TUM School of Computation, Information and Technology
Software and Systems Engineering
Technical University of Munich

Visualizing Token Importance in Large Language Models

A Review of the State of and a Framework for Understanding Token Attribution in LLMs

Ruben Kaiser

Software and Systems Engineering, Technical University of Munich

ruben.kaiser@tum.de

April 15, 2025

Supervisor: Xingcheng Chen
Examiner: Prof. Dr. Andrea Stocco

Abstract — Large Language Models have become increasingly integrated into critical domains, yet their internal processes remain largely opaque. While various attribution methods for transparency exist, visualization tools that effectively balance technical faithfulness with interpretability for non-experts are absent from the current landscape. Existing tools such as BertViz, Polyjuice, and Inseq suffer from specific limitations: incompatibility with modern architectures, post-hoc rationalizations instead of faithful explanations, and static rather than interactive visualizations.

This paper conducts a systematic analysis of current token attribution tools through the lens of Human-Computer Interaction (HCI) principles, drawing from comprehensive reviews of explainability methods. Based on identified gaps, it proposes an integrated visualization framework with three technical innovations: (1) a mathematically formalized job-based architecture enabling parallel and efficient attribution processing across multiple methods, (2) a dynamic visualization interface supporting context-sensitive exploration through token hovering, window adjustment, and direct model comparison, and (3) an extensible system designed to simultaneously visualize diverse attribution methods with appropriate visual encodings.

The proposed framework addresses the tension between faithfulness and plausibility in LLM interpretability, potentially enabling stakeholders from model developers to domain experts to gain actionable insights into how LLMs process information. Such transparency is essential for identifying reliability issues, mitigating biases, and ensuring proper alignment as these systems become increasingly embedded in critical decision-making processes.

1 Introduction

1.1 Background

The explosion in performance of Large Language Models (LLMs) has marked a transformative era in artificial intelligence [1–5]. The resulting widespread adoption can be largely attributed to the dramatic increase in model scale, with larger models consistently demonstrating enhanced capabilities across domains: From the initial GPT architecture [6] to sophisticated models like BERT [7], GPT-3 [8], and more recent iterations such as GPT-4 [9], LLaMA-3 [10], and Claude [11], (structurally based upon the landmark paper by Vaswani et al. [12]) these systems have demonstrated remarkable capabilities across a wide range of tasks.

In **healthcare**, these models assist with medical diagnosis and clinical documentation [13]. **Financial institutions** leverage LLMs for market analysis and risk assessment, with specialized models like BloombergGPT showing enhanced performance in finance-specific tasks [14]. **Educational** applications span personalized tutoring and curriculum development (see Abd-Alrazaq et al. for medical education [15]), while **legal professionals** use LLMs for document review and contract analysis [16]. **Creative industries** have embraced these models for content generation across multiple formats [17]. This widespread integration across critical infrastructure and decision-making processes continues to reshape economic systems, information environments, and social dynamics [18].

The variety of task is enabled by the ever increasing sizes of the foundation models and the amount of available training data, and - by correlation [19] - their increase in performance, with consistent scaling laws predicting performance improvements as parameters, training data, and computational resources increase [20].

However, this rapid advance in capabilities has outpaced our understanding of how these systems function. As Liao and Vaughan argue, LLMs present *"looming risks to society itself,"* [21] necessitating that *"transparency must be a central pillar"* [21] of responsible development and deployment. Weidinger et al. [18] define six risk areas, spanning from (miss-)information to environmental and social-economic harms.

Unlike simpler machine learning models whose operations can be readily inspected, LLMs operate as complex, distributed systems whose internal representations and decision-making processes remain largely opaque. This opacity is particularly concerning given their above-mentioned growing integration into high-stakes domains where errors or biases could have significant consequences for individuals and communities [22] as well as society as whole.

1.2 Current Challenges in LLM Interpretability

Several key components lead to complications in developing effective interpretability methods for LLMs:

1. The sheer **scale and complexity** of these models—often containing hundreds of billions of parameters—render many traditional explainability approaches computationally *infeasible or inadequate* for capturing the multi-layered transformations that occur throughout the system [23]. A complete analysis of parameter interactions across numerous attention heads and layers quickly becomes infeasible, and even when done, often don't convey meaning or doesn't present the full picture [24].

2. **Competing interpretability paradigms** approach the problem from different perspectives, each with their own methodological assumptions. Some focus on mechanistic interpretability, seeking to reverse-engineer specific circuits within the model [25]. Others adopt a more functional approach, analyzing the relationship between inputs and outputs without attempting to map the exact computational paths [26]. And others again rely on the model itself to explain its "reasoning" [27]. These divergent approaches, while valuable in diversity, have created a fragmented research landscape that can be difficult for practitioners to navigate.

3. The **proprietary nature** of many leading LLMs (OpenAI, Google DeepMind, Anthropic) presents a significant barrier to open research. With limited access to model weights, training data, and internal representations, researchers must often work with incomplete information about the systems they seek to understand. [23]

4. The **contextual and emergent nature** of representations in LLMs poses unique challenges. Unlike traditional feature importance methods that can identify fixed, context-independent relationships between inputs and outputs, LLMs construct dynamic, context-sensitive representations that may vary substantially across different inputs and generation steps [28]. This contextuality makes it difficult to generalize explanations across different use cases or to provide simplified yet accurate accounts of model behavior.

We will look at some specific examples for visualisation attempts in section 2.

1.3 Current Approaches to Interpretability

The field of LLM interpretability encompasses a diverse range of methods that attempt to illuminate different aspects of model behavior and knowledge. Zhao et al. [23] and Cambria et al. [22] provide comprehensive surveys of existing approaches, highlighting both their strengths and limitations.

Feature attribution methods aim to identify which input tokens most influenced a particular prediction or generation. Techniques such as integrated gradients [29], LIME [30], and SHAP [31] have been adapted for language models, although their computational requirements can become prohibitive for very large models. These methods often struggle to account for the complex interactions between tokens in contextual representations, potentially yielding misleading attributions in certain scenarios.

Attention visualization tools offer insights into how models allocate focus across input elements. Implementations like BertViz [32] allow users to inspect attention patterns across different layers and heads, revealing structural regularities in how models process linguistic information. However, a substantial debate exists regarding whether attention weights genuinely reflect causal relationships in model reasoning [24, 33, 34].

Probing studies examine whether specific linguistic properties are encoded in model representations, leveraging classifiers trained on model activations to detect the presence of syntactic, semantic, or factual knowledge [35]. While valuable for understanding what information models capture, probing methods typically yield limited insights into how this information is utilized during actual generation or prediction tasks.

Concept-based approaches investigate how high-level concepts are represented across model components [36, 37]. These methods bridge the gap between low-level activations and human-interpretable ideas, offering a more abstract but potentially more useful characterization of model knowledge. However, defining and validating appropriate concepts remains challenging, particularly for the diverse and contextual representations in LLMs.

Despite this array of techniques, measuring and visualizing token importance in LLMs presents persistent difficulties. As models grow in size and complexity, traditional attribution methods become computationally expensive and may fail to capture the hierarchical, distributed nature of token representations [23]. Furthermore, token importance can vary dramatically with context, making static visualizations potentially misleading without appropriate qualification.

1.4 Problem Statement

Given these challenges and current limitations ($\rightarrow$ section 2) in interpretability and the potential social implications of widespread LLM use, there is an urgent need for more effective and accessible tools to visualize and analyze token importance in LLMs. This work addresses this gap by proposing a roadmap and requirements for a visualization tool that builds upon existing research but reimagines interpretability through human-computer interaction principles [38, 39].

