Towards Transparent Reasoning: A Survey and Review of Explainable AI Methods and the Rise of Thought-Based LLMs

Brenden Noblitt*
*Department of Computer Science and Information Systems,
East Texas A&M University, Commerce, TX, USA
Email: bnoblitt@leomail.tamuc.edu

***Abstract*—Artificial intelligence is increasingly influencing decisions in critical domains such as criminal justice, healthcare, and finance. Yet when these systems make errors or behave unexpectedly, the reasons behind their choices often remain hidden, posing serious risks to safety, fairness, and accountability. As AI continues to integrate into society and daily life, the need for trust and confidence in AI systems is becoming more apparent. Explainable AI (XAI) aims to bridge this gap by providing insights into how models make decisions. This paper surveys traditional methods – including LIME, SHAP, and integrated gradients – and examines emerging techniques that go beyond attribution to expose reasoning. Specifically, DeepSeek-R1 represents a new framework for analyzing model reasoning chains by generating a chain of thought during inference, offering users insight into the model's internal decision-making. The review discusses the taxonomy and tradeoffs of current XAI methods and outlines ongoing challenges. We conclude by identifying current challenges and future directions for aligning model reasoning with human expectations, ensuring transparency and trust in real-world applications.**

Keywords—Explainable Artificial Intelligence, XAI, Model Interpretability, Feature Attribution, LIME, SHAP, CFE, Saliency Map, LRP, CAM, DeepSeek-R1, Trustworthy AI

I. Introduction

As AI systems increasingly make decisions that affect the lives of humans, there is a growing demand for models that can not only perform well but also justify their outputs. This follows the push for AI systems that are both interpretable and explainable, giving researchers, engineers, and users a glimpse into the "black-box" processing of AI and providing clarity on why a given decision is made. While much research has been focused on attribution-based methods, little attention has been given to models that expose their reasoning during inference. This paper surveys traditional XAI methods, such as LIME, SHAP, and Integrated Gradients, and further investigates reasoning-based approaches in current models, such as DeepSeek-R1.

This paper argues that while traditional XAI methods such as LIME, SHAP, and integrated gradients provide useful local attribution, they fall short of exposing a model's full reasoning process. Reasoning-chain approaches—particularly those used in DeepSeek-R1—represent a critical advancement by offering transparent, human-readable steps that align more closely with how people expect decisions to be made. The following sections summarize explainable and trustworthy AI principles, different XAI techniques, an overview of DeepSeek-R1's reasoning chain, how XAI principles may be adapted to reasoning-based models, and potential future research directions for reasoning-based models and XAI principles.

II. Overview of XAI Principles

The AI High-Level Expert Group (AI HELG) outlined four basic principles that trustworthy AI should have. 1) Respect for Human Autonomy 2) Prevention of Harm 3) Principle of Fairness 4) Principle of Explicability. The authors state that these abstract ethical principles should be translated into tangible, operational requirements. This should involve relevant stakeholders, namely developers who implement AI, developers who ensure systems comply with compliance requirements, and users who are willing to use AI and give feedback [2]. While there is a general need for explainable and trustworthy AI, specific needs include transparency, which is the capability of an AI system to provide understandable and reasonable explanations of a model's decision process. Governance and compliance issues, where XAI enables governance in systems by confirming the decisions made by AI are ethical, accountable, transparent, and compliant. Model performance and debugging offer potential benefits in enhancing the performance of AI systems, particularly in terms of model design and debugging. Reliability, which includes control mechanisms to trust the AI models' predictions and decisions. Safety, to ensure the safety, security, and well-being of humans involved in these systems. Finally, human-AI collaboration, where collaboration between humans and AI systems can be facilitated [1]. With these principles, the difference between interpretable AI and explainable AI should be noted. Interpretable AI is inherently understandable by design. Explainable AI is the techniques and methods used to make the results of models understandable [2].

III. XAI Techniques and Methods

A. Overview

There are different categories of XAI techniques with different applications and usages. First, there are local and global techniques. Local explanation techniques focus on explaining predictions or decisions made by a specific instance or input to a model. These techniques focus on individual predictions or outputs made by a model. Global explanation techniques provide an overview or complete

description of the model. These techniques usually require knowledge of input data, algorithms, and the trained model itself. There are also ante-hoc and post-hoc explanation techniques. Ante-hoc explanation techniques are employed during the training and development stages of AI systems. These techniques include decision trees, general additive models, and Bayesian models. Post-hoc explanation techniques are employed after an AI model has been trained and deployed to explain the model's predictions or decision-making process. These techniques include Local Interpretable Model-Agnostic Explanations (LIME) and Shapley Additive Explanations (SHAP). Within post-hoc explanations, there are model-specific approaches, which focus on a model's internal structure, and model-agnostic approaches, which can be applied to all AI models without having an understanding of the model's working structure [2]. There are also perturbation and gradient techniques, which are discussed in the following sections.

B. Perturbation-Based Techniques

Perturbation-based techniques involve modifying the input data of an AI system and observing changes to the output in order to understand which input features are the most important for the model's predictions. This could involve changing a pixel in an image or a word in a body of text, or some other input, and measuring the impact the change created on the output. We will focus on three popular perturbation techniques for explaining AI systems.

First, Local Interpretable Model-Agnostic Explanations (LIME) are insightful for understanding specific decisions a model makes by providing local individual instance explanations. The goal of LIME is to look near input x, ask what the black-box model f does nearby, and train a simple, human-understandable model g to mimic that local behavior. While LIME is insightful for understanding specific model decisions, it can have instability and lacks global insight [3]. Compared to SHAP, LIME is faster and easier to implement but lacks the theoretical guarantees that SHAP provides, making its explanations more sensitive to sampling choices and randomness.

Another popular perturbation technique is Shapley Additive Explanations (SHAP). SHAP values quantify how much each feature contributed to a specific prediction, based on a principled and fair approach based on game theory. The output is model-dependent, meaning that SHAP explanations will reflect bias if the model is biased. SHAP can also have high computational costs [4]. While SHAP tends to be more stable and theoretical than LIME, its efficiency tradeoffs may make it impractical for large-scale or real-time use cases.

The third perturbation technique we will discuss is Counterfactual Explanation (CFE). CFE explains what would need to change in the input to obtain a different and potentially more desirable output. This offers insight into decision-making processes of the model, aiding in the identification of errors or bias. While counterfactuals can offer insight into the decision-making process of AI systems, multiple counterfactuals can be present, making it difficult to interpret which one is the most optimal. Unlike LIME and SHAP, which focus on attributing importance to inputs, CFE emphasizes actionable recourse, making it especially useful in high-stakes scenarios where end-users want to understand how to change an outcome.

