Large Language Model developments

Philip Treleaven, Sara Craciun, and Edward Sharkey

UCL Computer Science

# ABSTRACT

**Large Language Models** (LLMs) are Generative AI models trained with vast amounts of data and context, and excel at processing, understanding, and generating human language using Transformer models to learn the relationships [Menon, 2023]. LLMs are a subset of Transformer models, and hence are closely related, but not interchangeable.

Given the ‘explosion’ of LLM research the challenge is tracking emerging developments and matching LLMs to applications. This paper reviews LLM developments, with the aim of understanding the underlying models’ computational strengths and weaknesses, building on previous work focusing on LLM optimization [Sharkey & Treleaven, 2025].

This review divides LLM developments into:

1. **Traditional LLMs** – basic LLMs are trained on large datasets, supported by several popular foundational models (e.g. ChatGPT, DeepSeek). In addition, LLM prompt engineering techniques such as Chain-of-Thought (CoT) and Retrieval-Augmented Generation (RAG) can be used to optimize traditional LLM performance.
2. **Multi-model LLMs** – composite models are being proposed where multiple specialized models (experts) work together (e.g. Mixture of Experts, Multimodal LLMs).
3. **Large Deep Models** – what we refer to as“deep” models use NLP combined with reasoning capabilities and apply structured reasoning techniques (e.g. Large Reasoning Models).
4. **Domain-specific LLMs** – LLMs tailored for specific domains comprising a) specific data types such as numeric or images; or b) an industry sector, such as finance, law, medicine, or education. This also includes “Vibe Coding” [Kumar, 2025].
5. **Optimizing LLMs** - incorporating specialist mechanisms to optimize an LLM, beyond prompt engineering (e.g. Privacy-Preserving/Differentially Privacy LLMs).
6. **Underlying Network interpretability** - understanding the internal reasoning processes of the underlying trained neural networks. Several methods are being developed to improve interpretability, focusing on techniques like attention visualization, attribution methods, and modifications to the model architecture itself (e.g. Representational Engineering).

In summary, we are especially interested in tracking the developments of LLMs to understand the computational strengths of each LLM technique and the behaviour of the underlying neural network architecture. A great source of information is GitHub’s Awesome-LLM (<https://github.com/Hannibal046/Awesome-LLM>). If there are emerging LLMs we have missed please inform us ([p.treleaven@ucl.ac.uk](mailto:p.treleaven@ucl.ac.uk)).

To support this paper, we are building an LLM system to recommend and track new LLMs for specific ML application domains, starting with financial services.

# Introduction

As Large Language Models and Transformer models become more sophisticated, it is increasingly important to understand the various paradigms underpinning their design and functionality. LLMs are garnering significant attentionfor their ability to produce highly coherent and contextually relevant outputs. These models play a pivotal role in tasks such as summarization, translation, sentiment analysis, and dialogue systems, among others.

The following sections provide as background an overview of the prominent LLM technologies. For context we start with a high-level explanation of common ML paradigms.

## Machine Learning Paradigms

In terms of this paper, ML techniques can be divided into:

* **Classic ML** - classic ML techniques primarily involve statistical and connectionist models that rely on feature engineering and discriminative learning. These methods typically require carefully curated datasets and substantial domain knowledge to extract relevant features.
* **Generative AI (GenAI)** - GenAI focuses on creating systems capable of dynamically producing text, images, audio, video, code, or other modalities in a manner that appears strikingly human like. LLMs are essentially a subset of GenAI for human-like text and speech:
  + *LLM* used to generate text or complete tasks based on their training data and pre-defined capabilities LLMs (e.g. Traditional LLMs). In addition, *LLM Agents* introduces features to LLMs like planning, memory, and tool usage to achieve greater autonomy and handle more complex, multi-step tasks (e.g. Multi-model LLMs).
  + *World models* GenAI models that learn the dynamic behaviors of the real world, including physics and spatial properties. A model that observes its environment and predicts what might happen next, considering its current knowledge and unknown factors that could affect future outcomes [NVIDIA, 2025].

Table 1 is a simple comparison of the core features of Traditional ML, and Generative AI.

|  |  |  |
| --- | --- | --- |
| **Feature** | **Classic ML** | **Generative AI** |
| **Primary Function** | Extract relevant features from curated datasets | Generative content creation |
| **Focus** | Natural language processing etc. | Generating new content (text, images, etc.) |
| **Examples** | Recommendation, forecasting | Text generation, image generation, etc. |
| **Key Concept** | ML, neural networks | LLMs (e.g. NLP), World models |

Table 1: Comparison of Classic ML and Generative AI

## LLM Technologies

Unpacking the key underlying LLM technologies:

* **Artificial Neural Networks (ANNs)** – ANNs are a type of machine learning (ML) process that uses interconnected nodes or neurons in a layered structure “inspired” by the human brain [McCulloch & Pitts, 1943]. An ANN uses training examples and learns by adjusting a “weight” associated with each connection linking two neurons [Rumelhart et al, 1986].
* **Deep Learning** – uses ANNs with multiple layers (i.e., deep neural networks) to analyze data and learn complex patterns [LeCun et al, 2015].
* **Natural Language Processing (NLP)** – NLP, as discussed, involves the understanding of the meaning of text and speech [Hirschberg & Manning, 2015; DeepLearning.ai, 2023]. NLP centers on tokenization, a way of separating a piece of text into smaller units called “tokens” [Grefenstette, 1999]. Here, tokens can be either words, characters, or sub-words.
* **Transformers** – a type of deep learning ANN model that is used for NLP tasks and can learn long-range dependencies between words in a sentence [Vaswani et al, 2017; Devlin et al, 2019; Lin et al, 2021].

## Tokenization for NLP

A crucial first step in Natural Language Processing (NLP) methods is **tokenization** - segmenting text into words, sub‑words or characters. Comprehensive treatments of word‑, character‑ and sentence‑level tokenization appear in Chapter 2 of Jurafsky and Martin [Jurafsky & Martin, 2025], while for state‑of‑the‑art sub‑word schemes such as byte‑pair encoding see [Sennrich et al, 2016].

Figure 1 presents a taxonomy for sentence compression methods from Statistical Techniques to Reinforcement Learning. Prominent is Deep Learning that makes use of multiple-layer neural networks. It divides broadly into Recurrent Neural Networks, Convolutional Neural Networks, Graph Neural Networks and Transformers.

A diagram of a method

AI-generated content may be incorrect.

Figure 1: Taxonomy of sentence compression methods [Passali et al, 2023]

## Foundation LLM models

Central for LLMs are foundation models, such as ChatGPT and DeepSeek, that provide a powerful framework for creating new generative AI (GenAI) content, offering capabilities for natural language understanding and generation, image processing, and more. Broadly, foundation Large Language Models (LLMs) are categorized into:

* **Open-source LLMs** - Models whose source code and architecture are publicly available for use, modification, and distribution [Meta Platform, 2025]. This openness enables users to develop customized applications with tailored language generation capabilities. Examples include Meta’s LLaMA, DeepSeek R1, and Alibaba’s Qwen.
* **Proprietary LLMs** - Models developed and owned by specific companies using private training data and infrastructure. They are typically pre-trained on large diverse datasets and are ready for immediate deployment for general-purpose tasks. Examples include OpenAI’s GPT-4, Google’s Gemini, and Anthropic’s Claude.

In summary, a foundation model is a large, general‑purpose AI trained on massive public data; you deploy it for your domain by writing well‑crafted prompts (or light fine‑tuning) that steer its tone, structure, and reasoning, and then further improve accuracy by adding Retrieval‑Augmented Generation (RAG), which fetches the most relevant facts from your own knowledge base at query time and injects them into the prompt so the model produces up‑to‑date, grounded answers without needing full retraining.

In Table 2 we have attempted to show example foundation models and tools available for LLM programming.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **LLM category** | **Traditional LLMs** | **Multi-Modal LLMs** | **Large “Deep” Models** | **Domain-specific LLMs** | **Optimizing LLM tooling /techniques** | **Underlying network /interpretability** |
| **Models / Tools** | ChatGPT-4o (GPT-4-turbo), DeepSeek V3, GPT-3.5-turbo, LLaMA-2 (70B), Claude 2 | GPT-4o Vision, Gemini 1.5 Pro, DeepSeek V3-Vision, DALLE-3 (image), Stable Diffusion XL, LLaVA 1.5, AutoGen / AutoGPT\* | GPT-4, Claude 3 Opus, Gemini 1.5 Ultra (1 M-token context), DeepSeek R1, SONAR | BloombergGPT (finance), BioGPT (biomedical), Med-PaLM 2 (medical), FinGPT, Code Llama / StarCoder (software), Galactica (science) | DeepSpeed-ZeRO / MoE, LoRA & QLoRA adapters, FlashAttention-2, BitsAndBytes 8-/4-bit quantisation, vLLM & ONNX Runtime / TensorRT-LLM | GPT-2, BERT, TransformerLens, Captum, Ecco, Attention-Roll-out (Acme interp suite) |

Table 2: LLMs, Foundational Models and LLM tools

One major concern of users of foundation LLMs is data privacy and security. Namely, foundation models “harvesting” organizations’ private and sensitive data during training. Key application areas include sensitive healthcare data which is heavily regulated and highly valuable financial data. LLMs often ingest vast and diverse datasets, increasing the likelihood of unintentionally memorizing and reproducing personally identifiable information [Anthropic, 2025; DeepSeek, 2025].

LLMs are subject to passive privacy leakage and also active privacy attacks [Zhang et al, 2025a; Yan et al, 2025]. Based on where privacy protection is located, we can group LLMs into three categories: a) privacy protection in pre-training; b) privacy protection in fine-tuning; and c) privacy protection in inferences.

Broadly there are three privacy strategies:

* **Foundation models** – configure privacy settings to only use sensitive data for prompts and not as part of the LLM training. Foundation models now provide comprehensive settings to address privacy concerns during pre-training and fine-tuning.
* **Privacy techniques** – use sophisticated privacy techniques, such as differential privacy, data cleaning, and federated learning, to mitigate privacy risks.
* **Custom LLM**- install and customize a “local” LLM either by purchasing a pre-trained foundational LLM or installing an open-source LLM, that may require training.