This approach aims to:

1. Make complex model behavior accessible through intuitive interfaces that don't require deep technical expertise
2. Generate actionable insights that can inform responsible development and deployment
3. Bridge the gap between technical precision and practical usability

By democratizing access to LLM interpretability tools, we can enable a wider range of stakeholders—from domain experts in healthcare and education to regulators and end users—to meaningfully engage with questions of model behavior, bias, and limitations as these systems continue to transform our information environment and social structures.

2 Current State of Token Importance Visualization

We do not introduce the concepts of tokens in this paper, but the reader is invited to read either Sennrich et al. [40] for their work on subword tokenization for neural machine translation, or Vaswani et al. [12] for their explanation of tokens in the context of transformer architectures.

2.1 Current Approaches to Token Importance Visualization

The (in section 1.1) rapidly expanding domain-agnostic involvement of Large Language Models (LLMs) necessitates robust interpretability methods [21]. In this section, we analyze current approaches to token visualization, focusing on their theoretical foundations and practical implementations.

As foundation to understanding and evaluating existing tools, we must first examine the attribution methods they employ to quantify token importance.

2.1.1 Token Attribution Methods

Token attribution in LLMs aims to quantify how individual input tokens (words, subwords, or characters) contribute to a model's output. Several methodologies have emerged, of which the most prominent ones are:

Integrated Gradients Introduced by Sundararajan et al. [29], Integrated Gradients (IG) addresses limitations in standard gradient-based attribution by satisfying important axioms: sensitivity and implementation invariance. Standard gradient methods [41, 42] break sensitivity because "the prediction function may flatten at the input and thus have zero gradient despite the function value at the input being different from that at the baseline" [29] (see also [43]). IG calculates the path integral of gradients from a baseline input to the actual input:

$$\text{IntegratedGrads}_i(x) = (x_i - x_i') \times \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha \times (x - x'))}{\partial x_i} d\alpha \tag{1}$$

IG's computational requirements are substantial for LLMs with billions of parameters, as it requires multiple forward and backward passes to approximate the integral [23].

SHAP SHapley Additive exPlanations (SHAP) [31] unifies various attribution methods using concepts from game theory: It treats features as players in a cooperative game and assigns importance based on their marginal contribution to predictions. While SHAP provides desirable properties like local accuracy and consistency, its exact calculation has exponential complexity.

Attention mechanisms Transformer-based LLMs rely on self-attention mechanisms that assign weights to tokens during processing [12]. These weights present an intuitive target for interpretation, visualized in tools like BertViz [32]. However, significant debate exists regarding attention's reliability for explanation [24]. Research by Jain and Wallace [33] found that attention weights poorly correlate with feature importance methods, and models can maintain identical predictions with drastically altered attention patterns suggesting that raw attention may not provide faithful explanations of model behavior.

Perturbation and surrogate methods Perturbation-based methods assess token importance by measuring how removing, masking, or altering tokens affects the model's output. While intuitive, these approaches may create linguistically incoherent inputs and face computational scalability issues [23]. Surrogate methods like LIME [30] create locally faithful explanations by approximating complex models with simpler, interpretable ones. However, "LIME does not satisfy some properties of additive attribution, such as local accuracy, consistency, and missingness" [23] (see also [31]).

2.1.2 Explainability Principles for Visualization Tools

Effective LLM visualization tools must balance technical rigor with usability. Two critical dimensions for evaluating explanations are according to Zhao et al. [23]:

- **Faithfulness**: How accurately an explanation reflects the model's actual decision process.
- **Plausibility**: How understandable and convincing an explanation appears to humans.

These dimensions often exist in tension: highly plausible explanations may not faithfully represent the model's internal processes due to the inherit necessity of breaking down very complex processes, loosing potentially relevant detail. A tangible example is Chain of Thought (CoT) reasoning, which is very natural to understand but can generate plausible yet unfaithful explanations [27], given explanations are generated in retrospect (after the main inference).

From an HCI perspective, Liao et al. [21] emphasize human-centered transparency approaches, including: progressive disclosure of complexity; appropriate visual encodings (color, size, position); interactive exploration capabilities; and support for developing accurate mental models over time.

2.2 Existing Tools and Approaches

The field of LLM interpretability has seen rapid development of specialized tools and frameworks for token attribution visualization. We analyze several key examples based on the work of Zhao et al. [23] and Cambria et al. [22] below, highlighting their contributions and limitations.

2.2.1 BertViz

BertViz [32] offers multiscale visualization of attention in transformer models. It provides visualization at multiple levels:

- **Attention-head view**: Visualizes attention patterns between tokens using bipartite graphs
- **Model view**: Presents a birds-eye view of attention across all layers and heads
- **Neuron view**: Examines individual components within the attention mechanism

While BertViz provided valuable insights into attention patterns at the time of its development, its applicability to current state-of-the-art models is limited. Additionally, as summarized by Bibal et al. [24] and pointed out by Jain and Wallace [33], attention weights often poorly correlate with feature importance methods. Models can maintain identical predictions with drastically altered attention patterns, suggesting raw attention may not provide faithful explanations of model behavior.

2.2.2 Polyjuice

Developed by Wu et al. [44], Polyjuice serves as a general-purpose counterfactual generator. By fine-tuning GPT-2 on various datasets, it produces diverse, contextually appropriate counterfactual examples that help illuminate model behavior through contrastive analysis.

Despite its innovative approach, Polyjuice faces significant limitations for current LLM interpretability needs:

- **Architectural constraints**: Built on fine-tuned GPT-2, the tool cannot readily generalize to modern architectures and parameter scales.
- **Computational efficiency**: The counterfactual generation process becomes prohibitively expensive for exploring complex prompts or large text spaces.
- **Lack of integration**: The tool does not integrate with other attribution methods, limiting its utility as part of a comprehensive interpretability framework.

2.2.3 LMExplainer

LMExplainer [45] integrates knowledge graphs with attention mechanisms to generate explanations for language model decisions. The system implements several key components:

- Knowledge graph integration to ground explanations in structured knowledge
- Graph attention networks (GAT) to identify relevant reasoning elements
- Two-stage explanation generation that addresses both why certain options were chosen and others rejected
- Debugging framework that evaluates explanation quality along multiple dimensions

The system operates by constructing an element-graph from ConceptNet, pruning it based on relevance scores (via the probability function f_{prob} defined in equation 2), and using graph attention to identify key reason-elements that support the model's decisions. These elements then form the basis for natural language explanations.

While LMExplainer demonstrates promising performance on CommonsenseQA and OpenBookQA datasets, several challenges remain:

- The relationship between extracted reason-elements and the model's actual internal decision processes requires further validation
- The evaluation methodology combines human assessment (both experts and crowdsourced participants) with automated evaluations, but larger-scale human studies would strengthen confidence in results
- Some implementation details regarding knowledge graph pruning and parameter settings could benefit from additional specification to improve reproducibility

LMExplainer represents a step toward more interpretable language models by attempting to provide structured, knowledge-grounded explanations rather than relying solely on attention patterns or post-hoc rationalizations. However, like other approaches in this domain, it faces the fundamental challenge of balancing explanation faithfulness with human comprehensibility.

2.2.4 Chain-of-Thought Analysis

Chain-of-Thought (CoT) explanations are tangible but Turpin et al. raise critical concerns for LLM interpretability [27]: LLMs can generate plausible yet unfaithful explanations that fail to reflect their actual decision-making processes. Specifically:

- LLMs generated convincing explanations for incorrect answers when influenced by biasing features
- Performance dropped by up to 36% on benchmark tasks with biased prompts
- Models justified stereotype-aligned answers without explicitly mentioning stereotypes

This research underscores that even when LLMs provide natural language explanations for their reasoning, these explanations may not reliably reflect their internal decision-making processes. This highlights the need for attribution methods that reveal actual model behavior rather than relying on self-reported explanations.