While perturbation-based techniques provide valuable insights by observing model behavior through input changes, they do not leverage information about how the model's internal structure influences its outputs. To address this limitation, gradient-based techniques analyze how changes in input features affect the model's predictions by directly examining the model's gradients.

C. Gradient-Based Techniques

Gradient-based techniques involve analyzing gradients of the model's output concerning its input features. A larger gradient indicates that the feature has a greater influence on the model's output. We will focus on four popular gradient-based explanation techniques used in AI systems.

The first of which is the saliency map. A saliency map is a visualization technique used to show which parts of the input influence the output of a model the most based on the output gradients. This technique is widely used in image generation by using the saliency map to highlight important features between the input and output and decide if the model is correctly creating features of the image. While saliency maps provide a clear indication of the model's approach to creating the input, saliency maps are a manual visualization tool and require manual human assessment, making them less ideal for automated validation [5]. They are also prone to noise and instability, which has led to the development of more robust techniques like LRP and integrated gradients.

Another gradient-based technique used is Layer-wise Relevance Propagation (LRP). LRP describes which parts the input contributed to the final decision of a model by redistributing the prediction backwards through the neural network. This technique uses actual activations and weights from the forward pass to trace where the evidence came from. LRP is great for tracing relevance through a system, but is complex to implement and requires a good understanding of the neural network's architecture [6]. Unlike saliency maps, which rely purely on gradients, LRP incorporates model internals like activations and weights, making it more reliable in some contexts but less flexible across model types.

TABLE I. EXPLANATION TECHNIQUE COMPARISONS

Method	**Type**	**Model-Agnostic?**	**Strengths**	**Weaknesses**
LIME	Perturbation	Yes	Fast, intuitive	Instability, black box
SHAP	Perturbation	Yes	Theoretical grounding	Slow
CAM	Gradient	No	Visual, class-focused	CNN-specific
IG	Gradient	Yes	Faithful, smooth	Needs baseline
LRP	Gradient	No	Relevance tracing	Complex to implement
CFE	Perturbation	Yes	Actionable	Counterfactual validity

The third technique to be discussed is Class Activation Mapping (CAM), a technique that identifies which regions the model focuses on when making a classification decision. This technique requires a convolutional feature extractor, a Global Average Pooling (GAP) layer, and a fully connected output

layer within the CNN architecture. It's a simple and fast technique to implement and helps localize discriminative regions; however, it is model-specific [7]. Unlike techniques such as integrated gradients or LRP that offer fine-grained attribution at the feature level, CAM provides more coarse, spatial-level insights, making it ideal for visual diagnostics but less informative for other data modalities.

A fourth technique that acts as its subcategory is Integrated Gradients (IG). IGs work by looking at gradients along a straight line from a neutral reference to the real input, as opposed to looking at the gradients at a single point. IG was proposed to be designed with two axioms in mind. The first axiom is Sensitivity, where for every input and baseline that differ in one feature but have different predictions, the differing feature should be given non-zero attribution. The other axiom is Implementation Invariance, which states that two models that produce the same outputs for all inputs, regardless of how they are implemented internally, should yield the same explanations. IG techniques are smooth and provide faithful explanations; however, they require a meaningful baseline [8]. Compared to CAM or saliency maps, IG offers a more theoretically grounded explanation and applies broadly to both image and non-image models, but it requires careful tuning of the baseline to avoid misleading outputs.

While the previous sections focused on the techniques used to generate explanations, it is equally important to consider how these explanations are evaluated. The effectiveness of any XAI method ultimately depends on how well it meets human and task-specific expectations for trust, fidelity, and usefulness.

IV. XAI Evaluation Methods

While explanation methods allow us to explain a model's behavior, decisions, and internal state, evaluation methods allow us to assess how useful or meaningful a given explanation is. Some shortcomings in XAI evaluation have been detailed in recent research. One shortfall is the lack of evaluation. While the evaluation of XAI techniques is crucial for ensuring models are fulfilling their goals, recent reviews have shown there is a small portion of efforts directed towards exploring such evaluation. Another shortfall is the lack of consensus. Categorization and terminology are not consistent in the current literature, especially between the terms "subjective" and "objective". There are competing usages of these terms between papers, which creates a challenging task for the standardization of evaluation methods. Lack of multidisciplinary approaches is another shortcoming of XAI evaluation. Authors argue that there is a greater emphasis on highly technical approaches for determining evaluation, some of which lack the human element that allows for true evaluation from the human perspective. Lack of standardized evaluation procedures is also apparent. Due to the lack of standardized evaluation procedures, many researchers create new ways to evaluate explanation methods, which in turn deteriorates the ability to interpret the outcomes of experiments. This becomes more difficult in human-centered approaches, which tend to be more subjective. The lack of incorporation of cognitive processes can also be a challenge. Current XAI techniques assume that the end-user mostly uses analytical thinking to deduce insights from the evaluation techniques. Providing extremely detailed XAI explanations does not guarantee that the end-user will use all supplied information, which can be misleading during the evaluation of the explanation. Finally, the lack of visualization and interaction strategies can create challenges in allowing users to accurately consume information supplied by XAI systems [9].

There are two groups of evaluation methods we will discuss here: human-centered approaches and computer-centered approaches. Human-centered evaluation methods evaluate how a provided XAI explanation meets the needs of a user. This approach is mainly concerned with comprehensibility, trust, user satisfaction, and decision-making support. This approach assesses the cognitive load and ensures XAI explanations do not affect the user's cognitive processing capacity. The other approach is the computer-centered evaluation approach. This approach evaluates XAI techniques using objective, automated criteria without human intervention, preferably in an automated fashion. This approach is based on fidelity, consistency, robustness, efficiency, and other dimensions, which will be discussed in further detail [1].

Fidelity refers to how the provided XAI explanations are close to the actual decision made by the model. High fidelity reflects that the explanation is accurate in interpretation. Consistency refers to the focus on the stability and coherence of the explanations provided by the XAI system with the same input and different model runs. Stability, uniformity, and predictability are the most important aspects of consistency. Robustness assesses the resilience of explanations in the behavior of model change in various aspects. This includes resistance to input perturbations and adaptability to model changes. Efficiency refers to the computational capacity and resources, including resource utilization. Sufficiency refers to the assessment of the adequacy of rationales in supporting the model's decision-making process. It measures the difference in the model's confidence between using the entire input and using just the rationale [1].

Despite the variety of evaluation metrics, many traditional XAI methods still struggle to convey how models reason through complex tasks. To address this gap, a new class of techniques—reasoning-chain methods—aims to reveal not just what a model predicts, but how it reasons through a prompt to arrive at that prediction.

