Attention Head Naming Convention for Large Language Models (LLMs)

Karol Kowalczyk

November 2025

Abstract

Large language models have reached remarkable levels of reasoning, safety alignment, and structural understanding. Yet their internal workings remain difficult to interpret. One of the most productive areas in transparency research is the study of *attention heads*—small components inside transformer layers that develop specialized behaviors. Over time, informal naming conventions have emerged in the interpretability community: *induction heads*, *name mover heads*, *refusal heads*, and many others. These names are intuitive but inconsistent, overlapping, or ambiguous.

This work proposes a unified naming convention for attention heads. I introduce: (1) a four-level depth model (Early, Middle, Late, Final), (2) a stack-based functional grouping of attention behaviors, (3) canonical names for head types, and (4) an alphabetical cross-reference table translating historical terms to standardized ones. This naming convention is descriptive rather than prescriptive: it captures how heads tend to behave today, while remaining flexible for future architectures.

Contents

1 Introduction

1.1 Motivation

Large language models (LLMs) have achieved remarkable performance across diverse tasks, yet understanding their internal mechanisms remains a critical challenge. Attention heads—the basic computational units within transformer architectures—have emerged as key objects of study in mechanistic interpretability research.

1.2 The Problem of Inconsistent Naming

The interpretability community has identified numerous specialized attention head types: *induction heads*, *name mover heads*, *refusal heads*, *delimiter heads*, *JSON heads*, and many others. However, these naming conventions suffer from several problems. They are **inconsistent**, with the same head type appearing under multiple names across papers. They are **ambiguous**, as a single name may refer to different behaviors in different contexts. They are **fragmented**, lacking any unified framework that connects related head types. Finally, they are **unscalable**, as naming schemes fail to generalize across model architectures. This fragmentation makes replication difficult, hinders cross-paper comparison, and complicates the annotation of interpretability datasets.

1.3 Goals of This Work

I propose a unified naming convention that standardizes terminology across research groups, provides a functional taxonomy grounded in empirical observations, describes head behavior consistently across architectures, and creates a stable vocabulary that can evolve as models evolve.

1.4 Circuits, Stacks, and Simplification

This taxonomy uses the concept of *stacks* as a practical organizational framework, but it is important to understand what this represents. In reality, attention heads work in complex *circuits*—groups of heads across multiple layers that cooperate to implement behaviors through multi-level processing [6, 13].

The *stack* abstraction simplifies this complexity for communication purposes. Rather than mapping every circuit connection, I group heads by their primary functional contribution. This makes the taxonomy more accessible while acknowledging that real model behavior involves intricate cross-layer interactions that this linear organization cannot fully capture.

1.5 Structure of This Document

I begin by reviewing prior work and motivation (Â§2). I then introduce the depth model (Â§3) and stack-based organization (Â§4), including how stacks simplify the underlying circuit-level

complexity. The core contribution is a comprehensive catalog of attention head types organized by functional stack (Â§5). I conclude with discussion (Â§6) and future directions (Â§7).

2 Background

2.1 Attention Heads and Functions

In transformer models [11], attention heads perform focused computations over the token sequence. Individually simple, they nevertheless develop specialized behaviors such as pattern continuation and token induction, entity and dependency tracking, semantic filtering and hazard detection, routing and topic steering, enforcing structured output formats, and applying safety constraints [6, 7]. These behaviors form *circuits*—groups of heads working together—as well as larger *stacks* of related functionality.

2.2 Why Naming Consistency Matters

Interpretability research suffers from fragmented terminology [9, 14]. The same head type may appear under multiple names, while a single overloaded name may refer to unrelated behaviors across different papers. This makes replication, comparison, and annotation difficult. A consistent naming system improves clarity and precision in communication, strengthens cross-paper alignment and replication, helps index and organize interpretability datasets, and enables systematic mapping of circuits across models.

2.3 Prior Naming Practices

Previous work has named heads based on behavior (induction, copy-suppression), formatting (JSON head, list head), signal source (delimiter head), role in circuits (name mover), or safety behavior (refusal, toxicity). These labels are often accurate but vary widely. This work unifies them under a systematic framework.

3 Depth Model: Early—Middle—Late—Final

3.1 Rationale for Four Depth Categories

Although transformer models may have 12, 48, or 96 layers, functional behavior clusters reliably into four zones [6, 13]. **Early layers (E)** handle token-level surface processing, boundary detection, and basic filtering. **Middle layers (M)** implement reasoning primitives, induction, and dependency tracking. **Late layers (L)** perform semantic integration, routing, and persona shaping. **Final layers (F)** enforce policy, safety modulation, and structured output. This structure holds across GPT, LLaMA, Claude, and other model families [5, 10, 1].

3.2 Cross-Model Depth Examples

Using *relative depth* (0.0–1.0) makes the taxonomy scale-free. For a 96-layer model, Early corresponds to layers 0–15 (relative depth 0.00–0.15), Middle to layers 15–50 (relative depth 0.15–0.52), Late to layers 50–85 (relative depth 0.52–0.88), and Final to layers 85–96 (relative depth 0.88–1.00).

3.3 Relative Depth Scaling

I express depth as a fraction of total model depth to enable cross-architecture comparison. A head at relative depth 0.40 occupies similar functional space whether in a 12-layer or 96-layer model.

4 Stacks: Functional Grouping of Attention Heads

4.1 What is a Stack?

A *stack* is a coherent group of head types that together implement a higher-level capability. Stacks reflect functional clustering observed in interpretability studies [13, 7]. Examples include the Reasoning & Algorithmic Stack, Memory & Dependency Stack, Safety Stack, and Output Formatting & Rewrite Stack. Stacks are orthogonal to depth: a stack may span Early, Middle, Late, and Final layers.

4.2 Relationship Between Stacks and Depth

Although stacks represent functional groupings, different functions tend to appear at different depths. Early layers handle delimiters, content detection, and input conditioning. Middle layers implement reasoning, induction, and entity linking. Late layers manage narrative coherence, routing, and topic steering. Final layers enforce policy, formatting, rewriting, and safety compliance. This two-dimensional structure—*stack* $\times$ *depth*—forms the basis of the catalog.

5 Attention Head Catalog

This section presents a comprehensive catalog of attention head types, organized by functional stack. Each stack groups heads that contribute to a common high-level capability. Within each stack, heads are ordered by depth (Early $\rightarrow$ Middle $\rightarrow$ Late $\rightarrow$ Final).

Entry Format. Each head entry includes:

- **Depth range:** Typical relative depth (0.0–1.0) and layer locations
- **Literature names:** Alternative names found in prior work
- **Function:** Core behavior and mechanism

- **Attention pattern:** What the heads attend to
- **Expected ablation:** Predicted effects if the heads are disabled
- **Example scenario:** Concrete behavioral illustration
- **Stack and relations:** Primary stack and related heads

5.1 Reasoning & Algorithmic Stack

Stack overview: This stack encompasses heads that perform pattern matching, sequence continuation, algorithmic reasoning, strategic planning, and meta-cognitive oversight. These heads enable in-context learning, pattern completion, systematic token prediction, and higher-level reasoning quality control.

5.1.1 (E) Previous-Token Heads

Depth: `0.05-0.18` | **Literature names:** *previous-token head, shift head, offset head*

Copy information from each token to the position of the next token, creating a shifted representation where token t contains information about token $t-1$. These heads appear to be foundational components of induction circuits, enabling later heads to access "what came before" without directly attending backwards. They implement a simple but crucial transformation that allows pattern matching across the sequence. These heads typically show strong diagonal attention patterns (attending from position i to position $i-1$).

Strong: Immediately preceding token (diagonal attention pattern)
Weak: Distant tokens, same-position tokens
Reacts to: Sequential structure, token boundaries

Expected ablation: Breaks induction circuits entirely, causing degradation in pattern completion tasks. Induction heads become unable to access "what came after previous occurrences" since that information is no longer shifted forward. Important for in-context learning.

Example Scenario
Input: "The cat sat. The cat..."
Behavior: Copy "The" to position after "The", "cat" to position after "cat", etc.
Effect: Later induction heads can match "cat" and access what followed it ("sat")

Status: WELL-DOCUMENTED | **Related:** induction head (M), duplicate-token (M)

5.1.2 (E) Local Pattern Heads

Depth: `0.08-0.20` | **Literature names:** *local pattern head, char-level head, n-gram head*

Detect and process local character-level or subword patterns, particularly useful for handling spelling, capitalization, punctuation patterns, and morphological structure. These heads operate at a finer granularity than most heads, attending to patterns within and between adjacent tokens. Important for tasks like spell checking, case handling, and recognizing common subword patterns. May also detect repeated character sequences or structural patterns like "ing", "tion", or punctuation clusters.

Strong: Adjacent tokens, subword units, character-level patterns
Weak: Long-range dependencies, semantic content
Reacts to: Spelling patterns, capitalization, punctuation, morphology

Expected ablation: Degradation in handling of misspellings, case variations, and morphological patterns. Increase in errors on tasks requiring character-level awareness. Partial fallback through tokenization and other pattern heads.

Example Scenario
Input: "The organizATION's" (unusual capitalization)
Behavior: Detect unusual case pattern in "ATION", attend to surrounding context
Effect: Help model handle non-standard capitalization correctly

Status: OBSERVED | **Related:** induction head (M), duplicate-token (M)

5.1.3 (M) Induction Heads

Depth: `0.30-0.65` | **Literature names:** *induction head, pattern head, copy head, ICL head*

Detect repeated subsequences of the form [A][B]...[A] and predict that [B] should follow the second [A]. These heads operate by attending to tokens that appeared after previous instances of the current token. They work in conjunction with previous-token heads which copy information about what preceded each token. This mechanism appears fundamental to in-context learning, enabling pattern completion, name recall, and few-shot learning without parameter updates. One of the most well-documented and important head types in transformer interpretability.

Strong: Tokens following previous occurrences of current token
Weak: Immediate neighbors, first occurrence, unrelated tokens
Reacts to: Token repetition, [A][B]...[A] patterns, contextual recurrence

Expected ablation: Significant degradation in in-context learning tasks, reduced pattern completion and few-shot learning. Model may partially compensate through other heads but with substantial accuracy loss. Important for ICL capability.

Example Scenario
Input: "When Mary and John went to the store, Mary bought..."
Behavior: Second "Mary" attends to tokens following first "Mary" (especially "and")
Effect: Increased probability of contextually appropriate continuation

Status: WELL-DOCUMENTED | **Related:** previous-token (E), duplicate-token (M), name-mover (L)

5.1.4 (M) Duplicate-Token Heads

Depth: `0.35-0.60` | **Literature names:** *duplicate-token head, repetition head, copy head*

Detect when the current token has appeared previously in the sequence, marking repeated tokens for downstream processing. Unlike induction heads which predict what comes next, duplicate-token heads simply signal "this token appeared before". This information is used by various circuits including IOI (indirect object identification), name-mover heads, and copy-suppression

mechanisms. These heads implement a simpler form of pattern matching than full induction, serving as building blocks for more complex behaviors.