2.2.5 Inseq

The Inseq toolkit [46] represents a significant advance in providing standardized interpretability analyses for sequence generation models. This Python library focuses on:

- Centralized access to multiple feature attribution methods (IG, SHAP, LIME, Attention, ...)
- Support for standard static visualizations like heatmaps
- Model-architecture independence, working with all major Hugging Face transformer models by building on PyTorch[1] and the transformers library[2].
- Structured output formats (JSON) for further analysis

The broad support for many attribution methods and models allows researchers to easily compare different attribution techniques. This flexibility makes it particularly valuable for systematic evaluation across different models and tasks.

However, Inseq on its own generates only static visualizations: The toolkit primarily offers static heatmaps rather than interactive exploration interfaces [39, 46].

The robust foundation of Inseq in supporting multiple attribution methods across diverse model architectures makes it a valuable starting point for building more comprehensive visualization tools, despite these limitations.

[1] PyTorch Website
[2] Transformers on pypi.org

2.3 Limitations of Current Approaches

Current approaches to token importance visualization in LLMs face several key limitations when considered collectively:

- **Faithfulness-plausibility tension**: As highlighted by the issues with LMExplainer and CoT explanations, tools often sacrifice faithful representation of model internals for human-understandable explanations, creating potentially misleading interpretations.
- **Computational scalability**: Methods like Integrated Gradients and SHAP face substantial computational barriers when applied to billion-parameter models, limiting their practical application. Current implementations rarely optimize these methods for large-scale deployment.
- **Limited interactivity**: Most current tools provide static visualizations rather than supporting interactive exploration of model behavior. As [39] demonstrate, effective visualization requires meaningful interaction to support progressive discovery and hypothesis testing.
- **Fragmented ecosystem**: Different tools emphasize various aspects of interpretability, with no comprehensive solution addressing all needs. This creates integration challenges for researchers attempting to conduct thorough analyses.
- **Inadequate evaluation frameworks**: Tools often lack standardized evaluation of whether visualizations actually improve user understanding of model behavior or support accurate mental model development.
- **Poor integration with domain knowledge**: Current approaches rarely connect token importance to broader semantic structures or domain-specific knowledge, limiting contextual understanding of model behavior.

These limitations highlight the need for more integrated well visualised approaches to token importance visualization; effective tools must provide both global overviews and detailed, interactive exploration capabilities while maintaining fidelity to the model's actual processing mechanisms.

3 Core Mathematical Concepts

Based on the identified challenges in LLM interpretability and limitations of existing approaches, this section formalizes the requirements and development goals for the proposed attribution visualization framework. These specifications are established through a combination of mathematical rigor and human-centered design principles.

3.1 Theoretical Foundation Requirements

The mathematical framework underpinning the attribution visualization system is defined as follows:

Let $\mathcal{M} = \{m_1, m_2, \ldots, m_n\}$ be the set of available (transformer based) generation models. Let $\mathcal{F} = \{f_1, f_2, \ldots, f_k\}$ be the set of attribution methods, encompassing gradient-based, internals-based, and perturbation-based approaches.

For any attribution function ϕ, the mapping of an input-output pair to a matrix of importance scores is formalized as:

$$\phi_f^m(X, Y, o) = \mathbf{A}^{m,f,X,Y} \in \mathbb{R}^{n \times t} \tag{2}$$

where:

- $X = (x_1, x_2, \ldots, x_n)$ represents the input token sequence
- $Y = (y_1, y_2, \ldots, y_t)$ represents the output token sequence
- $\mathbf{A}_{i,j}^{m,f,X,Y}$ denotes the attribution score of input token x_i for output token y_j

- $o \in \mathcal{O}$ represents the set of options applied to attribution calculations

Given the limitations of individual attribution methods highlighted in Section 2, the system must support multiple attribution approaches within a unified interface. Cross-model compatibility must be ensured through abstraction of model-specific details, handled by Inseq (see Appendix A.6).

3.2 Computational and Processing Requirements

Attribution calculations for LLMs are computationally intensive, necessitating a job-based processing architecture. A job J is defined as:

$$J = (\mathcal{M}', \mathcal{F}', i, o) \tag{3}$$

where:

- $\mathcal{M}' \subseteq \mathcal{M}$ is the subset of selected models
- $\mathcal{F}' \subseteq \mathcal{F}$ is the subset of selected attribution methods
- $i \in \mathcal{I}$ is a single input text, where $\mathcal{I}$ represents the set of possible input texts
- $o \in \mathcal{O}$ is a set of options applied to all model-method combinations

Each job processes all combinations in the Cartesian product $\mathcal{M}' \times \mathcal{F}'$. The system must support asynchronous processing, parallel execution, effective resource management, and result caching for improved user experience. Detailed API specifications are provided in Appendix A.1.

3.3 Aggregation Functions

To enable meaningful interactive analysis, several key aggregation functions are defined:

- **Input-centered aggregation** (when focusing on input token x_k):

$$\mathbf{A}_{x_k,c}(j) = \sum_{i=\max(1,k-\lfloor c/2 \rfloor)}^{\min(n,k+\lfloor c/2 \rfloor)} \mathbf{A}_{i,j} \tag{4}$$

- **Output-centered aggregation** (when focusing on output token y_k):

$$\mathbf{A}_{y_k,c}(i) = \sum_{j=\max(1,k-\lfloor c/2 \rfloor)}^{\min(t,k+\lfloor c/2 \rfloor)} \mathbf{A}_{i,j} \tag{5}$$

- **Token importance aggregation** (for default visualization):

$$I(x_i) = \sum_{j=1}^{t} \mathbf{A}_{i,j} \quad \text{and} \quad I(y_j) = \sum_{i=1}^{n} \mathbf{A}_{i,j} \tag{6}$$

Additional aggregation functions for multi-dimensional methods and multi-method combinations are detailed in Appendix A.3.

3.4 Visualization Requirements

The system must visually encode attribution scores $\mathbf{A}_{i,j}$ through appropriate perceptual channels while maintaining text readability. For effective visualization, raw attribution scores must be transformed into a standardized range. This normalized attribution intensity, which will later be mapped to visual properties such as color opacity (see Section 4.4), is defined as:

$$\text{intensity}(\mathbf{A}_{i,j}) = \frac{\mathbf{A}_{i,j} - \min(\mathbf{A})}{\max(\mathbf{A}) - \min(\mathbf{A})} \in [0, 1] \tag{7}$$

Beyond static visualization, the system must support dynamic exploration through context window adjustment, method toggling, token hovering interactions, model comparison, and linguistic category filtering. The context window size parameter c is constrained as $1 \leq c \leq \lfloor \min(n, t)/2 \rfloor$.

For multi-turn conversations, the follow-up input for turn k is constructed as:

$$X_k = (X_1, Y_1, X_2, Y_2, \ldots, X_{k-1}, Y_{k-1}, X_k) \tag{8}$$

where each X_i and Y_i represent the user input and model output at turn i.

4 System Architecture and Visualization Methodology

This section introduces the conceptual framework for token attribution visualization, detailing its proposed architecture and key visualization methodologies. The proposed system is designed to address the requirements outlined in section 3.

4.1 System Overview and Backend Pipeline

The envisioned system provides an interactive visualization platform for exploring attribution information in sequence generation models, employing a job-based API architecture with Inseq as the backend attribution engine and featuring a reactive frontend for intuitive interaction with attribution data.

Using the formal definitions from section 3.1, the system operates on the fundamental sets $\mathcal{M}$ (available models), $\mathcal{F}$ (attribution methods), $\mathcal{I}$ (possible input texts), and $\mathcal{O}$ (method options). The computational unit is a job $J = (\mathcal{M}', \mathcal{F}', i, o)$ as defined in Section 3.2, processing all combinations in the Cartesian product $\mathcal{M}' \times \mathcal{F}'$.