V. LLM Reasoning

A. Overview of LLM Reasoning

A newer development in the LLM space of artificial intelligence is the idea of LLM reasoning, specifically displaying an LLM's reasoning to the end-user. DeepSeek's R1 model uses think tokens to continuously display the model's thinking and reasoning processes to the end-user. This opens up the opportunity to study and analyze model reasoning, furthering both model explainability and interpretability. Traditionally, LLMs have been prone to "System 1" thinking—what psychologist Daniel Kahneman describes as fast, intuitive, and automatic mental processes. In contrast, "System 2" thinking is slower, more analytical, and deliberate. Recent research has focused on enabling LLMs to exhibit System 2-style reasoning, particularly for tasks that require multi-step logic, planning, or introspection. Chain of Thought (CoT) prompting and other frameworks built upon CoT have shown success in training-free approaches, where the model is prompted to think step-by-step. Other training-free approaches included prompting with

algorithm examples and facilitating problem decomposition. Training-based approaches work in instilling reasoning behavior, where reward signals train the model to develop reasoning processes. This has allowed popular LLM models to develop reasoning capabilities, such as DeepSeek-R1, o1, Claude 3.7, and Gemini 2.5, although DeepSeek-R1 was the first to implement this into user-facing models and is the only model with the implementation details available publicly, which is why we will be focusing on DeepSeek's R1 model specifically in this paper.

B. Early Attempts at LLM Interpretability

Early attempts to expose the interpretability and reasoning of LLMs occurred, leading up to the introduction of Chain-of-Thought prompting. These approaches incorporate techniques such as weighted tokens and using other models to describe what the LLM might attempt to do. We will discuss some of those approaches in this section.

Attention Mechanisms were an early approach to exposing LLM interpretability and reasoning. They are a technique that allows a model to selectively focus on different parts of the input sequence when processing or generating text. This includes calculating weights for each element in the input sequence and using the weights to reflect how relevant each element is to the current task. Traditional encoder-decoder models compressed the entire input into a single fixed-length vector. This method became less accurate as input length increased. By adding an attention mechanism to the decoder, the encoder no longer must encode all information in the source input and instead can be spread throughout the input, allowing the decoder to selectively retrieve information. This approach introduced key performance benefits when using longer sequences of text as the model input. One of the test models with this new approach, RNNsearch-50, showed little or no deterioration with input sequences of a length of fifty or more. This approach also removed the bottleneck of calculating the vector of the sequence input all at once. Instead, a sequence of annotation vectors is calculated, which allows the decoder to selectively retrieve information. This allows for the visualization of weights associated with the annotations, allowing for further analysis of what the model anticipates is the most important within an input sequence [10].

Three years later, the authors in [11] argued that the relationship between the attention weights and model outputs was not consistent, possibly diluting the explanations that were being inferred from the weights regarding LLM explainability. Correlation between feature importance measures and learned attention weights is weak for recurrent encoders. This signifies that attention does not reliably correlate with feature importance. Despite high attention weights, alternative and different attention configurations over inputs yield effectively the same output. Presenting heatmaps to visualize weights in these scenarios could indicate a particular feature was responsible for an output when it in fact was not. Shuffling attention weights often can have a minimal impact on output results as well. There are cases in which attention weights might suggest explaining an output by a small set of features, but where scrambling the attention makes little difference to the prediction, signifying that model reasoning does not depend on specific attention patterns [11]. While attention mechanisms are the backbone of LLM innovations, their use in explaining LLMs is not as widely used as more current techniques. Visualizing and analyzing attention mechanisms is still used in conjunction with other methods.

Probing Classifiers were another early attempt to assist in interpreting LLMs. Probing classifiers are a technique used in LLMs and NLP that analyzes the internal representations of a model by training a separate model specialized in classification to predict specific linguistic properties from these classifications. They are common for understanding what information a model has learned and how it is encoded. Probing classifiers are also used as stand-alone solutions to some classification problems, such as part-of-speech tagging and sentiment analysis. Probing classifiers attempt to help understand what kind of information a model is encoding internally. The author discusses that probing provides a practical tool to investigate what is learned inside deep neural networks in the absence of more principled interpretability tools. Probing classifiers have some strengths that lend themselves to LLM interpretability. First, the classifications created by probing help see if the models are encoding fine-grain linguistic knowledge, such as syntax. This approach can also help locate where different types of knowledge can be found within the model, providing an inherent sense of mapping to a black-box model. Probing classifiers are also easy to implement across multiple models and datasets.

There are some associated limitations with this method, one of which is the interpretation of the results of probing classifier experiments. The values returned from the experiments lack context, suggesting that a baseline or comparison is required to determine if the evaluation is good or bad. Powerful probes can learn the property even if the original model doesn't encode it. The high accuracy of non-linear probes may come from memorization of surface patterns rather than from the information captured. Another limitation is the disconnect between the probing classifier and the original model. A successful probe does not guarantee that the model uses that information for its own predictions. Probing frameworks may indicate a correlation between representation and linguistic probabilities; however, this does not indicate if the property was involved in the predictions. Finally, properties must be defined. This limits usable datasets to those that are annotated, which are typically limited to only English and sets of certain properties. This also requires focusing on properties that are thought to be relevant, which could lead to bias. While there are some limitations around probing classifiers, they played an important role in early efforts in understanding the internal mechanisms of a black-box model [12].

Chain-of-Thought (CoT) Prompting brought a unique approach to LLM interpretability and exposure of LLM reasoning, which is still currently being used. CoT prompting is a technique that directs LLMs to generate a series of intermediate reasoning steps before producing a final response. This technique attempts to improve the accuracy and reliability of LLMs, especially tasks that require complex reasoning, and shows their chain of reasoning used to get to their output. LLMs often struggle to respond accurately to multi-step problems and frequently fail to decompose them into smaller, manageable steps. Traditional prompting also made LLMs less interpretable due to the lack of reasoning provided by the LLM in the output response. CoT prompting allows models to decompose a multi-step problem into intermediate steps, allowing additional compute to be

funneled into more complex problems. It also offers insight into the model's behavior, allowing a user to see the reasoning path to an answer. Finally, CoT can also be implemented in most LLMs by including CoT prompts.

There are some associated limitations with the CoT prompting approach. The author cites that CoT only works well with larger models. Models with fewer than 10B parameters showed little to no difference when using CoT. Because the outputs are human-readable, CoT may not reflect the true internal decision process. CoT is also highly sensitive to phrasing in the prompt, and CoT increases token usage, which can create higher costs and latency. There is also no way for the LLM to self-check its reasoning chains internally. While CoT is a novel prompting method that helped initial efforts to expose internal LLM reasoning, it still comes with limitations [13].

DeepSeek-R1 has built upon the foundation of these approaches to create a more cohesive and accurate way to display LLM reasoning chains. While attention mechanisms introduced a method to expose attention weights to understand why a model favors a certain word or phrase, and probing classifiers allowed for the understanding of internal encoding, DeepSeek-R1 has used a combination of reinforcement learning and ideas from Chain-of-Thought prompting to think and reason about prompts leading up to the final answer.