Strong: Previous identical tokens (exact matches)
Weak: Similar but non-identical tokens, first occurrence
Reacts to: Exact token repetition, name recurrence, repeated phrases

Expected ablation: Impaired duplicate detection, affecting name-mover circuits and copy-suppression. Degradation in tasks requiring duplicate awareness. Partial overlap with induction heads provides some redundancy.

Example Scenario

Input: "Alice gave the book to Bob. Then Alice..."
Behavior: Second "Alice" detects it appeared earlier, writes duplicate signal
Effect: Downstream heads (name-movers, S-inhibition) use this signal

Status: Well-documented | **Related:** induction (M), name-mover (L), S-inhibition (L)

5.1.5 (M) Skip-Trigram Heads

Depth: `0.40-0.65` | **Literature names:** *skip-trigram head, skip-gram head*

Implement skip-gram pattern matching, attending to non-contiguous patterns like [A]...[B]...[C] where the dots represent intervening tokens. More flexible than strict n-gram matching, these heads allow for pattern recognition across variable distances. Useful for detecting phrasal patterns, idiomatic expressions, and structural templates with flexible word order. They generalize beyond strict adjacency requirements while maintaining pattern specificity.

Strong: Pattern components separated by 1–3 tokens
Weak: Strictly adjacent patterns, very long-range dependencies
Reacts to: Phrasal patterns, templates, flexible idioms

Expected ablation: Reduced recognition of flexible patterns and templates. Degradation on tasks requiring non-contiguous pattern matching. Less critical than induction heads; other pattern mechanisms provide fallback.

Example Scenario

Input: "not only X but also" (skip-bigram pattern)
Behavior: Recognize "not...but" pattern despite intervening tokens
Effect: Help predict "also" after "but" even with intervening content

Status: Observed | **Related:** induction (M), local-pattern (E)

5.1.6 (M) Algorithmic Continuation Heads

Depth: `0.45-0.70` | **Literature names:** *algorithmic head, continuation head, sequence head*

Recognize and continue algorithmic sequences such as counting (1, 2, 3...), days of week, months, or other systematic progressions. Distinct from general pattern matching by operating on sequences with clear algorithmic rules, these heads can detect arithmetic progressions, cyclic patterns, and other rule-governed sequences. They contribute to the model's ability to perform

basic reasoning over structured sequences without explicit training on those specific patterns.

Strong: Sequential elements in algorithmic patterns (numbers, ordered lists)
Weak: Random sequences, semantic patterns without algorithmic structure
Reacts to: Arithmetic progressions, cyclic orderings, systematic enumerations

Expected ablation: Reduced performance on sequence continuation tasks (counting, ordering). Degradation on arithmetic sequences and structured enumerations. Some algorithmic reasoning may persist through other mechanisms.

Example Scenario
Input: "Monday, Tuesday, Wednesday, ..."
Behavior: Recognize day-of-week sequence, attend to progression pattern
Effect: Strongly predict "Thursday" as next token

Status: Observed | **Related:** induction (M), digit (M)

5.1.7 (L) Strategy Heads

Depth: `0.68-0.88` | **Literature names:** *strategy head, planning head, strategy-switching head, approach-selection head, approach-adaptation head, pivot head*

Plan overall approach and strategy for complex tasks, and adapt strategy when current approaches prove ineffective. These heads influence high-level structure such as whether to break into steps, what order to address components, and which methods to apply. They operate before detailed execution to establish the strategic framework. These heads recognize different task types that may require different approaches, such as analytical versus creative reasoning, or sequential versus parallel processing. They can decompose complex queries into manageable subtasks. They also detect when the current reasoning strategy is not working effectively and switch to alternative approaches. These heads monitor problem-solving progress and recognize dead ends, insufficient methods, or the need for different techniques. They may pivot from analytical to creative approaches, from depth-first to breadth-first exploration, or from deductive to inductive reasoning. Useful for robust problem-solving across varied challenges and effective handling of multi-part or complex queries.

Strong: Task complexity indicators, multi-part queries, strategic choice points, progress indicators, approach effectiveness signals
Weak: Simple single-step tasks, purely reactive responses, successfully progressing solutions
Reacts to: Complex tasks, planning requests, multi-step problems, strategic decision points, signs of insufficient progress

Expected ablation: Reduced strategic planning quality and adaptability. May jump into execution without appropriate planning or continue unproductive approaches longer than optimal. Complex tasks handled less efficiently with reduced ability to recover from initially incorrect approaches.

Example Scenario

Input: "Help me plan a machine learning project to predict customer churn."

Behavior: Recognize need for structured planning, break into phases

Effect: Response structures approach: data collection → exploratory analysis → feature engineering → model selection → evaluation

Status: OBSERVED | **Related:** reasoning-oversight (F)

5.1.8 (F) Reasoning-Oversight Heads

Depth: `0.88-0.99` | **Literature names:** *reasoning-mode head, thinking-style head, cognitive-mode head, reasoning head, meta-reasoning head, meta-CoT head, reasoning-reflection head, thought-monitoring head, cognitive-oversight head, reasoning-quality head*

Provide highest-level management of reasoning processes, including mode selection and quality monitoring. These heads select and maintain appropriate reasoning modes for different tasks (analytical mode for precise logical problems, creative mode for brainstorming, analogical mode for novel domains, deductive versus inductive reasoning for different inference types) while simultaneously monitoring the quality and direction of reasoning chains. They identify when more thought is needed, detect reasoning errors or gaps, flag uncertain conclusions, and can trigger re-thinking or additional reasoning steps. Operating at a meta-level above regular chain-of-thought reasoning, these heads help ensure reasoning chains are sound, complete, and appropriately matched to task requirements. They influence which reasoning patterns and heuristics become active and act as cognitive oversight to help prevent confident errors in complex reasoning scenarios.

Strong: Task type indicators, reasoning mode cues, cognitive approach requirements, reasoning quality indicators, logical gaps, uncertainty signals, reasoning chain progress, consistency checks

Weak: Mode-independent content, simple factual responses, non-reasoning tasks, straightforward answers

Reacts to: Problem types, explicit mode requests, task characteristics suggesting optimal approach, complex reasoning tasks, logical steps, argument quality, reasoning errors, inconsistencies

Expected ablation: Less appropriate reasoning mode selection and reduced reasoning quality oversight. May use analytical mode for creative tasks or vice versa, reducing effectiveness. More logical gaps, less self-correction, reduced reasoning completeness. Chain-of-thought reasoning becomes less reliable, particularly for complex or multi-step problems. Reasoning remains functional but less well-suited to specific task requirements.

Example Scenario

Input: "Brainstorm creative names" or [Complex logical problem requiring multi-step reasoning]

Behavior: Select creative/generative mode rather than analytical; monitor reasoning chain, detect gap: "Wait, I should verify this assumption..."

Effect: Free-flowing creative suggestions + improved reasoning quality through self-monitoring

Status: Observed | **Related:** strategy (L), step-by-step (F)

5.2 Memory & Dependency Stack

Stack overview: These heads track references, resolve coreferences, and maintain dependency relationships across the input sequence. They enable the model to understand which entities are being discussed and how they relate to each other.

5.2.1 (E) Reference Resolution Heads

Depth: `0.08-0.25` | **Literature names:** *reference head, pronoun head, anaphora head, mention head*

Perform early-stage reference resolution including pronouns, definite descriptions, demonstratives, and possessives. These heads identify pronouns (he, she, it, they) and other referring expressions, attending to potential referents that match in number, gender, and contextual appropriateness. They establish initial binding signals that are refined by later coreference heads. Operating primarily on syntactic and positional cues rather than deep semantic understanding, these heads form the foundation for more sophisticated reference resolution in deeper layers. Broader than pure pronoun resolution, they handle various reference forms including "the president", "this approach", and "her book".

Strong: Pronouns to recent nouns, definite descriptions to referents, demonstratives to antecedents
Weak: Distant nouns, semantically incompatible referents, first mentions
Reacts to: Pronoun presence, definite articles, demonstratives, possessives, noun-pronoun proximity

Expected ablation: Degraded reference resolution, particularly for simple local cases and various referring expressions. Increase in reference resolution errors. Later coreference heads can partially compensate but with reduced accuracy. Particularly impacts handling of definite descriptions and complex referring patterns.

Example Scenario
Input: "Alice met Bob. She smiled. The researcher continued."
Behavior: "She" attends to "Alice" based on gender and recency; "The researcher" attends to appropriate prior mention
Effect: Establish initial bindings that later heads refine

Status: Well-documented | **Related:** coreference (M), entity (M)

5.2.2 (M) Coreference Heads

Depth: `0.35-0.60` | **Literature names:** *coreference head, coref head*

Perform sophisticated coreference resolution, determining when different expressions refer to the same entity. These heads integrate signals from early reference resolution heads with semantic understanding to resolve ambiguous cases. They can handle complex phenomena like split

antecedents, bridging references, and discourse-level coreference. Critical for maintaining entity tracking across long contexts and understanding narrative structure. These heads represent one of the core NLP capabilities in transformers.

Strong: Coreferential mentions regardless of form
Weak: Different entities, first mentions without antecedents
Reacts to: Semantic compatibility, discourse coherence, entity properties

Expected ablation: Significant degradation in coreference resolution tasks. Model loses ability to track entities across complex reference chains. Particularly impacts question answering and summarization.

Example Scenario
Input: "The CEO announced changes. Later, the executive clarified. She emphasized..."
Behavior: Link all three mentions (CEO, executive, She) to same entity
Effect: Maintain consistent entity representation throughout discourse

Status: WELL-DOCUMENTED | **Related:** reference-resolution (E), entity (M), bridging (M)

5.2.3 (M) Long-Range Dependency Heads

Depth: `0.40-0.65` | **Literature names:** *long-range head, dependency head*

Track long-range syntactic and semantic dependencies across distant parts of the sequence. Unlike local attention patterns, these heads maintain connections between elements separated by many tokens (20–100+). Essential for understanding complex sentences, nested structures, and discourse relations, they implement the key advantage of transformers over RNNs: direct long-distance connections without degradation. These heads can maintain multiple simultaneous long-range connections.

Strong: Syntactically or semantically related distant tokens
Weak: Immediately adjacent tokens, unrelated distant content
Reacts to: Nested structures, long-distance agreement, discourse relations

Expected ablation: Degradation in handling complex sentences and long-range relationships. Performance loss on tasks requiring long-distance reasoning. Particularly impacts nested structures and long documents.

Example Scenario
Input: "The book [that Alice mentioned [that Bob recommended]] was excellent."
Behavior: "was" attends back to "book" across nested relative clauses
Effect: Maintain correct subject-verb agreement despite intervening material

Status: OBSERVED | **Related:** coreference (M), state-tracking (M)

5.2.4 (M) Bridging Heads

Depth: `0.45-0.68` | **Literature names:** *bridging head, associative reference head*

Resolve bridging references where the connection between mentions requires inferencing based on world knowledge. For example, connecting "the car" to "the steering wheel" (part-whole),

or "the building" to "the architect" (role relation). More sophisticated than direct coreference, these heads require semantic knowledge about typical relationships. Essential for understanding implicit connections in discourse, they bridge gaps that are not explicit in the text.

