


THE RISE OF LARGE LANGUAGE MODELS: A BEGINNER'S SURVEY

A ASCENSÃO DOS GRANDES MODELOS DE LINGUAGEM: UM PANORAMA PARA INICIANTES

EL AUGE DE LOS GRANDES MODELOS DE LENGUAJE: UNA ENCUESTA PARA PRINCIPIANTES

 <https://doi.org/10.56238/arev7n11-118>

Submitted on: 10/13/2025

Publication date: 11/13/2025

Gustavo de Aquino Mouzinhos¹, Leandro Youiti Silva Okimoto², Leonardo Yuto Suzuki Camelo³, Nádila da Silva de Azevedo⁴, Hendrio Luis de Souza Bragança⁵, Rubens de Andrade Fernandes⁶, Fabricio Ribeiro Seppe⁷, Raimundo Cláudio Souza Gomes⁸, Fábio de Sousa Cardoso⁹

ABSTRACT

Large Language Models (LLMs) have rapidly reshaped Natural Language Processing by shifting the field from task-specific systems to general-purpose models capable of following instructions and handling diverse tasks with minimal adaptation. This beginner-oriented survey traces the rise of LLMs, outlining the historical path from statistical methods and early neural architectures to Transformer-based models and the emergence of in-context learning. We introduce the core ingredients that make LLMs work—pre-training on large corpora, optional fine-tuning and alignment, and decoding strategies for generation—emphasizing how scale and self-attention enable broad generalization. At its core, this paper offers a compact, newcomer-friendly narrative of the LLM era, distilling the minimum set of concepts and milestones needed to build intuition about how modern models are trained and used.

Keywords: Large Language Models. Generative AI. Natural Language Processing. Deep Learning.

RESUMO

Os Grandes Modelos de Linguagem (LLMs) transformaram rapidamente o Processamento de Linguagem Natural, ao deslocar o campo de sistemas específicos para tarefas em direção a modelos de uso geral, capazes de seguir instruções e lidar com diversas tarefas com mínima adaptação. Esta pesquisa voltada para iniciantes traça a ascensão dos LLMs, delineando o percurso histórico desde os métodos estatísticos e as primeiras arquiteturas

¹ Dr. in Electrical Engineering. Universidade Federal do Amazonas. E-mail: gustavoaquino@gmail.com

² Dr. in Computer Science. Universidade Federal do Amazonas. E-mail: leandrookimoto@gmail.com

³ Graduated in Electrical Engineering. Universidade do Estado do Amazonas.

E-mail: leonardo.yuto@icomp.ufam.edu.br

⁴ Graduated in Computer Engineering. Universidade do Estado do Amazonas.

E-mail: nadila.azevedo@icomp.ufam.edu.br

⁵ Dr. in Computer Science. Universidade Federal do Amazonas. E-mail: hendrio.luis@icomp.ufam.edu.br

⁶ Dr. of Electrical Engineering. Universidade do Estado do Amazonas. E-mail: rfernandes@uea.edu.br

⁷ Specialist in Smart Industry Management. Universidade do Estado do Amazonas.

E-mail: fab.seppe@gmail.com

⁸ Dr. of Electrical Engineering. Universidade do Estado do Amazonas. E-mail: rsgomes@uea.edu.br

⁹ Dr. of Mechanical Engineering. Universidade do Estado do Amazonas. E-mail: fcardoso@uea.edu.br

neurais até os modelos baseados em Transformer e o surgimento do aprendizado em contexto (in-context learning). Apresentamos os principais elementos que fazem os LLMs funcionarem — pré-treinamento em grandes corpora, fine-tuning e alinhamento opcionais, e estratégias de decodificação para geração — enfatizando como a escala e o mecanismo de autoatenção permitem ampla generalização. Em essência, este artigo oferece uma narrativa concisa e acessível sobre a era dos LLMs, condensando o conjunto mínimo de conceitos e marcos necessários para construir uma compreensão intuitiva sobre como os modelos modernos são treinados e utilizados.

Palavras-chave: Grandes Modelos de Linguagem. IA Generativa. Processamento de Linguagem Natural. Aprendizado Profundo.

RESUMEN

Los Grandes Modelos de Lenguaje (LLMs) han transformado rápidamente el Procesamiento del Lenguaje Natural, pasando de sistemas específicos para tareas a modelos de propósito general capaces de seguir instrucciones y manejar diversas tareas con una adaptación mínima. Esta revisión orientada a principiantes traza el auge de los LLMs, delineando el camino histórico desde los métodos estadísticos y las primeras arquitecturas neuronales hasta los modelos basados en Transformer y el surgimiento del aprendizaje en contexto (in-context learning). Presentamos los componentes fundamentales que hacen que los LLMs funcionen — el preentrenamiento en grandes corpus, el fine-tuning y la alineación opcionales, y las estrategias de decodificación para la generación — destacando cómo la escala y la autoatención permiten una amplia generalización. En esencia, este artículo ofrece una narrativa breve y accesible sobre la era de los LLMs, destilando el conjunto mínimo de conceptos y hitos necesarios para comprender cómo se entrenan y utilizan los modelos modernos.

Palabras clave: Grandes Modelos de Lenguaje. IA Generativa. Procesamiento del Lenguaje Natural. Aprendizaje Profundo.

1 INTRODUCTION

Large Language Models (LLMs), such as GPT-4, have reshaped Natural Language Processing (NLP) by enabling general-purpose systems that converse, follow instructions, and perform a wide range of tasks with minimal or no task-specific training (1,2). Before this shift, NLP advances largely relied on pipelines or specialized models tailored to narrow objectives—machine translation, sentiment analysis, or named-entity recognition—each demanding labeled data and separate training (3). The emergence of large, Transformer-based models—popularized by GPT-3 and its in-context learning capability—showed that a single pre-trained model can adapt to diverse tasks simply by conditioning on input examples and instructions (3,4). This paradigm change accelerated both research and productization across many domains.

Despite their versatility, LLMs face practical constraints. Finite context windows can hinder multi-step reasoning over long or interdependent materials (5). Fixed knowledge cutoffs also limit awareness of post-training information, contributing to outdated or incomplete answers in dynamic domains (5,7). Retrieval-Augmented Generation (RAG) addresses these issues by coupling LLMs with information retrieval so that responses are grounded in relevant, up-to-date sources (6,7).

Traditional RAG pipelines typically perform vector-similarity search over document corpora and assemble retrieved passages into the model's context for generation (7,8). However, these approaches may falter when handling complex queries that require understanding relationships between disparate pieces of information (9). Recent research has integrated knowledge graphs into the RAG framework, resulting in Graph-Based RAG methods (9,10,11,12).