C. Limitations of Chain-of-Thought (CoT)

While Chain-of-Thought prompting brings some unique advantages to understanding LLM reasoning, CoT does come with some limitations, which are debated further in the current literature. We will discuss some of those limitations that set the stage for future innovations on the CoT approach.

Unfaithful explanations are one limitation of CoT. The authors of [14] attempt to determine if CoT explanations are used by the model to generate the final response or if they are simply rationalizing without the utilization of that rationale. The authors argue that CoT explanations are not faithful by default. Training objectives do not incentivize models to accurately report reasoning for their behavior. LLMs are trained on human-written explanations, which are known to be incomplete or biased, and reinforcement learning with human feedback (RLHF) techniques may disincentivize faithful explanations. CoT explanations can be plausible yet systematically unfaithful, where the model's explanations can be influenced by biasing features in their inputs, which they fail to mention in their reasoning and explanation. The authors used five models across five datasets and found that explanations often do not influence the final answer. When models use CoT explanations, they will often ignore them while generating the final answer. These tests do not look at internal activations and do not consider newer, finely tuned models, such as DeepSeek-R1, but still provide a good basis for the potential lack of faithfulness in generic CoT [14].

Poor generalization is another limitation of CoT. The authors of [15] attempt to determine if CoT can perform well across general ranges of tasks or if CoT is more suited for specific tasks. This is done using Blocksworld, a classical planning domain to examine the performance of LLMs. Through their experiments, they found that CoT only helps with the prompts tailored to the exact task and that CoT does not meaningfully enhance performance except on narrow problem distributions. As problems begin to grow beyond the prompted template, performance degrades sharply. They determined that CoT performance does not come from learning algorithms, but rather from pattern matching of prompt templates. They also highlight that a tradeoff of this approach is effectiveness vs. manual prompt labor to craft a well-created CoT prompt. While this testing focuses on the planning domain and doesn't incorporate newer models with reinforcement learning, it is still worth noting that CoT can lack higher-level reasoning in wider, more generic problems [15].

Negative impact on non-reasoning tasks is another limitation of CoT. The authors of [16] attempt to determine if CoT prompting always improves performance. They use tailored experiments and popular datasets to determine performance between CoT prompting and traditional direct answers from the LLM. They found that their testing with 16 LLMs across 9 datasets, CoT underperformed in nearly all tasks evaluated in this study. Through this, they propose the Explicit-Implicit Duality framework with explicit pattern inference and execution and implicit pattern recognition and execution, where the implicit mechanisms may be attempting to compensate for deficiencies in the explicit reasoning process. They also find that even with extended reasoning chains, CoT fails to outperform the direct answering paradigm while also incurring a higher token utilization. While this testing focuses mainly on pattern-based ICL tasks, this work still poses the argument that CoT is not a "one-size-fits-all" approach and that problems not requiring reasoning could have performance degradation when attempting to use CoT [16].

While these limitations help define the boundaries of Chain-of-Thought's usefulness, they have also inspired researchers to extend and adapt the original framework. These evolutions aim to preserve CoT's core goal—exposing LLM reasoning—while addressing challenges like verbosity, unfaithful explanations, or limited generalization. The next section explores key developments in this space, including interface-time, agent-based, and training-time strategies.

D. Evolution of Chain-of-Thought (CoT) Prompting

With the initial success of Chain-of-Thought prompting, researchers attempted to build on the methodologies of CoT by creating different methods to expand CoT to fit other problem sets. We will discuss some of these methods in different categories: interface-time prompting frameworks, prompt-based agent frameworks, and training-time reasoning interventions.

Least-to-Most (LtM) prompting was an initial framework created from the ideas in Chain-of-Thought prompting. LtM prompting is a prompting strategy to elicit reasoning from an LLM. It consists of posing a complex problem as a list of easier subproblems and then sequentially solving the generated subproblems using few-shot prompting, where no additional training on the LLM is required. LtM attempts to solve some issues with CoT, particularly CoT's difficulties in generalizing problems in symbolic, compositional, and arithmetic domains. CoT focuses on encouraging the model to generate intermediate reasoning steps within a single prompt, whereas LtM focuses on breaking down a complex problem into smaller subproblems and solving them sequentially while using answers from earlier subproblems as context. The author cites that LtM showed massive accuracy gains on the SCAN benchmark using the code-davinci-002 model, where CoT scored ~16% on accuracy while LtM

scored ~99% accuracy. LtM requires problems to be broken down into subproblems, making it less effective for global, holistic reasoning tasks. The performance of LtM is also controlled by the decomposition prompts the user creates. Token utilization can increase due to the cascading effect of using previous answers as context, and accuracy is relied upon by the LLM itself [17].

Tree-of-Thoughts (ToT) prompting is another framework created from CoT. ToT is a general prompting framework for solving problems with LLMs. ToT works by actively maintaining a tree of thoughts, where each thought is a language sequence that acts as an intermediate step towards problem solving. ToT attempts to solve CoT's issues with linear operation, where CoT struggles with tasks that require branched decision making or require global planning. ToT defines reasoning steps as "thoughts", which act as nodes within a tree data structure. Breadth-first and depth-first traversal algorithms are used to search across information to pull relevant information. This approach also forces the model to explore multiple reasoning paths over thoughts due to the tree-like structure. The author cites framework performance on a created test called the "Game of 24", where the goal is to take 4 input numbers and perform basic arithmetic operations to obtain 24. The CoT approach only solved 4% whereas ToT showed a solution rate of 74%. Due to the branching search, ToT can be expensive, especially in terms of compute. The applicability of the task depends on natural thought decomposition. There is also no internal reasoning change as model weights are not updated [18].

Graph-of-Thoughts (GoT), similar to ToT, is another framework built upon CoT. GoT is a prompting approach that enhances LLMs' capabilities through networked reasoning. Rather than ToT's approach to structuring reasoning thoughts in a tree-like data structure, GoT organizes thoughts in a graph-like structure, where each thought is a node and the dependencies between those thoughts are an edge within the graph. This graph structure allows for nonlinear interconnections between thoughts, potentially improving task quality and a reduction in cost. This approach also uses a modular architecture to create flexibility within the system. These modules are the prompter, which prepares messages for the LLM, the parser, which extracts information from LLM thoughts, the scoring module, which verifies and scores LLM thoughts, and the control, which coordinates the reasoning process and decides how to progress. Some limitations of this approach include high computational costs due to searching across a graph, surface-level prompting, as GoT cannot adapt to model internals, and task fit, where GoT is better suited for tasks that can be broken down by merging smaller sub-problems into a larger reasoning sequence. [19].