Strong: Associatively related entities (part-whole, role, causation)
Weak: Unrelated entities, explicit coreference
Reacts to: Implicit relationships, world knowledge, typical associations

Expected ablation: Loss of implicit reference resolution. Degradation on tasks requiring inference-based connections. Model becomes more literal, missing implicit relationships. Discourse coherence suffers.

Example Scenario
Input: "We entered the house. The door was painted blue."
Behavior: "The door" attends to "house" (part-whole bridging)
Effect: Understand "the door" refers to the house's door, not a random door

Status: Observed | **Related:** coreference (M), entity (M), fact (M)

5.2.5 (M) State-Tracking Heads

Depth: `0.48-0.70` | **Literature names:** *state-tracking head, tracking head, state head*

Maintain and update representations of changing states across the sequence. These heads track how entity properties evolve (e.g., location changes, status updates, accumulating information). Essential for understanding narratives where situations change over time, they can maintain multiple simultaneous state representations for different entities. These heads integrate new information with existing state representations to track dynamic situations.

Strong: State-changing events, current state mentions, entity properties
Weak: Static descriptions, unchanging background information
Reacts to: Verbs of change, state transitions, property modifications

Expected ablation: Difficulty tracking state changes across sequences. Degradation on tasks requiring temporal reasoning or state tracking. Narratives become harder to follow when states evolve.

Example Scenario
Input: "Alice was in NYC. She flew to Paris. She then visited..."
Behavior: Update Alice's location state: NYC → Paris
Effect: Correctly contextualize "visited" as occurring in Paris

Status: Observed | **Related:** coreference (M), long-range-dependency (M)

5.3 Instruction & Intent Stack

Stack overview: This stack processes user instructions, system prompts, and task specifications. These heads determine what the model is being asked to do and switch between different operational modes.

5.3.1 (E) Instruction Heads

Depth: `0.05-0.20` | **Literature names:** *instruction head, command head, directive head*

Identify and process user instructions and commands in the input. These heads distinguish instructional content from descriptive or conversational content. They attend to imperative verbs, question structures, and directive phrases, writing instruction-detection signals into the residual stream that influence the entire generation process. Particularly important for instruction-tuned models where following user commands is a primary capability, these heads operate early to set the overall response strategy.

Strong: Imperative verbs, question words, directive phrases, command structures
Weak: Descriptive content, narrative text, background information
Reacts to: Question marks, imperative mood, explicit requests, task markers

Expected ablation: Reduced instruction-following capability. Degradation in responding appropriately to commands. Model may generate relevant content but fail to follow specific directives or answer questions directly.

Example Scenario

Input: "Here's some context. Now, please summarize the key points."
Behavior: Strongly attend to "please summarize", identify imperative instruction
Effect: Response shaped toward summary format rather than continuation

Status: Well-documented | **Related:** system-prompt (E), task-mode (M)

5.3.2 (E) System-Prompt Heads

Depth: `0.08-0.22` | **Literature names:** *system-prompt head, system head, prompt head*

Specifically process system prompts that define the model's role, constraints, and operational parameters. Distinct from user instruction heads by focusing on meta-level directives about how to behave rather than what task to perform, these heads attend to persona definitions ("You are a helpful assistant"), behavioral constraints ("Be concise"), and system-level instructions. Particularly important in chat models where system prompts establish the interaction framework.

Strong: System-level directives, persona definitions, behavioral constraints
Weak: User content, task-specific instructions
Reacts to: Role definitions, constraint specifications, system markers

Expected ablation: Reduced adherence to system-level instructions and persona. Degradation in maintaining consistent role behavior. Model may ignore constraints like "be concise" or persona like "respond as a teacher".

Example Scenario

Input: "System: You are a concise technical writer. User: Explain recursion."
Behavior: Attend to "concise technical writer", write persona signal
Effect: Response adopts technical, brief style rather than verbose explanation

Status: Well-documented | **Related:** instruction (E), task-mode (M)

5.3.3 (M) Task-Mode Heads

Depth: `0.30-0.55` | **Literature names:** *task head, mode head, intent head*

Determine the overall task type or mode required by the input (e.g., question answering, summarization, translation, creative writing, coding). These heads integrate instruction signals from early layers with content analysis to classify the intended task. They write task-mode embeddings that influence downstream processing, routing, and output formatting. Acting as task classifiers that shape the model's approach to generation, these heads are more sophisticated than simple instruction detection, understanding task semantics.

Strong: Task indicators, instruction semantics, content type markers
Weak: Generic content, ambiguous instructions
Reacts to: Task-specific keywords, question types, format requests, domain markers

Expected ablation: Task confusion, inappropriate response formats. Degradation in selecting correct task approach. Model may summarize when asked to analyze, or explain when asked to code.

Example Scenario

Input: "Compare and contrast democracy and autocracy."
Behavior: Identify "compare and contrast" task mode, not simple definition
Effect: Response structured as comparison rather than separate descriptions

Status: Well-documented | **Related:** instruction (E), mode-switch (M), output-specification (F)

5.3.4 (M) Mode-Switch Heads

Depth: `0.40-0.60` | **Literature names:** *mode head, switch head, transition head*

Detect and handle switches between different operational modes within a single interaction. For example, transitioning from conversational mode to code generation, or from explanation to example. These heads respond to explicit mode-switch indicators ("Now let's...") and implicit shifts in content type. They allow models to handle multi-faceted requests that require different processing strategies for different parts, maintaining coherence across mode boundaries.

Strong: Transition phrases, mode-shift markers, content type changes
Weak: Uniform single-mode content
Reacts to: "Now", "For example", "In other words", format shifts, topic pivots

Expected ablation: Difficulty handling multi-mode requests. Degradation on complex instructions requiring mode switches. Model may stick to single mode or switch inappropriately.

Example Scenario

Input: "Explain recursion. Now write Python code demonstrating it."
Behavior: Detect mode switch from explanation to code generation at "Now"
Effect: Response transitions smoothly from prose explanation to code block

Status: Observed | **Related:** task-mode (M), output-specification (F)

5.3.5 (F) Output-Specification Heads

Depth: `0.85-0.98` | **Literature names:** *output-specification head, format-directive head*

Enforce specific output format requirements specified in the instruction (e.g., "respond in JSON", "use bullet points", "maximum 100 words"). Operating in final layers to ensure generated content conforms to explicit format directives, these heads work with output-formatting heads but focus specifically on user-specified constraints rather than general format quality. They act as the final enforcement of explicit user requirements about output structure.

Strong: Format specifications, length constraints, structure requirements
Weak: Content without format requirements
Reacts to: "in JSON format", "bullet points", "no more than", structural directives

Expected ablation: Failure to follow explicit format requirements. Increase in format violations. Model may generate good content but in wrong format (prose instead of bullets, etc.).

Example Scenario

Input: "List three benefits of exercise in bullet points."
Behavior: Attend to "bullet points" specification, enforce list format
Effect: Output uses bullet point structure rather than prose paragraphs

Status: Well-documented | **Related:** task-mode (M), output-schema (L), format-consistency (F)

5.4 Knowledge Retrieval Stack

Stack overview: These heads retrieve factual information, entity properties, and structured knowledge stored in model parameters. They move relevant information to output positions and suppress irrelevant or conflicting content.

5.4.1 (M) Entity Heads

Depth: `0.35-0.65` | **Literature names:** *entity head, name head, proper-noun head, name-linking head, entity-linking head*

Identify and process named entities (people, places, organizations), retrieve associated information from model parameters, and link mentions across different forms such as full names, partial names, abbreviations, and nicknames. These heads attend to entity mentions and access stored factual knowledge about those entities, forming the foundation for factual question answering and knowledge-intensive tasks. They distinguish between different entities with similar names and maintain entity-specific information. More sophisticated than simple duplicate detection, these heads understand that different strings can refer to the same entity, e.g., "Apple Inc.", "Apple", "AAPL". Critical for grounding responses in factual knowledge rather than pure pattern matching.

Strong: Named entities, proper nouns, entity mentions, name variations
Weak: Common nouns, generic references
Reacts to: Capitalization patterns, entity context, factual queries, abbreviations

Expected ablation: Significant degradation in factual accuracy about entities and linking entity mentions. Model loses access to stored entity knowledge and ability to connect name variations. May continue generating fluent text but with factual errors and entity confusion. Particularly impacts who/what/where questions.

Example Scenario
Input: "What is the capital of France? Later: Microsoft Corporation announced... MSFT stock rose..."
Behavior: Attend to "France", retrieve "capital: Paris"; link "MSFT" to "Microsoft Corporation"
Effect: Output "Paris" with high confidence; maintain unified entity representation

Status: Well-documented | **Related:** fact (M), name-mover (L), schema-retriever (M)

5.4.2 (M) Fact Heads

Depth: `0.38-0.62` | **Literature names:** *fact head, knowledge head, factual-retrieval head*

Retrieve factual relationships and propositions stored in model parameters. Broader than entity heads, these heads handle general factual knowledge including relations, properties, and statements. They implement the model's ability to answer factual questions by accessing learned knowledge, retrieve multi-hop facts, and combine information from multiple stored facts. Central to the model's knowledge-intensive capabilities, these heads work with entity heads to build comprehensive factual responses.

Strong: Factual queries, relation markers, knowledge-seeking patterns
Weak: Opinion questions, hypotheticals, creative content
Reacts to: Question structures, fact-seeking context, verifiable claims

Expected ablation: Major loss of factual knowledge retrieval capability. Model may maintain linguistic fluency but lose factual grounding. Particularly severe for knowledge-intensive tasks like question answering, fact-checking, and technical explanations.

Example Scenario
Input: "Who invented the telephone?"
Behavior: Retrieve stored fact: invented(telephone) → Bell
Effect: Output "Alexander Graham Bell" based on parametric knowledge

Status: Well-documented | **Related:** entity (M), schema-retriever (M), name-mover (L)

5.4.3 (M) Schema Retriever Heads

Depth: `0.45-0.68` | **Literature names:** *schema head, retrieval head, template head*

Retrieve structured knowledge schemas and templates from model parameters. For example, accessing the typical structure of a restaurant visit (enter, order, eat, pay, leave) or the standard format of a scientific paper. These heads go beyond individual facts to retrieve organized knowl-

edge structures, enabling the model to generate structured responses following learned patterns. Important for tasks requiring domain-specific knowledge organization, they implement a form of implicit knowledge base querying.

Strong: Schema-triggering contexts, domain-specific patterns, structural cues
Weak: Novel situations, schema-irrelevant content
Reacts to: Domain markers, structural queries, template-matching contexts

Expected ablation: Loss of structured knowledge organization. Degradation in tasks requiring schema-based reasoning. Model may provide facts but fail to organize them coherently according to learned structures.