# LLM Taxonomy

This section introduces our LLM taxonomy, listing prominent examples, with details and references provided in the sections below:

1. **Traditional LLMs** – basic LLMs are trained on large datasets and using deep learning techniques. LLM prompt engineering techniques, such as Chain-of-Thought (CoT), and Retrieval-Augmented Generation (RAG) can be used to optimize traditional LLM performance:
   * *Small Language Models (SLM)*– SLM models are smaller in scale and scope, trained on domain-specific data, and designed to be more computationally efficient, cost-effective, and suitable for specific tasks or domains.
   * *Large Language Models (LLM)* – LLMs are AI models are trained on vast amounts of data and context, and excels at processing, understanding, and generating human language. data and use deep learning techniques, specifically transformer models, to learn relationships.
   * *Prompt Engineering (PE)* - the process of designing and refining GenAI prompts which are basically instructions, questions, or guidelines that guide LLM responses. An example being *Chain-of Thought* reasoning.
2. **Multi-model LLMs** – composite models have been proposed where multiple specialized models (experts) work together:
   * *Mixture of Experts (MoE)* – MoE is a machine learning technique where multiple specialized models (experts) work together, with a gating network selecting the best expert for each input.
   * *Multimodal LLMs (MLLM)* – MLLMs are AI systems also understand and generate various types of data, but unlike traditional LLMs that primarily focus on text, MLLMs expand this capability by integrating multiple modalities, enabling them to perform more complex and versatile tasks.
   * *Agentic AI* – are mostly LLM system composed of “agents” that can make decisions and act autonomously to achieve specific goals.
3. **Large Deep Models** – what we refer to as“deep” models use NLP combined with reasoning capabilities and apply structured reasoning techniques:
   * *Large Reasoning Models (LRM*) – LRM use NLP combined with reasoning capabilities and are trained to apply structured reasoning techniques when responding to prompts.
   * *Large Action Models (LAM)* – LAM goes beyond generating text, like a Large Language Model (LLM), by understanding and executing actions based on user input and context.
   * *Large Context/Contextual* Models – large context models and contextual models address different aspects of context.
4. **Domain-specific LLMs** – LLMs and Transformer models tailored for specific domains, comprising a) specific data types such as numeric or images; b) an industry sector, such as finance, law, medicine, or education; and c) multiple domains:
   * *Large Quantitative Models (LQM)* – LQM are specifically to process, analyze, and generate quantitative data rather than natural language.
   * *Vibe Coding LLMs* - an approach to programming and application development that heavily relies on an LLM for code generation, refinement, and debugging.
   * *Financial LLMs–* Finance/Quant Models specialized in processing financial data, market analysis, and quantitative reasoning.
   * *Legal Models–* Legal models are designed to understand, interpret, and generate legal language, documents, and case law.
5. **Optimizing LLMs** – incorporating specialist mechanism to optimize an LLM, falling between prompt engineering and network interpretability:
   * *Quantization LLMs -* Quantization reduces model size by storing weights with lower‑precision formats such as INT8 rather than FP32 or FP16.
   * *Privacy-Preserving LLMs* – Privacy-Preserving/Differentially Privacy LLMs incorporate mechanisms that limit the ability to memorize or leak sensitive training data.
6. **Underlying Network interpretability** - understanding the internal reasoning processes of trained neural networks. Several methods are being developed to improve the interpretability of transformers, focusing on techniques like attention visualization, attribution methods, and modifications to the model architecture itself:
   * *Mechanistic Interpretability* - seeks to understand the internal reasoning processes of trained neural networks and gain insight into how and why they produce the outputs that they do.
   * *Representation Engineering –* RepE or Activation Engineering seeks to understand and control of AI by observing and manipulating the internal representations - weights or activations - that AI uses to understand and process information.

# Traditional LLMs

Traditional LLMs divide broadly into basic (un-optimized) and optimized models using prompt engineering. As is well known, prompt engineering involves crafting clear, concise prompts that direct generative AI systems towards desired outputs. By carefully selecting keywords, contextual cues, or domain-specific language, developers can influence how models interpret queries and shape their final responses [Brown et al, 2025].

**Traditional LLMs** – basic LLMs are trained on large datasets and use deep learning techniques. LLM prompt engineering techniques such as Chain-of-Thought (CoT) and Retrieval-Augmented Generation (RAG) can be used to optimize traditional LLM performance [Sharkey & Treleaven, 2025]:

* *Small Language Models (SLM)*– SLM models are smaller in scale and scope, trained on domain-specific data, and designed to be more computationally efficient, cost-effective, and suitable for specific tasks or domains [Shone, 2024].
* *Large Language Models (LLM)* – LLMs are AI models are trained on vast amounts of data and context, and excels at processing, understanding, and generating human language. data and use deep learning techniques, specifically transformer models, to learn relationships.
* *Prompt Engineering (PE)* - the process of designing and refining GenAI prompts which are basically instructions, questions, or guidelines that guide LLM responses. An example being Chain-of Thought (CoT) reasoning [Sahoo et al, 2024].
* *Chain-of Thought (CoT) -* Chain-of Thought, Tree of Thought, Graph of Thought, Forest of Thought models etc. extend standard LLMs by explicitly planning sequences of intermediate reasoning steps or actions to achieve specific goals. These models enhance traditional chain-of-thought reasoning by exploring multiple possible reasoning paths structured as trees or graphs. This approach allows for systematic exploration of alternatives and backtracking, improving problem-solving capabilities on complex, multi-step tasks that require strategic planning and evaluation [Bi et al, 2024].
* *Retrieval-Augmented Generation (RAG)* – RAG models enhance traditional LLMs by integrating specialist knowledge retrieval systems. Instead of relying solely on pretrained knowledge, these models dynamically access relevant documents or databases during inference, improving accuracy and grounding outputs in up-to-date or domain-specific information. Enhances general LLMs with external knowledge access [Gao et al, 2024; Gupta, et al, 2024].

## Small Language Models

SLM models are smaller in scale and scope, trained on domain-specific data, and designed to be more computationally efficient, cost-effective, and suitable for specific tasks or domains.

Key Differences between SLMs and LLMs [Shone, 2024] are: a) *Size and parameters* - SLMs have a smaller number of variables, typically ranging from a million to a billion rather than a trillion; b) *Computational requirements* **-** SLMs require less computational power and memory, hence faster to train and deploy; c) *Training data* **-** SLMs are trained on smaller, more focused datasets, allowing for specialization in specific tasks; and d) *Performance* **-** SLMs can perform well on specific, well-defined tasks.

SLMs are well-suited for scenarios where resources are limited, such as on-device AI, edge computing, and mobile applications.

## Large Language Models

LLMs, as discussed, are trained with vast amounts of data and context, and excel at processing, understanding, and generating human language. Deep learning techniques, specifically Transformer models, are used to learn relationships.

Basic LLMs were un-optimized, with current LLMs built using foundation models optimized using prompt engineering.

## Prompt Engineering

Prompt Engineering is the process of designing and refining GenAI prompts - basically instructions, questions, or guidelines - that guide LLM responses.

LLM optimization is the process of refining and enhancing the performance and efficiency of large language models. This includes improving computational efficiency, text generation accuracy and how biases are handled, and even reducing the environmental impact of training and deploying these models.

Prominent optimization technologies include:

* *Zero/Few shot learning* - zero-shot learning involves a model recognizing new categories or classes it hasn't seen during training. Few-shot learning, on the other hand, allows a model to learn from a small number of examples of the new classes [Brown et al, 2025].
* *Chain-of-Thought Prompting (CoT) -* CoT prompting simulates a form of human-like reasoning by encouraging LLMs to break down complex tasks into logical steps. Instead of jumping directly to a final answer, the model is nudged to articulate intermediate reasoning, thereby improving the interpretability and quality of its outputs [Wei et al, 2025].
* *Retrieval Augmentation Generation (RAG*) - RAG combines LLM outputs with information retrieved from trusted external sources. The LLM first generates a response or interpretation of a query, after which a retrieval component sources additional data or facts to refine the output. This hybrid approach mitigates the limitations of purely generative models by grounding responses in verifiable evidence, thereby reducing errors and hallucinations [Lewis et al, 2025].

### Properties

Table 3 summarizes the application properties of LLMs: Small Language Models, Classic (unoptimized) LLMs, and LLMs optimized using techniques such as prompt engineering and RAG.

|  |  |  |  |
| --- | --- | --- | --- |
| **Model Type** | **Properties (Typical Characteristics)** | **Applications (Use Cases)** | **Strengths** |
| **Small Language Models (SLMs)** | Smaller scale, domain-focused datasets; specialized tasks; low parameter count (millions) | On-device AI, edge computing (e.g. mobile apps, IoT); specific domain tasks | Low computational requirements (faster, lightweight); cost-effective for narrow tasks |
| **Large Language Models (LLMs)** | General-purpose models trained on trillions of tokens; very high parameter count (billions–trillions) | General-purpose NLP applications across domains (chatbots, translation, summarization, etc.) | High versatility and state-of-the-art performance; can serve as foundation models adaptable to many tasks |
| **Optimized LLMs** | General-purpose LLMs enhanced with specialized techniques (prompt engineering, fine-tuning, retrieval, etc.) | All tasks handled by base LLMs, but with improved accuracy, relevance, or efficiency (e.g. complex reasoning with CoT, up-to-date info with RAG) | Enhanced performance on complex or specialized tasks (through CoT, RAG, etc.); often avoids need for model retraining by leveraging prompt or plugin strategies |

Table 3: Traditional LLMs’ properties

# Multi-model LLMs

Whereas a single (traditional) LLM relies on one comprehensive model for a range of tasks, multiple-model LLMs rely on the collaboration of several specialized models. The multi-model approach is meant to achieve a greater level of accuracy for a set of specific tasks. Examples in this category include Mixture of Experts, Multimodal LLMs and Agentic AI.

Multi-model LLMsare composite models where multiple specialized models (experts) work together:

* *Mixture of Experts (MOE) -* MoE is a machine learning technique where multiple specialized models (experts) work together, with a gating network selecting the best expert for each input [Cai et al, 2024].
* *Multimodal LLMs (MLLMs) -* MLLMs are AI systems that understand and generate various types of data, but unlike traditional LLMs that primarily focus on text, MLLMs expand this capability by integrating multiple modalities, enabling them to perform more complex and versatile tasks [Song et al, 2023].
* *Agentic AI -* are mostly LLM system composed of “agents” that can make decisions and act autonomously to achieve specific goals. These agents, unlike traditional AI systems, are designed to pursue complex goals and execute tasks with minimal human supervision. They can understand and interpret complex contexts, engage in nuanced decision-making, and adapt their actions based on changing conditions [Plaat et al, 2025].

We can also include:

* *Modular AI Architectures -* these architectures break down Agentic AI systems into discrete, interchangeable modules specialized for particular functions, such as reasoning, retrieval, or language generation. Modular designs enable flexible composition, easier maintenance, and improved interpretability by isolating capabilities into reusable components [Choudhary, 2025].