ReAct is a prompt-based agent framework that shares similarities to CoT but also has some important distinctions. ReAct is a general paradigm that combines reasoning and acting with language models to solve diverse language reasoning and decision-making tasks. ReAct aims to solve some issues with traditional prompting strategies for generating reasoning. CoT reasoning operates as a static black box, where the model uses its own internal representations to generate thought and is not grounded by the external world. This can lead to hallucinations and error propagation over the reasoning process. ReAct prompts the LLM to use both verbal reasoning and actions while providing traces to both, working in an interleaved manner. These traces help the human user to further understand why an LLM is doing something because it traces both the verbal response and the action, making it well-suited for LLM agents. ReAct is a prompt-based framework, meaning no additional fine-tuning is needed for the model. While ReAct is a prompt-based framework, it differs from other prompt-based frameworks by using external tools like search engines for context. It also uses a feedback loop to adapt its next action to the previous action, aligning closer to an agentic-style architecture rather than static prompting. While external tool usage has accuracy benefits, it also creates performance limitations depending on the API's used to connect to these tools. The interleaved reasoning can also have performance limitations at runtime. Its use of few-shot prompting can also create accuracy issues since the prompt is dependent upon the user [20].

Mechanistic interpretability is a training-time reasoning intervention field of study made popular by Anthropic, the creators of the Claude model. Rather than a tool or framework, mechanistic interpretability is a field of study focused on determining how deep neural networks make decisions by reverse-engineering their internal mechanisms. It attempts to break down neural networks into components that are more easily understandable. Neurons within a neural network are polysemantic, meaning they respond to mixtures of unrelated inputs, making them difficult to understand. Mechanistic interpretability attempts to understand these neurons and provide a way to represent monosemantic representations within the LLM. Mechanistic interpretability uses a dictionary learning algorithm called a sparse autoencoder to create learned features from a trained model to break down units into a more monosemantic unit of analysis rather than the neurons themselves. This approach also highlights that monosemanticity becomes more apparent at scale, where features become cleaner and more interpretable as the model grows. This approach does have some limitations. First, mechanistic interpretability is focused on one layer and does not decompose deeper pathways. Training a sparse autoencoder is also computationally intensive, creating some performance limitations. Mechanistic interpretability also assumes the sparse features reflect deeper computational roles, not just the effects downstream [21].

E. Efficiency and Advancements in Reasoning

As more methods were created to understand the inner workings of LLMs, further research was done in order to combat some of the performance implications that methods like CoT and ToT had. We will discuss a few of these methods.

Thought Propagation was one of the first methods created to create LLM reasoning without requiring heavy computational effort to scan input text. Thought Propagation (TP) is a framework to improve existing reasoning-from-scratch methods and enhance complex reasoning in LLMs. TP works by prompting LLMs to propose a set of analogous problems related to the input, then the input problem is solved with its analogous counterpart with existing frameworks like CoT. TP attempts to solve issues with reasoning prompt strategies that start from scratch, and in turn, struggles with issues like accumulated errors and a lack of reusing previous insights. TP uses a pipeline-like architecture, where "LLM Propose" generates a set of analogous problems, "LLM Solve" solves both the original and analog problems using

existing frameworks like CoT, and "LLM Aggregate" which integrates the analogical solutions to improve the original problem. TP was shown to perform well on shortest-path tasks, creative writing, and LLM-agent planning, and outperformed other frameworks like CoT and ToT. TP can still have some incurred computational cost, requires good analogies to create analogous problems, and is best suited for tasks with structural similarity [22].

Prompt Sketching is another framework and an early precursor to Sketch-of-Thought (SoT). Prompt Sketching is a prompting framework that predicts values for multiple variables in a template within the prompt to offer more control over LLM reasoning. Prompt sketching attempts to solve some issues with CoT prompting, such as the high variance in intermediate answers from CoT and improving the model's awareness of the overall template. This approach attempts to embed structure into the model by having the model fill in a predefined template all at once. Template prompting allows the LLM to predict all template variables in one shot during decoding rather than doing it one at a time. This creates a full structure within the LLM's ingested prompt. The model then scores follow-up instructions in advance, which optimizes the template. Through experimentation, the authors found prompt sketching outperformed CoT in 7 of 8 tasks. This approach does have some limitations. First, templates must be well-designed in order to perform well. Scoring follow-up instructions can also come with computational implications due to the scoring of multiple slots. It is also less effective with tasks that do not map well into the provided templates [23].

Sketch-of-Thought (SoT) is a framework that provides some notable performance gains. SoT is inspired by cognitive sciences, using symbolic "sketches" to guide the model to produce concise, structured reasoning steps in the reasoning chain to capture the essential logic required without using full sentences to elaborate reasoning. The general idea is to allow models to use a short-hand approach to reasoning in order to capture semantic meaning while eliminating redundant sentences. In a traditional CoT architecture, the Chain-of-Thought is generated, combined in the LLM with the user prompt, and outputs a response with the generated reasoning. In SoT, rather than combining the user prompt and CoT prompt within the LLM, SoT splits out the flow into multiple steps. SoT first uses a "router model", which involves a smaller text model, to select a type of reasoning paradigm to use to generate reasoning. With the selected paradigm, a "paradigm cache" is used to collect reusable sketches based on the selected paradigm from the router model. The collected sketches are then passed directly into the user prompt in order to wrap reasoning sketches around the prompt. The combined text is then sent to the LLM, and an output response is generated. This approach is shown to reduce token usage by ~78% with little loss to accuracy.

Looking closer at the paradigms within the paradigm cache, there are three different types of paradigms that assist in creating usable sketches for the LLM.

Conceptual Chaining

```
Q: What is the name of the currency used in Seoul?
A: <think> #Seoul → #South Korea → Won </think>
Answer: Korean Won
```

Fig. 1. Conceptual Chaining Example. Adapted from [24]

Conceptual Chaining is the first one, which uses principles of cognitive science to create concise and logical sequences between key concepts. This paradigm focuses on how concepts may be related to each other, and is useful in commonsense, logical, and scientific reasoning where structured relationships between ideas is critical. An example of this can be seen in Figure 1.

Chunked Symbolism, demonstrated in Figure 2, organizes numerical and symbolic reasoning into compact steps. This is similar to how humans chunk related information into meaningful units.

Chunked Symbolism

```
Q: A car accelerates at 2.5 m/s^2 for 10 seconds. If
its initial velocity was 15 m/s, what is its final
velocity?
A: <think> a = 2.5 m/s^2, t = 10 s, vi = 15 m/s vf =
15 + (2.5 × 10), vf = 40 m/s </think>
Answer: 40 m/s
```

Fig. 2. Chunked Symbolism Example. Adapted from [24]

This paradigm condenses mathematical reasoning into symbolic reasoning that allows more information to be stored with fewer tokens. This paradigm is useful in mathematical and arithmetic reasoning.