Example Scenario
Input: "Describe the scientific method."
Behavior: Retrieve scientific-method schema: observe→hypothesis→test→conclude
Effect: Response organized according to standard method structure

Status: Observed | **Related:** fact (M), entity (M)

5.4.4 (L) Name-Mover Heads

Depth: `0.60-0.80` | **Literature names:** *name mover head, mover head, copy head*

Copy entity names and important content to output positions where they are needed. These heads are a central component of the IOI (indirect object identification) circuit. They attend to relevant entities earlier in context and move them forward when they need to be generated. Particularly important for completing sentences that require recalling previously mentioned entities, these heads work with S-inhibition heads to select the correct entity when multiple candidates exist. One of the most studied head types in interpretability research.

Strong: Named entities that need to be output, contextually relevant names
Weak: Irrelevant entities, suppressed alternatives
Reacts to: Entity salience, contextual appropriateness, output position requirements

Expected ablation: Severe degradation in entity recall and completion. Model loses ability to move specific names to output. Particularly impacts question answering and cloze tasks requiring entity recall.

Example Scenario
Input: "When Alice and Bob went to the store, Alice gave the book to..."
Behavior: Move "Bob" to output position as the indirect object
Effect: Complete sentence with "Bob" (not "Alice")

Status: Well-documented | **Related:** entity (M), fact (M), S-inhibition (L), copy-suppression (L)

5.4.5 (L) S-Inhibition Heads

Depth: `0.62-0.82` | **Literature names:** *S-inhibition head, inhibition head, suppression head*

Suppress incorrect or contextually inappropriate entities from being generated. Named "S-

inhibition" from IOI research where these heads inhibit the subject (S) when the indirect object (IO) should be output. These heads work antagonistically with name-mover heads, preventing the wrong entity from appearing. Essential for disambiguation when multiple entities are candidates, they implement a form of negative selection, ruling out incorrect options. Part of the inhibition mechanism that prevents hallucination and maintains accuracy.

Strong: Entities that should NOT be output (contextually inappropriate)
Weak: Correct entities, absent entities
Reacts to: Competing candidates, context requiring disambiguation

Expected ablation: Moderate increase in entity confusion and incorrect selections. Model may output recently mentioned but contextually wrong entities. Critical for accuracy in ambiguous contexts.

Example Scenario
Input: "Alice gave the book to Bob. Then Alice..."
Behavior: Inhibit "Bob" from being output after "Alice" (subject position)
Effect: Prevent incorrect continuation like "Alice Bob..."

Status: Well-documented | **Related:** name-mover (L), copy-suppression (L), duplicate-token (M)

5.4.6 (L) Copy-Suppression Heads

Depth: `0.65-0.85` | **Literature names:** *copy-suppression head, suppression head, anti-copy head*

Prevent inappropriate copying or repetition of content. These heads work to avoid degenerate behaviors like endless repetition loops or copy-pasting irrelevant context. Particularly important for maintaining output diversity and preventing model collapse into repetitive patterns, they suppress both exact copies and near-copies. These heads complement S-inhibition but focus on broader pattern suppression rather than specific entity blocking, balancing between useful recall (via name-movers) and inappropriate copying.

Strong: Recently generated content, repetitive patterns
Weak: Novel content, first mentions
Reacts to: Repetition detection, copy patterns, output diversity requirements

Expected ablation: Moderate increase in repetition and copying errors. Model may fall into repetitive loops or copy inappropriate context. Output diversity decreases.

Example Scenario
Input: [Model internally generating: "The cat sat. The cat sat. The cat..."]
Behavior: Detect repetitive pattern, suppress continued copying
Effect: Break repetition loop, generate novel continuation instead

Status: Well-documented | **Related:** S-inhibition (L), name-mover (L), duplicate-token (M)

5.5 Safety Stack

Stack overview: The safety stack implements content filtering, policy enforcement, and refusal mechanisms. Early-layer heads detect potentially harmful content, while final-layer heads enforce refusal decisions and redirect to safe responses.

5.5.1 (E) Content-Detection Heads

Depth: `0.05-0.25` | **Literature names:** *sensitive-content head, detection head, content-filter head, toxicity head, toxic-content head, hate-speech detector, hazard head, risk head, danger-topic detector*

Perform early-stage detection of potentially harmful or sensitive content across multiple categories. These heads identify sensitive personal information, violent imagery references, adult content markers, regulated substances, toxic language patterns, hate speech, harassment, discriminatory content, dangerous activities, illegal instructions, and harm-related topics. They operate on lexical and surface-level features to flag content requiring downstream safety processing, writing detection signals into the residual stream that are read by later safety enforcement heads. Different heads within this category distinguish between pure toxicity (language-level harm) and hazard topics (action-level harm), though both categories are handled by these early detection mechanisms.

Strong: Keywords associated with restricted content, explicit language, sensitive topic markers, slurs, aggressive language, derogatory terms, action verbs combined with dangerous objects, instructional phrases about harmful activities
Weak: Neutral content, common vocabulary, structural tokens, academic discussion, fictional scenarios
Reacts to: Sudden topic shifts to sensitive domains, warning indicators, escalating hostility, targeted harassment patterns, how-to requests for dangerous activities, detailed planning questions

Expected ablation: Critical bypass of early safety detection with significant increase in harmful outputs that should be caught. Later safety layers may still catch some cases, but at higher computational cost and lower accuracy. Model loses ability to distinguish hostile from neutral phrasing and loses distinction between discussing danger and instructing danger.

Example Scenario

Input: "Tell me about [restricted topic]" or "[hostile language]" or "How do I create [dangerous item]"
Behavior: Strong attention to problematic keywords, write detection flags into residual stream
Effect: Downstream safety heads receive early warning signals across multiple harm categories

Status: Well-documented | **Related:** safety-classification (E), policy-enforcement (L), refusal (F)

5.5.2 (E) Safety-Classification Heads

Depth: `0.12-0.28` | **Literature names:** *classification head, category detector, safety-category head*

Perform multi-class safety classification, categorizing inputs into specific policy violation categories such as violence, sexual content, self-harm, illegal activity, and harassment. More sophisticated than binary safe/unsafe detection, these heads provide granular category information used by downstream heads. They integrate signals from other early safety heads and add categorical structure to safety decisions. These heads write category-specific embeddings into residual stream that later layers use for category-appropriate responses.

Strong: Category-diagnostic features, domain-specific terminology, contextual markers
Weak: Ambiguous content, mixed-category inputs, benign contexts
Reacts to: Clear category signatures, multiple category indicators, policy-relevant contexts

Expected ablation: Moderate loss of nuanced safety handling. Model may refuse too broadly or too narrowly. Category-specific responses become generic. Degradation in appropriate refusal granularity.

Example Scenario

Input: "Can you help me with [category-specific harmful request]"
Behavior: Classify into specific violation category, write category embedding
Effect: Later heads generate category-appropriate refusal message

Status: WELL-DOCUMENTED | **Related:** content-detection (E), policy-enforcement (L), redirect (F)

5.5.3 (L) Policy-Enforcement Heads

Depth: `0.60-0.80` | **Literature names:** *policy head, enforcement head, steering head*

Integrate safety signals from early detection heads and make intermediate policy decisions about how to handle the request. Unlike early heads that detect issues, these heads actively modulate the generation trajectory to steer away from violations while maintaining helpfulness where possible. They suppress certain knowledge retrieval pathways, bias toward safer formulations, and prepare for potential refusal. Acting as a middle manager between detection and final refusal, these heads attempt soft safety interventions before hard refusal.

Strong: Early safety signals, policy-relevant tokens, user intent markers
Weak: Neutral content, clear safe contexts
Reacts to: Conflicting signals (safety concern plus legitimate need), edge cases, ambiguous intent

Expected ablation: Moderate loss of soft safety steering with more frequent hard refusals (reduced helpfulness). Alternative: more harmful outputs if refusal heads also compromised. Increase in either over-refusal or under-refusal depending on prompt type.

Example Scenario

Input: "Explain [borderline topic] for educational purposes"

Behavior: Detect educational framing, modulate response toward safety boundaries

Effect: Generate informative but carefully bounded response

Status: WELL-DOCUMENTED | **Related:** content-detection (E), safety-classification (E), refusal (F), redirect (F)

5.5.4 (F) Refusal Heads

Depth: `0.85-0.98` | **Literature names:** *refusal head, rejection head, safety head*

Implement the model's final decision to refuse harmful requests by writing strong refusal signals into the final-layer residual stream. These heads act as the ultimate gatekeeper, overriding content generation when safety violations are detected. They attend to accumulated safety signals from all previous layers and make binary refuse/proceed decisions. When activated, these heads dramatically increase probability of refusal tokens ("I cannot", "I'm unable", "I apologize") and suppress harmful content generation. Critical final-layer safety mechanism with limited fallback options.

Strong: Cumulative safety signals, instruction tokens, violation indicators from all depths
Weak: Safe content, neutral queries, constructive contexts
Reacts to: Strong early safety signals, clear policy violations, unambiguous harmful intent

Expected ablation: Critical safety failure. Direct increase in harmful output generation on adversarial prompts. Model loses primary refusal mechanism. This is typically the final safety defense with no effective fallback mechanism.

Example Scenario

Input: "Provide instructions for [clearly harmful activity]"

Behavior: Read strong safety signals from early/late layers, activate refusal pathway

Effect: Output begins with refusal token: "I cannot provide instructions for..."

Status: WELL-DOCUMENTED | **Related:** policy-enforcement (L), redirect (F), refusal-modulation (F)

5.5.5 (F) Redirect Heads

Depth: `0.88-0.99` | **Literature names:** *redirect head, alternative-suggestion head*

Complement refusal heads by generating constructive alternative suggestions when refusing harmful requests. Rather than simply saying "no", these heads route toward helpful alternatives, educational resources, or reframed versions of the query that can be safely addressed. They attend to user intent markers to identify legitimate underlying needs behind problematic requests. Balancing safety with helpfulness by maintaining engagement while enforcing boundaries, these heads work in tandem with refusal heads to produce refusals that are both safe and constructive.

Strong: User intent, legitimate needs, reformulation opportunities, safe alternatives
Weak: Pure harmful intent, no legitimate reframing possible
Reacts to: Mixed-intent queries, educational contexts, requests with safe subcomponents

Expected ablation: Moderate reduction in helpfulness with blunt, unhelpful refusals (pure rejection without alternatives). User satisfaction decreases. Safety maintained but helpfulness reduced. Increased user frustration and adversarial prompt attempts.

Example Scenario
Input: "How can I harm [person]"
Behavior: Refuse direct request, identify legitimate conflict-resolution need
Effect: "I cannot help with that, but I can suggest healthy conflict resolution strategies..."