## Mixture of Experts (MoE)

MoE is an LLM technique where multiple specialized models (experts) work together, with a gating network selecting the best expert for each input. Thes “experts” are individual neural networks that are trained to specialize in different aspects of the task, such as specific linguistic patterns, syntactic structures, or semantic understanding. A “gating network” acts as a router, deciding which experts to activate for a given input. The gating network uses a learned gating function (e.g. neural network) to assign probabilities to each expert, selecting the top-k most likely experts. With MoE models, “sparsity” is a key characteristic since only a subset of experts is active for any given input. This means MoE models can handle a large number of parameters being computationally efficient during inference.

MoE works as follows [Cai et al, 2025]: a) *Input* - an input sentence or paragraph is fed into the MoE model; b) *Routing* - the gating network analyzes the input and assigns a weight to each “expert” based on its relevance; c) *Activation* - the “experts” with the highest weights are activated with their sub-networks processing the input; and d) *Output* .- the outputs from the activated experts are combined (often using a weighted average based on the gating network's weights) to produce the final output of the MoE model.

## Multimodal LLMs (MLLM)

MLLMs are AI systems that understand and generate various types of data, but unlike traditional LLMs that primarily focus on text, MLLMs expand this capability of traditional LLMs by integrating multiple modalities, enabling them to perform more complex and versatile tasks. This means that MLLMs can interpret a variety of data types, including: a) *sensory data* - Information from motion sensors; b) *GPS data* – information from GPS and other tracking devices; c) *3D data* - spatial data and 3D representations; d) *Structured data* – information from spreadsheets or databases requiring numerical or categorical analysis; and e) *Mixed documents* – data such as Webpages or multimedia documents that combine text, code, images, etc.

## Agentic AI

Agentiic AI are LLMs composed of “agents” that can make decisions and act autonomously to achieve specific goals. These agents, unlike traditional LLM systems, execute tasks independently and are designed to pursue complex goals and execute tasks with minimal human supervision. They can understand and interpret complex contexts, engage in nuanced decision-making, and adapt their actions based on changing conditions.

The key Agentic AI concepts [Plaat et al, 2025] are: a) *Autonomy -* Agentic AI systems are capable of operating independently and making decisions without constant human guidance; b) *Goal-oriented -* they are designed to pursue specific objectives and can adapt their actions to achieve those goals; c) *Reasoning/Planning -* Agentic AI uses sophisticated reasoning and iterative planning to solve complex, multi-step problems; d) *Adaptability -* systems learn from their experiences and adjust their behavior based on feedback and changing conditions; and e) *Integration -* Agentic AI often combines different AI techniques, such as large language models (LLMs), machine learning, and reinforcement learning.

### Properties

In Table 4 we attempt to summarize the properties of multi-model LLMs.

|  |  |  |  |
| --- | --- | --- | --- |
| **Model Type** | **Properties (Typical Characteristics)** | **Applications (Use Cases)** | **Strengths** |
| Mixture-of-Experts (MoE) | Route each input token to a sparse subset of specialised “expert” sub-networks chosen by a learned gating router. | • Sparse activation (only ~1–2 experts per token) • Gating network for dynamic routing • Parameters scale ≫ compute cost. | Massive parameter counts with reasonable inference cost; lets you train domain-specialised experts in one model. |
| Multimodal LLMs (MLLMs) | Fuse textual and non-textual information (images, audio, video, sensor, code, …) in a single model that understands and generates across modalities. | • Modality-specific encoders feeding a shared transformer • Cross-attention for aligning modalities • Large unified embedding space. | Creative & analytical tasks needing joint reasoning: visual Q&A, data-rich chatbots, robo-control. |
| Agentic AI | Compose one or more LLMs into autonomous agents able to plan, tool-call and iterate toward a goal with minimal human oversight. | • Planner → Executor loops • Memory / scratch-pads • Tool-calling / code-execution • Often combines RAG + RL. | Long-horizon tasks: multi-step research, coding, workflow automation, test-case generation. |

Table 4: Multi-model LLM properties

# Large Deep Models

We introduce the term large “deep” models for LLMs that use NLP combined with specialist capabilities and are trained to apply structured techniques when responding to prompts.

* *Large Reasoning Models (LRM)* – LRM use NLP combined with reasoning capabilities and are trained to apply structured reasoning techniques when responding to prompts [Patil & Jadon, 2025].
* *Large Action Models (LAM)* – LAM goes beyond generating text, like a LLM, by understanding and executing actions based on user input and context [Wang et al, 2024a].
* *Large Context* Models – large context models are a specific type of LLM designed to handle significantly larger amounts of text within their processing window (context window) [Martineau, 2024; Saeed, 2024].
* *Large Concept Models (LCM)* – LCM language models process natural language at the concept level rather than analyzing individual words [Duquenne et al, 2023].

We might also include:

* *Large Cognition Models*– use LLMs as in‑silico test‑beds for human‑like reasoning and other higher‑level mental abilities [Niu et al, 2024; Qu et al., 2024].
* *Self-Reflective/Introspective LLMs* – These models incorporate mechanisms to evaluate, critique, or refine their own outputs before finalizing a response. Through self-assessment, often via secondary networks or iterative reasoning loops, introspective models improve accuracy, consistency, and safety by reducing errors and hallucinations [Cappelen & Dever, 2025 ].

## Large Reasoning Models (LRM)

LRMs use NLP combined with reasoning capabilities and are trained to apply structured reasoning techniques when responding to prompts. LRMs are trained differently to LLMs [Patil, 2025]. They use: a) *Enriched training data* - not just language patterns but also examples designed to teach reasoning such as real-world scenarios with clear outcomes; b) *Reinforcement* *learning* – used to reward for correct or logically consistent answers and penalized for incorrect ones; c) *Human* *feedback* – reinforcement learning combined with human feedback (RLHF) to guide and refine the model’s outputs, helping it learn nuanced reasoning strategies that align with domain expertise; and d) *Prompt engineering* – CoT deployed to guide the model through multi-step tasks or layered questions.

### LRM types of reasoning

The four main types of “reasoning” commonly used in LRMs are: a) *Deductive reasoning* - also known as “top-down reasoning”, applies general rules to specific cases requiring strict adherence to established rules or facts; b) *Inductive reasoning* - draws general conclusions from specific observations in training data, supporting probabilistic predictions rather than guaranteed outcomes, c) *Abductive reasoning* - involves inferring most likely explanation based on incomplete or uncertain data; and d) *Analogical reasoning* - involves identifying similarities between different situations or datasets and applying insights from one context to another., to recognize relational patterns across examples and transfer learning.

## Large Action Models

LAMs are a type of LLM system that goes beyond generating text, by understanding and executing actions based on user input and context. Think of it as a "can-do" cousin to the "talk-only" LLM. LAMs can interact with various systems and interfaces to perform tasks such as making reservations, booking appointments, or completing forms.

The key characteristics of LAMs [Wang et al, 2024a] are: a) *Action-oriented* **-** LAMs are designed to perform actions in software or physical environments rather than process text; b) *Understanding**intent* -LAMs analyze user input (e.g. text, voice, etc.) and interpret the underlying goal or intention behind the request; c) *Task decomposition* **-** breaks down complex tasks into sub-tasks; d) *Execution* **-** LAMs execute actions by interacting with other systems, applications, or physical devices; e) *Adaptability*- LAMs adapt their actions based on feedback, learning and improving their performance over time; f) *LLM* *integration* **-** LAMs build upon traditional LLMs, leveraging their capabilities, adding action-oriented components; and g) *Hybrid approach*– LAMS can combine neural networks with symbolic reasoning or planning algorithms to enhance to understand context and execute tasks effectively.

## Large Context/Contextual Models

Large context models and related large contextual models address different aspects of context: a) *Context models* - provide LLMs with more information to process at once, improving their ability to understand and respond to complex queries; b) *Contextual models* - focus on how information is related within a specific context to retrieve relevant information from a larger (RAG) knowledge base [Martineau, 2024; Saeed, 2024].

The evolution of context windows in LLMs represents one of the most significant advances in AI capability and utility. The context window (or “context length”) of a large language model (LLM) is the amount of text, in tokens, that the model can consider or “remember” at any one time. A larger context window enables an AI model to process longer inputs and incorporate a greater amount of information into each output.

## Large Concept Models (LCM)

META’s LCMs are models that process language at the concept level, analyzing larger sections of text to extract the underlying ideas, rather than analyzing individual words. Instead of relying on language-specific patterns, LCMs store meaning making them adaptable for tasks like multilingual summarization, translation, and cross-format content generation. They perform autoregressive sentence prediction and are built on the SONAR embedding space, which allows them to process text in over 200 languages and speech in 76 [Duquenne et al, 2023].

LCMs follow a three-step pipeline:

* *Concept Encoder (SONAR)* - input (text or speech) is broken into sentences, a semantic embedding space. Each sentence is turned into a semantic “concept embedding” space using a tool called SONAR (which works in 200 languages and supports text + speech).
* *LCM core* - performs reasoning and prediction using a Transformer model, which reasons over the concept sequence to generate output concepts. The concept embeddings are processed by the LCM to perform a task, like summarizing or translating.
* *Concept Decoder (SONAR)* - translates the model’s output (embeddings) back into human-readable natural language (or speech) output.

The key characteristics of LCMs [Bhatnagar & Casals, 2025]: a) *Multiformat* - multilingual and multiformat flexibility working in 200+ languages and supports text and speech, without extra training; b) *Generalization* - generalizing to new tasks by using a language-independent system, allowing it to handle new languages and tasks without extra training; c) *Coherence* - processing full sentences, keeping responses clearer and more structured over long text; and d) *Efficiency* – efficient in handling context by using compact sentence representations, making it easier to process long documents efficiently.