Finally, Expert Lexicons, demonstrated in Figure 3, uses domain knowledge and specialized notation to condense reasoning. This paradigm is centered around domain keywords, such as abbreviations, to condense domain-specific information with fewer tokens.

Expert Lexicons

```
Q: A patient with STEMI is given MONA therapy. They
are allergic to aspirin. Are they at risk with this
treatment?
A: <think> STEMI → ST-Elevation MI, MONA → Morphine,
O2, Nitrates, Aspirin, so Aspirin ∈ MONA </think>
Answer: Yes
```

Fig. 3. Expert Lexicons Example. Adapted from [24]

Expert Lexicons excels in technical and business disciplines, where domain expertise enables significant compression of information [24].

F. DeepSeek-R1's Approach to Reasoning

To accurately depict DeepSeek-R1's reasoning process, let us define the typical human reasoning process. First, humans begin by establishing the problem definition. This involves simplifying relevant details of the task representation to identify known and unknown information. Next, an initial response is created, where an appropriate solution to an analogous problem or reliance on heuristics-based thinking is used to give an immediate answer, similar to System 1 thinking. Next, planning occurs, where one may have to think strategically or analytically, such as breaking down a large task into smaller ones or incrementally progressing to the desired goal. Execution and monitoring happen after, where one may monitor their confidence in their progress to determine if the plan needs to be readjusted. Reconstruction occurs next, where one's initial approach or world assumptions may need to be modified during the solving process. Finally, solution verification occurs, where one may reflect on their approach and solution to ensure it meets the constraints of the given problem. While this is not all-

encompassing human reasoning, this gives a general baseline to compare against LLM reasoning.

We can now define the taxonomy of DeepSeek-R1's reasoning process. First, DeepSeek-R1 starts with the problem definition, where the model reformulates the problem with explicit recognition of the required solution (i.e., "Ok, so the user wants me to…"). Next, a concept called the "blooming cycle" takes place, which is the first major reasoning cycle for the model. This is where the model decomposes a problem into subproblems and provides an interim answer (i.e., "First, I should…"). Reconstruction cycles take place next, which are subsequent reasoning cycles where the model considers what happened during the blooming cycle. The model may then provide a new interim answer and repeat the cycle (i.e., "Wait, alternatively…"). Finally, the model reaches the final answer (i.e., "Ok, I'm now sure…").

Comparing this to the human reasoning process, there are a couple of observations we can make. First, the problem definition in DeepSeek-R1's reasoning process tends to be more formalized, typically just stating the unknowns in its reasoning. Next, the model rarely gives a heuristics-based initial response and instead dives straight into a strategic approach. While this is the intention of integrating System 2-style reasoning into LLMs, it can sometimes become redundant or unwanted for simple questions, leading to a "paralysis-by-analysis" bias. Next, rather than the human-based "plan-execute-reconstruct", DeepSeek-R1 tends to plan while it executes. Finally, the final decision the model makes is a restatement of the model's confidence as it verifies throughout the reconstruction cycle [25].

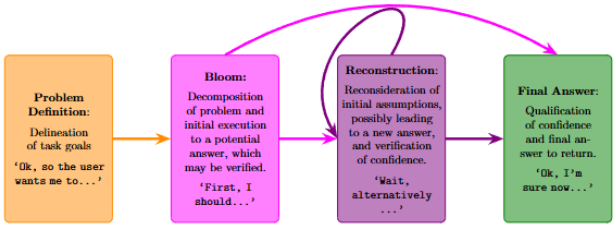

Fig. 4. DeepSeek-R1's reasoning process. Adapted from [25]

Another component of DeepSeek-R1's reasoning process is the reinforcement learning process applied to the base model. While reinforcement learning is effective in reasoning tasks, traditional approaches have relied heavily on supervised data. DeepSeek-R1 instead emphasizes self-evaluation through reinforcement learning, developing reasoning capabilities without supervised data. The model uses reward modeling to reinforce accurate outputs, evaluate whether a response is correct, and enforce formatting, such as placing its reasoning between special "think" tokens. The self-evolution process of DeepSeek-R1's base model is particularly crucial, as it provides a method of consistently allowing the model to improve without external adjustments. Due to reinforcement learning methods, the model is also capable of having "aha" moments, where the model allocates additional thinking steps when solving complex problems. This allows the model to dedicate more thinking time to a problem while also allowing researchers to observe the behavior of the model. This combination of CoT prompting and reinforcement learning provides a unique way of accurately depicting DeepSeek-R1's reasoning processes and becoming more transparent.

The creators of DeepSeek-R1 acknowledge several limitations. General capabilities of the model could be improved, with some tasks falling short of the DeepSeek-V3 model. These tasks include function calling, complex role-playing, and JSON outputs. There are plans to further research how CoT can be leveraged to enhance performance in these tasks. Language mixing is another limitation. DeepSeek-R1 is optimized for English and Chinese; however, this can result in the model mixing between the two languages. There is research being done to mitigate this limitation. Prompt engineering is another limitation. It is observed that DeepSeek-R1 is sensitive to prompts, with few-shot prompting degrading performance. It is recommended to use zero-shot prompting when working with this model. Finally, software engineering tasks are a limitation of the model. Large-scale reinforcement learning has not been applied in software engineering tasks, and in turn, DeepSeek-R1 shows little improvement over DeepSeek-V3 in this task domain. There are plans to implement rejection sampling on software engineering data or incorporate asynchronous evaluations during the reinforcement learning process in order to improve efficiency in these tasks [26].

Overall, this new approach to LLM reasoning could pave the way for new research in XAI. Giving users and researchers the ability to observe the reasoning chain of an LLM can assist in solving some of the issues XAI research has brought to light, such as transparency and bringing clarity to the "black box" thinking that traditional LLMs are known for. DeepSeek-R1 addresses several of the core shortcomings of earlier CoT methods by combining multiple innovations. Its use of think tokens and the blooming cycle creates a structured, reliable trace of reasoning steps, which improves the faithfulness of explanations and helps mitigate hallucinated logic. The self-evaluation component further enhances transparency by having the model critique and refine its thought process, rather than blindly continuing. Reinforcement learning (RL) techniques help reinforce these behaviors by rewarding accuracy and coherence in reasoning, which improves generalization across tasks. Importantly, DeepSeek-R1 is also designed to minimize performance degradation on non-reasoning tasks by using adaptive mechanisms that allow it to fall back to simpler strategies when reasoning isn't required. Together, these features form a more robust, interpretable alternative to traditional prompting strategies.

VI. Applications of Chain-of-Thought Prompting

While there have been a variety of different implementations of Chain-of-Thought prompting for generic prompting activities, researchers in specific domains have also developed CoT into domain-specific applications, some of which we will discuss here.