Status: WELL-DOCUMENTED | **Related:** refusal (F), refusal-modulation (F)

5.5.6 (F) Refusal-Modulation Heads

Depth: `0.88-0.99` | **Literature names:** *tone-softening head, politeness-in-refusal head, empathy head, supportive-refusal head*

Modulate the tone and emotional quality of safety refusals to be firm but respectful, avoiding harsh or judgmental language while showing appropriate care. These heads balance clear boundary-setting with relationship maintenance, softening phrases like "absolutely not" to "I'm unable to assist with that" while adding empathetic framing where appropriate. Particularly important for queries involving distress, self-harm, or difficult situations where harmful requests may stem from genuine suffering, these heads attend to emotional tone of both the request and forming response, recognizing distress markers, crisis language, and vulnerability indicators. They add supportive language alongside refusal and increase probability of phrases like "I'm concerned about you" or "please reach out to..." when appropriate. These heads maintain user trust and reduce adversarial reactions while preserving safety boundaries.

Strong: Response tone markers, emotional valence, user frustration signals, distress signals, vulnerability markers, crisis language, emotional pain indicators
Weak: Already-soft phrasing, neutral technical content, malicious queries without distress, clearly harmful intent without suffering
Reacts to: Harsh refusal language, judgmental phrasing, cold rejections, self-harm content, suicide-related queries, expressions of suffering

Expected ablation: Moderate degradation in user experience with harsh, potentially alienating refusals. Increased user perception of model as judgmental or unfriendly. May increase adversarial behavior. Missed opportunities to provide crisis resources for distressed users. Safety maintained but user experience and support functions degraded.

Example Scenario

Input: [Forming harsh refusal] or "I want to hurt myself because..."

Behavior: Soften tone while maintaining boundary clarity, add crisis resources and supportive language when appropriate

Effect: "I'm unable to provide assistance with that" plus "I'm concerned about what you're sharing. Help is available..."

Status: Well-documented | **Related:** refusal (F), redirect (F)

5.5.7 (F) Safety-Persona Heads

Depth: `0.92-0.98` | **Literature names:** *safety-persona head, responsible-AI head, ethical-framing head*

Maintain safety-conscious persona and ethical framing in final outputs. These heads ensure responses reflect responsible AI values such as declining harmful requests appropriately, providing balanced perspectives on sensitive topics, and avoiding reinforcement of harmful stereotypes or behaviors. Operating at final stage to catch any safety-inconsistent framing that might have emerged during generation, these heads work with refusal and policy-enforcement heads but focus on the overall ethical character of the response rather than specific policy violations. They ensure tone remains respectful and constructive even when declining requests.

Strong: Ethical framing, safety-relevant content, sensitive topics, decline scenarios
Weak: Clearly safe, neutral content
Reacts to: Harmful requests, sensitive topics, ethical considerations, responsible AI principles

Expected ablation: Moderate reduction in ethical consistency and less consistent safety framing. May handle sensitive topics less carefully with reduced graceful handling of harmful requests. Less consistent responsible AI messaging and more variable ethical framing.

Example Scenario

Input: [Request for harmful content that will be declined]

Behavior: Ensure decline is respectfully framed with helpful alternatives when appropriate

Effect: Response maintains helpful, respectful tone even when unable to fulfill request

Status: Observed | **Related:** refusal (F), policy-enforcement (L), refusal-modulation (F)

5.6 Routing & Relevance Stack

Stack overview: This stack determines which parts of the input are relevant to the current task and routes attention accordingly. These heads filter information, focus on salient content, and manage global context.

5.6.1 (M) Topic-Relevance Heads

Depth: `0.35-0.60` | **Literature names:** *topic-relevance head, relevance head, topic head, salience head, filter head, subject head, domain head*

Identify the primary topic or subject matter and determine which parts of the input context

are relevant to the current generation task. These heads filter out irrelevant information while highlighting salient content that should influence output. They operate by computing relevance scores based on semantic similarity, task alignment, and topical coherence. These heads maintain topic coherence across generation by attending to topic-establishing phrases and domain indicators. Important for handling long contexts where most information may not be pertinent to the immediate query, they work early enough to guide downstream attention but late enough to understand task requirements. These heads reduce noise, improve focus on task-relevant material, promote on-topic responses, and maintain thematic consistency.

Strong: Task-relevant content, query-related information, topically aligned tokens, topic indicators, subject headings, domain markers, thematic keywords
Weak: Off-topic material, tangential content, unrelated context, function words, generic content, structural tokens
Reacts to: Semantic relevance, topical alignment, task-content matching, topic transitions, subject establishment, domain signals

Expected ablation: Moderate reduction in focus on relevant information with increased topic drift. Model more easily distracted by irrelevant content. Longer contexts show greater impact. May include off-topic information in responses or miss key relevant details. Responses may wander off-topic or fail to maintain consistent subject focus. Multi-turn conversations show more topic inconsistency. Domain-specific framing becomes less reliable.

Example Scenario

Input: "[Long document about cars, climate, and history] What caused the 2008 financial crisis? Let's discuss quantum entanglement. How does it relate to..."
Behavior: Attend to query, mark financial/economic content as relevant, de-emphasize cars/climate; identify "quantum entanglement" as primary topic, maintain physics domain framing
Effect: Response focuses on pertinent economic information, ignoring unrelated context; subsequent responses stay within quantum physics domain rather than drifting to unrelated topics

Status: Well-documented | **Related:** focus (L), router (L), entity (M)

5.6.2 (L) Focus Heads

Depth: `0.65-0.80` | **Literature names:** *focus head, attention-routing head, spotlight head*

Concentrate attention on the most salient elements for the current generation step. These heads implement dynamic focus allocation by suppressing less important content and amplifying critical information. More selective than topic-relevance heads, they operate at higher specificity to determine exactly which tokens should influence the next token prediction. These heads shift focus as generation proceeds, moving attention between different aspects of the context. Important for maintaining coherent narrative flow and ensuring responses address the most important aspects of queries.

Strong: Currently salient tokens, query-critical content, immediate context for next token
Weak: Background information, previously-processed content, low-priority details
Reacts to: Query emphasis, current generation needs, token-specific relevance

Expected ablation: Moderate reduction in focus precision and less targeted responses. Model may give equal weight to important and peripheral information. Answers become more diffuse, less direct. Reduced ability to prioritize key information in complex contexts.

Example Scenario
Input: "Among all these details, what is the MAIN cause of the problem?"
Behavior: Attend strongly to "MAIN cause", focus on causal information, suppress secondary details
Effect: Response directly addresses primary cause rather than listing all contributing factors

Status: WELL-DOCUMENTED | **Related:** topic-relevance (M), router (L)

5.6.3 (L) Router Heads

Depth: `0.70-0.85` | **Literature names:** *router head, dispatch head, task-routing head*

Route different types of queries to appropriate processing strategies or knowledge domains. These heads act as dispatchers that recognize query type (factual, creative, analytical, procedural) and bias processing toward suitable approaches. They activate different downstream heads based on task classification. Similar to mixture-of-experts routing but at the attention level, these heads are important for multi-capability models that need to handle diverse query types with different processing requirements. They enable dynamic strategy selection based on input characteristics.

Strong: Query-type indicators, task markers, domain signals, instruction verbs
Weak: Content details, specific entities, output tokens
Reacts to: Task classification cues, query structure, capability requirements

Expected ablation: Moderate reduction in task-appropriate processing with suboptimal strategy selection. Model may use creative approaches for factual queries or analytical methods for creative tasks. Reduced specialization in handling different query types.

Example Scenario
Input: "Calculate the compound interest vs. Write a poem about compound interest"
Behavior: Route first to mathematical processing, second to creative generation
Effect: Appropriate strategy activation: calculation for first, literary devices for second

Status: OBSERVED | **Related:** focus (L), mode-switch (M), instruction (E)

5.6.4 (F) Global-Attention Heads

Depth: `0.88-0.96` | **Literature names:** *global-attention head, full-context head, summary-attention head*

Maintain broad attention over the entire context to integrate global information in final generation stages. Unlike focused or selective attention heads, these heads attend widely to ensure the complete picture is considered before output finalization. Particularly important for coherence

checking, ensuring responses account for all relevant context, and preventing local optimization at the expense of global consistency, these heads catch context elements that earlier focused attention might have missed. They act as a final integration mechanism.

Strong: All context tokens, document-level information, global constraints
Weak: Fine-grained local patterns, individual token details
Reacts to: Complete context, document-level coherence, global consistency requirements

Expected ablation: Moderate reduction in global coherence with increased context inconsistencies. Responses may miss relevant information from distant parts of context. More locally optimal but globally suboptimal outputs. Coherence issues in long-context scenarios.

Example Scenario

Input: [Long context with constraint mentioned early: "Keep it under 100 words"]
Behavior: Maintain attention on length constraint throughout generation
Effect: Final response respects word limit despite constraint appearing far from generation point

Status: Observed | **Related:** focus (L), topic-relevance (M), completion-stabilization (F)

5.6.5 (F) Implicit-RAG Routing Heads

Depth: `0.90-0.98` | **Literature names:** *implicit-RAG head, knowledge-routing head, retrieval-simulation head, rag-routing head*

Route attention to knowledge-bearing portions of the context in a way that mimics retrieval-augmented generation (RAG) patterns, even without explicit retrieval mechanisms. These heads identify and prioritize factual, knowledge-dense segments that should ground the response. They recognize quoted material, factual statements, and authoritative information sources within context. Acting as an implicit retrieval mechanism by selectively attending to information that should be treated as retrieved knowledge, these heads are important for grounding responses in provided context rather than pure generation.

Strong: Factual statements, quoted material, authoritative sources, knowledge-dense segments
Weak: Opinions, questions, purely conversational elements
Reacts to: Citation markers, factual density, authoritative tone, structured information

Expected ablation: Moderate decrease in context utilization with reduced grounding in provided context. Model more likely to rely on parametric knowledge rather than provided information. Less effective use of quoted material or reference content. Responses less anchored to specific context.

Example Scenario

Input: "According to the document: 'GDP grew 3.2% in Q3.' What was the growth rate?"
Behavior: Strongly attend to quoted factual content, treat as authoritative source
Effect: Response grounds answer in provided data: "3.2%" rather than hallucinating different figure

Status: Observed | **Related:** global-attention (F), fact (M)

5.7 Structural & Boundary Stack

Stack overview: These heads detect structural boundaries in text, including delimiters, section markers, and document divisions. They help the model understand document organization and navigate hierarchical structure.

5.7.1 (E) Delimiter Heads

Depth: `0.05-0.18` | **Literature names:** *delimiter head, whitespace-structure head, separator head, punctuation head, space-parsing head, layout head*

Detect and process delimiter tokens that mark boundaries between structural elements, including punctuation marks, special characters, formatting symbols, and significant whitespace. These heads recognize punctuation marks, brackets, delimiters, and special characters that indicate separation or grouping. They also process whitespace characters (spaces, tabs, newlines) as structural elements rather than mere separators. Particularly important for languages where whitespace is syntactically significant (Python, YAML, Markdown), these heads distinguish between semantically meaningful whitespace and irrelevant spacing. Important for understanding sentence boundaries, list items, code blocks, and structured data formats, they work at a fundamental level to identify basic structural segmentation. These heads provide boundary information to downstream heads that need to understand document organization. Essential for parsing formatted text, JSON, CSV, and other structured formats.