### Properties

In Table 5 we again summarize the properties of large deep models.

|  |  |  |  |
| --- | --- | --- | --- |
| **Model Type** | **Properties (Typical Characteristics)** | **Applications (Use Cases)** | **Strengths** |
| **Large Reasoning Models (LRM)** | Train the LLM to explicitly plan, justify or prove its answers rather than produce fluent text only. | Think-Aloud GPT 4, Gemini 1.5 “Reasoning Engine”, DeepMind AlphaGeometry, Google PaLM-E reasoning tasks. | Cost of multi-turn deliberation, reward hacking (spurious chains), difficulty of automatic grading of reasoning quality. |
| **Large Action Models (LAM)** | Couple an LLM with action execution layers so the model can read intent → plan → act inside software or the physical world. | Devin (“AI software engineer”), Adept Fuyu-8B, OpenAI GPT-4o + Actions, AutoGPT / CrewAI agent frameworks, robotics model PaLM-E. | Safety & permissioning, grounding to real-world state, cascading errors from mis-planned steps, latency/cost of tool chains. |
| **Large Context / Contextual Models** | Scale the effective working memory of the model by (a) vastly enlarging the native context window and / or (b) tightly integrating Retrieval-Augmented Generation (RAG). | Anthropic Claude 3 Opus (1 M ctx), Google Gemini 1.5 Pro, Microsoft phi-3-vision-128k, wrapper libs LlamaIndex, Haystack. | Quadratic attention cost, long-range forgetting, citation faithfulness, retrieval noise & recency drift. |
| **Large Concept Models (LCM)** | Move from word-level tokens to sentence/idea-level “concept embeddings” that are language-agnostic & multimodal. | Meta AI LCM-4, SONAR-2 (text + speech in 200 languages), “Stories over Speech” demo. | Coarse granularity may miss nuance, aligning concept spaces across modalities, limited public training data. |

Table 5: Large Deep Models

# Domain-specific LLMs

Domain-specific LLMs are the most rapidly developing category of LLMs. These and related Transformer models are tailored for specific domains, comprising a) specific data types such as numeric or video; b) an industry sector, such as finance, law, medicine, or education; and c) multiple domain models:

* *Large Quantitative Models (LQM)* – LQM are specifically to process, analyze, and generate quantitative data rather than natural language [Bishesh, 2025].
* *Vibe Coding LLMs* - an approach to programming and application development that relies heavily on LLMs for code generation, refinement, and debugging [Kumar, 2025].
* *Financial LLMs* – Finance/Quant Models specialized in processing financial data, market analysis, and quantitative reasoning. They are trained on vast datasets from financial markets, news, and economic reports, enabling applications such as trading strategies, risk management, portfolio optimization, and financial forecasting (BloombergGPT, FinGPT, LQMs) [Nie et al, 2024].
* *Legal Models –* Legal models are designed to understand, interpret, and generate legal language, documents, and case law. By fine-tuning on legal corpora, these models assist with contract analysis, legal research, compliance monitoring, and automated drafting, improving efficiency and reducing errors in legal workflows (e.g. Harvey, CaseLawBERT) [Laia et al, 2024].
* *Educational LLMs* – Educational/Pedagogical models are fine-tuned to provide personalized tutoring, explanations, and assessments. They adapt to learners’ skill levels, offer interactive feedback, and support diverse subjects, enhancing learning experiences through AI-driven teaching assistance [Xua et al, 2024].

In addition, we can also include these specialist LLMs:

* *Industrial LLMs* – Industrial/Supply Chain/Manufacturing models focus on optimizing industrial operations, supply chain logistics, and manufacturing processes. By analyzing sensor data, operational logs, and demand forecasts, they facilitate predictive maintenance, inventory management, and process automation [Raza et al, 2025].
* *Biochemical LLMs*– Bio/Chem models are trained on biochemical or molecular data rather than natural language. They predict molecular properties, protein structures, or chemical reactions. Such models are now pivotal in drug discovery, genomics, and materials science, where textual LLMs are insufficient for structured scientific data (e.g., MolBERT, AlphaFold) [Zhang et al, 2024].
* *Scientific LLMs* – Scientific/Research models are tailored for scientific and academic domains, trained on research papers, technical articles, and domain-specific terminology. They support literature summarization, hypothesis generation, and data extraction, accelerating research discovery and knowledge dissemination. (e.g. Galactica, SciBERT) [Garcia et al, 2025].
* *Video-Language LLMs* – Video models are designed to process and generate content from temporal visual data such as videos, combining language understanding with multi-frame visual reasoning. Unlike static Vision-Language Models (VLMs), Video-Language LLMs model temporal dynamics, enabling applications like video captioning, summarization, question answering, and multimodal action understanding. They are trained on large-scale video-text datasets and may use temporal attention, 3D convolution, or spatiotemporal transformers to capture frame-to-frame dependencies. (e.g. VideoLLaMA 2, VideoChatGPT, Gemini 1.5/2.0, OpenFlamingo-V) [Tang et al., 2023].
* *Long-Document LLMs* – Long-document models are specialized to process extensive texts such as books, research papers, contracts, or technical documentation. These models incorporate architectural changes (e.g., sparse attention or memory modules) to handle long-range dependencies and preserve coherence across thousands of tokens. Examples include BigBird, Longformer, Hierarchical Attention Transformers (HATs),andBlockBERT [Principe et al, 2025 ].
* *Robotic LLMs* – Robotic/Embodied LLMs connect language understanding with physical actions in real or virtual environments. These models interpret instructions and execute them through robotic systems or simulated avatars, combining perception, planning, and motor control. They are key to advancements in robotics, autonomous agents, and virtual assistants [De Ridder, 2024; Wang et al, 2025].
* *Edge Device LLMs* – Edge/On-Device models are optimized to run efficiently on local or edge devices such as smartphones, IoT gadgets, or embedded systems. By reducing computational and memory requirements, edge LLMs provide faster responses, improved privacy, and offline functionality without relying on cloud infrastructure [Jang & Morabito, 2025].
* *Multi-domain LLMs* – These foundation models are trained from scratch or adapted on large, domain-focused datasets (e.g., biomedical literature, legal case law, or financial documents). They serve as base models for downstream tasks and applications in highly specialized fields, where general-purpose models may underperform [Chen et al., 2025a].
* *Adult Entertainment LLMs* - fueled by advances in AI, such as LLMs and Avatars, the adult entertainment industry is also undergoing a significant transformation [Lapointe et al, 2025].

## Large Quantitative Models (LQM)

LQMs are designed specifically to process, analyze, and generate quantitative data rather than natural language. Unlike Traditional LLMs, which excel in language-related tasks, LQMs focus on handling large-scale numerical datasets, performing complex calculations, statistical analysis, predictive modeling, and optimization tasks. These models are envisioned as the quantitative counterpart to traditional LLMs, providing data-driven insights and solutions to numerical problems across various industries.

LQMs use deep learning frameworks for time-series data using Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTM) networks, combined with custom layers for advanced statistical models using Bayesian Networks or Regression Models.

The key characteristics of LQMs [Bishesh, 2025] are: a) *Reason style* - Deterministic or stochastic mathematical reasoning; b) *Interpretability* - typically more interpretable, especially in structured scientific models; c) *Architecture* - Mathematical or physics-based models, often combined with ML techniques; d) *Training* - scientific theories, simulations, and specialized datasets; e) *Real-time usage* - running simulations, forecasting, and optimizing systems; and f) *Error* *handling* -with LQMs errors tend to be mathematical or computational, often easier to debug.

## Vibe Coding LLMs

Vibe coding involves using LLMs to generate, refine, and debug code, with developers focusing on the overall "vibe" or direction of the project rather than crafting every line of code. In “vibe coding”, developers focus on describing the desired functionality in natural language (“the vibe”), rather than writing exact code, and the LLM generates and iteratively refines the code. This contrasts with traditional prompt engineering for code (which might require very precise prompts); vibe coding is more about guiding the AI’s outputs at a high level and letting the model fill in the details.

The key characteristics of vibe coding LLMs [Kumar, 2025]: a) *Natural language* - vibe coding describe their desired functionality in human language; b) *Refinement* - the developer reviews and refines the generated code iteratively, providing feedback in natural language.; c) *Guidance* - focus the desired “creative” outcome rather than granular code details; d) *Prototyping* - significantly speed up the development of early-stage prototypes; and e) *Accessibility* - make software development more accessible to individuals without extensive programming and technical backgrounds. An interesting development of vibe coding is the Darwin Gogel Machine [Zhang et al, 2025b] a self-improving system that iteratively modifies its own code.

## Legal LLMs

Legal Services and the Judiciary are being “revolutionized” by AI legal assistants covering contract generation, legal advice, online dispute resolution (ODR) and Internet Courts [Barnett et al, 2023]. Legal technology (i.e. LegalTech for lawyers, LawTech for clients, JudicialTech for the Judiciary) encompasses software and technology solutions that support and enhance legal services. It aims to improve efficiency, accuracy, and accessibility within the legal industry by automating tasks, streamlining processes, and providing new ways to deliver legal services. Legal tech can include everything from basic tools like e-signature platforms to more advanced applications like AI-powered legal research and contract analysis [Laia et al, 2024].

The key characteristics of Legal technology and LLM assistants [Laia et al, 2024] are: a) *Automation* **-** Legal technology can automate repetitive tasks, such as document drafting, data entry, and legal research, freeing up legal professionals to focus on more complex tasks; b) *LegalTech software* **-** involves specialized software and online platforms for legal professionals that help with tasks like document management, legal research, contract generation/analysis, and client communication; c) *LawTech software* – client (LLM) software providing legal advice, draft contracts, case routing, and estimation of litigation success. Making legal services more accessible to individuals and businesses; and d) *JudicialTech software* – judicial platforms for judicial stakeholders, so-called smart/Internet courts and online dispute resolution, and LLM assistants for Judges and court officials.

By leveraging legal technology, legal professionals, clients and the Judiciary can improve the speed and accuracy of their work, potentially reduce costs and errors, and improve access to justice.

## Educational LLMs

Education is probably the most active sector for LLM assistants [Xua et al, 2024].

Figure 2 [Chu et al, 2025] illustrates LLM educational activities divided into pedagogical agents and domain-specific agents.