For newcomers, understanding how we arrived at this moment is essential. The trajectory from task-specific systems to large, instruction-following models reflects progress in representations learning, scalable architectures, and data-driven pre-training. This survey offers a guided entry point: it (i) traces the historical path culminating in Transformer-based LLMs; (ii) explains foundational

ideas behind pre-training, fine-tuning, and alignment to situate model capabilities and limits (1–3); and (iii) introduces retrieval-augmented extensions that mitigate knowledge staleness and expand practical utility (6–8).

2 FUNDAMENTALS OF LARGE LANGUAGE MODELS

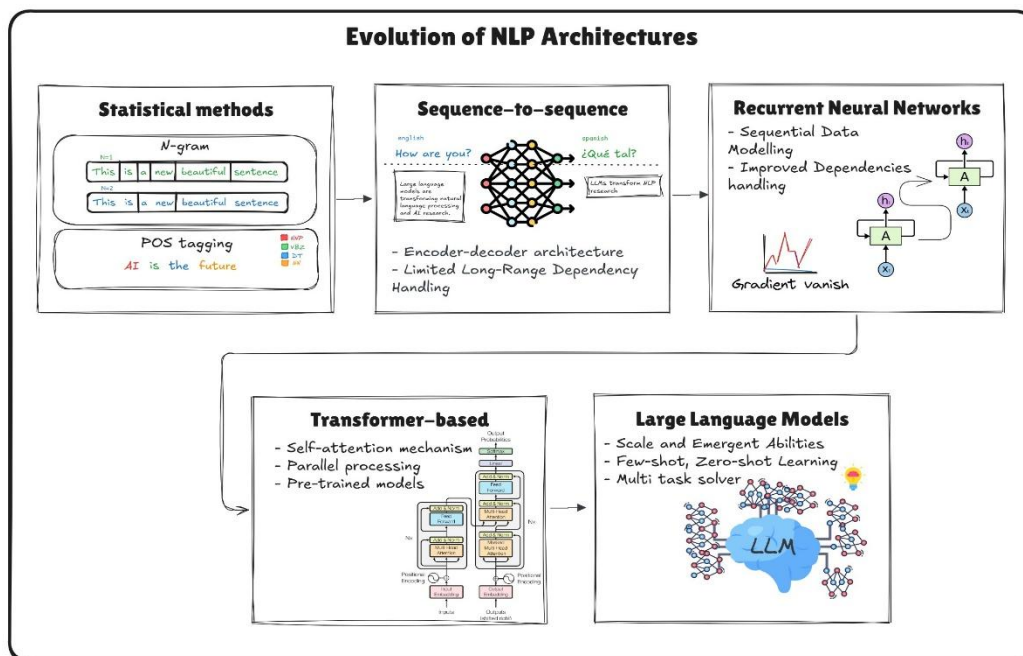
LLMs represent a transformative leap in natural language processing, building upon decades of advancements in statistical methods, neural architectures, and representation learning. Despite earlier models that relied on task-specific designs and limited contextual understanding, LLMs harness large-scale pre-training and self-attention mechanisms to capture nuanced language relationships. This evolution enables LLMs to generalize remarkably well, handling diverse tasks with minimal fine-tuning or additional training. By scaling model parameters and training on vast datasets, LLMs exhibit emergent properties such as few-shot learning and contextual reasoning, making them indispensable across various domains. Although the paper focuses on LLM architecture, this section reviews historical advancements, architectural improvements, and application impacts that underpin LLM functionalities.

2.1 HISTORICAL CONTEXT AND EVOLUTION OF LANGUAGE MODELS

Language modeling has evolved from simple count-based methods to sophisticated neural architectures that drive today's LLMs. This progression highlights the motivations, breakthroughs, and design choices that paved the way for modern language models. Figure 1 offers a visual overview of NLP system evolution, highlighting key developments from N-gram models to LLMs. This illustration demonstrates how each generation builds upon its predecessor by incorporating more advanced techniques for enhanced language understanding and versatility.

Figure 1

An overview of NLP's evolution, from early statistical methods and seq2seq architectures to RNNs, culminating in Transformer-based LLMs



Source: Created by the author

The evolution of NLP began with statistical methods, such as N-gram models and part-of-speech tagging, which relied on predefined rules and statistical correlations. Later, sequence-to-sequence (seq2seq) (14) models introduced encoder-decoder architectures for tasks like translation. However, these models struggled to capture long-range dependencies. RNNs improved dependency modeling but faced issues such as gradient vanishing. The introduction of Transformer-based architectures was a significant advancement, as self-attention, parallel processing, and pre-training addressed these challenges. Building on this foundation, LLMs have pushed boundaries by using scale to reveal emergent capabilities, including few-shot and zero-shot learning, while efficiently addressing complex multitask problems.

2.1.1 Early approaches

The first generation of language models, exemplified by N-gram models (4), utilized statistical methods to estimate word sequence probabilities. An N-gram model predicts the next word based on the preceding $N-1$ words, capturing local dependencies. This approach uses the Markov assumption, positing that a word's probability depends only on a

limited history. These models were task-specific and limited in scope, relying heavily on structured data and extensive feature engineering for tasks such as spelling correction (15), machine translation (16), and part-of-speech tagging (4,16). For example, a trigram model estimates a word's likelihood by considering the two preceding words.

Although straightforward and computationally efficient, N-gram approaches struggled to capture long-range dependencies because predictions were based on a fixed-size context window (4). Moreover, they suffered from data sparsity (4), as many plausible word sequences appear rarely or not at all in training corpora, complicating accurate probability estimation. This limitation paved the way for neural network-based approaches, which learn continuous representations of words and larger context windows.

These early approaches were designed for well-defined tasks such as sentiment analysis, part-of-speech tagging, named entity recognition, and language translation (17-20).

2.1.2 Neural networks pre-transformers

The second generation of language models revolutionized NLP by enabling the learning of hierarchical language representations. Recurrent Neural Networks (RNNs) (3) and later Long Short-Term Memory (LSTM) networks improved the handling of sequential data by preserving contextual information over longer sequences (3).

1. LSTM uses a gating mechanism to control information flow, retaining relevant details over longer spans and better modeling long-range dependencies.
2. Gated Recurrent Unit (GRU) simplifies the LSTM architecture by using fewer gates, reducing computational overhead while retaining much of the capacity to capture temporal dependencies.

These models enabled the creation of static word representations with models such as Word2Vec and GloVe (21,22). They generate fixed vector representations for each word, capturing semantic relationships based on word co-occurrence. Although unable to dynamically adjust to context, these embeddings laid the groundwork for more sophisticated NLP tasks by providing reusable word representations.

RNN-based approaches, however, still struggled with vanishing gradients, limiting their ability to capture long-range dependencies (7), and required sequential processing, which limited parallelization during training and inference.