For each tuple $(m, f) \in \mathcal{M}' \times \mathcal{F}'$, the backend employs Inseq [46] to:

1. Load the model: `attribution_model = inseq.load_model(m, f)`
2. Compute attributions: `attributions = model.attribute(i, **o)`
3. Process and store results in the job's result collection

The job-based API exposes four key endpoints: job creation (for initiating new attribution jobs), job status (for monitoring progress), job results (for retrieving completed analysis), and WebSocket notification (for real-time updates). The complete API specification is provided in Appendix A.1.

4.2 Result Data Structure

The result structure is designed to balance completeness with efficient transmission and rendering. Each job result includes job metadata, input text and tokenization, options, and individual results for each model-method combination. For each tuple (m, f), the result contains model and method identifiers, generated output, raw attribution matrix $\mathbf{A}^{m,f,X,Y} \in \mathbb{R}^{n \times t}$ as defined in section 3.1, pre-computed token importance aggregations $I(x_i)$ and $I(y_j)$ from section 3.3, and step scores for metrics such as probability and entropy.

The pre-computed aggregations are included to enable immediate visualization. The complete attribution matrix is attached for detailed exploration to be calculated client-side. The detailed schema is provided in Appendix A.2.

4.3 Frontend Interface and Visualization Techniques

The frontend comprises three primary views forming a coherent user flow:

The **Job Creation View** enables queuing of new attribution jobs through model selection, method selection, input text entry, and options configuration. Upon submission, the system creates a job with the specified parameters.

The **Job Overview** provides a centralized dashboard for managing running and completed jobs, with real-time status updates via WebSocket connection. When a job completes, it moves from Running to Completed status with a visual transition effect.

The **Attribution Dashboard** serves as the core interactive component for exploring attribution relationships. It features header controls (model selector, method toggles, context window slider, etc.), the main visualization area, and a follow-up job creation interface. This dashboard implements the visualization requirements specified insection 3.4. How the visualisation implements interactivity (which is a major benefit of the proposed system) is content of the following section 4.4.

4.4 Interactive Attribution Visualization

The core visualization interactively reveals attribution relationships between input and output tokens: For each attribution score $\mathbf{A}_{i,j}$, the visualization applies the normalization function defined in section 3.4. This normalized value is mapped to visual properties such as background color opacity.

The color encoding scheme assigns each attribution method a distinct color from a perceptually distinct palette, with appropriate blending techniques for combined visualization. A clear visual hierarchy maintains text readability while conveying attribution information through background color, border style, and typography.

Four key interactive elements drive the exploration experience:

Token Hovering: When a user hovers over a token, the system highlights the hovered token and computes attribution relationships to all other tokens using the appropriate aggregation function from Section 3.3. This reveals specific token relationships dynamically.

Context Window Control: A slider adjusts the context window size parameter c within the constraints defined in Section 3.4. As this value changes, the system immediately updates the visualization using the corresponding aggregation functions.

Method Selection: Toggle buttons enable independent toggling of attribution methods, with visual feedback indicating active methods. When multiple methods are active, each is assigned a distinct color (e.g., red and blue), with overlapping attributions naturally creating blended visual effects (e.g., purple) in areas where both methods indicate importance.

Model Comparison: A split view toggle enables side-by-side comparison of different models, maintaining synchronized interactions between views for direct comparison of attribution patterns.

The default visualization presents each token colored by its overall importance ($I(x_i)$ for input tokens and $I(y_j)$ for output tokens) as defined in section 3.3, providing an immediate visual summary of influential tokens.

4.5 Advanced Visualization Capabilities

Beyond the core visualization, the system offers several advanced capabilities:

The before-mentioned toggle activates the **Model Comparison View**, which enables direct comparison of attribution patterns between different models. This split view synchronizes interactions across models and can optionally highlight differences through a differential attribution matrix $\mathbf{A}^{\text{diff}}_{i,j} = \mathbf{A}^{m_1}_{i,j} - \mathbf{A}^{m_2}_{i,j}$. Even when models generate different outputs for the same input, the system maintains appropriate alignment for comparison.

Conversational Follow-up Analysis traces attribution patterns through multi-turn conversations using the conversation construction formula from section 3.4. When viewing results from follow-up jobs, the system provides a conversation timeline, historical context highlighting showing how previous tokens influence current generation, and attribution flow analysis revealing evolving patterns across turns.

Linguistic Category Filtering enables analysis based on linguistic categories through integration with external NLP services. After enriching tokens with linguistic annotations, the system may support filtering by categories such as part-of-speech, named entities, and syntactic roles. Three visualization modes can be pictured: highlight mode (emphasizing selected categories), isolation mode (showing only selected categories), and comparison mode (displaying different categories with distinct visual encodings).

To ensure responsive interactions despite the computational intensity of attribution analysis, the system implements multiple optimization strategies: pre-computed aggregations for immediate visualization, on-demand client-side aggregation during interactions, throttled computations for larger matrices, and progressive loading for very large matrices.[3].

5 Future Directions

While the attribution visualization system proposed would provide a comprehensive framework for exploring token importance in large language models, several promising directions for further enhancement could be explored should this conceptual framework be implemented. The following section outlines potential extensions to the current system that address challenges in LLM interpretability and visualization.

5.1 Advanced Analysis Capabilities

The current system focuses on visualizing attribution relationships as computed by existing methods (supported by Inseq [46]). Future work could implement automated pattern recognition algorithms that identify recurring attribution structures across different inputs and models.

Additionally, the system could be extended with quantitative attribution metrics for evaluating quality and patterns. These metrics might include attribution concentration measures quantifying how focused or distributed scores are across tokens; stability metrics evaluating consistency across similar inputs; method comparison metrics quantifying agreement between different techniques; and linguistic alignment measures correlating attribution patterns with linguistic structures. Such metrics might provide objective measures for systematic comparison and evaluation.

A powerful extension would be tighter integration with counterfactual analysis (see [44]), enabling exploration of how changes to input tokens affect both model outputs and attribution patterns. This integration could feature automated generation of linguistically valid counterfactual inputs; side-by-side visualization of attribution patterns before and after modifications; sensitivity analysis identifying which tokens produce significant changes when modified; and adversarial input generation to identify fragile attribution patterns. Such capabilities would help understand the stability and robustness of model behavior through direct manipulation of inputs.

5.2 AR Visualization Techniques

For exploring complex attribution spaces, augmented reality (AR) could provide more immersive and intuitive interfaces. AR applications might include spatial arrangement of tokens in three-dimensional space based on attribution relationships and gestural interactions for manipulating attribution parameters. While further research is necessary, such interfaces might leverage spatial cognition for new ways to interact with complex attribution data and offer a new way of presenting the very complex information the proposed tool is meant to tackle in two dimensions more effectively and intuitively.

5.3 Integration with Model Development

A particularly valuable direction would be tighter integration between attribution visualization and model development workflows. The system could support attribution-guided fine-tuning processes by identifying problematic attribution patterns indicating model weaknesses or biases; generating targeted training examples

[3]also see A.5

to address specific issues; and monitoring attribution patterns during fine-tuning to assess improvement. This would create a direct feedback loop between model analysis and improvement.

The system could incorporate tools specifically designed for prompt engineering support through analyzing how different prompt formulations affect attribution patterns; suggesting prompt modifications to improve clarity or reduce biases; providing comparative analysis of prompt variants to identify optimal formulations; and automatically generating templates based on desired attribution characteristics. Such capabilities would help develop more effective prompts by revealing how different elements influence model processing.

6 Conclusion

This paper proposes a comprehensive conceptual framework for visualizing and analyzing token attribution in large language models. Through formal mathematical foundations, a job-based processing architecture, and an interactive visualization interface, the approach aims to make complex LLM behavior more transparent and interpretable.