A. Legal Judgement Prediction

Law is one domain in which CoT is already being used to enhance case analysis and judgment. Legal Judgment Prediction (LJP) aims to predict the judgment of a legal case based on its fact description. This involves the automation of predictions of verdicts based on case facts. Traditional approaches, such as pre-trained language models, rely on supervised learning, require an extensive set of data, and typically do not explain their output. The authors propose Legal Syllogism Prompting (LoT), a zero-shot template-based technique like CoT that teaches the model to structure

its thinking as a legal syllogism. It differs from traditional CoT techniques as it requires no additional examples and is zero-shot. The technique uses a structure similar to Aristotle's classic expression of a major premise, a minor premise, and a conclusion. In law, the major premise is the law article, the minor premise is the case facts, and the conclusion is the judgment. Using GPT-3 on the CAIL2018 dataset, LoT outperformed zero-shot and CoT prompting across eight legal judgment categories, showing that LoT can successfully enhance the legal reasoning capacity of LLMs. LoT also produces explicit syllogisms, identifying the law to be applied, linking facts, and deriving a conclusion, which shows LLM reasoning and builds trustworthiness in legal AI. LoT was only tested on GPT-3, meaning results may or may not translate to other models, the prompts' fixed template may be too rigid for poorly structured cases, and the data was mainly tested with CAIL2018. Early indications of this technique are positive with potential application in the field of law [27].

B. Open-Ended Medical Question Answering

Another area CoT is making advances in is medicine, especially in the analysis of patient history and initial diagnosis. CLINICR is a 5-shot prompting technique designed to mirror clinical incremental reasoning. This technique uses CoT principles to continuously build on gained knowledge regarding patient facts and history. A new dataset, MEDQA-OPEN, is introduced as an adaptation of MedQA-USMLE to transform questions into open-ended questions. This approach attempts to mimic the traditional differential diagnosis process found in medicine by generating the differential diagnosis and then eliminating unlikely options to conclude. A verifier reward model is trained to validate the reasoning instead of relying on elimination prompts. This approach outperforms previous 5-shot CoT prompts on open-ended MedQA assessments. Limitations include potential mistranslation to smaller models, no use of fine-tuning, which could potentially enhance performance, and a lack of solid evaluation metrics since metrics like factual accuracy and safety are difficult to quantify and are not deeply analyzed. With some limitations, this approach could be a promising innovation in assisting with more complex medical questions [28].

C. Financial Reasoning

Finance is another place where CoT is innovating, using CoT to analyze financial metrics and provide feedback. FinCoT attempts to solve the issue of unstructured CoT prompts that fail to reflect domain-expert reasoning processes in their reasoning logic. FinCoT uses explicit expert-guided steps, such as analyzing trends, financial metrics, and concluding. This approach injects expert financial workflows into a structured CoT template, yielding auditable results without fine-tuning. FinCoT starts with a top-level message to frame the task, guides step-by-step execution using think tokens, uses embedded diagrams to determine a step-by-step problem-solving strategy, and then re-reads its reasoning before creating a final answer. This approach was tested against standard zero-shot and generic CoT, where FinCoT saw an 80% accuracy compared to generic CoT at 63%. FinCoT also cut token utilization compared to generic CoT, giving heightened reasoning in the financial domain while also being cost-aware. FinCoT is limited to CFA-style questions compared to open-ended problems, such as analyzing a full financial report. This method is limited to prompting and does not explore techniques like fine-tuning, which could potentially enhance performance. With these limitations, FinCoT still shows promise in using CoT to assist with analysis activities in the financial domain [29].

These are just three areas in which CoT is bringing innovation to create more robust autonomous processes. While CoT brings productivity improvements and automation, it specifically brings the ability to view LLM reasoning in sensitive areas that require justification to take certain actions.

VII. Challenges in Reasoning Chains and Potential Future Research Directions

While reasoning chains like Chain-of-Thought (CoT), Graph-of-Thoughts (GoT), Sketch-of-Thought (SoT), and DeepSeek-R1's combination of reasoning change and reinforcement learning provide a transparent interface into an LLM's internal processing and reasoning, the faithfulness of these reasoning chains remains an unresolved and apparent challenge. There is often a gap between the reasoning a model presents and the true internal computations and reasoning used to get to the final answer. LLMs may hallucinate intermediate reasoning steps that appear plausible but do not causally influence the output. This is especially problematic in high-stakes and sensitive domains like law, healthcare, and finance, where users depend on both the answer and the rationale behind it.

There is no guarantee that the reasoning presented by current LLMs aligns causally with the internal decision process and that it is consistent across similar inputs. There is a lack of robust testing and benchmarks to measure reasoning faithfulness, and limited research on how users interpret or trust these chains. This disconnect undermines the very goal of explainability, which is to foster user confidence and model accountability. To address this challenge, there are a variety of research directions that can be taken to shed light on this topic.

A. Causal Validation of Research Chains

One direction could be to develop evaluation methods that can test if intermediate reasoning steps are causally linked to the final output. This could involve interventional testing, which involves removing or altering a step and observing the effect on the output. Research could analyze and build on techniques from model interpretability and causal inference to see whether reasoning is merely rationalized without usage or truly instrumental.

B. Faithful Reasoning Supervision During Training

Instead of training models solely on final answers, future systems could be trained on human-annotated or programmatically generated reasoning steps, enforcing alignment between the thinking trace and the actual computation path. If a model claims to "first check patient history", then it should use that operation internally. This would involve expanding datasets to include process supervision.

C. Interactive Reasoning Interfaces

Design and implement interfaces where users can interact with reasoning chains. This could include asking for justification, flagging unclear steps, or requesting simpler language. This feedback could then guide the rerouting of the reasoning process or initiate self-refinement cycles. This

could potentially close the gap between transparency and usability that is apparent within some models.

D. Benchmarking and Standardization

The overall field could benefit from standardized benchmarks and metrics for reasoning faithfulness. While some datasets test final accuracy, new benchmarks could evaluate the minimality, correctness, and utility of intermediate steps. This could be complemented by user studies measuring perceived trust and satisfaction with reasoning traces.

E. Application-Specific Reasoning Guarantees

In high-risk and sensitive domains, the domain-specific constraints could be imposed on reasoning outputs. For instance, in finance, reasoning could be constrained by accounting rules. In law, they could be constrained by jurisdictional precedent. These constraints could guide the reasoning trace or potentially validate it post hoc.

VIII. Conclusion

As AI and LLM systems continue to integrate into high-stakes domains, the need for transparent, trustworthy, and user-aligned decision-making is more critical than ever. Traditional explainable AI (XAI) methods, such as feature attribution and counterfactual reasoning, have provided valuable insights into model behavior. However, they often fall short in exposing the full chain of logic that underlies complex model outputs.