Convolutional neural networks (CNNs), more common in computer vision, have also been applied in text modeling. In this context, 1D convolutions slide across word embeddings

to capture local context. While CNNs allow easier parallelization than RNNs, they often require stacking multiple layers to capture very long-range dependencies without added complexity (23).

Building on LSTMs, the seq2seq paradigm emerged, pioneered by Sutskever et al. (2014). In seq2seq, an RNN encoder processes the input sequence into a fixed-dimensional representation, which an RNN decoder then converts into an output sequence (e.g., translating a sentence). However, a single vector was insufficient for long sequences, leading Bahdanau (2015) to introduce attention, which allows the decoder to focus on different parts of the input at each step. This bypassed the fixed-size hidden state bottleneck and improved performance on tasks like machine translation and summarization (13). The transition from seq2seq to LLMs involved adopting the Transformer architecture (25), which significantly improved training efficiency and scalability.

2.1.3 The transformer era

The third generation of language models marked a leap forward with the Transformer architecture introduced by Vaswani (2017), summarized by the phrase “Attention is all you need.” The key innovation of the Transformer is self-attention, which allows the model to focus on different parts of a sequence at various positions. This approach offers several advantages:

- **Parallelization:** Unlike RNNs that process tokens sequentially, self-attention computes all positions simultaneously, speeding up training and modeling long-range dependencies directly.
- **Long-Range Dependencies:** Self-attention connects distant tokens directly, overcoming the limitations of RNNs and providing dynamic, context-sensitive embeddings.
- **Modular Design:** Composed of repeated encoder and decoder blocks, the Transformer architecture is highly scalable and adaptable, inspiring variants like BERT, GPT, and T5.

The Transformer quickly became the de facto architecture for many NLP tasks, transforming language modeling, question answering, and machine translation (26,27). Early successes included ELMo, which used bidirectional LSTMs for context-sensitive word

representations (28), and Transformer-based models such as BERT (1,2), which introduced masked language modeling and next-sentence prediction. Although BERT is not typically labeled an LLM due to its encoder-only design and focus on classification rather than free-form text generation, it demonstrated that pre-trained models could be fine-tuned for multiple tasks.

Despite these advances, Transformer-based models have limitations. The computational and memory costs remain high, restricting accessibility. Fixed-length context windows prevent seamless processing of very long documents or dialogues. Additionally, these models require massive datasets for optimal performance, posing challenges for low-resource languages or specialized domains. Finally, while Transformers excel at pattern recognition, they can struggle with deeper reasoning, logical consistency, and factual grounding, sometimes producing errors or hallucinations (3). Motivated by these issues, the leap to fourth-generation LLMs has focused on scaling, improved training objectives, and enhanced reasoning.

Transformers quickly became the foundation for powerful language models that redefined state-of-the-art performance across numerous NLP tasks, ushering in the era of large language models, as discussed in the next section.

2.2 THE RISE OF LARGE LANGUAGE MODELS

Recent Large Language Models (LLMs), often referred to as fourth-generation models, build on the Transformer architecture to offer broad, flexible capabilities across numerous tasks. Instead of requiring extensive labeled datasets or parameter adjustments, these models perform tasks by conditioning on input examples and instructions within a single prompt (1). This makes prototyping new applications more efficient (2).

This paradigm mirrors human learning: a person can tackle a novel assignment by reviewing examples or instructions rather than undergoing extensive retraining. For instance, LLMs such as T5 (Text-To-Text Transfer Transformer) (29) consolidate various NLP tasks into a unified text-to-text format, while GPT-3 (1) introduced in-context learning, where tasks are performed by providing illustrative examples. Other notable models include LLaMA (Large Language Model Meta AI), which made smaller yet efficient models more accessible, and GPT-4, which refined coherence, reasoning, and contextual understanding (2,3,30,31).

In practice, LLMs are now used for tasks such as machine translation, text summarization, and question answering. For example, by including sentence pairs in

different languages, an LLM can translate new sentences (32). Similarly, providing text passages with their summaries helps guide the model to summarize new input (33), and including question–answer pairs trains the system to respond contextually to new queries (34).

Below are some examples of tasks managed through LLMs:

- Machine Translation: Providing sentence examples in one language and their translations in another enables the model to translate new sentences (32).
- Text Summarization: Including text passages and corresponding summaries guides the model in analyzing and summarizing new input (34)
- Question Answering: Presenting question-answer pairs allows the model to generate contextually relevant responses to new questions (34).

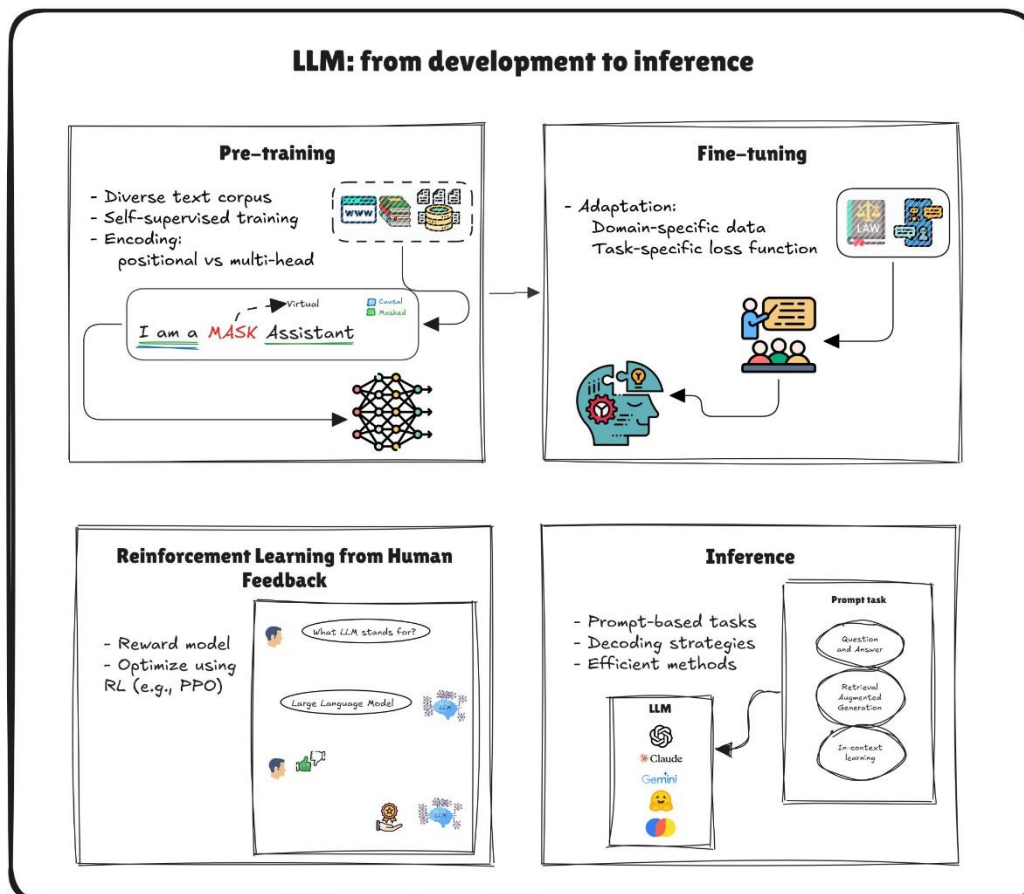
3 LLM: FROM TRAINING TO INFERENCE OVERVIEW

The development of LLMs involves a multiphase training process designed to enable them to understand and generate human-like text.