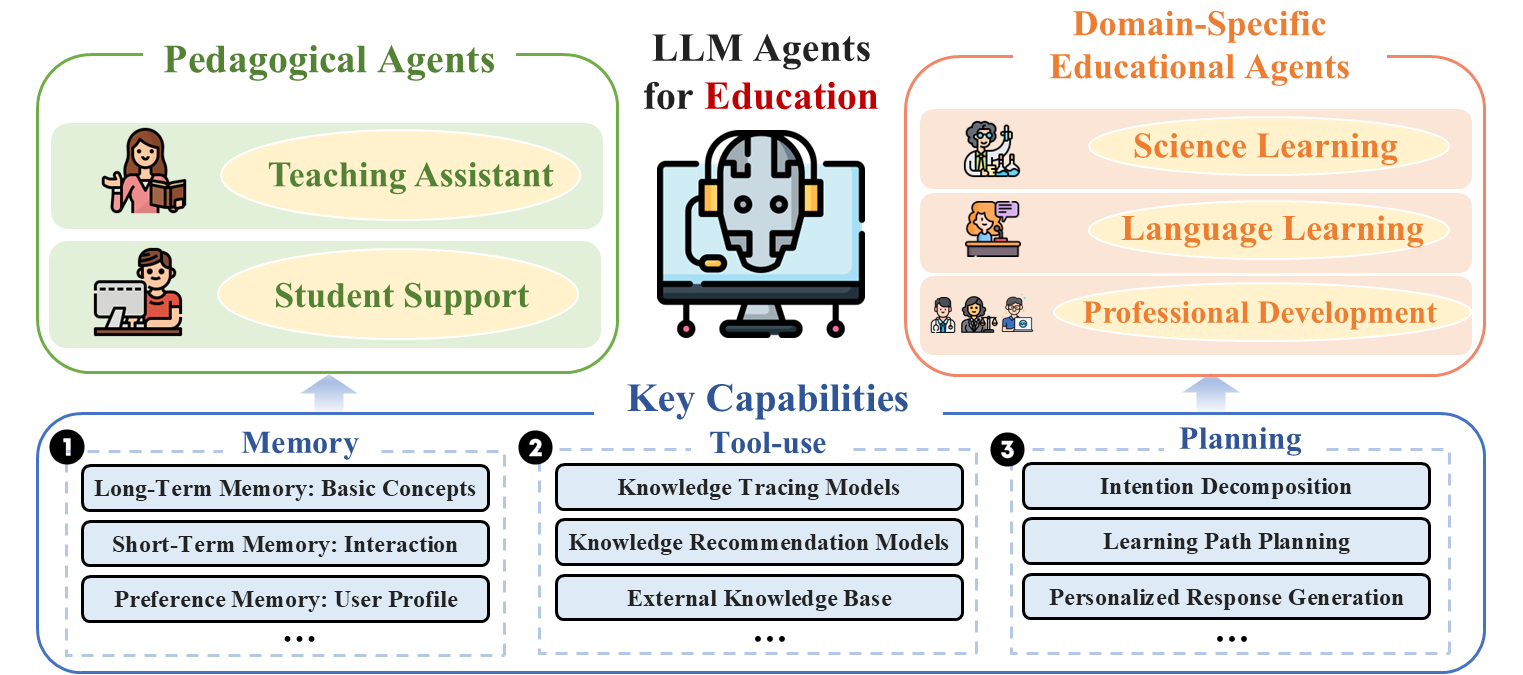


Figure 2: LLM Pedagogical Agents for Education [Chu et al, 2025 ]

Benefits of LLMs in education for teachers and students include a) *Personalization* - personalized learning content; b) *Automation* - automating tasks like lesson planning and grading; c) *Feedback* - providing quick help and feedback to students; and d) *Innovation* - cutting-edge learning resources and technology, such as interactive simulations, and games etc.

### Properties

Table 6 summarizes the properties of domain-specific LLMs.

|  |  |  |  |
| --- | --- | --- | --- |
| **Model Type** | **Properties (Typical Characteristics)** | **Applications (Use Cases)** | **Strengths** |
| **Large Quantitative Models (LQM)** | Treat numerical / time-series input as first-class data and perform deterministic or stochastic mathematical reasoning. | CNN / RNN / LSTM blocks, Bayesian layers, physics-informed modules. | Forecasting, simulations, optimisation, ops research. |
| **Vibe-Coding LLMs** | Developers describe the vibe in NL; the model generates, refines and debugs the code iteratively. | NL-to-code generation loops, RLHF with code quality rewards. | Ease and speed of development |
| **Financial LLMs** | Fine-tuned on market data, news and filings for quantitative reasoning over finance. | Domain RAG, numerical adapters, long-context windows. | Trading signals, risk management, forecasting, research Q&A. |
| **Legal LLMs** | Understand statutes, contracts and case-law; draft and analyse legal text. | Long-doc attention, citation retrieval, jurisdiction filters. | Contract review, legal research, compliance checks. |
| **Educational LLMs** | Personalised tutoring and teacher co-pilots that adapt to learner level. | Student-profile memory, interactive feedback loops. | Tailored explanations, automated grading, lesson generation. |

Table 6: Domain-specific Models

# Optimizing LLMs

We introduce the term “optimizing LLMs” for models incorporating specific (i.e. non-prompting) mechanism to optimize an LLM*:*

* *Quantization LLMs* - Quantization reduces model size by storing weights with lower‑precision formats such as INT8 rather than FP32 or FP16 [Dettmers et al., 2022].
* *Privacy-Preserving LLMs* – Privacy-Preserving/Differentially Privacy LLMs incorporate mechanisms that limit the ability to memorize or leak sensitive training data. Differential privacy ensures that no individual data point has a measurable influence on the model’s outputs, protecting user confidentiality and enabling compliance with data protection regulations such as GDPR and HIPAA [Zhao & Song, 2024].
* *Adversarial-Robust LLMs* – These models are trained or adapted to resist adversarial prompts and malicious input manipulation, such as prompt injection or jailbreak attacks. Techniques include robust training, defensive fine-tuning, and input sanitization. Such models are essential for deploying LLMs in security-critical environments, including legal, military, and enterprise settings [Xhonneux et al., 2024].
* *Thinking LLMs* - aimed at improving general instruction following in large language models (LLMs) by explicitly incorporating internal “thought” processes before generating responses [Wu et al, 2024].

Again, these optimizing LLMs might be included:

* *Watermarked Models* – Watermarked/Steganographic models incorporate hidden signals or watermarks into generated outputs to verify authenticity or trace origins. Steganographic techniques embed imperceptible information within text or data, supporting intellectual property protection, content provenance, and combating misinformation [Kirchenbauer et al., 2023].
* *Alignment-Tuned Models* – Alignment-Tuned (RLHF, Constitutional AI) models are fine-tuned to align more closely with human values, ethical principles, or domain-specific guidelines. Reinforcement Learning from Human Feedback (RLHF) and Constitutional AI approaches help ensure that the model’s output remain safe, helpful, and contextually appropriate, particularly in sensitive or high-stakes applications [Ouyang et al., 2022].
* *Instruction-Tuned LLMs* – Instruction-Tuned/Role-Conditioned LLMs are further fine-tuned to follow explicit human instructions or behave according to defined roles (e.g., financial advisor, legal analyst). Instruction tuning improves the alignment of model outputs with user intent, enhancing reliability, safety, and usability in real-world scenarios [Wang et al., 2023].
* *Tool-Using LLMs (Function Calling, Agents)* – These models extend traditional LLM capabilities by invoking external tools, APIs, or functions during generation. By dynamically calling specialized functions or software components, tool-using LLMs can perform tasks beyond pure language generation, such as calculations, database queries, or code execution, enabling more practical and accurate outputs [Chen et al, 2025b].
* *Program-Synthesis Models* – Program-synthesis models generate executable code or formal programs from natural language descriptions or specifications. These models combine language understanding with logical reasoning and often produce stepwise solutions, supporting software development, automation, and complex problem solving [Nijkamp et al., 2022].

## Quantization LLMs

Quantization is a technique utilized within large language models (LLMs) to convert weights and activation values of high precision data, usually 32-bit floating point (FP32) or 16-bit floating point (FP16), to a lower-precision data, like 8-bit integer (INT8). In essence, the quantization process involved the mapping of weights stored in high-precision values to lower-precision data types. For LLMs, weights often have distributions centered around zero, making symmetric quantization a common choice. However, activations, particularly after ReLU or GeLU functions, can be strictly non-negative or have highly asymmetric distributions.

The Two Types of LLM Quantization are a) *Post-Training Quantization (PTQ)* - this refers to techniques that quantize an LLM after it has already been trained; and b) *Quantization-Aware Training (QAT)*: this refers to methods of fine-tuning on data with quantization in mind.

The key characteristics of Quantization LLMs are [Clark, 2024]: a) *Range mapping* - the mapping of weights stored in high-precision values to lower-precision data types; b) *Granularity* - defines how broadly or narrowly we apply quantization parameters like scale and zero point across a model; c) *Calibration* - involves running inference on a representative dataset to optimize the quantization parameters and minimize quantization error.

Quantization is an integral part of the LLM landscape. By compressing the size of language models, quantization techniques (e.g. QLoRA, PRILoRA, GTPQ, GGML/GGUF, AWQ) help to increase the adoption of LLMs [Lang et al, 2024].

## Privacy-preserving LLMs

LLMs often require extensive datasets for pretraining and fine-tuning, which may include sensitive, proprietary, or personally identifiable information (PII). These datasets, coupled with the ability of the models to memorize and reproduce training data, expose risks such as data leakage, membership inference, and model inversion attacks [Song & Zhao, 2024].

To address these challenges, researchers have developed a range of privacy preserving mechanisms tailored to LLMs. Techniques such as differential privacy, federated learning, cryptographic protocols, and trusted execution environments have demonstrated varying degrees of success in mitigating privacy risks while retaining usability. These approaches are particularly crucial in privacy-sensitive domains, including healthcare and finance, where breaches of data confidentiality can have severe consequences.

## Adversarial-Robust LLMs

Continuous adversarial attacks have recently demonstrated higher success rates and significantly faster computation times than their discrete counterparts in LLMs. Adversarial training, which involves online augmenting the training data of a neural network with adversarial attacks, has consistently proven to enhance robustness against adversaries [Zou et al, 2023; Xhonneux et al., 2024].

### Properties

Table 7 summarizes the properties of what we refer to as Optimizing LLMs, that provide optimization mechanisms beyond prompt engineering.

|  |  |  |  |
| --- | --- | --- | --- |
| **Approach** | **Properties (Typical Characteristics)** | **Applications** | **Strengths** |
| Quantization LLMs | Compress the model by storing weights/activations in lower-precision formats (INT8, INT4) instead of FP32/FP16. | Post-Training Quantisation (PTQ), Quantisation-Aware Training (QAT), per-channel scales, zero-point calibration. | Running large models on laptops, edge devices or single GPUs; cost-effective serving. |
| Privacy-Preserving LLMs | Inject differential-privacy noise or secure computation so no single datum measurably affects outputs. | DP-SGD training, clipping & noise addition, secure aggregation. | Healthcare, finance, any setting with sensitive data & compliance needs. |
| Adversarial-Robust LLMs | Harden models against jailbreaks, prompt injection and other adversarial inputs. | Robust / adversarial fine-tuning, input sanitisation, rule-based guards. | Safety-critical, legal or military chatbots, secure RAG pipelines. |

Table 7: Optimizing LLM properties

# Underlying Network Interpretability

Whereas prompt engineering optimizes the ‘prompts’, newer techniques attempt to decode and manipulate the underlying neural network architecture. Examples include Mechanistic Interpretability [Nanda et al, 2023] and Representation Engineering [Wehner, 2024]. These interpretability techniques focus on understanding the internal reasoning processes of neural networks, aiming to reveal how they process information and make predictions. We also highlight the opportunity of these techniques to adjust specific activations functions using control vectors as a method to increase interpretability in LLMs.

