Attention Head Naming Convention for Large Language Models (LLMs)

Karol Kowalczyk

November 2025

Abstract

Large language models exhibit remarkable reasoning, safety alignment, and structural understanding, yet their internal workings remain opaque. Attention heads—specialized components within transformer layers—have emerged as key objects of study in interpretability research. The community has developed informal names: *induction heads*, *name mover heads*, *refusal heads*, but these terms are inconsistent, overlapping, and ambiguous.

This work proposes a unified naming convention for attention heads: (1) a four-level depth model (Early, Middle, Late, Final), (2) stack-based functional grouping, (3) canonical names for head types, and (4) cross-reference tables mapping historical terms to standardized ones. This taxonomy is descriptive rather than prescriptive, capturing current head behaviors while remaining flexible for future architectures.

Contents

1 Introduction

1.1 Motivation

Attention heads—the basic computational units within transformer architectures—have emerged as key objects of study in mechanistic interpretability research despite achieving remarkable performance across diverse tasks.

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*, and *JSON heads*. These naming conventions are **inconsistent** (same head type, multiple names), **ambiguous** (single name, different behaviors), **fragmented** (no unified framework), and **unscalable** (fail across architectures). This fragmentation complicates replication, comparison, and dataset annotation.

1.3 Goals of This Work

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

1.4 Circuits, Stacks, and Simplification

This taxonomy uses *stacks* as organizational framework. Attention heads work in complex *circuits*—groups across layers cooperating through multi-level processing [6, 13].

The *stack* abstraction simplifies this complexity for communication. Rather than mapping every circuit connection, I group heads by primary functional contribution, making the taxonomy accessible while acknowledging that real model behavior involves intricate cross-layer interactions.

1.5 Structure of This Document

I review prior work (§2), introduce the depth model (§3) and stacks (§4), present a comprehensive catalog organized by functional stack (§5), and conclude with discussion (§6) and future directions (§7).

2 Background

2.1 Attention Heads and Functions

In transformer models [11], attention heads perform focused computations over token sequences. Though individually simple, they develop specialized behaviors: pattern continuation, entity tracking, semantic filtering, routing, format enforcement, and safety constraints [6, 7]. These behaviors form *circuits* and larger *stacks* of related functionality.

2.2 Why Naming Consistency Matters

Interpretability research suffers from fragmented terminology [9, 14]. The same head type appears under multiple names, while overloaded names refer to unrelated behaviors. Consistent naming improves communication clarity, strengthens cross-paper alignment, helps index interpretability datasets, and enables systematic circuit mapping.

2.3 Prior Naming Practices

Previous work named heads by behavior (induction, copy-suppression), formatting (JSON, list), signal source (delimiter), circuit role (name mover), or safety function (refusal, toxicity). Though often accurate, these labels vary widely. This work unifies them under a systematic framework.

3 Depth Model: Early—Middle—Late—Final

3.1 Rationale for Four Depth Categories

Functional behavior clusters reliably into four zones [6, 13]: **Early (E)** layers handle token-level processing, boundary detection, and filtering. **Middle (M)** layers implement reasoning primitives, induction, and dependency tracking. **Late (L)** layers perform semantic integration, routing, and persona shaping. **Final (F)** layers enforce policy, safety, and structured output. This structure holds across GPT, LLaMA, and Claude [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 = layers 0–15 (0.00–0.15), Middle = 15–50 (0.15–0.52), Late = 50–85 (0.52–0.88), Final = 85–96 (0.88–1.00).

3.3 Relative Depth Scaling

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

4 Stacks: Functional Grouping of Attention Heads

4.1 What is a Stack?

A *stack* groups head types that together implement a higher-level capability. Stacks reflect functional clustering observed in studies [13, 7]. Examples: Reasoning & Algorithmic, Memory & Dependency, Safety, Output Formatting. Stacks span Early, Middle, Late, and Final layers.