Figure 2 illustrates the pipeline for LLM development and deployment, comprising four key phases (1). The first phase, pre-training, exposes the model to a diverse text corpus using self-supervised objectives such as causal and masked language modeling. The second phase, fine-tuning, uses domain-specific data and task-specific loss functions to specialize the model. Next, Reinforcement Learning from Human Feedback (RLHF) integrates human evaluations to optimize the model's alignment with human preferences. Finally, in the inference phase, the trained model generates responses for prompt-based tasks using advanced decoding strategies and efficient computation.

Figure 2

Illustration of the four major phases in the LLM lifecycle: Pre-training



Source: Created by the author

3.1 LLM PRE TRAINING

Pre-training builds foundational linguistic understanding by exposing the model to massive amounts of text data from web crawls, encyclopedias, and other large corpora. This extensive knowledge base supports diverse downstream tasks. A robust pre-trained model can be adapted for specific tasks, saving substantial computational resources (29). Exposure to varied contexts enables the model to recognize patterns across multiple domains.

Two widely adopted objectives at this stage are causal language modeling and masked language modeling:

- Causal Language Modeling (CLM): In this autoregressive framework, the model predicts the next token based on all preceding tokens. Models such as GPT excel in text generation and completion tasks (35).
- Masked Language Modeling (MLM): By analyzing billions of words, the model internalizes linguistic and factual knowledge, capturing grammar, vocabulary, and syntax patterns. Randomly selected tokens are masked, and the model learns to predict them using both left and right context. BERT employs this bidirectional strategy to improve performance in tasks like question answering and sentence classification (36).

With billions of parameters, pre-trained models capture extensive statistical patterns, including grammar, world knowledge, and basic reasoning capabilities (29). Two fundamental techniques—positional encoding and multi-head self-attention—are critical to these capabilities (25). Since Transformer architectures do not inherently account for token order, positional encoding injects sequence structure by augmenting each token embedding with a unique positional vector, often derived from sinusoidal functions. Multi-head self-attention then projects input embeddings into several attention heads, each capturing different aspects of token interdependence via scaled dot-product attention. These mechanisms work in tandem to learn long-range dependencies, forming the backbone of modern LLMs.

Although pre-training imparts broad capabilities, the model may not excel in specialized tasks or capture domain-specific nuances. Fine-tuning addresses these shortcomings.

It is important to mention the stopping criteria for token generation. Although the model is trained via next-token prediction, inference systems often specify \emph{stopping criteria} such as:

- Special stop tokens (e.g., <EOS>) that signal an endpoint.
- Maximum generation lengths or timeouts to prevent unbounded text output.
- Custom-defined sentinel tokens (e.g., <STOP>, </s>) introduced during fine-tuning or prompt engineering.

These strategies ensure that the model terminates generation gracefully and aligns output with user or system requirements. Advantages of Large-Scale Pre-training

- Versatility: Exposure to varied contexts helps the model recognize patterns across multiple domains.

- **Efficiency:** A robust pre-trained model can be adapted for specific tasks, saving substantial computational resources.
- **Rich Internal Representations:** Ingesting vast corpora enables the model to learn linguistic norms, factual knowledge, and core reasoning capabilities (29).

Despite acquiring extensive general capabilities, a pre-trained LLM may not excel in specialized tasks or capture domain-specific nuances. This shortfall motivates further refinement through fine-tuning.

3.2 LLM FINE-TUNING

Once a model has been pre-trained as a next-token predictor, fine-tuning adapts it to specific tasks, data domains, or styles. By training on labeled datasets aligned with the target application, the model retains much of its general pre-trained knowledge while gaining specialized proficiency (37).

Task-Specific Adaptation

Fine-tuning can be tailored to:

- **Domain Adaptation:** Specializing in a field such as law or medicine by training on domain-specific corpora (38).
- **Task-Specific Objectives:** Optimizing performance on classification, summarization, or other tasks by adjusting parameters based on labeled examples.

Regularization techniques, such as dropout, weight decay, and learning rate scheduling, help avoid overfitting and maintain generalization (39).

Control Tokens and Special Instructions

During fine-tuning, additional tokens or instructions guide generation:

- **Instruction Tokens:** Special tokens (e.g., <SUMMARY> or <TRANSLATE>) that signal desired behavior.
- **Role Indicators:** Tokens like <USER>, <SYSTEM>, and <ASSISTANT> structure multi-turn dialogue.
- **Stop Tokens or Sequences:** Training the model to end output upon encountering specific tokens ensures controlled generation.

Introducing these tokens during training helps the model comply with them at inference time without manual post-processing.

Alignment and Further Refinements

Beyond task-specific fine-tuning, many large language models (LLMs) undergo additional alignment:

- **Reinforcement Learning from Human Feedback (RLHF):** The model's outputs are rated based on quality and adherence to guidelines, and it is fine-tuned to maximize positive feedback.
- **Safety and Ethical Constraints:** Fine-tuning may integrate policies to mitigate harmful or biased content.

Balancing Generality and Specialization

Fine-tuning channels the broad linguistic understanding from pre-training into high-value narrow tasks. However, overfitting on small or biased datasets can reduce the model's generality. Techniques like parameter-efficient fine-tuning or prompt-based finetuning (e.g., LoRA, prefix-tuning) help preserve versatility while achieving strong task performance.

3.3 REINFORCEMENT LEARNING FROM HUMAN FEEDBACK

RLHF refines the fine-tuned model by incorporating direct human judgments on output quality. By comparing model outputs with human evaluations, RLHF iteratively adjusts the model's behavior to enhance quality, safety, and alignment with user preferences (e.g., reducing bias or harmful content) (40). The RLHF process includes:

1. **Collecting Human Feedback:** Human annotators rate model outputs based on criteria such as coherence, factual accuracy, and helpfulness (41).
2. **Training a Reward Model:** A reward model is trained to predict human preferences from the collected feedback (42), serving as the objective for reinforcement learning.
3. **Policy Optimization:** Using reinforcement learning algorithms such as Proximal Policy Optimization (PPO) (43), the language model is optimized to maximize the predicted reward (44).