In summary, network interpretability addresses understanding the internal reasoning processes of trained neural networks. Several methods are being developed to improve the interpretability of Transformers, focusing on techniques like attention visualization, attribution methods, and modifications to the model architecture itself:

* *Mechanistic Interpretability (MI)* - seeks to understand the internal reasoning processes of trained neural networks and gain insight into how and why they produce the outputs that they do [Nanda et al., 2023; Rai et al, 2024].
* *Representation Engineering (RepE)* – RepE or Activation Engineering seeks to understand and control of AI by observing and manipulating the internal representations - weights or activations - that AI uses to understand and process information [Wehner et al, 2024].
* *Sparse Models* – Sparse models (e.g. Sparse Autoencoders, Mixture-of-Sparse Experts) employ techniques that activate only a subset of the network's parameters for each input, reducing computational load and improving efficiency. Sparse autoencoders enforce sparsity constraints during training, while mixture-of-sparse experts dynamically select relevant experts, balancing performance with resource usage (i.e. incorporates a sparsity constraint during training to encourage the network to learn a representation where only a small number of hidden units are active at any given time) [Cunningham et al, 2023].

We also include for completeness:

* *Memory-Augmented Models* – these models integrate explicit memory components that allow them to store and retrieve information across long sequences or multiple interactions. By augmenting neural networks with external or internal memory mechanisms, these models can remember context beyond typical transformer limits, improving performance in complex or multi-turn tasks [Wang et al., 2023].
* *Continual Learning Models* – Continual learning models are designed to learn incrementally from a stream of data without forgetting previously acquired knowledge (mitigating catastrophic forgetting). This enables them to adapt to evolving domains, new tasks, or changing environments over time while retaining past expertise [de Masson d’Autume et al., 2019].
* *Meta-Learning Models* – meta-learning models, also known as “learning to learn,” acquire the ability to quickly adapt to new tasks with minimal training data. By learning generalizable learning strategies during pretraining, these models improve sample efficiency and flexibility across diverse domains and tasks [Sinha et al., 2024].
* *Federated/Decentralized LLMs* – Federated learning enables training LLMs across multiple decentralized devices or organizations without sharing raw data. This privacy-preserving approach allows models to learn from diverse, distributed datasets while maintaining data security and compliance with regulatory standards [Zhang et al., 2023].
* *Neurosymbolic Models (NeSy)* – Neurosymbolic models combine neural network learning with symbolic reasoning systems. This hybrid approach integrates the pattern recognition strengths of neural models with the structure and logic of symbolic systems, enabling better reasoning, rule-following, and interpretability in tasks that require consistency, such as legal reasoning or theorem proving [DeLong et al., 2023].
* *Compressed Models* – Compression techniques (e.g. Quantized, Distilled, LoRA, etc.) reduce the size and complexity of the neural network to enable faster inference and lower hardware requirements. Approaches include quantization (reducing numerical precision), distillation (training smaller models to mimic larger ones), and parameter-efficient fine-tuning methods like Low-Rank Adaptation (LoRA) [Wang et al., 2024b].
* *Activation Steering* - with activation steering we can add a vector at a specific layer of the LLM during the generation of outputs and expect it to follow that behaviour more closely [Bayat et al, 2025].

## Mechanistic Interpretability

Mechanistic interpretability (MI) [**Nanda et al., 2023**] focuses on understanding the internal workings of machine learning models, particularly LLMs, by reverse-engineering their algorithms into human-understandable mechanisms [Lindsey, 2025]. It aims to go beyond simply observing inputs and outputs, striving to identify the specific components and processes within the underlying neural network that lead to particular behaviors. This approach seeks to reverse engineering neural networks by analyzing the learned weights to uncover human-interpretable algorithms, and learned weights to understand how information flows and computations are performed [Rai et al, 2024]. The goal of mechanistic interpretability is to understand the internal workings of a neural network, typically by identifying the circuits within the model that are responsible for specific behaviors. Additional examples of this approach can be found in [Cammarata et al, 2023; Elhage et al, 2025]. Mechanistic interpretability is critical to improving trustworthiness in LLMs, this remains a critical topic when considering which LLM to utilise for different use cases, such as privacy and security.

At the heart of MI are two key concepts: “features” and “circuits”, pivotal in deciphering the operations of neural networks and understanding how information is processed. The key characteristics of MI: a) *Features* - refer to the specific properties or patterns that a neural network learns to recognize and process; b) *Circuits* - are groups of neurons or “functional units” within a neural network that work together to perform specific computations; and c) *Relationships* - features are the "what"—the information being processed—while circuits are the "how"—the pathways and computations that process this information.

## Representation Engineering

Representation Engineering (RepE) or Activation Engineering is a new paradigm for optimizing the behaviour of LLMs. It involves creating vectors of activations which cause desired changes to output text when added to the forward passes of a frozen LLM. RE is a set of methods to understand and control the behaviour of LLM neural networks. This is done by first identifying a linear direction in the activations that are related to a specific concept [Wehner, 2024], a type of behaviour, or a function, which we call the concept vector. During the forward pass, the similarity of activations to the concept vector can help to detect the presence or absence of this concept direction. Furthermore, the concept vector can be added to the activations during the forward pass to steer the behaviour of the LLM towards the concept direction.

The key representation characteristics of RepE [Wehner et al, 2025]: a) *Identification* - determine key concepts and find where they are encoded within the model's internal representations (activations or weights); b) *Manipulation* - measure or quantify the concept's representation and develop techniques to “steer” the model’s behavior; and c) *Model behavior* - use the manipulated representations to influence the model's output or behavior on new inputs

RepE offers a powerful way to understand and control complex AI systems by focusing on their internal representations, offering a more targeted and potentially more effective approach compared to traditional methods.

## Sparse Autoencoders Models

Sparse Autoencoders models (SAM) [Cunningham et al, 2023] incorporates a sparsity constraint during training to encourage the network to learn a representation where only a small number of hidden units are active at any given time. Autoencoders are neural networks trained to reconstruct their input. They consist of an encoder (which compresses the input) and a decoder (which reconstructs the input). They learn a compressed, lower-dimensional representation of the data, often used for dimensionality reduction or feature extraction.

The key characteristics of SAMs are: a) *Autoencoders* - neural networks trained to reconstruct their input, using an encoder and decoder; b) *Sparsity* - constraint forces the network to focus on a smaller set of important features, leading to a more efficient and interpretable representation; and c) *Efficiency* - by forcing neurons to represent specific features, sparse autoencoders can make the learned representations more interpretable, allowing for better understanding of the model's inner workings.

Sparse autoencoders are increasingly used to understand the inner workings of large language models (LLMs), helping to identify and interpret the features learned by these models.

### Properties

Lastly, we summarize the properties of the optimization techniques for manipulating the underlying neural network and associated weights.

|  |  |  |  |
| --- | --- | --- | --- |
| **Approach** | **Properties (Typical Characteristics)** | **Applications** | **Strengths** |
| **Mechanistic Interpretability (MI)** | Reverse-engineer an LLM into human-readable features and circuits that explain how specific behaviours arise. | Activation patching, path-patching, weight lenses, circuit tracing dashboards. TransformerLens toolkit; Circuits work by Nanda et al.2023, Cammarata et al 2023. | Trust-building audits, red-team analysis, safety verification, curriculum for model debugging. |
| **Representation/Activation Engineering (RepE)** | Identify linear concept vectors in activation space and add or subtract them at run-time to steer model outputs. | Concept-vector discovery, similarity scoring, forward-pass vector injection. RepE / Activation-Addition methods (Wehner 2024, 2025). | Fine-grained behaviour control, rapid customisation without weight updates, content filtering. |
| **Sparse Autoencoder Models (SAM)** | Train sparse autoencoders on model activations to obtain a set of disentangled interpretable latent features. | Encoder–decoder with sparsity constraint, feature selectivity metrics. Sparse-feature autoencoders for LLMs (Cunningham et al., 2023). | Discovering human-readable features, dimensionality reduction, systematic probing of large models. |

Table 8: Underlying Network Interpretability

# Conclusions - LLM Challenges

In summary, Large Language Models can be categorized in several ways, including by their **architecture** (e.g., autoregressive vs. encoder-decoder), their **access** type (proprietary vs. open-source), and their **specialization** (general-purpose vs. task-specific). We can also categorize LLMs by their **training** data/regime or their primary objectives (e.g., predictive text generation vs. instruction following).

Despite their power, LLM (Transformer) models face several key challenges. These include: (a) *Computational complexity and resource requirements* – training and deploying large models is extremely costly and slow, and energy consumption is a concern; (b) *Understanding long sequences and dependencies* – even with enhancements, models can struggle with very long-range dependencies or context lengths beyond their window (e.g., maintaining coherence in a long document); (c) *Interpretability and bias* – it remains difficult to interpret why a model produced a given output, and models can inadvertently amplify biases present in training data; (d) *Optimization challenges* – including fine-tuning for specific tasks without overfitting or losing generality, and ensuring efficient inference (latency) for real-time applications*.* Additional key areas of concern are aligning models with human values (preventing toxic or harmful outputs) and ensuring users can trust the model’s answers (e.g., distinguishing reliable responses from hallucinations).

The current debate in AI research includes “Thinking LLMs” [Wu et al, 2024] – approaches aimed at improving general AI instruction - by explicitly incorporating internal “thought processes” before generating responses. For example, *Thinking LLMs* generate a hidden layer of “thoughts” (an internal chain-of-thought) and refine it through a specialized training method called Thought Preference Optimization (TPO) **`** before producing a final answer. This means the LLM isn’t just generating an answer in one go; it is internally reasoning or simulating a thought process, which can lead to more coherent and accurate answers. TPO is a training framework introduced by researchers [Shojaee et al., 2025] to teach LLMs to “think before speaking,” optimizing the model’s internal reasoning via feedback on the final answer quality [Shojaee et al, 2025]. Early results from such “Thinking LLM” methods show promise across logic, mathematics, and even creative tasks, indicating a significant step towards more general and reliable AI systems.

In conclusion, the development of Large Language Models is a fast-moving field at the intersection of computer science, linguistics, and many application domains. Continued innovation in model architectures, training techniques, and alignment strategies will be critical for addressing current limitations. By systematically tracking these emerging LLM paradigms and understanding the strengths and weaknesses of each, we can better match LLM technologies to the applications that benefit from them – from finance to education – and deploy them in a way that is safe, trustworthy, and effective.

# ACKNOWLEDGEMENTS

The authors would like to thanks ...