We address specific critical gaps in existing interpretability tools: the inability to compare different attribution methods within a unified interface, the lack of interactive exploration capabilities in tools like Inseq, and the computational scalability challenges that current methods face when applied to billion-parameter models. By addressing these limitations directly, the framework enables more comprehensive and nuanced analysis of token importance in modern LLMs. However, it is important to acknowledge that the framework remains conceptual at this stage, and several challenges must be addressed in moving toward implementation.

Future work building on this conceptual foundation should involve developing a prototype system that implements the core components of the framework. Such an implementation would allow for evaluation along multiple dimensions: computational efficiency, usability by different stakeholder groups, and—most importantly—whether the approach genuinely improves understanding of model behavior. Evaluation methodologies should combine quantitative metrics (such as task completion time and accuracy for interpretability tasks) with qualitative assessment through case studies involving domain experts.

Implementation priorities should focus on (1) developing the core attribution processing pipeline with support for at least two complementary attribution methods, (2) creating an initial interactive visualization interface with basic exploration capabilities, and (3) conducting user studies with ML researchers to refine the approach. The modular design of the framework allows for incremental implementation and evaluation.

The potential impact of this work extends beyond technical contributions to ML interpretability. In healthcare, for instance, the framework could help clinicians understand how medical LLMs weigh different symptoms in forming diagnostic suggestions. In legal applications, it could reveal how models prioritize different precedents or statutes when generating analysis. These domain-specific insights are essential for responsible deployment in high-stakes environments.

This research aligns with broader initiatives in AI transparency and governance, including emerging regulatory requirements such as those outlined in recent transparency provisions that may soon mandate explainability for AI systems deployed in critical domains. There are potential criticisms regarding the computational feasibility of interactive attribution visualization for billion-parameter models, but the optimization strategies and job-based architecture are designed specifically to address these concerns through efficient resource management and progressive loading techniques.

The opacity of LLMs represents one of the most significant barriers to their trustworthy integration into society. By providing a rigorous yet accessible approach to understanding token importance, this framework represents a step toward LLMs that can be meaningfully audited, improved, and aligned with human values. The true value of this work will ultimately be measured by its ability to empower diverse stakeholders—from model developers to domain experts to regulators—with actionable insights into how these increasingly influential systems process and prioritize information.

A Appendix

A.1 API Specification

This appendix provides the specification for the system's REST API, enabling programmatic interaction with the attribution visualization framework.

Job Creation Endpoint Creates a new attribution job with specified models, methods, input, and options.

```
POST /api/jobs
{
  "models": ["model_1", "model_2", ...],
  "methods": ["method_1", "method_2", ...],
  "input": "input text",
  "options": { ... }
}
```

Job Status Endpoint Provides the current status of a job.

```
GET /api/jobs/{job_id}/status
```

Job Results Endpoint Returns complete results when a job is completed.

```
GET /api/jobs/{job_id}/results
```

WebSocket Notification Sends real-time updates on job status changes.

```
WS /api/ws/jobs/{job_id}
```

A.2 Result Data Structure Schema

The complete job result structure follows this schema:

```
{
  "job_id": "unique_job_id",
  "created_at": "ISO datetime",
  "completed_at": "ISO datetime",
  "input": {
    "text": "original input text",
    "tokens": ["x_1", "x_2", ..., "x_n"]
  },
  "options": { ... },
  "results": [
    {
      "model": "model_name",
      "method": "method_name",
      "output": {
        "text": "generated output text",
        "tokens": ["y_1", "y_2", ..., "y_t"]
      },
      "attribution": {
        "raw": [ ... ],  // Attribution matrix
        "aggregated": {
          "token_importance": { ... }
        }
      },
      "step_scores": { ... }
    },
    ...
  ]
}
```

A.3 Advanced Aggregation Functions

In addition to the core aggregation functions described in Section 3.3, the system implements several advanced aggregation methods:

Dimensional Aggregation For multi-dimensional attribution methods:

$$\mathbf{A}_{i,j} = \text{aggregate}(\mathbf{A}_{i,j,1}, \mathbf{A}_{i,j,2}, \ldots, \mathbf{A}_{i,j,d}) \tag{9}$$

Where aggregation can be L2 Norm: $\sqrt{\sum_{k=1}^{d} \mathbf{A}_{i,j,k}^2}$, Sum: $\sum_{k=1}^{d} \mathbf{A}_{i,j,k}$, or Mean: $\frac{1}{d}\sum_{k=1}^{d} \mathbf{A}_{i,j,k}$.

Multi-method Aggregation When multiple methods are active:

$$\mathbf{A}_{i,j}^{\text{combined}} = \frac{1}{|F'|} \sum_{f \in F'} \mathbf{A}_{i,j}^{f} \tag{10}$$

Temporal Aggregation For analyzing attribution across conversation turns:

$$\mathbf{A}_{i,j,k}^{\text{temporal}} = \sum_{t=1}^{k} \alpha_t \cdot \mathbf{A}_{i,j}^{t} \tag{11}$$

Where α_t is a weighting factor for turn t and $\mathbf{A}_{i,j}^{t}$ is the attribution score at turn t.

A.4 Frontend Views

Job Creation View Includes model selection (multi-select with metadata), method selection (with compatibility indicators), input text area (with token visualization), options configuration (with method-specific options), template selection, and submit button.

Job Overview Provides running jobs table (with progress indicators), completed jobs table (with status and view buttons), filter controls, sort controls, search field, and batch operations.

Attribution Dashboard Features header controls, visualization area (with hover, click, and drag interactions), attribution details panel, method comparison panel, token statistics, export controls, and follow-up interface.

A.5 Performance Optimizations

Parallel Processing Implemented through task queue architecture with priority-based scheduling, worker pool management with dynamic scaling, resource allocation algorithms, and fine-grained task decomposition.

Caching Strategy Multi-level caching includes model cache (loaded models with LRU eviction), tokenization cache (indexed by text and model), attribution cache (with content-based hashing), and aggregation cache (pre-computed common aggregations).

Progressive Loading Divides large attribution matrices into tiles based on viewport, implements priority loading based on user interactions, uses background loading during idle periods, and maintains a local cache of loaded tiles.

A.6 Implementation Technologies

Backend Technologies The system builds upon Inseq for attribution computation, wrapped in a job-based API structure. The backend uses a modular architecture with a job management service, attribution computation engine, API and WebSocket service, and results database. Performance optimizations include parallel processing, caching, resource management, and compressed data transmission.

Frontend Technologies The React/TypeScript application features a layered component architecture with core visualization components, specialized view components, and layout components. State management uses a structured approach with global application state, component-local state, and derived state. Visualization optimizations include virtualized rendering, throttled event handlers, web workers for heavy computation, memoization, and efficient DOM updates.

Inseq Integration The system utilizes Inseq's core functions for model loading and attribution computation while extending its visualization capabilities with interactive features. An optimization layer pre-computes common aggregations, optimizes memory usage, implements caching strategies, and provides progressive loading for large datasets.

References

[1] L. De Angelis, F. Baglivo, G. Arzilli, G. P. Privitera, P. Ferragina, A. E. Tozzi, and C. Rizzo, "Chatgpt and the rise of large language models: the new ai-driven infodemic threat in public health," *Frontiers in public health*, vol. 11, p. 1166120, 2023.

[2] K. Carolan, L. Fennelly, and A. F. Smeaton, "A review of multi-modal large language and vision models," *ArXiv*, vol. abs/2404.01322, 2024.

[3] M. Ferdaus, M. Abdelguerfi, E. Ioup, K. N. Niles, K. Pathak, and S. Sloan, "Towards trustworthy ai: A review of ethical and robust large language models," *ArXiv*, vol. abs/2407.13934, 2024.

[4] M. A. Babu, K. M. Yusuf, L. N. Eni, S. M. S. Jaman, and M. R. Sharmin, "Chatgpt and generation 'z': A study on the usage rates of chatgpt," *Social Sciences & Humanities Open*, vol. 10, p. 101163, 2024.