The outcome is a more trustworthy and user-friendly LLM that aligns better with ethical norms and practical use cases, ultimately improving the safety and utility of its outputs. Once the model is adequately fine-tuned and aligned, it moves to the inference stage.

3.4 LLM INFERENCE

After pre-training and optional refinement (e.g., fine-tuning or RLHF), the LLM enters the inference stage, processing user prompts to generate context-aware responses. At its core, inference relies on next-token prediction: given a prompt, the model probabilistically selects the most plausible next token, one step at a time. Additional strategies are often applied to control and optimize this process.

Decoding Strategies for Text Generation

The decoding strategy determines how tokens are sampled from the output distribution at each step. Common techniques include:

- Greedy Decoding: Selects the token with the highest probability at each step. It is computationally efficient and often produces coherent outputs, but may become repetitive or get stuck in suboptimal segments (45).
- Beam Search: Explores multiple "beams" (partial hypotheses) in parallel, periodically pruning unlikely candidates. This method can yield more creative or higher-probability sequences at the cost of increased computation (46).
- Sampling-Based Methods: Introduce randomness during token selection to foster diversity and avoid repetitive loops. Two popular variants are:
 - Top-k Sampling: Limits choices to the k most probable tokens at each step, then samples from this reduced set.
 - Nucleus (Top-p) Sampling: Samples from a dynamic shortlist of tokens whose cumulative probability is below a threshold p (47).

Tuning hyperparameters such as temperature, top-k, and top-up helps balance creativity and consistency in the generated output (2).

Efficiency and Deployment Considerations

As LLMs grow in size, running inference on consumer-grade devices or modest servers becomes challenging. Several techniques help mitigate these constraints:

- Model Quantization: Reduces numerical precision (e.g., from FP32 to INT8) to lower memory usage and accelerate operations, with slight accuracy trade-offs.

- Knowledge Distillation: Trains a smaller "student" model to mimic a larger teacher LLM, retaining much of the teacher's performance with fewer parameters (48).
- Hardware Acceleration: Uses specialized devices such as GPUs, TPUs, or custom ASICs to handle large matrix multiplications. Frameworks like TensorFlow and PyTorch support distributed training and inference (49,50).
- Inference Pipelines and Caching: Caching or partial re-evaluation of repeated prefixes reduces latency, and large-scale deployments may employ model or pipeline parallelism.

Controlling Generation and Stopping Criteria

During inference, special tokens (e.g., <EOS>, </s>) or user-defined sentinel tokens signal the end of output. Developers often impose maximum token limits or specific formatting requirements to ensure responses remain on-topic and within practical lengths, mitigating issues like runaway generation.

Real-World Usage

Inference may involve elaborate prompt engineering, providing instructions or examples to guide the model's output, and tool integrations for advanced interactions (e.g., retrieving external data or calling APIs). With proper inference infrastructure, users can issue prompts like:

- Summaries ("Summarize the following scientific paper in 200 words")
- Translations ("Translate this paragraph to French.")
- Q&A requests ("Explain the difference between supervised and unsupervised learning.")

The model then generates responses by leveraging the tailored knowledge and safeguards established in previous stages.

LLM inference bridges the gap between training and practical deployment. Through strategic decoding, hardware optimizations, and controlled output boundaries, modern LLMs effectively and efficiently respond to a broad range of real-world tasks, underpinning the interactivity and adaptability observed in AI chatbots, advanced search engines, and multimodal applications.

Impact of LLMs on Applications

LLMs have transformed a wide range of applications by generating contextually nuanced and interactive responses.

In Information Retrieval (IR), AI chatbots such as ChatGPT offer context-sensitive conversational experiences that challenge conventional search engines (51,52). These chatbots interpret user intent more accurately and support multi-turn dialogues, enhancing the search experience. Microsoft's New

Bing is an example of using LLMs to integrate conversational AI, thereby improving user engagement and satisfaction (53).

In Computer Vision (CV), researchers are developing vision-language models similar to ChatGPT to support multimodal dialogues. These models integrate visual and textual data to interpret and interact with various media types. Notable examples include BLIP-2, InstructBLIP, Otter, and MiniGPT-4, which combine instruction tuning with vision-language understanding for tasks such as image captioning, visual question answering, and interactive multimedia content creation (54,55,56,57).

This new wave of technology has led to a robust ecosystem of applications based on LLMs. For example, Microsoft 365 is being enhanced by LLMs (e.g., Copilot) to streamline office tasks, improve productivity, and assist with email writing, report generation, and data-driven recommendations (58). Similarly, Google Workspace is integrating LLMs to offer smart suggestions, automate routine tasks, and facilitate more efficient collaboration (52).

Beyond text generation, LLMs now incorporate code interpretation and function calling, enabling dynamic interactions. These features allow models not only to understand and generate natural language but also to execute code, bridging the gap between human instructions and machine operations.

Tools such as the CoRE Code Interpreter facilitate tasks like data analysis and visualization by enabling users to execute and debug code within the interface (59,60). Meanwhile, platforms like GitHub Copilot assist developers with intelligent autocompletion, error detection, and automating repetitive coding tasks (61). Function calling capabilities further extend LLM utility by interfacing with external tools and APIs, retrieving real-time data, executing automated workflows, and controlling devices (62). For instance, integrating LLMs with Internet of Things (IoT) devices enables smart home systems that can execute complex user commands, enhancing automation and convenience (63).

LLMs also play significant roles in specialized sectors. In healthcare, they assist with medical documentation, patient interaction, and even preliminary diagnostics by analyzing patient data and providing evidence-based recommendations (38). In education, LLMs serve

as personalized tutors, offering customized learning experiences, answering student queries, and generating educational content tailored to individual learning styles (64).

3.5 COMMON LIMITATIONS AND CHALLENGES OF LLMs

LLMs have revolutionized natural language processing, yet their adoption brings technical, security, and ethical challenges:

- Hallucination: LLMs may generate outputs that are coherent yet factually incorrect.
- Context Limitations: Fixed context windows restrict the processing of large documents or multi-turn dialogues.
- Bias and Fairness: Models can perpetuate harmful stereotypes present in their training data.
- Knowledge Freshness: Relying on pre-training data may result in outdated information.

A chief concern is hallucination, where LLMs produce outputs that appear coherent but are factually wrong. Since LLMs rely on learned statistical patterns rather than real-time validation, they can confidently produce erroneous or fabricated content (3). Addressing this issue requires deliberate evaluation methods, including human oversight, adversarial testing, and integration with external knowledge bases for dynamic fact-checking (65).

Another challenge is context limitation and the associated knowledge cutoff. LLMs have fixed context windows, restricting the amount of text processed at once. For example, GPT-3 can handle approximately 2,048 tokens, while some GPT-4 variants support up to 8,192 tokens (2,66). Tasks involving lengthy documents or multi-turn conversations may require chunking or retrieval-augmented generation to reconstruct relevant context (12,67). Additionally, since LLMs do not automatically update after their training date, they may provide inaccurate answers over time if not refreshed with newer data (1,68).

