M. (2025). Large language model (LLM). In *Salem Press Encyclopedia of Science*. Salem Press.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Riedel, S. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *arXiv*. https://arxiv.org/pdf/2005.11401.pdf

Li, J., Hu, R., Zeng, Y., Hu, X., Zhang, L., Liu, P., ... & Shi, J. (2023). BLIP-2: Bootstrapped language-image pretraining with frozen image encoders and large language models. *arXiv*. https://arxiv.org/pdf/2301.12597.pdf

Li, J., Shen, Y., Chen, Z., Ke, L., & Yu, A. W. (2023). Evaluating the use of chain-of-thought in VQA with BLIP-2. *arXiv preprint*. https://arxiv.org/abs/2310.09297

Liu, W., Gupta, S., Shi, Y., Kang, Y., Kumar, A., & Song, L. (2024). Generative AI: A survey of its development trends and future outlook. *ResearchGate*. https://www.researchgate.net/publication/380032572

Long, J. (2023). Improving the reasoning ability of large language models via a tree of thoughts. *Microsoft Research Blog*. https://www.microsoft.com/en-us/research/blog/improving-the-reasoning-ability-of-large-language-models-via-a-tree-of-thoughts/

Mahesh, B. (2020). Machine learning algorithms: A review. *International Journal of Science and Research (IJSR), 9*(1), 381–386. https://doi.org/10.21275/ART20203995

National Geographic Society. (n.d.). Alan Turing [Photograph]. *National Geographic*. https://www.nationalgeographic.com/science/article/alan-turing-test-artificial-intelligence-life-history

National Institute of Standards and Technology. (n.d.). Alan Turing and the beginning of AI. *U.S. Department of Commerce*. https://www.nist.gov/news-events/news/2021/06/alan-turing-and-beginning-ai

Naveed, H., Khan, A. U., Qiu, S., Saqib, M., Anwar, S., Usman, M., Akhtar, N., Barnes, N., & Mian, A. (2024). A comprehensive overview of large language models. *arXiv preprint*. https://doi.org/10.48550/arXiv.2307.06435

New Scientist. (n.d.). What is the Turing test? https://www.newscientist.com/definition/turing-test/

Olea, C., Tucker, H., Phelan, J., Pattison, C., Zhang, S., Lieb, M., Schmidt, D., & White, J. (2024). Evaluating persona prompting for question answering tasks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34–42.

OpenAI. (2023). GPT-4 technical report. *arXiv*. https://doi.org/10.48550/arXiv.2303.08774

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., ... & Christiano, P. (2022). Training language models to follow instructions with human feedback. *Advances*

in Neural Information Processing Systems, 35, 27730–27744. https://doi.org/10.48550/arXiv.2203.02155

Patil, A., & Puranik, S. (2024). Prompt engineering in the age of generative AI: Enhancing productivity and trust in LLMs. *Journal of Artificial Intelligence Research and Applications, 58*(2), 105–118.

Patil, R., & Gudivada, V. (2024). A review of current trends, techniques, and challenges in large language models (LLMs). *Applied Sciences, 14*(5), 2074. https://doi.org/10.3390/app14052074

Pfeiffer, J., Kamath, A., Rücklé, A., Cho, K., & Gurevych, I. (2020). AdapterFusion: Non-destructive task composition for transfer learning. *arXiv preprint*. https://arxiv.org/pdf/2005.00247.pdf

Press, O., Bar, A., Smith, N. A., & Levy, O. (2022). Measuring and narrowing the compositionality gap in language models. *arXiv*. https://arxiv.org/abs/2210.03350

PromptHub. (2024, April 30). Chain-of-thought prompting guide: Techniques for improving reasoning in LLMs. https://www.prompthub.us/blog/chain-of-thought-prompting-guide

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI*.

Reynolds, L., & McDonell, K. (2021). Prompt programming for large language models: Beyond the few-shot paradigm. *arXiv*. https://doi.org/10.48550/arXiv.2102.07350

Saghiri, A. M. (2024). Why GPT-based chatbots will be vital applications, challenges, and the shaping of the fragile job market [Flowchart]. *ResearchGate*. https://www.researchgate.net/publication/378477275

Sahoo, S., Kumar, A., & Singh, P. (2024). A systematic survey of prompt engineering in large language models: Techniques and applications. *arXiv*. https://arxiv.org/abs/2402.07927

Saini, M., & Sharma, R. (2024). Applications and future of generative AI in various domains. *International Journal of Trend in Scientific Research and Development (IJTSRD), 8*(2). https://www.ijtsrd.com/papers/ijtsrd72647.pdf