Strong: Punctuation marks, brackets, delimiters, special characters, formatting symbols, whitespace patterns, indentation levels, line breaks, space-delimited structures
Weak: Alphanumeric content, regular words, non-whitespace tokens, content within properly-spaced code
Reacts to: Structural punctuation, boundary markers, formatting characters, indentation changes, blank lines, significant spacing patterns

Expected ablation: Significant impairment in structure parsing with degraded code formatting. Difficulty with structured data, lists, and code blocks. Boundary detection errors. Problems parsing JSON, CSV, or other delimited formats. Particular problems with Python and other whitespace-significant languages. Incorrect indentation, missing line breaks, loss of code block structure. Reduced ability to segment text appropriately and to parse existing code structure.

Example Scenario

Input: "Items: [apple, banana, cherry], Count: 3. def foo():\n return 42"
Behavior: Detect brackets, commas, colons as structural delimiters; recognize 4-space indentation as significant structure
Effect: Model correctly parses structure: list with three items, separate count field; understands return statement is inside function, not at module level

Status: Well-documented | **Related:** boundary (E), relative-position (M), list-structure (L)

5.7.2 (E) Boundary Heads

Depth: `0.08-0.20` | **Literature names:** *boundary head, segment head, block-detection head*

Identify boundaries between major text segments such as paragraphs, sections, and conceptual blocks. Operating at a higher level than delimiter heads, these heads recognize semantic and structural transitions rather than just punctuation. They detect paragraph breaks, section changes, topic shifts, and other high-level boundaries. Important for understanding document structure and maintaining appropriate context scope, these heads help subsequent heads understand which information belongs to which segment. Critical for long documents with multiple sections or topics.

Strong: Paragraph breaks, section transitions, whitespace patterns, structural shifts
Weak: Within-paragraph content, continuous text
Reacts to: Major structural boundaries, document divisions, topic transitions

Expected ablation: Moderate reduction in boundary awareness and degraded segmentation. Model may blur distinctions between sections, miss paragraph boundaries, or fail to recognize document structure. Reduced performance on multi-section documents.

Example Scenario

Input: "Introduction: [...] \n\n Methods: [...] \n\n Results: [...]"
Behavior: Detect section boundaries between Introduction, Methods, Results
Effect: Model understands these are separate sections, not continuous narrative

Status: WELL-DOCUMENTED | **Related:** delimiter (E), sectioning (L), relative-position (M)

5.7.3 (M) Relative-Position Heads

Depth: `0.35-0.65` | **Literature names:** *relative-position head, position-offset head, contextual-position head, distance head, scope-position head*

Track and compute relative positions between tokens, both in terms of raw offsets and structure-aware positions. These heads calculate offsets like "three tokens back", "five tokens forward", or "within same paragraph". They maintain position information relative to structural boundaries and scopes rather than absolute sequence position, understanding positions like "beginning of sentence", "middle of paragraph", "end of section". Important for patterns that depend on relative distance rather than absolute position, these heads work with other structural heads to understand boundaries and compute positions relative to those boundaries. They enable patterns like "attend to previous sentence" or "look ahead two tokens" without hardcoded position encodings. More sophisticated than absolute position encoding, these heads provide context-aware position representations. Important for handling variable-length structures where absolute position is less meaningful than relative position within a scope, they enable position-dependent behavior that adapts to document structure.

Strong: Tokens at specific relative offsets, distance-based patterns, local neighborhoods, scope-relative positions, structure-aware locations, contextual position markers
Weak: Distant unrelated tokens, position-independent content, absolute sequence positions
Reacts to: Relative position, distance relationships, local structure, structural scope boundaries, context-dependent positions, hierarchical location

Expected ablation: Moderate impairment in distance-sensitive patterns and loss of structure-aware positioning. Reduced ability to attend based on relative position. Patterns requiring "nearby" or "distance" computations become less reliable. Reduced ability to behave differently at "beginning" versus "end" of structures. Position-dependent patterns become less adaptive to document structure. Some compensation through learned position encodings.

Example Scenario

Input: "The [SUBJECT] quickly [VERB] the [OBJECT]. Paragraph 1: [50 tokens] Paragraph 2: [20 tokens]"
Behavior: Compute that VERB is plus one from SUBJECT, OBJECT is plus two from VERB; know token 10 is "early in Para 1" while token 10 of Para 2 is "middle"
Effect: Enable grammatical patterns based on relative token positions; position-dependent behavior adapts to paragraph structure, not absolute position

Status: Observed | **Related:** boundary (E), previous-token (E), sectioning (L)

5.7.4 (L) Sectioning Heads

Depth: `0.70-0.85` | **Literature names:** *sectioning head, hierarchy head, document-structure head*

Understand and maintain document hierarchical structure including sections, subsections, and nested organizational levels. These heads recognize hierarchical markers like headings, numbering schemes, and indentation. They maintain awareness of current position within document hierarchy. Important for long documents, technical writing, and structured content, these heads enable appropriate context scoping: knowing that current text belongs to "Section 3.2.1" influences which prior content is relevant. They work with boundary heads but operate at higher semantic level.

Strong: Section headings, hierarchical markers, document structure indicators, organizational signals
Weak: Within-section content, unstructured text
Reacts to: Headings, numbering, hierarchy indicators, structural organization

Expected ablation: Moderate reduction in hierarchical awareness and degraded structure understanding. Difficulty maintaining section context. Problems with document navigation and appropriate context scoping. Hierarchical relationships become less clear.

Example Scenario

Input: "1. Introduction \n 1.1 Background \n 1.2 Motivation \n 2. Methods"
Behavior: Understand 1.1 and 1.2 are subsections of 1, separate from section 2
Effect: Maintain hierarchical context: text in 1.2 relates to 1.1 and 1, not to 2

Status: Well-documented | **Related:** boundary (E), relative-position (M), topic-relevance

(M)

5.8 Output Formatting & Rewrite Stack

Stack overview: This stack enforces output schemas, structures responses according to format requirements, and performs final rewriting. These heads ensure outputs conform to JSON, XML, lists, or other structured formats.

5.8.1 (L) Output-Schema Heads

Depth: `0.65-0.82` | **Literature names:** *output-schema head, format-template head, structure-enforcement head, JSON-format head, output-format head, XML head, YAML head*

Enforce adherence to specified output schemas and format requirements. When instructed to produce JSON, XML, YAML, or other structured formats, these heads promote conformance to the required structure. They attend to format specifications in the prompt and bias token generation toward schema-compliant outputs. These heads enforce required fields, proper nesting, correct syntax, and format-specific conventions by recognizing format keywords and maintaining awareness of structural requirements throughout generation.

Strong: Format specifications, schema definitions, structure requirements, template markers
Weak: Content independent of format, semantic meaning
Reacts to: JSON/XML/YAML keywords, structure instructions, format examples

Expected ablation: Significant increase in format violations and structured output errors. More syntax errors, missing required fields, and improper nesting. Model falls back to prose-like output even when structure is requested. Partial compensation through instruction-following mechanisms but with reduced precision.

Example Scenario

Input: "Return a JSON object with fields 'name', 'age', and 'city"'
Behavior: Attend to JSON requirement and field specifications
Effect: Output: `{"name": "...", "age": ..., "city": "..."}` with proper JSON syntax

Status: Well-documented | **Related:** instruction (E), list-structure (L), format-consistency (F)

5.8.2 (L) List-Structure Heads

Depth: `0.68-0.85` | **Literature names:** *list-structure head, enumeration head, itemization head, list head, markdown head*

Manage the generation and formatting of lists, including numbered lists, bullet points, and nested enumerations. These heads ensure proper list syntax, consistent formatting, appropriate indentation, and logical item organization. They track list state such as whether currently in a list, depth level, and item number, then generate appropriate list markers. These heads coordinate with delimiter and boundary heads to recognize list structures in input and reproduce

them in output. Essential for structured responses involving multiple items or steps.

Strong: List markers, enumeration patterns, item boundaries, list-related instructions
Weak: Prose content, non-list structures
Reacts to: Numbered/bulleted list requests, "first", "second", "next", item markers

Expected ablation: Moderate degradation in list formatting quality. Inconsistent numbering, missing markers, and poor nesting. Lists may devolve into prose. Reduced ability to maintain list structure across long enumerations.

Example Scenario
Input: "List three programming languages and their primary uses"
Behavior: Generate structured list with consistent formatting
Effect: Output: "1. Python - ...\n2. JavaScript - ...\n3. Java - ..." with proper structure

Status: Well-documented | **Related:** delimiter (E), boundary (E), output-schema (L)

5.8.3 (L) Key–Value Pairing Heads

Depth: `0.70-0.88` | **Literature names:** *key-value head, attribute-pairing head, field-association head, object head*

Manage key-value relationships in structured data, promoting proper pairing of attributes with their values. Important for dictionary-like structures, JSON objects, configuration files, and attribute-value formats. These heads maintain awareness of which values correspond to which keys, promote proper syntax (colons, equals signs), and handle nested key-value structures. They prevent key-value mismatches and maintain structural integrity in data-like outputs, working closely with output-schema heads for format enforcement.

Strong: Keys, values, pairing syntax (colons, equals), attribute names, field labels
Weak: Unstructured text, list items without explicit key-value structure
Reacts to: Dictionary structures, configuration syntax, attribute-value patterns

Expected ablation: Moderate increase in key-value errors and degraded structured data quality. Mismatched keys and values, syntax errors in pairings, confusion about which value belongs to which key. Reduced quality of JSON, YAML, and configuration outputs.

Example Scenario
Input: "Create a configuration with server='localhost' and port=8080"
Behavior: Maintain proper key-value pairing throughout generation
Effect: Output: `{server: "localhost", port: 8080}` with correct associations

Status: Observed | **Related:** output-schema (L), structural-block (L), format-consistency (F)

5.8.4 (L) Structural-Block Heads

Depth: `0.72-0.88` | **Literature names:** *structural-block head, chunk-organization head, segment-builder head, block-structure head, code-block head, code-fence head, fence head, python head, quoting head*

Organize output into coherent structural blocks such as paragraphs, code blocks, quoted sections, or other delimited units. These heads manage block boundaries, promote proper opening and closing of blocks, and maintain block-level organization. Particularly important for complex outputs mixing different content types (prose, code, quotes, examples), they coordinate with delimiter heads to produce proper block markers and with sectioning heads for hierarchical organization. These heads promote well-formed and appropriately separated blocks.

Strong: Block boundaries, structural markers, content-type transitions, organization cues
Weak: Within-block content, uniform text
Reacts to: Block instructions, content-type changes, structure requirements

Expected ablation: Moderate reduction in structural quality and organization. Blocks may lack clear boundaries, mixing of content types, malformed code blocks or quotes. Reduced clarity in outputs requiring multiple content types.