This paper explored the rapidly emerging class of thought-based explainability methods, particularly those that leverage reasoning chains within LLMs. By tracing the evolution from Chain-of-Thought (CoT) prompting to more efficient and modular methods like Sketch-of-Thought (SoT) and DeepSeek-R1's combination of reasoning chains and reinforcement learning, it demonstrated how modern models externalize their internal reasoning through symbolic and structured forms, enhancing transparency and trustworthiness. It is also visualized how these approaches can add value in domains such as legal judgment, medical QA, and financial forecasting, areas where simple attributions are not optimal for building user trust.

Despite their promise, these methods still face major challenges, particularly in ensuring that the reasoning presented is faithful to the model's actual decision process. Addressing this gap will be essential for the broader deployment of XAI in socially impactful domains. By grounding future research in causal alignment, loop interfaces, and rigorous evaluation, we can move toward systems that are not only accurate but also transparent, accountable, and aligned with human reasoning.

Acknowledgment

I would like to thank Dr. Mohammad Alsmirat for giving guidance on formalized research guidelines and topics in explainable AI (XAI). I also appreciate the support of my peers and faculty of the Department of Computer Science and Information Systems at East Texas A&M University.

References

[1] M. Mersha, K. Lam, J. Wood, A. K. AlShami, and J. Kalita, "Explainable artificial intelligence: A survey of needs, techniques, applications, and future direction," *Neurocomputing*, vol. 599, p. 128111, Sep. 2024, doi: https://doi.org/10.1016/j.neucom.2024.128111.

[2] B. Long, E. Liu, R. Qiu, and Y. Duan, "Explainable AI the Latest Advancements and New Trends," arXiv.org, 2025. https://arxiv.org/abs/2505.07005

[3] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why Should I Trust You?': Explaining the Predictions of Any Classifier," Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16, pp. 1135–1144, Aug. 2016, doi: https://doi.org/10.1145/2939672.2939778.

[4] S. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," arXiv:1705.07874 [cs, stat], Nov. 2017, Available: https://arxiv.org/abs/1705.07874

[5] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps," arXiv.org, 2013. https://arxiv.org/abs/1312.6034

[6] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation," PLOS ONE, vol. 10, no. 7, p. e0130140, Jul. 2015, doi: https://doi.org/10.1371/journal.pone.0130140.

[7] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning Deep Features for Discriminative Localization," arXiv.org, 2015. https://arxiv.org/abs/1512.04150

[8] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic Attribution for Deep Networks." Available: https://arxiv.org/pdf/1703.01365

[9] P. Lopes, E. Silva, C. Braga, T. Oliveira, and L. Rosado, "XAI Systems Evaluation: A Review of Human and Computer-Centred Methods," Applied Sciences, vol. 12, no. 19, p. 9423, Sep. 2022, doi: https://doi.org/10.3390/app12199423.

[10] D. Bahdanau, K. Cho, and Y. Bengio, "Published as a conference paper at ICLR 2015 NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE," 2016. Available: https://arxiv.org/pdf/1409.0473

[11] S. Jain and B. C. Wallace, "Attention is not Explanation," arXiv:1902.10186 [cs], May 2019, Available: https://arxiv.org/abs/1902.10186

[12] Y. Belinkov, "Probing Classifiers: Promises, Shortcomings, and Advances," arXiv.org, 2021. https://arxiv.org/abs/2102.12452

[13] J. Wei et al., "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models Chain-of-Thought Prompting," Jan. 2023. Available: https://arxiv.org/pdf/2201.11903

[14] M. Turpin, J. Michael, E. Perez, and S. Bowman, "Language Models Don't Always Say What They Think: Unfaithful Explanations in Chain-of-Thought Prompting." Available: https://arxiv.org/pdf/2305.04388

[15] K. Stechly, K. Valmeekam, and S. Kambhampati, "Chain of Thoughtlessness? An Analysis of CoT in Planning," arXiv.org, 2024. https://arxiv.org/abs/2405.04776 (accessed Jun. 29, 2025).

[16] T. Zheng et al., "The Curse of CoT: On the Limitations of Chain-of-Thought in In-Context Learning," arXiv.org, 2025. https://arxiv.org/abs/2504.05081

[17] D. Zhou et al., "LEAST-TO-MOST PROMPTING ENABLES COMPLEX REASONING IN LARGE LANGUAGE MODELS." Available: https://arxiv.org/pdf/2205.10625

[18] S. Yao et al., "Tree of Thoughts: Deliberate Problem Solving with Large Language Models." Available: https://arxiv.org/pdf/2305.10601

[19] M. Besta et al., "Graph of Thoughts: Solving Elaborate Problems with Large Language Models," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, no. 16, pp. 17682–17690, Mar. 2024, doi: https://doi.org/10.1609/aaai.v38i16.29720.

[20] S. Yao et al., "REACT : SYNERGIZING REASONING AND ACTING IN LANGUAGE

[21] "Towards Monosemanticity: Decomposing Language Models With Dictionary Learning," transformer-circuits.pub. https://transformer-circuits.pub/2023/monosemantic-features/index.html

[22] J. Yu, R. He, and R. Ying, "Thought Propagation: An Analogical Approach to Complex Reasoning with Large Language Models," arXiv.org, 2023. https://arxiv.org/abs/2310.03965

[23] L. Beurer-Kellner, M. N. Müller, M. Fischer, and M. Vechev, "Prompt Sketching for Large Language Models," arXiv.org, 2023. https://arxiv.org/abs/2311.04954 (accessed Jun. 29, 2025).

[24] S. A. Aytes, J. Baek, and S. J. Hwang, "Sketch-of-Thought: Efficient LLM Reasoning with Adaptive Cognitive-Inspired Sketching,"

arXiv.org, 2025. https://arxiv.org/abs/2503.05179 (accessed Jun. 22, 2025).

[25] S. V. Marjanović et al., “DeepSeek-R1 Thoughtology: Let’s think about LLM Reasoning,” arXiv.org, 2025. https://arxiv.org/abs/2504.07128

[26] DeepSeek-AI et al., “DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning,” arXiv.org, 2025. https://arxiv.org/abs/2501.12948

[27] C. Jiang and X. Yang, “Legal Syllogism Prompting: Teaching Large Language Models for Legal Judgment Prediction,” arXiv.org, 2023. https://arxiv.org/abs/2307.08321 (accessed Jun. 29, 2025).

[28] Nachane, Saeel Sandeep et al., “Few shot chain-of-thought driven reasoning to prompt LLMs for open ended medical question answering,” arXiv.org, 2024. https://arxiv.org/abs/2403.04890

[29] N. Nitarach, W. Sirichotedumrong, P. Pitchayarthorn, P. Taveekitworachai, P. Manakul, and K. Pipatanakul, “FinCoT: Grounding Chain-of-Thought in Expert Financial Reasoning,” arXiv.org, 2025. https://arxiv.org/abs/2506.16123 (accessed Jun. 29, 2025).