Bias and fairness are critical ethical issues when LLMs perpetuate harmful stereotypes from their training data. Underrepresentation or misrepresentation of specific groups can lead to skewed or discriminatory outputs (69,70). These biases can have harmful consequences, particularly in high-stakes fields such as healthcare or finance. Ensuring balanced datasets and applying targeted debiasing strategies are essential for equitable model outcomes (71).

The quality of domain-specific datasets is another challenge. Insufficient or poorly curated data may cause the model to overfit, performing well on training examples but poorly on unseen data (69). This risk intensifies in specialized fields, where limited training data might not capture critical nuances (72). Even extensive datasets require careful curation, as both data size and quality significantly influence model performance (73).

Compounding these issues are computational constraints. Fine-tuning large-scale models requires powerful GPUs and substantial memory to optimize massive parameter sets (42). These requirements can be prohibitively expensive for smaller organizations, limiting innovation and inclusivity in LLM research. Although techniques like Low-Rank Adaptation (LoRA) (74) and prefix tuning (75) reduce the number of trainable parameters, optimizing hardware usage and algorithmic strategies remains an active research focus.

Finally, outdated knowledge can degrade performance over time, especially in rapidly evolving fields

(12). Stale or incomplete information contributes to hallucinations and factual inconsistencies, reducing user trust and limiting applicability in specialized settings (76,77). In domains like law or medicine, where recent developments are crucial, LLMs may struggle to provide accurate or current answers unless they are periodically updated or augmented with external resources (78,79).

Addressing these multifaceted challenges requires a multidisciplinary approach. Advanced retrieval-augmented strategies and dynamic updating protocols help mitigate hallucinations and outdated knowledge, while diverse data collection and ongoing bias detection are necessary to ensure fairness. Complementary governance frameworks also provide robust oversight, especially in contexts where misinformation or data privacy pose significant risks.

3.6 RECENT STRATEGIES TO IMPROVE LLMS

Recent developments in LLMs have focused on addressing challenges such as hallucination, bias, and limited context handling. Prompt engineering involves crafting the input text (or “prompt”) to guide and optimize the model’s output (80). This technique nudges the model toward a particular style, content focus, or reasoning pathway by carefully selecting the words, formatting, and structure of the prompt.

Techniques such as few-shot and zero-shot prompting enable LLMs to tackle new tasks with minimal or no additional labeled data, increasing flexibility and reducing domain adaptation costs. Chain-of-thought prompts enhance transparency and correctness by encouraging the model to provide intermediate reasoning steps, thereby reducing errors and building user trust.

To counteract the out-of-date knowledge inherent in pre-trained LLMs, researchers have introduced RAG (81). RAG integrates an information retrieval module that fetches relevant data (e.g., from web documents or curated databases) before generation, providing up-to-date, grounded information. Expanding on this, Graph-RAG incorporates knowledge graphs or other structured databases to ensure that the model's answers are both current and logically consistent with known entity relationships (82).

Meanwhile, specialized Agents are being developed to coordinate LLMs with external tools and APIs, creating a dynamic environment where the model can break down tasks, consult external resources, and perform specialized calculations. Together, these improvements enhance factual grounding and expand the range of tasks that modern LLMs can effectively tackle.

4 CONCLUSION

This survey traced the progression from early statistical approaches to Transformer-based Large Language Models, highlighting how scale, pre-training on broad corpora, and self-attention enabled general-purpose language understanding and generation with minimal task-specific adaptation. We organized the core mechanics of the LLM lifecycle—pre-training, optional fine-tuning and alignment, and inference—into a compact narrative aimed at newcomers, emphasizing the conceptual foundations rather than exhaustive implementation details. In doing so, the paper's contribution is to provide an accessible on-ramp: a coherent account of the ideas and milestones that define the LLM era, sufficient to build intuition for how contemporary models are constructed, adapted, and used.

4.1 SCOPE AND LIMITATIONS

This work is intentionally introductory. It abstracts away many engineering choices (e.g., large-scale optimization strategies, data curation pipelines, distributed training systems) and does not attempt a comprehensive taxonomy of models or benchmarks. Likewise, it focuses on the conceptual role of retrieval for maintaining currency without

elaborating full system designs. Ethical, legal, and societal dimensions are acknowledged in outline but not surveyed in depth. These boundaries are deliberate, keeping attention on the minimal set of principles needed to understand the rise and basic operation of LLMs.

4.2 DIRECTIONS FOR FURTHER STUDY

Several areas warrant deeper treatment beyond this primer. First, systematic comparisons of training objectives and data regimes could clarify how specific design choices shape generalization and robustness. Second, richer accounts of alignment methods and their theoretical underpinnings would complement the operational overview provided here. Third, longitudinal perspectives on model evaluation, dataset shift, and domain transfer remain open and important. Finally, integrative studies of retrieval-supported generation—spanning indexing strategies, context assembly, and failure modes—would help translate foundational understanding into mature, reliable applications.

REFERENCES

- Agarwal, V., Jin, Y., Chandra, M., De Choudhury, M., Kumar, S., & Sastry, N. (2024). MedHalu: Hallucinations in responses to healthcare queries by large language models. arXiv. <https://doi.org/10.48550/arXiv.2409.19492>
- Arslan, M., Ghanem, H., Munawar, S., & Cruz, C. (2024). A survey on RAG with LLMs. *Procedia Computer Science*, 246, 3781–3790. <https://doi.org/10.1016/j.procs.2024.09.178>
- Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kern7, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., et al. (2022). Constitutional AI: Harmlessness from AI feedback. arXiv. <https://doi.org/10.48550/arXiv.2212.08073>
- Bahdanau, D. (2014). Neural machine translation by jointly learning to align and translate. arXiv. <https://doi.org/10.48550/arXiv.1409.0473>
- Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. (2021). On the opportunities and risks of foundation models. arXiv. <https://doi.org/10.48550/arXiv.2108.07258>
- Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. (2022). On the opportunities and risks of foundation models. arXiv. <https://doi.org/10.48550/arXiv.2108.07258>
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS 2020)*. Curran Associates.