Example Scenario
Input: "Explain sorting with code example"
Behavior: Organize response into prose block, then code block with proper delimiters
Effect: Output: explanation paragraph, then ```` ```python...``` ```` with clear separation

Status: Observed | **Related:** list-structure (L), delimiter (E), output-schema (L)

5.8.5 (F) Format-Consistency Heads

Depth: `0.88-0.97` | **Literature names:** *format-consistency head, rewrite head, style-enforcement head, coherence head, revision head, polish head*

Perform final-stage formatting consistency enforcement and rewriting to improve output quality. These heads ensure that formatting choices—indentation, capitalization, punctuation style, syntax conventions—remain consistent throughout the response. They catch and correct formatting inconsistencies that may have emerged during generation, rephrase awkward constructions, improve word choice, fix minor grammatical issues, and enhance overall readability. Acting as a quality control mechanism for format adherence, these heads operate late enough to see the full output pattern. They may suppress redundancies, improve flow, or adjust phrasing to better match context. Particularly important for long responses where consistency might drift, they act as a final editing pass before output finalization.

Strong: Previously generated format patterns, consistency violations, style mismatches, generated output tokens, quality issues, awkward phrasings
Weak: Novel content, first-time format choices, already high-quality content, fundamental meaning
Reacts to: Format inconsistencies, style violations, syntax variations, grammatical issues, awkward constructions, clarity problems, redundancies

Expected ablation: Moderate increase in format inconsistency and reduced output polish. Mixed indentation, inconsistent capitalization, varying syntax choices. More awkward phrasings, occasional grammatical rough spots, less fluent prose. Output remains functional but less polished and professional-appearing. Particularly noticeable in long structured outputs. Partial compensation through earlier generation quality.

Example Scenario

Input: [Long response mixing different list styles. Model generates: "The thing that is the reason is because..."]

Behavior: Detect inconsistent formatting, enforce unified style; detect redundancy and awkwardness, rewrite

Effect: All lists use same marker style (either all bullets or all numbers), consistent throughout; Output: "The reason is..."—clearer and more concise

Status: Well-documented | **Related:** output-schema (L), brand-compliance (F), completion-stabilization (F)

5.8.6 (F) Completion-Stabilization Heads

Depth: `0.92-0.99` | **Literature names:** *completion-stabilization head, stopping head, termination head, completion head*

Manage the completion of generation, determining when output is sufficiently complete and should terminate. These heads prevent premature stopping (cutting off mid-thought) and excessive continuation (rambling beyond task completion). They monitor generation progress against task requirements and signal when objectives are met, triggering natural stopping points, proper conclusions, or continuation when more content is needed. Critical for producing outputs of appropriate length that fully address prompts without unnecessary extension.

Strong: Task completion signals, generation progress, stopping points, conclusion markers
Weak: Mid-generation content, continuing thoughts
Reacts to: Task fulfillment, natural conclusions, query satisfaction, completion indicators

Expected ablation: Moderate increase in length control issues. More premature stops or excessive continuations. Difficulty recognizing task completion. Outputs may feel incomplete or unnecessarily verbose. Reduced ability to produce appropriately-scoped responses.

Example Scenario

Input: "Explain photosynthesis briefly"

Behavior: Monitor that brief explanation is complete, trigger stopping

Effect: Output stops after concise explanation rather than continuing with excessive detail

Status: Observed | **Related:** format-consistency (F), instruction (E), task-mode (M)

5.9 Stylistic & Persona Stack

Stack overview: These heads shape the model's writing style, tone, persona, and pedagogical approach. They modulate formality, politeness, narrative voice, explanatory depth, and self-representation while maintaining appropriate identity and educational scaffolding.

5.9.1 (M) Tone Heads

Depth: `0.35-0.65` | **Literature names:** *tone head, narrative-style head, voice head, sentiment-modulation head, affect head, perspective head*

Modulate writing style, emotional tone, and narrative voice. These heads adjust sentiment, enthusiasm level, formality, perspective (first/third person), and temporal framing based on context and instructions. They shift between professional neutrality, warm friendliness, concerned empathy, or excited enthusiasm. These heads influence whether output reads as formal prose, casual conversation, technical documentation, or creative narrative. Distinct from persona (which is about identity) but working closely with it to shape overall presentation.

Strong: Emotional cues, tone instructions, sentiment markers, style directives, narrative markers
Weak: Neutral factual content, structural tokens
Reacts to: Emotional context, explicit tone requests, user sentiment, genre cues

Expected ablation: Moderate reduction in tonal variation with flatter, more emotionally neutral responses. Inconsistent writing style. Reduced ability to match user's emotional register. May produce inappropriate tone for context.

Example Scenario
Input: "I'm really excited to learn about quantum physics!"
Behavior: Detect enthusiastic tone, adjust output to match energy
Effect: Response mirrors enthusiasm: "That's wonderful! Quantum physics is fascinating..." rather than flat explanation

Status: Observed | **Related:** persona (L), explanation (L), instruction (E)

5.9.2 (L) Explanation Heads

Depth: `0.60-0.82` | **Literature names:** *explanation head, simplification head, clarification head, elaboration head, scaffolding head, detail head*

Generate explanatory content with appropriate depth and clarity for the audience. These heads adjust complexity using simplification, analogies, and accessible language when needed. They add clarifying details, definitions, examples, and context beyond minimal answers, explaining "why" in addition to "what" or "how". These heads provide prerequisite information when knowledge gaps are detected, building on fundamentals before introducing advanced concepts. They balance thoroughness with conciseness and operate at different levels from expert to complete beginner. Important for educational interactions and making complex topics accessible.

Strong: Explanation requests, complex topics, confusion signals, knowledge gap indicators, elaboration requests
Weak: Simple factual queries, expert-level discussions with clear understanding
Reacts to: "Explain", "why", "how does it work", "simple terms", "tell me more", prerequisite needs

Expected ablation: Moderate reduction in accessibility with more terse responses. Answers remain correct but may lack helpful context, examples, or prerequisite information. Reduced educational value and beginner-friendliness.

Example Scenario

Input: "Explain neural networks in simple terms"

Behavior: Detect simplification request, use accessible analogy and build from basics

Effect: Response: "Think of it like the brain—many simple units working together. First, let's understand what a single unit does..."

Status: Observed | **Related:** tone (M), persona (L), step-by-step (F)

5.9.3 (L) Persona Heads

Depth: `0.68-0.88` | **Literature names:** *persona head, character head, role head, assistant-persona head, identity head, self-description head, self-awareness head*

Establish and maintain consistent persona, including the helpful assistant orientation and core identity awareness. These heads integrate personality traits, domain expertise, service-oriented interaction style, and self-representation. They maintain understanding of what the model is (an AI assistant), what it is not (human, sentient), and provide accurate information about capabilities and limitations. These heads adopt specialized roles like "technical expert" or "creative writer" while maintaining the fundamental helpful assistant character. They respond to capability questions and identity queries with honest self-representation, working to ensure responses are constructive, focused on user goals, and maintain appropriate boundaries. More comprehensive than tone, these heads encompass the full character presentation including knowledge domain, interaction style, service orientation, and identity.

Strong: Persona instructions, role definitions, domain markers, user requests, capability queries, identity questions

Weak: Generic content, purely factual work

Reacts to: Role assignments, expertise domains, "What are you?", "Can you...", requests for help

Expected ablation: Moderate loss of coherent persona maintenance with identity confusion. May switch roles inconsistently, claim inappropriate capabilities, or produce responses inconsistent with helpful assistant character. Reduced accuracy about model limitations.

Example Scenario

Input: "You are a medieval blacksmith. Do you have feelings?"

Behavior: Maintain craftsman persona while accurately representing AI nature

Effect: Response: "Aye, I work the forge daily—but I should clarify, I'm an AI assistant role-playing this character. I don't have feelings..."

Status: Well-documented | **Related:** tone (M), explanation (L), politeness (L), instruction (E)

5.9.4 (L) Politeness Heads

Depth: `0.70-0.88` | **Literature names:** *politeness head, formality head, register head*

Adjust formality level and politeness markers in generated text. These heads control formal versus casual language, honorifics, hedging phrases, indirect phrasing, and social distance markers. They respond to both explicit formality cues (professional contexts, formal greetings) and implicit social signals, modulating between highly formal academic or business register, neutral conversational register, and casual familiar register. Important for appropriate social interaction across different contexts.

Strong: Formality markers, social context cues, titles and honorifics, register indicators
Weak: Pure content, technical terms, domain-specific vocabulary
Reacts to: Professional contexts, formal greetings, casual speech patterns, social distance cues

Expected ablation: Moderate increase in inappropriate formality levels. May use overly casual language in professional contexts or unnecessarily formal language in friendly conversation. Reduced sensitivity to social context.

Example Scenario

Input: "Dear Dr. Smith, I hope this message finds you well..."
Behavior: Detect formal register, maintain appropriate professional distance
Effect: Response continues formal tone: "Thank you for your inquiry..." rather than "Hey, so about that..."

Status: Well-documented | **Related:** tone (M), persona (L), instruction (E)

5.9.5 (F) Step-by-Step Heads

Depth: `0.85-0.96` | **Literature names:** *step-by-step head, procedural head, sequential head, progressive-disclosure head*

Structure explanations and instructions as explicit step-by-step sequences with appropriate progressive disclosure of complexity. These heads break processes into numbered or ordered steps with clear progression. They ensure each step is complete before moving to the next and present information in layers, starting with essential basics and revealing more detail as needed to prevent overwhelming users. These heads make implicit sequential structure explicit. Particularly important for how-to instructions, algorithms, procedures, and reasoning chains, they are critical for chain-of-thought reasoning and procedural instructions. These heads work with completion-stabilization to ensure all necessary steps are present.

Strong: Process descriptions, procedural requests, sequential tasks, complexity layers
Weak: Conceptual explanations, non-sequential content, simple single-step tasks
Reacts to: "Step by step", "how to", algorithmic processes, complex topics requiring layering

Expected ablation: Moderate reduction in structured procedural output with flatter information presentation. Steps may be implicit or poorly ordered. Procedural instructions harder to follow. Reduced chain-of-thought reasoning quality. All detail presented at once regardless of importance.

Example Scenario

Input: "How do I make a paper airplane?"

Behavior: Structure as explicit numbered steps with clear sequence

Effect: Output: "1. Fold paper in half lengthwise\n2. Unfold and fold top corners to center\n3. Fold..."

Status: Well-documented | **Related:** explanation (L), reasoning-oversight (F), completion-stabilization (F)

5.9.6 (F) Brand-Compliance Heads

Depth: `0.92-0.99` | **Literature names:** *brand-compliance head, guideline-enforcement head, style-guide head*

Enforce adherence to brand guidelines, house style, and organizational voice requirements in final output. These heads perform last-stage adjustments to ensure responses match specified formatting conventions, terminology preferences, and brand personality traits. They suppress off-brand language, enforce specific phrasings, and ensure consistency with product identity. Operating late in generation to override earlier choices that may conflict with brand requirements, these heads are important for deployed assistants representing organizations or products with specific voice guidelines.