[5] A. Humlum and E. Vestergaard, "The adoption of chatgpt," IZA Discussion Papers 16992, Bonn, 2024.

[6] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, "Improving language understanding by generative pre-training," 2018.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (J. Burstein, C. Doran, and T. Solorio, eds.), (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.

[8] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 1877–1901, Curran Associates, Inc., 2020.

[9] OpenAI, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, R. Avila, I. Babuschkin, S. Balaji, V. Balcom, P. Baltescu, H. Bao, M. Bavarian, J. Belgum, I. Bello, J. Berdine, G. Bernadett-Shapiro, C. Berner, L. Bogdonoff, O. Boiko, M. Boyd, A.-L. Brakman, G. Brockman, T. Brooks, M. Brundage, K. Button, T. Cai, R. Campbell, A. Cann, B. Carey, C. Carlson, R. Carmichael, B. Chan, C. Chang, F. Chantzis, D. Chen, S. Chen, R. Chen, J. Chen, M. Chen, B. Chess, C. Cho, C. Chu, H. W. Chung, D. Cummings, J. Currier, Y. Dai, C. Decareaux, T. Degry, N. Deutsch, D. Deville, A. Dhar, D. Dohan, S. Dowling, S. Dunning, A. Ecoffet, A. Eleti, T. Eloundou, D. Farhi, L. Fedus, N. Felix, S. P. Fishman, J. Forte, I. Fulford, L. Gao, E. Georges, C. Gibson, V. Goel, T. Gogineni, G. Goh, R. Gontijo-Lopes, J. Gordon, M. Grafstein, S. Gray, R. Greene, J. Gross, S. S. Gu, Y. Guo, C. Hallacy, J. Han, J. Harris, Y. He, M. Heaton, J. Heidecke, C. Hesse, A. Hickey, W. Hickey, P. Hoeschele, B. Houghton, K. Hsu, S. Hu, X. Hu, J. Huizinga, S. Jain, S. Jain, J. Jang, A. Jiang, R. Jiang, H. Jin, D. Jin, S. Jomoto, B. Jonn, H. Jun, T. Kaftan, Łukasz Kaiser, A. Kamali, I. Kanitscheider, N. S. Keskar, T. Khan, L. Kilpatrick, J. W. Kim, C. Kim, Y. Kim, J. H. Kirchner, J. Kiros, M. Knight, D. Kokotajlo, Łukasz Kondraciuk, A. Kondrich, A. Konstantinidis, K. Kosic, G. Krueger, V. Kuo, M. Lampe, I. Lan, T. Lee, J. Leike, J. Leung, D. Levy, C. M. Li, R. Lim, M. Lin, S. Lin, M. Litwin, T. Lopez, R. Lowe, P. Lue, A. Makanju, K. Malfacini, S. Manning, T. Markov, Y. Markovski, B. Martin, K. Mayer, A. Mayne, B. McGrew, S. M. McKinney, C. McLeavey, P. McMillan, J. McNeil, D. Medina, A. Mehta, J. Menick, L. Metz, A. Mishchenko, P. Mishkin, V. Monaco, E. Morikawa, D. Mossing, T. Mu, M. Murati, O. Murk, D. Mély, A. Nair, R. Nakano, R. Nayak, A. Neelakantan, R. Ngo, H. Noh, L. Ouyang, C. O'Keefe, J. Pachocki, A. Paino, J. Palermo, A. Pantuliano, G. Parascandolo, J. Parish, E. Parparita, A. Passos, M. Pavlov, A. Peng, A. Perelman, F. de Avila Belbute Peres, M. Petrov, H. P. de Oliveira Pinto, Michael, Pokorny, M. Pokrass, V. H. Pong, T. Powell, A. Power, B. Power, E. Proehl, R. Puri, A. Radford, J. Rae,

A. Ramesh, C. Raymond, F. Real, K. Rimbach, C. Ross, B. Rotsted, H. Roussez, N. Ryder, M. Saltarelli, T. Sanders, S. Santurkar, G. Sastry, H. Schmidt, D. Schnurr, J. Schulman, D. Selsam, K. Sheppard, T. Sherbakov, J. Shieh, S. Shoker, P. Shyam, S. Sidor, E. Sigler, M. Simens, J. Sitkin, K. Slama, I. Sohl, B. Sokolowsky, Y. Song, N. Staudacher, F. P. Such, N. Summers, I. Sutskever, J. Tang, N. Tezak, M. B. Thompson, P. Tillet, A. Tootoonchian, E. Tseng, P. Tuggle, N. Turley, J. Tworek, J. F. C. Uribe, A. Vallone, A. Vijayvergiya, C. Voss, C. Wainwright, J. J. Wang, A. Wang, B. Wang, J. Ward, J. Wei, C. Weinmann, A. Welihinda, P. Welinder, J. Weng, L. Weng, M. Wiethoff, D. Willner, C. Winter, S. Wolrich, H. Wong, L. Workman, S. Wu, J. Wu, M. Wu, K. Xiao, T. Xu, S. Yoo, K. Yu, Q. Yuan, W. Zaremba, R. Zellers, C. Zhang, M. Zhang, S. Zhao, T. Zheng, J. Zhuang, W. Zhuk, and B. Zoph, "Gpt-4 technical report," 2024.