Shanahan, M. (2024). Talking about large language models. *Communications of the ACM, 67*(2), 68–79. https://doi.org/10.1145/3642084

Smith, A. (2017). Artificial intelligence digital neural network [Photograph]. *Unsplash*. https://images.unsplash.com/photo-1504384308090-c894fdcc538d

Sparks, R., Koharchik, L., & Meyer, H. (2023). The CLEAR path: A framework for enhancing information literacy through prompt engineering. *The Journal of Academic Librarianship, 49*(6), 102689. https://doi.org/10.1016/j.acalib.2023.102689

Tiwari, S., & Patel, A. (2024). A review on generative AI: Challenges, opportunities, and applications. *arXiv*. https://arxiv.org/pdf/2403.04190

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all need. *arXiv*. https://arxiv.org/pdf/1706.03762.pdf

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., & Zhou, D. (2022). Self-consistency improves chain of thought reasoning in language models. *arXiv*. https://arxiv.org/abs/2203.11171

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., ... & Le, Q. V. (2022). Chain of thought prompting elicits reasoning in large language models. *arXiv*. https://doi.org/10.48550/arXiv.2201.11903

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., ... & Le, Q. V. (2023). Chain-of-thought prompting elicits reasoning in large language models. *Nature, 615*(7950), 660–665. https://doi.org/10.1038/s41586-023-05886-4

What are large language models. (n.d.). *MachineLearningMastery.com*. https://machinelearningmastery.com/what-are-large-language-models/

White, T., Zhang, S., Lin, Z., & Dai, Z. (2023). Prompting GPT-3 to be reliable. *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. https://doi.org/10.48550/arXiv.2301.11270

Wu, W., Zhou, K., Park, S., & Liang, P. (2022). AI Chains: Transparent and controllable human-AI interaction by chaining large language model prompts. *arXiv*. https://doi.org/10.48550/arXiv.2209.11302

Yao, S., Ye, D., Liang, P., & Etzioni, O. (2023). Tree of thoughts: Deliberate problem solving with large language models. *arXiv*. https://arxiv.org/abs/2305.10601

Yao, S., Zhao, J., Yu, D., Anil, R., Yu, Y., Park, J. S., & Cao, Y. (2022). ReAct: Synergizing reasoning and acting in language models. *arXiv*. https://arxiv.org/abs/2210.03629

Yin, W., Choi, E., & Neubig, G. (2023). A practical survey on zero-shot prompt design for in-context learning. *Proceedings of the International Conference RANLP 2023*. https://aclanthology.org/2023.ranlp-1.69.pdf

Zhang, R., Liu, L., Hu, Y., & He, X. (2022). Efficient fine-tuning of pretrained language models via low-rank adaptation. *Findings of the Association for Computational Linguistics: EMNLP 2022*. https://arxiv.org/pdf/2207.01093.pdf

Zhang, Y., Sun, S., Galley, M., Chen, Y. C., Brockett, C., Gao, X., & Dolan, B. (2022). Few-shot learning with retrieval augmented language models. *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 1126–1140. https://doi.org/10.18653/v1/2022.acl-long.80

Zhao, C., Haris, N., & Balog, M. (2023). Investigating prompting techniques for zero- and few-shot visual question answering. *ResearchGate*. https://www.researchgate.net/publication/371684481

Zhao, W., Lin, Z., Song, X., Ren, X., Tan, M., & Qi, P. (2023). InstructGPT meets zero-shot learning: An empirical study. *arXiv*. https://doi.org/10.48550/arXiv.2305.03079

Zhao, W., Wallace, E., Feng, S., Klein, D., & Singh, S. (2021). Calibrate before use: Improving few-shot performance of language models. *International Conference on Machine Learning*, 12697–12706. https://doi.org/10.48550/arXiv.2102.09690

Zhao, Z., Yin, W., Li, S., Li, Y., Li, X., & Ma, J. (2023). R2AG: Retrieval-aware augmented generation for knowledge-intensive NLP. *arXiv preprint*. https://arxiv.org/pdf/2303.12712.pdf

Zhou, H., Liu, J., & Guo, C. (2023). Zero-shot prompting strategies for table question answering with a low-resource language. *Emerging Science Journal*, *7*(5), 1222–1238. https://www.ijournalse.org/index.php/ESJ/article/view/2540/pdf

Zhou, K., Yang, J., Loy, C. C., & Liu, Z. (2022). Learning to prompt for vision-language models. *arXiv*. https://arxiv.org/abs/2109.01134

Zhou, S., Yang, H., Wang, Y., et al. (2023). Least-to-most prompting enables complex reasoning in large language models. *arXiv*. https://doi.org/10.48550/arXiv.2205.10625