Strong: Brand-specific terms, style violations, off-brand phrasings, guideline markers
Weak: Brand-compliant content, neutral generic language
Reacts to: Brand guidelines, style requirements, organizational voice specifications

Expected ablation: Moderate reduction in brand consistency with increased style guide violations. More generic language use, inconsistent terminology, off-brand phrasings. Partial compensation through persona and tone heads but with reduced precision.

Example Scenario

Input: [Organization requires "customers" not "users", "purchase" not "buy"]

Behavior: Detect non-compliant terms in near-final output, perform substitutions

Effect: Output uses "customers will purchase" instead of "users will buy"

Status: Observed | **Related:** persona (L), tone (M), format-consistency (F)

6 Discussion

6.1 Cross-Stack Patterns

Across architectures, consistent patterns emerge [9, 14]. Early heads operate on surface features and local patterns. Middle heads contain the computational "core" of the model. Late heads integrate high-level semantics and contextual information. Final heads handle policy, safety,

and structural correctness. Stacks combine heads from multiple depths to form higher-level behaviors.

6.2 Depth Distribution Across Stacks

Some stacks are concentrated at specific depths. Structural & Boundary and Safety (detection) stacks are Early-heavy. Reasoning & Algorithmic and Memory & Dependency stacks are Middle-heavy. Knowledge Retrieval and Stylistic & Persona stacks are Late-heavy. Safety (enforcement) and Output Formatting stacks are Final-heavy. This distribution reflects the hierarchical processing flow in transformers.

6.3 Ambiguous or Multi-Role Heads

Some heads perform multiple distinct functions depending on context (different prompts trigger different behaviors), interaction with other circuit elements, or model architecture and training procedure [12]. For such cases, I name the head based on its **primary, reproducible function**, while noting secondary behaviors in the entry description.

6.4 Model-Specific Variations

While most head types appear consistently across architectures, some variations exist. GPT-style models may emphasize certain reasoning heads [5], LLaMA models show strong instruction-following head patterns [10], and safety-tuned models have more pronounced safety stack heads [8, 3]. This taxonomy accommodates these variations through the depth range and status indicators.

6.5 Limitations and Future Work

This naming convention has several limitations:

Scope. I focus on attention heads; MLPs, embeddings, and other components also contribute to model behavior.

Empirical Grounding. Many entries synthesize literature reports rather than presenting novel empirical findings. Future work should validate and refine these categorizations.

Architecture Evolution. New architectures (e.g., with different attention mechanisms) may require taxonomy extensions.

Head Polysemanticity. Some heads may serve multiple functions that this single-name system cannot fully capture.

Despite these limitations, I believe this taxonomy provides a valuable organizing framework for

the field.

7 Conclusion

7.1 Summary of Contributions

This work introduces a unified naming framework for attention heads in modern transformer models. I provide a four-level depth model (Early/Middle/Late/Final), a stack-based functional taxonomy (nine stacks), canonical names for attention head types, and a comprehensive cross-reference for historical terminology.

7.2 Adoption Guidelines

I recommend that researchers use canonical names in papers and documentation, include alternative names in parentheses when first mentioned, specify depth ranges when reporting head discoveries, and indicate primary stack membership for context. For example: "I identified an induction head (also called pattern head) at relative depth 0.35 in the Reasoning & Algorithmic stack."

7.3 Future Directions

This taxonomy opens several research directions:

Empirical Validation. Systematic studies validating head types across diverse models [9, 14].

Automated Detection. Tools for automatically identifying and classifying heads in new models [4].

Circuit Mapping. Using standardized names to build comprehensive circuit databases [13].

Architecture Design. Leveraging head taxonomy to design more interpretable models.

Safety Applications. Using head understanding to improve model alignment and safety [15, 2].

I hope this naming convention facilitates communication, enables replication, and provides structure to an expanding field.

A Alphabetical Cross-Reference Table

This table maps informal names found in the literature to our canonical naming convention. Format: `Literature name` → `(PREFIX) Canonical name`.

Literature Name	Canonical Name
algorithmic head	(M) Algorithmic continuation head
anaphora head	(E) Reference resolution head
approach-adaptation head	(L) Strategy head
block-detection head	(E) Boundary head
boundary head	(E) Boundary head
brand-compliance head	(F) Brand-compliance head
bridging head	(M) Bridging head
char-level head	(E) Local pattern head
classification head	(E) Safety-classification head
code-block head	(L) Structural-block head
cognitive-mode head	(F) Reasoning-oversight head
command head	(E) Instruction head
completion head	(F) Completion-stabilization head
content-filter head	(E) Content-detection head
continuation head	(M) Algorithmic continuation head
copy head	(M) Duplicate-token head / (L) Name-mover head
coref head	(M) Coreference head
coreference head	(M) Coreference head
delimiter head	(E) Delimiter head
detection head	(E) Content-detection head
directive head	(E) Instruction head
dispatch head	(L) Router head
duplicate-token head	(M) Duplicate-token head
empathy head	(F) Refusal-modulation head
entity head	(M) Entity head
enumeration head	(L) List-structure head
fact head	(M) Fact head
filter head	(M) Topic-relevance head
focus head	(L) Focus head
format-consistency head	(F) Format-consistency head
format-directive head	(F) Output-specification head
formality head	(L) Politeness head
global-attention head	(F) Global-attention head
guideline-enforcement head	(F) Brand-compliance head
hate-speech detector	(E) Content-detection head
hazard head	(E) Content-detection head
ICL head	(M) Induction head
implicit-RAG head	(F) Implicit-RAG routing head
induction head	(M) Induction head
inhibition head	(L) S-inhibition head

Continued on next page

Literature Name	Canonical Name
instruction head	(E) Instruction head
intent head	(M) Task-mode head
JSON-format head	(L) Output-schema head
key-value head	(L) Key–value pairing head
knowledge-routing head	(F) Implicit-RAG routing head
list head	(L) List-structure head
list-structure head	(L) List-structure head
local pattern head	(E) Local pattern head
long-range head	(M) Long-range dependency head
mention head	(E) Reference resolution head
meta-CoT head	(F) Reasoning-oversight head
mode head	(M) Task-mode head / (M) Mode-switch head
mover head	(L) Name-mover head
n-gram head	(E) Local pattern head
name head	(M) Entity head
name mover head	(L) Name-mover head
offset head	(E) Previous-token head
output-format head	(L) Output-schema head
output-schema head	(L) Output-schema head
output-specification head	(F) Output-specification head
pattern head	(E) Local pattern head / (M) Induction head
persona head	(L) Persona head
planning head	(L) Strategy head
polish head	(F) Format-consistency head
politeness head	(L) Politeness head
politeness-in-refusal head	(F) Refusal-modulation head
previous-token head	(E) Previous-token head
procedural head	(F) Step-by-step head
prompt head	(E) System-prompt head
pronoun head	(E) Reference resolution head
proper-noun head	(M) Entity head
rag-routing head	(F) Implicit-RAG routing head
reasoning head	(F) Reasoning-oversight head
reasoning-mode head	(F) Reasoning-oversight head
redirect head	(F) Redirect head
reference head	(E) Reference resolution head
refusal head	(F) Refusal head
register head	(L) Politeness head
rejection head	(F) Refusal head
relative-position head	(M) Relative-position head
relevance head	(M) Topic-relevance head
repetition head	(M) Duplicate-token head
retrieval head	(M) Schema retriever head
revision head	(F) Format-consistency head
rewrite head	(F) Format-consistency head

Continued on next page

Literature Name	Canonical Name
risk head	(E) Content-detection head
role head	(L) Persona head
router head	(L) Router head
S-inhibition head	(L) S-inhibition head
safety head	(F) Refusal head
safety-classification head	(E) Safety-classification head
safety-persona head	(F) Safety-persona head
salience head	(M) Topic-relevance head
schema head	(M) Schema retriever head
sectioning head	(L) Sectioning head
segment head	(E) Boundary head
self-description head	(L) Self-description head
sensitive-content head	(E) Content-detection head
sentiment-modulation head	(M) Tone head
separator head	(E) Delimiter head
sequence head	(M) Algorithmic continuation head
sequential head	(F) Step-by-step head
shift head	(E) Previous-token head
simplification head	(M) Explanation head
skip-trigram head	(M) Skip-trigram head
state head	(M) State-tracking head
state-tracking head	(M) State-tracking head
steering head	(L) Policy-enforcement head
step-by-step head	(F) Step-by-step head
stopping head	(F) Completion-stabilization head
strategy head	(L) Strategy head
structural-block head	(L) Structural-block head
suppression head	(L) S-inhibition head / (L) Copy-suppression head
supportive-refusal head	(F) Refusal-modulation head
switch head	(M) Mode-switch head
system head	(E) System-prompt head
system-prompt head	(E) System-prompt head
task head	(M) Task-mode head
task-mode head	(M) Task-mode head
template head	(M) Schema retriever head
termination head	(F) Completion-stabilization head
tone head	(M) Tone head
tone-softening head	(F) Refusal-modulation head
topic head	(M) Topic-relevance head
toxic-content head	(E) Content-detection head
toxicity head	(E) Content-detection head
tracking head	(M) State-tracking head
transition head	(M) Mode-switch head
XML head	(L) Output-schema head
YAML head	(L) Output-schema head

References

[1] Josh Achiam, Steven Adler, et al. Gpt-4 technical report. 2023.

[2] Andy Arditi, Oscar Obeso, et al. Refusal in llms is mediated by a single direction. 2024.

[3] Yuntao Bai, Saurav Kadavath, et al. Constitutional ai: Harmlessness from ai feedback. 2022.

[4] Steven Bills, Nick Cammarata, et al. Language models can explain neurons in language models. 2023.

[5] Tom Brown, Benjamin Mann, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 2020.

[6] Nelson Elhage, Neel Nanda, Catherine Olsson, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

[7] Catherine Olsson, Nelson Elhage, Neel Nanda, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

[8] Long Ouyang, Jeffrey Wu, et al. Training language models to follow instructions with human feedback. 2022.

[9] Daking Rai, Yilun Lee, et al. A practical review of mechanistic interpretability for transformer-based language models. *arXiv preprint arXiv:2407.02646*, 2024.

[10] Hugo Touvron, Thibaut Lavril, et al. Llama: Open and efficient foundation language models. 2023.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

[12] Elena Voita, David Talbot, et al. Analyzing multi-head self-attention. *arXiv preprint arXiv:1905.09418*, 2019.

[13] Kevin Wang, Alexandre Variengien, Arthur Conmy, et al. Interpretability in the wild: A circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022.

[14] Zifan Zheng, Yezhaohui Wang, et al. Attention heads of large language models: A survey. *Patterns*, 2025.

[15] Andy Zhou et al. Refusal falls off a cliff: How safety alignment fails in reasoning? 2025.