- Carroll, A. J., & Borycz, J. (2024). Integrating large language models and generative artificial intelligence tools into information literacy instruction. *The Journal of Academic Librarianship*, 50, Article 102899. <https://doi.org/10.1016/j.acalib.2024.102899>
- Chen, Y.-C., Hsu, P.-C., Hsu, C.-J., & Shan Shiu, D. (2024). Enhancing function-calling capabilities in LLMs: Strategies for prompt formats, data integration, and multilingual translation. *arXiv*. <https://doi.org/10.48550/arXiv.2412.01130>
- Cheng, W., Sun, K., Zhang, X., & Wang, W. (2025). Security attacks on LLM-based code completion tools. *arXiv*. <https://doi.org/10.48550/arXiv.2408.11006>
- Christiano, P., Leike, J., Brown, T. B., Martic, M., Legg, S., & Amodei, D. (2023). Deep reinforcement learning from human preferences. *arXiv*. <https://doi.org/10.48550/arXiv.1706.03741>
- Dai, W., Li, J., Li, D., Tiong, A. M. H., Zhao, J., Wang, W., Li, B., Fung, P., & Hoi, S. (2023). InstructBLIP: Towards general-purpose vision-language models with instruction tuning. *arXiv*. <https://doi.org/10.48550/arXiv.2305.06500>
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of NAACL-HLT (Vol. 1, pp. 2)*. Association for Computational Linguistics.
- Ding, J., Nguyen, H., & Chen, H. (2024). Evaluation of question-answering based text summarization using LLM (Invited Paper). *Proceedings of the 2024 IEEE International Conference on Artificial Intelligence Testing (AITest) (pp. 142–149)*. IEEE. <https://doi.org/10.1109/AITest62860.2024.00025>
- Dong, Y., Zhang, H., Li, C., Guo, S., Leung, V. C., & Hu, X. (2024). Fine-tuning and deploying large language models over edges: Issues and approaches. *arXiv*. <https://doi.org/10.48550/arXiv.2408.10691>
- Edge, D., Trinh, H., Cheng, N., Bradley, J., Chao, A., Mody, A., Truitt, S., & Larson, J. (2024). From local to global: A graph RAG approach to query-focused summarization. *arXiv*. <https://doi.org/10.48550/arXiv.2404.16130>
- Franceschelli, G., & Musolesi, M. (2024). Creative beam search: LLM-as-a-judge for improving response generation. *arXiv*. <https://doi.org/10.48550/arXiv.2405.00099>
- Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M., & Wang, H. (2024). Retrieval-augmented generation for large language models: A survey. *arXiv*. <https://doi.org/10.48550/arXiv.2312.10997>
- Goyal, T., Li, J. J., & Durrett, G. (2023). News summarization and evaluation in the era of GPT-3. *arXiv*. <https://doi.org/10.48550/arXiv.2209.12356>
- Guo, T., Chen, X., Wang, Y., Chang, R., Pei, S., Chawla, N. V., Wiest, O., & Zhang, X. (2024). Large language model based multi-agents: A survey of progress and challenges. *arXiv*. <https://doi.org/10.48550/arXiv.2402.01680>
- Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv*. <https://doi.org/10.48550/arXiv.1503.02531>

- Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., de Melo, G., Gutierrez, C., Kirrane, S., Labra Gayo, J. E., Navigli, R., Neumaier, S., et al. (2021). Knowledge graphs. *ACM Computing Surveys*, 54(s/n), 1–37. <https://doi.org/10.1145/3447772>
- Holtzman, A., Buys, J., Du, L., Forbes, M., & Choi, Y. (2020). The curious case of neural text degeneration. *arXiv*. <https://doi.org/10.48550/arXiv.1904.09751>
- Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. *arXiv*. <https://doi.org/10.48550/arXiv.1801.06146>
- Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2021). LoRA: Low-rank adaptation of large language models. *arXiv*. <https://doi.org/10.48550/arXiv.2106.09685>
- Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., et al. (2023). A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv*. <https://doi.org/10.48550/arXiv.2311.05232>
- Ji, B., Duan, X., Zhang, Y., Wu, K., & Zhang, M. (2024). Zero-shot prompting for LLM-based machine translation using in-domain target sentences. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, s/n, 1–12. <https://doi.org/10.1109/TASLP.2024.3519814>
- Johnson, L. E., & Rashad, S. (2024). An innovative system for real-time translation from American Sign Language (ASL) to spoken English using a large language model (LLM). *Proceedings of the 2024 IEEE 15th Annual UEMCON* (pp. 605–611). IEEE. <https://doi.org/10.1109/UEMCON62879.2024.10754690>
- Jung, S. J., Kim, H., & Jang, K. S. (2024). LLM based biological named entity recognition from scientific literature. *Proceedings of the 2024 IEEE International Conference on Big Data and Smart Computing (BigComp)* (pp. 433–435). IEEE. <https://doi.org/10.1109/BigComp60711.2024.00095>
- Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., & Amodei, D. (2020). Scaling laws for neural language models. *arXiv*. <https://doi.org/10.48550/arXiv.2001.08361>
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., et al. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33, 9459–9474.
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., et al. (2021). Retrieval-augmented generation for knowledge-intensive NLP tasks. *arXiv*. <https://doi.org/10.48550/arXiv.2005.11401>
- Li, B., Zhang, Y., Chen, L., Wang, J., Yang, J., & Liu, Z. (2023). Otter: A multi-modal model with in-context instruction tuning. *arXiv*. <https://doi.org/10.48550/arXiv.2305.03726>
- Li, J., Li, D., Savarese, S., & Hoi, S. (2023). BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv*. <https://doi.org/10.48550/arXiv.2301.12597>