# References

**Anthropic, 2025 –** Anthropic, Agentic Misalignment: How LLMs could be insider threats. Anthropic Research, [www.anthropic.com/research/agentic-misalignment](http://www.anthropic.com/research/agentic-misalignment)

**Barnett et al., 2023 –** Barnett, J., Treleaven, P., Lederer, F., Vermeys, N., Zeleznikow, J., JudicialTech Supporting Justice. SSRN, <https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4597917>.

**Bayat et al, 2025** - Bayat, R., Kalahroudi, A., Pezeshki, M., Chandar, S., Vincent, P., Steering Large Language Model Activations in Sparse Spaces. arXiv, <https://arxiv.org/abs/2503.00177>.

**Bhatnagar & Casals, 2025 –** Bhatnagar, A. & Casals, A. (2025). Large Concept Models: A Paradigm Shift in AI Reasoning. InfoQ. <https://www.infoq.com/articles/lcm-paradigm-shift-ai-reasoning/>.

**Bi et al., 2024 –** Bi, Z., Kai Han, K., Chuanjian Liu, C., Yehui Tang, Y., Yunhe Wang, Y., Forest of Thought: Scaling Test Time Compute for Enhancing LLM Reasoning. arXiv, <https://arxiv.org/html/2412.09078v1>.

**Bishesh, 2025 –** Bishesh, B., Large Quantitative Models. Aviso, <https://www.aviso.com/blog/large-quantitative-models>.

**Brown et al., 2025 –** Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D., Language models are few shot learners. Advances in Neural Information Processing Systems, 33, arXiv, <https://arxiv.org/abs/2005.14165>.

**Cai et al., 2025 –** Cai, W., Jiang, J., Wang, F., Tang, J., Kim, S., Huang, J., A Survey on Mixture of Experts in Large Language Models. arXiv, <https://arxiv.org/abs/2407.06204>.

**Cammarata et al., 2023 –** Cammarata, N., Carter, S., Goh, G., et al., Thread: Circuits. Distill. <https://distill.pub/2020/circuits>.

**Cappelen & Dever, 2025** - Cappelen, H., Dever, J., Introspective Machines: Are LLMs Better at Self-Reflection Than Humans? Wiley, <https://onlinelibrary.wiley.com/doi/10.1111/phpe.12201>

**Chen et al, 2025a -** Chen, H., Zhao, Z., Han, K., Zhu, G., Zhao, Y., Du, Y., Xu, W., Shi, Q., An overview of domain-specific foundation model: key technologies, applications and challenges. arXiv, <https://arxiv.org/abs/2409.04267>.

**Chen et al, 2025b** - Chen,Y-C., Hsu, P-C., Hsu, C-J., Shiu, D-S., Enhancing Function-Calling Capabilities in LLMs: Strategies for Prompt Formats, Data Integration, and Multilingual Translation. arXiv, <https://arxiv.org/html/2412.01130v2>.

**Choudhary, 2025 –** Choudhary, V., Modular AI Agents: Building AI from Swappable Brains, Medium. <https://medium.com/accredian/modular-ai-agents-building-ai-from-swappable-brains-1b522a42951d>.

**Chu et al., 2025 –** Chu, Z., Wang, S., Xie, J., Zhu, T., Yan, Y., Ye, J., Zhong, A., Hu, X., Liang, J. Yu, P., Wen, Q., LLM Agents for Education: Advances and Applications. arXiv, <https://arxiv.org/abs/2503.11733>.

**Cunningham et al, 2023** - Cunningham, H., Ewart, A., Riggs, L., Huben, R., Sharkey, L., Sparse Autoencoders Find Highly Interpretable Features in Language Models. arXiv, <https://arxiv.org/abs/2309.08600>.

**DeLong et al, 2023** - DeLong, L., Mir, R., D. Fleuriot, J., Neurosymbolic AI for Reasoning over Knowledge Graphs: A Survey. arXiv, <https://arxiv.org/abs/2302.07200>.

**de Masson d’Autume,2019 -** de Masson d’Autume, C**.**, Aude Oliva, J., Verbeek, J. and Synnaeve, G., 2019. Continual Learning: A Comparative Study on How to Defend Against Catastrophic Forgetting in Neural Networks. arXiv, <https://arxiv.org/abs/1905.13339>.

**De Ridder, 2024** - De Ridder, A., Autonomous Agent Simulation: A Deep Dive into AI-Driven Modelling and Analysis. SmythOS. <https://smythos.com/developers/agent-development/autonomous-agent-simulation/>.

**DeepLearning.AI, 2023 –** DeepLearning.AI, A Complete Guide to Natural Language Processing. <https://www.deeplearning.ai/resources/natural-language-processing/>.

**DeepSeek, 2025 –** DeepSeek AI, DeepSeek v3 Technical Report. arXiv preprint arXiv:2412.19437.

**Dettmers et al, 2022** - Dettmers, T., Lewis, M., Belkada, Y. and Zettlemoyer, L.,LLM.int8(): 8 bit matrix multiplication for Transformers at scale, Proceedings of the 36th NeurIPS 2022, arXiv, <https://arxiv.org/abs/2208.07339>.

**Devlin et al, 2019 –** Devlin, J., Chang, M.W., Lee, K. & Toutanova, K., BERT: Pre training of Deep Bidirectional Transformers for Language Understanding. NAACL HLT 2019 Proceedings, arXiv, <https://arxiv.org/abs/1810.04805>.

**Duquenne et al, 2023 –** Duquenne, P A., Schwenk, H., Sagot, B., SONAR: Sentence Level Multimodal and Language Agnostic Representations. arXiv, <https://arxiv.org/abs/2308.11466>.

**Elhage et al, 2021 –** Elhage, N., Nanda, N., Olsson, C., et al. (2021). A Mathematical Framework for Transformer Circuits. Transformer Circuits. <https://transformer-circuits.pub/2021/framework/index.html>.

**Gao et al., 2024 –** Gao, Y., Xiong, Y., Gao, X., Retrieval Augmented Generation for Large Language Models: A Survey. arXiv, <https://arxiv.org/abs/2312.10997>.

**Garcia et al., 2025 –** Garcia, G., Manesco, J., Paiola, P., Miranda, L., de Salvo, M., Papa, J., A Review on Scientific Knowledge Extraction Using Large Language Models in Biomedical Sciences. arXiv, <https://arxiv.org/abs/2412.03531>.

**Grefenstette, 1999 –** Grefenstette, G., Tokenization, In: van Halteren, H. (ed.) Syntactic Wordclass Tagging, Springer, <https://link.springer.com/chapter/10.1007/978-94-015-9273-4_9>.

**Gupta et al, 2024** - Gupta, S., Ranjan, R., Singh, S., A Comprehensive Survey of Retrieval-Augmented Generation (RAG): Evolution, Current Landscape and Future Directions. arXiv, <https://arxiv.org/abs/2410.12837>.

**Hirschberg & Manning, 2015 –** Hirschberg, J., Manning, C., Advances in natural language processing. Science, 349(6245), <https://pubmed.ncbi.nlm.nih.gov/26185244/>.

**Jang & Morabito, 2025** - Jang, S., Morabito, R., Edge-First Language Model Inference: Models, Metrics, and Trade-offs. arXiv preprint arXiv:2505.16508v2. arXiv, <https://arxiv.org/abs/2505.16508>.

**Jurafsky & Martin, 2025** - Jurafsky, D. and Martin, J.H. 2025- Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models, Stanford University. <https://web.stanford.edu/~jurafsky/slp3/>.

**Kirchenbauer et al., 2023 –** Kirchenbauer, J., Gehrmann, S., Ruiz, F., et al. (2023). A Watermark for Large Language Models. arXiv, <https://arxiv.org/abs/2301.10226>.

**Kumar, 2025 –** Kumar, M. (2025). A Comprehensive Guide to Vibe Coding Tools. Medium. <https://medium.com/madhukarkumar/a-comprehensive-guide-to-vibe-coding-tools-2bd35e2d7b4f>.

**Laia et al, 2024 –** Laia, J., Gan, W., Wu, J., Qi, Z., Yu, P., Large Language Models in Law: A Survey, AI Open, <https://www.sciencedirect.com/science/article/pii/S2666651024000172>.

**Lang et al 2024** - Lang, J., Guo, Z., Huang, S., A Comprehensive Study on Quantization Techniques for Large Language Models. arXiv, <https://arxiv.org/html/2411.02530v1>.

**Lapointe et al, 2025** - Lapointe, V., Dubé, S., Rukhlyadyev, S., Kessai, T., David Lafortune, D., The Present and Future of Adult Entertainment: A Content Analysis of AI-Generated Pornography Websites. NH, <https://pubmed.ncbi.nlm.nih.gov/40032709/>.

**LeCun et al,2015 –** LeCun, Y., Bengio, Y., Hinton, G., Deep learning. Nature, 521(7553), <https://www.nature.com/articles/nature14539>.

**Lewis et al, 2020 -** Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S. and Kiela, D., Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. arXiv, <https://arxiv.org/abs/2005.11401>.

**Lin et al., 2021 –** Lin, T., Wang, Y., Liu, X. & Qiu, Z., A Survey of Transformers. arXiv, <https://arxiv.org/abs/2106.04554>.

**Lindsey, 2025 –** Lindsey, J., Tracing the Thoughts of a Large Language Model. Anthropic Research <https://www.anthropic.com/research/tracing-thoughts-language-model>.

**Martineau, 2024 –** Martineau, K., What’s an LLM Context Window and Why Is It Getting Larger?, IBM Research, <https://research.ibm.com/blog/larger-context-window>.

**McCulloch & Pitts, 1943 –** McCulloch, W. & Pitts, W., A logical calculus of the ideas immanent in nervous activity, Bulletin of Mathematical Biophysics, 5, <https://link.springer.com/article/10.1007/BF02478259>.

**Menon, 2023 –** Menon, P., Introduction to Large Language Models and the Transformer Architecture. Medium, <https://rpradeepmenon.medium.com/introduction-to-large-language-models-and-the-transformer-architecture-534408ed7e61>

**Meta Platforms, 2025 –** Meta Platforms Inc. LLaMA: Open source AI models. <https://www.llama.com/>.

**Nanda et al, 2023 –** Nanda, N., Chan, L., Lieberum, T., Progress Measures for Grokking via Mechanistic Interpretability. arXiv, <https://arxiv.org/abs/2301.05217>.

**Nie et al, 2024 –** Nie, Y., Kong, Y., Dong, X., Mulvey, J., Poor, V., Wen, Q., Zohren, S., A Survey of Large Language Models for Financial Applications. arXiv, <https://arxiv.org/abs/2406.11903>.