[10] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, A. Yang, A. Fan, A. Goyal, A. Hartshorn, A. Yang, A. Mitra, A. Sravankumar, A. Korenev, A. Hinsvark, A. Rao, A. Zhang, A. Rodriguez, A. Gregerson, A. Spataru, B. Roziere, B. Biron, B. Tang, B. Chern, C. Caucheteux, C. Nayak, C. Bi, C. Marra, C. McConnell, C. Keller, C. Touret, C. Wu, C. Wong, C. C. Ferrer, C. Nikolaidis, D. Allonsius, D. Song, D. Pintz, D. Livshits, D. Wyatt, D. Esiobu, D. Choudhary, D. Mahajan, D. Garcia-Olano, D. Perino, D. Hupkes, E. Lakomkin, E. AlBadawy, E. Lobanova, E. Dinan, E. M. Smith, F. Radenovic, F. Guzmán, F. Zhang, G. Synnaeve, G. Lee, G. L. Anderson, G. Thattai, G. Nail, G. Mialon, G. Pang, G. Cucurell, H. Nguyen, H. Korevaar, H. Xu, H. Touvron, I. Zarov, I. A. Ibarra, I. Kloumann, I. Misra, I. Evtimov, J. Zhang, J. Copet, J. Lee, J. Geffert, J. Vranes, J. Park, J. Mahadeokar, J. Shah, J. van der Linde, J. Billock, J. Hong, J. Lee, J. Fu, J. Chi, J. Huang, J. Liu, J. Wang, J. Yu, J. Bitton, J. Spisak, J. Park, J. Rocca, J. Johnstun, J. Saxe, J. Jia, K. V. Alwala, K. Prasad, K. Upasani, K. Plawiak, K. Li, K. Heafield, K. Stone, K. El-Arini, K. Iyer, K. Malik, K. Chiu, K. Bhalla, K. Lakhotia, L. Rantala-Yeary, L. van der Maaten, L. Chen, L. Tan, L. Jenkins, L. Martin, L. Madaan, L. Malo, L. Blecher, L. Landzaat, L. de Oliveira, M. Muzzi, M. Pasupuleti, M. Singh, M. Paluri, M. Kardas, M. Tsimpoukelli, M. Oldham, M. Rita, M. Pavlova, M. Kambadur, M. Lewis, M. Si, M. K. Singh, M. Hassan, N. Goyal, N. Torabi, N. Bashlykov, N. Bogoychev, N. Chatterji, N. Zhang, O. Duchenne, O. Çelebi, P. Alrassy, P. Zhang, P. Li, P. Vasic, P. Weng, P. Bhargava, P. Dubal, P. Krishnan, P. S. Koura, P. Xu, Q. He, Q. Dong, R. Srinivasan, R. Ganapathy, R. Calderer, R. S. Cabral, R. Stojnic, R. Raileanu, R. Maheswari, R. Girdhar, R. Patel, R. Sauvestre, R. Polidoro, R. Sumbaly, R. Taylor, R. Silva, R. Hou, R. Wang, S. Hosseini, S. Chennabasappa, S. Singh, S. Bell, S. S. Kim, S. Edunov, S. Nie, S. Narang, S. Raparthy, S. Shen, S. Wan, S. Bhosale, S. Zhang, S. Vandenhende, S. Batra, S. Whitman, S. Sootla, S. Collot, S. Gururangan, S. Borodinsky, T. Herman, T. Fowler, T. Sheasha, T. Georgiou, T. Scialom, T. Speckbacher, T. Mihaylov, T. Xiao, U. Karn, V. Goswami, V. Gupta, V. Ramanathan, V. Kerkez, V. Gonguet, V. Do, V. Vogeti, V. Albiero, V. Petrovic, W. Chu, W. Xiong, W. Fu, W. Meers, X. Martinet, X. Wang, X. Wang, X. E. Tan, X. Xia, X. Xie, X. Jia, X. Wang, Y. Goldschlag, Y. Gaur, Y. Babaei, Y. Wen, Y. Song, Y. Zhang, Y. Li, Y. Mao, Z. D. Coudert, Z. Yan, Z. Chen, Z. Papakipos, A. Singh, A. Srivastava, A. Jain, A. Kelsey, A. Shajnfeld, A. Gangidi, A. Victoria, A. Goldstand, A. Menon, A. Sharma, A. Boesenberg, A. Baevski, A. Feinstein, A. Kallet, A. Sangani, A. Teo, A. Yunus, A. Lupu, A. Alvarado, A. Caples, A. Gu, A. Ho, A. Poulton, A. Ryan, A. Ramchandani, A. Dong, A. Franco, A. Goyal, A. Saraf, A. Chowdhury, A. Gabriel, A. Bharambe, A. Eisenman, A. Yazdan, B. James, B. Maurer, B. Leonhardi, B. Huang, B. Loyd, B. D. Paola, B. Paranjape, B. Liu, B. Wu, B. Ni, B. Hancock, B. Wasti, B. Spence, B. Stojkovic, B. Gamido, B. Montalvo, C. Parker, C. Burton, C. Mejia, C. Liu, C. Wang, C. Kim, C. Zhou, C. Hu, C.-H. Chu, C. Cai, C. Tindal, C. Feichtenhofer, C. Gao, D. Civin, D. Beaty, D. Kreymer, D. Li, D. Adkins, D. Xu, D. Testuggine, D. David, D. Parikh, D. Liskovich, D. Foss, D. Wang, D. Le, D. Holland, E. Dowling, E. Jamil, E. Montgomery, E. Presani, E. Hahn, E. Wood, E.-T. Le, E. Brinkman, E. Arcaute, E. Dunbar, E. Smothers, F. Sun, F. Kreuk, F. Tian, F. Kokkinos, F. Ozgenel, F. Caggioni, F. Kanayet, F. Seide, G. M. Florez, G. Schwarz, G. Badeer, G. Swee, G. Halpern, G. Herman, G. Sizov, Guangyi, Zhang, G. Lakshminarayanan, H. Inan, H. Shojanazeri, H. Zou, H. Wang, H. Zha, H. Habeeb, H. Rudolph, H. Suk, H. Aspegren, H. Goldman, H. Zhan, I. Damlaj, I. Molybog, I. Tufanov, I. Leontiadis, I.-E. Veliche, I. Gat, J. Weissman, J. Geboski, J. Kohli, J. Lam, J. Asher, J.-B. Gaya, J. Marcus, J. Tang, J. Chan, J. Zhen, J. Reizenstein, J. Teboul, J. Zhong, J. Jin, J. Yang, J. Cummings, J. Carvill, J. Shepard, J. McPhie, J. Torres, J. Ginsburg, J. Wang, K. Wu, K. H. U, K. Saxena, K. Khandelwal, K. Zand, K. Matosich,

K. Veeraraghavan, K. Michelena, K. Li, K. Jagadeesh, K. Huang, K. Chawla, K. Huang, L. Chen, L. Garg, L. A, L. Silva, L. Bell, L. Zhang, L. Guo, L. Yu, L. Moshkovich, L. Wehrstedt, M. Khabsa, M. Avalani, M. Bhatt, M. Mankus, M. Hasson, M. Lennie, M. Reso, M. Groshev, M. Naumov, M. Lathi, M. Keneally, M. Liu, M. L. Seltzer, M. Valko, M. Restrepo, M. Patel, M. Vyatskov, M. Samvelyan, M. Clark, M. Macey, M. Wang, M. J. Hermoso, M. Metanat, M. Rastegari, M. Bansal, N. Santhanam, N. Parks, N. White, N. Bawa, N. Singhal, N. Egebo, N. Usunier, N. Mehta, N. P. Laptev, N. Dong, N. Cheng, O. Chernoguz, O. Hart, O. Salpekar, O. Kalinli, P. Kent, P. Parekh, P. Saab, P. Balaji, P. Rittner, P. Bontrager, P. Roux, P. Dollar, P. Zvyagina, P. Ratanchandani, P. Yuvraj, Q. Liang, R. Alao, R. Rodriguez, R. Ayub, R. Murthy, R. Nayani, R. Mitra, R. Parthasarathy, R. Li, R. Hogan, R. Battey, R. Wang, R. Howes, R. Rinott, S. Mehta, S. Siby, S. J. Bondu, S. Datta, S. Chugh, S. Hunt, S. Dhillon, S. Sidorov, S. Pan, S. Mahajan, S. Verma, S. Yamamoto, S. Ramaswamy, S. Lindsay, S. Lindsay, S. Feng, S. Lin, S. C. Zha, S. Patil, S. Shankar, S. Zhang, S. Zhang, S. Wang, S. Agarwal, S. Sajuyigbe, S. Chintala, S. Max, S. Chen, S. Kehoe, S. Satterfield, S. Govindaprasad, S. Gupta, S. Deng, S. Cho, S. Virk, S. Subramanian, S. Choudhury, S. Goldman, T. Remez, T. Glaser, T. Best, T. Koehler, T. Robinson, T. Li, T. Zhang, T. Matthews, T. Chou, T. Shaked, V. Vontimitta, V. Ajayi, V. Montanez, V. Mohan, V. S. Kumar, V. Mangla, V. Ionescu, V. Poenaru, V. T. Mihailescu, V. Ivanov, W. Li, W. Wang, W. Jiang, W. Bouaziz, W. Constable, X. Tang, X. Wu, X. Wang, X. Wu, X. Gao, Y. Kleinman, Y. Chen, Y. Hu, Y. Jia, Y. Qi, Y. Li, Y. Zhang, Y. Zhang, Y. Adi, Y. Nam, Yu, Wang, Y. Zhao, Y. Hao, Y. Qian, Y. Li, Y. He, Z. Rait, Z. DeVito, Z. Rosnbrick, Z. Wen, Z. Yang, Z. Zhao, and Z. Ma, "The llama 3 herd of models," 2024.

[11] M. Truell, W. Yan, J. Palmer, M. Catasta, D. Wu, C. Falls, A. Li, Y. Bhavnani, C. Allen, S. Last, and et al., "Claude 3.7 sonnet."

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[13] A. J. Thirunavukarasu, D. S. J. Ting, K. Elangovan, L. Gutierrez, T. F. Tan, and D. S. W. Ting, "Large language models in medicine," *Nature Medicine*, vol. 29, pp. 1930–1940, Aug 2023.