- Li, X. L., & Liang, P. (2021). Prefix-tuning: Optimizing continuous prompts for generation. arXiv. <https://doi.org/10.48550/arXiv.2101.00190>
- Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., Zhang, Y., Narayan, D., Wu, Y., Kumar, A., et al. (2023). Holistic evaluation of language models. arXiv. <https://doi.org/10.48550/arXiv.2211.09110>
- Lin, H. (2024). Designing domain-specific large language models: The critical role of fine-tuning in public opinion simulation. arXiv. <https://doi.org/10.48550/arXiv.2409.19308>
- Ling, C., Zhao, X., Lu, J., Deng, C., Zheng, C., Wang, J., Chowdhury, T., Li, Y., Cui, H., Zhang, X., et al. (2023). Domain specialization as the key to make large language models disruptive: A comprehensive survey. arXiv. <https://doi.org/10.48550/arXiv.2305.18703>
- Manerba, M. M., Stańczak, K., Guidotti, R., & Augenstein, I. (2023). Social bias probing: Fairness benchmarking for language models. arXiv. <https://doi.org/10.48550/arXiv.2311.09090>
- Mariño, J. B., Banchs, R. E., Crego, J. M., de Gispert, A., Lambert, P., Fonollosa, J. A. R., & Costa-jussà, M. R. (2006). N-gram-based machine translation. *Computational Linguistics*, 32(4), 527–549. <https://doi.org/10.1162/coli.2006.32.4.527>
- Mehta, H., Kumar Bharti, S., & Doshi, N. (2024). Comparative analysis of part of speech (POS) tagger for Gujarati language using deep learning and pre-trained LLM. *Proceedings of the 2024 3rd International Conference for Innovation in Technology (INOCON)* (pp. 1–3). IEEE. <https://doi.org/10.1109/INOCON60754.2024.10511678>
- Meng, X., Yan, X., Zhang, K., Liu, D., Cui, X., Yang, Y., Zhang, M., Cao, C., Wang, J., Wang, X., et al. (2024). The application of large language models in medicine: A scoping review. *iScience*, 27, Article 109713. <https://doi.org/10.1016/j.isci.2024.109713>
- Miah, Md. S. U., Kabir, Md. M., Sarwar, T. B., Safran, M., Alfarhood, S., & Mridha, Md. F. (2024). A multimodal approach to cross-lingual sentiment analysis with ensemble of transformer and LLM. *Scientific Reports*, 14, Article 9603. <https://doi.org/10.1038/s41598-024-60210-7>
- Mialon, G., Dessi, R., Lomeli, M., Nalmpantis, C., Pasunuru, R., Raileanu, R., Rozière, B., Schick, T., Dwivedi-Yu, J., Celikyilmaz, A., et al. (2023). Augmented language models: A survey. arXiv. <https://doi.org/10.48550/arXiv.2302.07842>
- Microsoft. (2023). Introducing Microsoft 365 Copilot: A whole new way to work. Microsoft News/Blog.
- Mikolov, T. (2013). Efficient estimation of word representations in vector space. arXiv. <https://doi.org/10.48550/arXiv.1301.3781>
- 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. <https://doi.org/10.48550/arXiv.2307.06435>
- Navigli, R., Conia, S., & Ross, B. (2023). Biases in large language models: Origins, inventory, and discussion. *ACM Journal of Data and Information Quality*, 15, 1–21.

- OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., et al. (2024). 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., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. arXiv. <https://doi.org/10.48550/arXiv.2203.02155>
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). PyTorch: An imperative style, high-performance deep learning library. arXiv. <https://doi.org/10.48550/arXiv.1912.01703>
- Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 1532–1543). Association for Computational Linguistics.
- Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. arXiv. <https://doi.org/10.48550/arXiv.1802.05365>
- Prabhu, S. P. (2024). PEDAL: Enhancing greedy decoding with large language models using diverse exemplars. arXiv. <https://doi.org/10.48550/arXiv.2408.08869>
- Procko, T. T., & Ochoa, O. (2024). Graph retrieval-augmented generation for large language models: A survey. Proceedings of the 2024 Conference on AI, Science, Engineering, and Technology (AIxSET) (pp. 166–169). IEEE. <https://doi.org/10.1109/AIxSET62544.2024.00030>
- Qu, C., Dai, S., Wei, X., Cai, H., Wang, S., Yin, D., Xu, J., & Wen, J.-R. (2024). Tool learning with large language models: A survey. arXiv. <https://doi.org/10.1007/s11704-024-40678-2>
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI Blog/Technical report.
- Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2023). Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv. <https://doi.org/10.48550/arXiv.1910.10683>
- Rong, B., & Rutagemwa, H. (2024). Leveraging large language models for intelligent control of 6G integrated TN-NTN with IoT service. IEEE Network, 38, 136–142. <https://doi.org/10.1109/MNET.2024.3384013>
- Sahoo, P., Singh, A. K., Saha, S., Jain, V., Mondal, S., & Chadha, A. (2024). A systematic survey of prompt engineering in large language models: Techniques and applications. arXiv. <https://doi.org/10.48550/arXiv.2402.07927>
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv. <https://doi.org/10.48550/arXiv.1707.06347>
- Shenoy, N., & Mbaziira, A. V. (2024). An extended review: LLM prompt engineering in cyber defense. Proceedings of the 2024 International Conference on Electrical, Computer and Energy Technologies (ICECET) (pp. 1–6). IEEE. <https://doi.org/10.1109/ICECET61485.2024.10698605>

- Singhal, K., Azizi, S., Tu, T., Mahdavi, S. S., Wei, J., Chung, H. W., Scales, N., Tanwani, A., Cole-Lewis, H., et al. (2023). Large language models encode clinical knowledge. *Nature*, 620, 172–180.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15, 1929–1958.
- Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *arXiv*. <https://doi.org/10.48550/arXiv.1409.3215>
- Tonmoy, S., Zaman, S., Jain, V., Rani, A., Rawte, V., Chadha, A., & Das, A. (2024). A comprehensive survey of hallucination mitigation techniques in large language models. *arXiv*. <https://doi.org/10.48550/arXiv.2401.01313>
- Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. (2023). LLaMA: Open and efficient foundation language models. *arXiv*. <https://doi.org/10.48550/arXiv.2302.13971>
- Vaswani, A. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*.
- Vilar, D., Freitag, M., Cherry, C., Luo, J., Ratnakar, V., & Foster, G. (2023). Prompting PaLM for translation: Assessing strategies and performance. *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 15406–15427). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2023.acl-long.859>
- Wang, X., Wang, Z., Gao, X., Zhang, F., Wu, Y., Xu, Z., Shi, T., Wang, Z., Li, S., Qian, Q., et al. (2024). Searching for best practices in retrieval-augmented generation. *arXiv*. <https://doi.org/10.48550/arXiv.2407.01219>
- Xiong, H., Bian, J., Li, Y., Li, X., Du, M., Wang, E., Yin, D., & Helal, S. (2024). When search engine services meet large language models: Visions and challenges. *arXiv*. <https://doi.org/10.48550/arXiv.2407.00128>
- Xu, H., Gan, W., Qi, Z., Wu, J., & Yu, P. S. (2024). Large language models for education: A survey. *arXiv*. <https://doi.org/10.48550/arXiv.2405.13001>
- Xu, S., Li, Z., Mei, K., & Zhang, Y. (2024). AIOS compiler: LLM as interpreter for natural language programming and flow programming of AI agents. *arXiv*. <https://doi.org/10.48550/arXiv.2405.06907>
- Ye, H., Liu, T., Zhang, A., Hua, W., & Jia, W. (2023). Cognitive mirage: A review of hallucinations in large language models. *arXiv*. <https://doi.org/10.48550/arXiv.2309.06794>
- Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., et al. (2024). A survey of large language models. *arXiv*. <https://doi.org/10.48550/arXiv.2303.18223>
- Zhu, D., Chen, J., Shen, X., Li, X., & Elhoseiny, M. (2023). MiniGPT-4: Enhancing vision-language understanding with advanced large language models. *arXiv*. <https://doi.org/10.48550/arXiv.2304.10592>

Zhu, Y., Yuan, H., Wang, S., Liu, J., Liu, W., Deng, C., Chen, H., Liu, Z., Dou, Z., & Wen, J.-R. (2024). Large language models for information retrieval: A survey. arXiv. <https://doi.org/10.48550/arXiv.2308.07107>