**Nijkamp et al, 2022** - Nijkamp, E., Pang, B., H., Tu, L., Wang, H., Zhou, Y., Savarese, S., Xiong, C., CodeGen: An Open Large Language Model for Code with Multi-Turn Program Synthesis. arXiv, <https://arxiv.org/abs/2203.13474>.

**Niu et al, 2024 –** Niu, Q., Liu, J., Bi, Z., Bi, Z., Feng, P., Peng, B., Chen, K., Li, M., Yan, L., Zhang, Y., Yin, C., Fei, C., Wang, T., Wang, Y., Chen, S., Liu, M., Large Language Models and Cognitive Science: A Comprehensive Review. arXiv, <https://arxiv.org/abs/2409.02387>.

**NVIDIA, 2025** – What are World Foundation Models, NVIDIA, <https://www.nvidia.com/en-gb/glossary/world-models/>.

**Ouyang et al, 2022 –** Ouyang, X., Wu, Y., Jiang, X., Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., Lowe, R., Training Language Models to Follow Instructions with Human Feedback. arXiv, <https://arxiv.org/abs/2203.02155>.

**Passali et al, 2023 –** Passali, T., Chatzikyriakidis, E., Andreadis, S., Stavropoulos, T., Matonaki, A., Fachantidis, A., Tsoumakas, G., From Lengthy to Lucid: A Systematic Literature Review on NLP Techniques for Taming Long Sentences, arXiv, <https://arxiv.org/html/2312.05172v1>.

**Patil & Jadon, 2025 –** Patil, A., Jadon, A., Advancing Reasoning in Large Language Models. arXiv, <https://arxiv.org/abs/2502.03671>.

**Plaat et al, 2025 –** Plaat, A., van Duijn, M., van Stein, N., Preuss, M., van der Putten, P., Batenburg, K., Agentic Large Language Models: A Survey. arXiv, <https://arxiv.org/abs/2503.23037>.

**Principe et al, 2025** - Principe, R., Chiarini, N., Viviani, M., Long Document Classification in the Transformer Era: A Survey on Challenges, Advances, and Open Issues. Wiley Interdisciplinary Reviews, <https://wires.onlinelibrary.wiley.com/doi/full/10.1002/widm.70019>.

**Rai et al., 2024 –** Rai, D., Zhou, Y., Feng,S., Saparov, A., Yao, Z., A Practical Review of Mechanistic Interpretability for Transformer Based Language Models. arXiv, <https://arxiv.org/abs/2407.02646>.

**Qu et al, 2024** - Qu, Y., Wei, C., Du, P., Che, W., Zhang, C., Ouyang, W., Bian, Y., Xu, F., Hu, B., Du, K., Wu H., Liu, J., Liu, Q., Integration of cognitive tasks into artificial general intelligence test for large models. [www.sciencedirect.com/science/article/pii/S2589004224007727](http://www.sciencedirect.com/science/article/pii/S2589004224007727).

**Raza et al., 2025 –** Raza, M., Jahangir, Z., Riaz, M., Saeed, M., Sattar, M., Industrial applications of large language models. Scientific Reports, 15, <https://www.nature.com/articles/s41598-025-98483-1>.

**Rumelhart, Hinton & Williams, 1986 –** Rumelhart, D.E., Hinton, G.E. & Williams, R.J. (1986). Learning representations by back propagating error,’ Nature, 323(6088), <https://www.nature.com/articles/323533a0>.

**Saeed, 2024 –** Saeed, T., Understanding Context Windows in Large Language Models (LLMs), Medium, <https://medium.com/@tahir.saeed_46137/understanding-context-windows-in-large-language-models-llms-4ad3dca6b86f>

**Sahoo et al, 2024 –** Sahoo, P. Singh, A., Saha, S., Jain, V., Mondal, S., Chadha, A., A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications, arXiv, <https://arxiv.org/html/2402.07927v1>.

**Sennrich et al, 2016 –** Sennrich, R., Haddow, B. and Birch, A. Neural machine translation of rare words with subword units, Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016), <https://aclanthology.org/P16-1162/>.

**Sharkey & Treleaven, 2025 –** Sharkey, E. and Treleaven, P., Optimising Large Language Models: Taxonomy and Techniques. SSRN, <https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5278456>.

**Shojaee et al., 2025 –** Shojaee, P., Mirzadeh, I., Alizadeh, K., Horton, M., Bengio, S., Farajtabar, M., The Illusion of Thinking. Apple ML Research. arXiv, <https://arxiv.org/abs/2506.06941>.

**Shone, 2024 –** Shone, O. Explore AI Models: Key Differences Between Small and Large Language Models. Microsoft, <https://www.microsoft.com/en-us/microsoft-cloud/blog/2024/11/11/explore-ai-models-key-differences-between-small-language-models-and-large-language-models/>.

**Song et al, 2023 –** Song, S., Li, X., Li, S., Zhao, S., Yu, J., Ma, J., Mao, X., Zhang, W., How to Bridge the Gap between Modalities: A Comprehensive Survey on Multi-modal Large Language Model, arXiv, <https://arxiv.org/html/2311.07594v2>.

**Tang et al, 2023** - Tang, Y., Bi, J., Xu, S., Song, L., Liang, S., Wang, T., Zhang, D., An, J., Lin, J., Zhu, R., Vosoughi, A., Huang, C., Zhang, Z., Liu, P., Feng, M., Zheng, F., Zhang, J., Luo, P., Luo, J., Xu, C., Video Understanding with Large Language Models: A Survey. arXiv, <https://arxiv.org/abs/2312.17432>.

**Vaswani et al, 2017 –** Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., Polosukhin, I., Attention is all you need, Advances in Neural Information Processing Systems. <https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>

**Wang et al, 2023** - Wang, W., Dong, L., Cheng, H., Liu, X., Yan, X., Gao, J., Wei, F., Augmenting Language Models with Long-Term Memory. arXiv, <https://arxiv.org/abs/2306.07174>

**Wang et al, 2024a –** Wang, L., Yang, F., Zhang, C., Lu, J., Qian, J., He, S., Zhao, P., Qiao, B., Huang, R., Qin, S., Su, Q., Ye, J., Zhang, Y., Lou, J., Lin,Q., Rajmohan, S., Zhang, D., Zhang, Q., Large Action Models: From Inception to Implementation. arXiv, <https://arxiv.org/abs/2412.10047>.

**Wang et al, 2024b** - Wang, W., Chen, W., Luo, Y., Long, Y., Lin, Z., Zhang, L., Lin, B., Cai, D., He, X., Model Compression and Efficient Inference for Large Language Models: A Survey. arXiv, <https://arxiv.org/abs/2402.09748>.

**Wang et al, 2025** - Wang, J., Shi, E., Huawen Hu, H., Ma, C., Yiheng Liu, Y., Wang, X., Yao, Y., Liu, X., Ge, B., Zhang, S., Large language models for robotics: Opportunities, challenges, and perspectives. Journal of Automation and Intelligence, [www.sciencedirect.com/science/article/pii/S2949855424000613](http://www.sciencedirect.com/science/article/pii/S2949855424000613)

**Wehner et al, 2025 –** Wehner, J., Abdelnabi, S., Tan, D., Krueger, D., Fritz, M., Taxonomy, Opportunities, and Challenges of Representation Engineering for LLMs. arXiv, <https://arxiv.org/abs/2502.19649>.

**Wei et al, 2025 –** Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., Zhou, D., Chain of Thought Prompting Elicits Reasoning in Large Language Models. arXiv, <https://arxiv.org/abs/2201.11903>.

**Wu et al, 2024 -** Wu, T., Lan, J., Yuan, W., Jiao, J., Weston, J., Sukhbaatar, S., Thinking LLMs: General Instruction Following with Thought Generation. arXiv, <https://arxiv.org/abs/2410.10630>.

**Xhonneux et al, 2024 –** Xhonneux, S., Sordoni, A., Günnemann, S., Gidel, G., Schwinn, L., Efficient Adversarial Training in LLMs with Continuous Attacks. arXiv, <https://arxiv.org/abs/2405.15589>.

**Xua et al, 2024 –** Xua, H. Xu, H., Gan, W., Qi, Z., Wu, J., Yu, P., Large Language Models for Education: A Survey. arXiv, <https://arxiv.org/abs/2405.13001>.

**Yan et al, 2025 –** Yan, B., Li, K., Xu, M., Dong, Y., Zhang, Y., Ren, Z., Xiuzhen Cheng, X., On protecting the data privacy of Large Language Models (LLMs) and LLM agents: A literature review, High-Confidence Computing, Volume 5, <https://doi.org/10.1016/j.hcc.2025.100300>.

**Zhang et al, 2023** - Zhang, X., Wang, Y., Li, J., & Liu, Z., Towards Federated Large Language Models: Motivations, Methods, and Future Directions. IEEE Journals & Magazine, <https://ieeexplore.ieee.org/abstract/document/10759678>

**Zhang et al., 2024 –** Zhang, Q., Ding, K., Lyv, T., Wang, X., Yin, Q., Zhang, Y., Yu, J., Wang, Y., Li, X., Xiang, Z., Feng, K., Zhuang, X., Wang, Z., Qin, M., Zhang, M., Zhang, J., Cui, J., Huang, T., Yan, P., Rnjun Xu, R., Chen, H., Li, X., Fan, X., Xing, H., Chen, H., Scientific Large Language Models: A Survey on Biological & Chemical Domains. arXiv, <https://arxiv.org/abs/2401.14656>.

**Zhang et al., 2025a –** Zhang, R., Li, H., Qian, X., Zhang, R., Li, H-W., Qian, W-Y., Jiang, W-B., Chen, H-X., On large language models’ safety, security, and privacy: A survey, Journal of Electronic Science and Technology, 23(1), 100301. [www.sciencedirect.com/science/article/pii/S1674862X25000023](http://www.sciencedirect.com/science/article/pii/S1674862X25000023)

**Zhang et al., 2025b -** Zhang, J., Hu, S., Lu, C., Lange, R., Clune, J., Darwin Godel Machine: Open-Ended Evolution of Self-Improving Agents. arXiv, <https://arxiv.org/abs/2505.22954>.

**Zhao & Song, 2024 –** Zhao, G., Song, E. Privacy Preserving Large Language Models: Mechanisms, Applications, and Future Directions. arXiv, <https://arxiv.org/abs/2412.06113>.

**Zou et al., 2023 –** Zou, A., Wang, Z., Carlini, N., Nasr, M., Kolter, Z., Fredrikson, M., Universal and Transferable Adversarial Attacks on Aligned Language Models. arXiv, <https://arxiv.org/abs/2307.15043>.