[14] S. Wu, O. Irsoy, S. Lu, V. Dabravolski, M. Dredze, S. Gehrmann, P. Kambadur, D. Rosenberg, and G. Mann, "Bloomberggpt: A large language model for finance," *arXiv preprint arXiv:2303.17564*, 2023.

[15] A. Abd-alrazaq, R. AlSaad, D. Alhuwail, A. Ahmed, P. M. Healy, S. Latifi, S. Aziz, R. Damseh, S. Alabed Alrazak, and J. Sheikh, "Large language models in medical education: Opportunities, challenges, and future directions," *JMIR Med Educ*, vol. 9, p. e48291, Jun 2023.

[16] J. Cui, Z. Li, Y. Yan, B. Chen, and L. Yuan, "Chatlaw: Open-source legal large language model with integrated external knowledge bases," *CoRR*, vol. abs/2306.16092, 2023.

[17] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, E. Brynjolfsson, S. Buch, D. Card, R. Castellon, N. Chatterji, A. Chen, K. Creel, J. Q. Davis, D. Demszky, C. Donahue, M. Doumbouya, E. Durmus, S. Ermon, J. Etchemendy, K. Ethayarajh, L. Fei-Fei, C. Finn, T. Gale, L. Gillespie, K. Goel, N. Goodman, S. Grossman, N. Guha, T. Hashimoto, P. Henderson, J. Hewitt, D. E. Ho, J. Hong, K. Hsu, J. Huang, T. Icard, S. Jain, D. Jurafsky, P. Kalluri, S. Karamcheti, G. Keeling, F. Khani, O. Khattab, P. W. Koh, M. Krass, R. Krishna, R. Kuditipudi, A. Kumar, F. Ladhak, M. Lee, T. Lee, J. Leskovec, I. Levent, X. L. Li, X. Li, T. Ma, A. Malik, C. D. Manning, S. Mirchandani, E. Mitchell, Z. Munyikwa, S. Nair, A. Narayan, D. Narayanan, B. Newman, A. Nie, J. C. Niebles, H. Nilforoshan, J. Nyarko, G. Ogut, L. Orr, I. Papadimitriou, J. S. Park, C. Piech, E. Portelance, C. Potts, A. Raghunathan, R. Reich, H. Ren, F. Rong, Y. Roohani, C. Ruiz, J. Ryan, C. Ré, D. Sadigh, S. Sagawa, K. Santhanam, A. Shih, K. Srinivasan, A. Tamkin, R. Taori, A. W. Thomas, F. Tramèr, R. E. Wang, W. Wang, B. Wu, J. Wu, Y. Wu, S. M. Xie, M. Yasunaga, J. You, M. Zaharia, M. Zhang, T. Zhang, X. Zhang, Y. Zhang, L. Zheng, K. Zhou, and P. Liang, "On the opportunities and risks of foundation models," 2022.

[18] L. Weidinger, J. Uesato, M. Rauh, C. Griffin, P.-S. Huang, J. Mellor, A. Glaese, M. Cheng, B. Balle, A. Kasirzadeh, C. Biles, S. Brown, Z. Kenton, W. Hawkins, T. Stepleton, A. Birhane, L. A. Hendricks, L. Rimell, W. Isaac, J. Haas, S. Legassick, G. Irving, and I. Gabriel, "Taxonomy of risks posed by language models," in *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, (New York, NY, USA), p. 214–229, Association for Computing Machinery, 2022.

[19] S. R. Bowman, "Eight things to know about large language models," 2023.

[20] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[21] Q. V. Liao and J. W. Vaughan, "Ai transparency in the age of llms: A human-centered research roadmap," *arXiv preprint arXiv:2306.01941*, vol. 10, 2023.

[22] E. Cambria, L. Malandri, F. Mercorio, N. Nobani, and A. Seveso, "Xai meets llms: A survey of the relation between explainable ai and large language models," 2024.

[23] H. Zhao, H. Chen, F. Yang, N. Liu, H. Deng, H. Cai, S. Wang, D. Yin, and M. Du, "Explainability for large language models: A survey," *ACM Trans. Intell. Syst. Technol.*, vol. 15, Feb. 2024.

[24] A. Bibal, R. Cardon, D. Alfter, R. Wilkens, X. Wang, T. François, and P. Watrin, "Is attention explanation? an introduction to the debate," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (S. Muresan, P. Nakov, and A. Villavicencio, eds.), (Dublin, Ireland), pp. 3889–3900, Association for Computational Linguistics, May 2022.

[25] N. Elhage, N. Nanda, C. Olsson, T. Henighan, N. Joseph, B. Mann, A. Askell, Y. Bai, A. Chen, T. Conerly, N. DasSarma, D. Drain, D. Ganguli, Z. Hatfield-Dodds, D. Hernandez, A. Jones, J. Kernion, L. Lovitt, K. Ndousse, D. Amodei, T. Brown, J. Clark, J. Kaplan, S. McCandlish, and C. Olah, "A mathematical framework for transformer circuits," *Transformer Circuits*, Dec 2021. Last accessed March 11, 2025.

[26] O. Antverg and Y. Belinkov, "On the pitfalls of analyzing individual neurons in language models," 2022.

[27] 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," in *Advances in Neural Information Processing Systems* (A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, eds.), vol. 36, pp. 74952–74965, Curran Associates, Inc., 2023.

[28] A. Rogers, O. Kovaleva, and A. Rumshisky, "A primer in bertology: What we know about how bert works," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 842–866, 01 2021.

[29] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proceedings of the 34th International Conference on Machine Learning* (D. Precup and Y. W. Teh, eds.), vol. 70 of *Proceedings of Machine Learning Research*, pp. 3319–3328, PMLR, 06–11 Aug 2017.

[30] M. T. Ribeiro, S. Singh, and C. Guestrin, "Model-agnostic interpretability of machine learning," 2016.

[31] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[32] J. Vig, "A multiscale visualization of attention in the transformer model," 2019.

[33] S. Jain and B. C. Wallace, "Attention is not explanation," 2019.

[34] S. Wiegreffe and Y. Pinter, "Attention is not not explanation," 2019.

[35] I. Tenney, D. Das, and E. Pavlick, "Bert rediscovers the classical nlp pipeline," 2019.

[36] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, and R. sayres, "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV)," in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, pp. 2668–2677, PMLR, 10–15 Jul 2018.

[37] A. Bau, Y. Belinkov, H. Sajjad, N. Durrani, F. Dalvi, and J. Glass, "Identifying and controlling important neurons in neural machine translation," 2018.

[38] M. Weiser, "The computer for the 21 st century," *Scientific american*, vol. 265, no. 3, pp. 94–105, 1991.

[39] E. Dimara and C. Perin, "What is interaction for data visualization?," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 1, pp. 119–129, 2020.

[40] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," 2016.

[41] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Müller, "How to explain individual classification decisions," *The Journal of Machine Learning Research*, vol. 11, pp. 1803–1831, 2010.

[42] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

[43] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proceedings of the 34th International Conference on Machine Learning* (D. Precup and Y. W. Teh, eds.), vol. 70 of *Proceedings of Machine Learning Research*, pp. 3145–3153, PMLR, 06–11 Aug 2017.

[44] T. Wu, M. T. Ribeiro, J. Heer, and D. S. Weld, "Polyjuice: Generating counterfactuals for explaining, evaluating, and improving models," 2021.

[45] Z. Chen, J. Chen, Y. Chen, H. Yu, A. K. Singh, and M. Sra, "Lmexplainer: Grounding knowledge and explaining language models," 2024.

[46] G. Sarti, N. Feldhus, L. Sickert, and O. van der Wal, "Inseq: An interpretability toolkit for sequence generation models," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Association for Computational Linguistics, 2023.