4.2 Relationship Between Stacks and Depth

Different functions appear at different depths. Early: delimiters, content detection, input conditioning. Middle: reasoning, induction, entity linking. Late: narrative coherence, routing, topic steering. Final: policy, formatting, rewriting, safety. This *stack* × *depth* structure forms the catalog basis.

5 Attention Head Catalog

This section catalogs attention head types by functional stack. Each stack groups heads contributing to common high-level capability, ordered by depth (Early → Middle → Late → Final).

Entry Format. Each head entry includes:

- **Depth range:** Typical relative depth (0.0–1.0)
- **Literature names:** Alternative names from prior work
- **Function:** Core behavior and mechanism
- **Attention pattern:** What the heads attend to
- **Expected ablation:** Predicted effects if disabled
- **Example scenario:** Concrete behavioral illustration
- **Stack and relations:** Primary stack and related heads

5.1 Reasoning & Algorithmic Stack

Stack overview: Heads performing pattern matching, sequence continuation, algorithmic reasoning, and meta-cognitive oversight. Enable in-context learning, pattern completion, and 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 token t to position $t+1$, creating shifted representation where each position contains information about the previous token. Foundational for induction circuits, enabling later heads to access "what came before" without attending backwards. Show strong diagonal attention patterns ($i \to i-1$).

Strong: Immediately preceding token
Weak: Distant tokens, same-position
Reacts to: Sequential structure, boundaries

Expected ablation: Breaks induction circuits entirely. Induction heads lose access to "what came after previous occurrences". Critical for in-context learning.

Example Scenario

Input: “The cat sat. The cat...”

Behavior: Copy tokens forward one position

Effect: Induction heads match “cat” and access “sat”

Status: Well-documented | **Related:** induction (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 character-level and subword patterns for spelling, capitalization, punctuation, and morphology. Operate at finer granularity than most heads, attending within and between adjacent tokens. Recognize patterns like “ing”, “tion”, or punctuation clusters.

Strong: Adjacent tokens, subword units

Weak: Long-range dependencies, semantics

Reacts to: Spelling, capitalization, morphology

Expected ablation: Degraded handling of misspellings, case variations, morphology. Errors on character-level tasks. Partial compensation through tokenization.

Example Scenario

Input: “organizATION’s”

Behavior: Detect unusual case pattern

Effect: Handle non-standard capitalization

Status: Observed | **Related:** induction (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 [A][B]...[A] patterns and predict [B] follows the second [A]. Attend to tokens after previous instances of current token. Work with previous-token heads to enable pattern completion, name recall, and few-shot learning. Fundamental to in-context learning.

Strong: Tokens following previous occurrences

Weak: Immediate neighbors, first occurrence

Reacts to: Token repetition, [A][B]...[A] patterns

Expected ablation: Significant degradation in in-context learning and pattern completion. 30–50% accuracy loss on few-shot tasks. Partial compensation through other heads.

Example Scenario

Input: “Mary and John went to the store, Mary bought...”

Behavior: Second “Mary” attends to tokens after first “Mary”

Effect: Increased probability of 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 current token appeared previously, marking repeats for downstream processing. Unlike induction heads (which predict next token), these simply signal "seen before". Used by IOI circuits, name-movers, and copy-suppression.

Strong: Previous identical tokens

Weak: Similar non-identical, first occurrence

Reacts to: Exact repetition, name recurrence

Expected ablation: Impaired duplicate detection. Degraded name-mover and copy-suppression circuits. Overlap with induction heads provides redundancy.

Example Scenario

Input: "Alice gave the book to Bob. Then Alice..."

Behavior: Second "Alice" writes duplicate signal

Effect: Name-movers and S-inhibition use 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*

Match non-contiguous patterns [A]...[B]...[C] with intervening tokens. More flexible than strict n-grams. Detect phrasal patterns, idioms, and structural templates with flexible word order.

Strong: Components separated by 1–3 tokens

Weak: Adjacent patterns, long-range

Reacts to: Phrasal patterns, flexible idioms

Expected ablation: Reduced flexible pattern recognition. Less critical than induction heads; other mechanisms compensate.

Example Scenario

Input: "not only X but also"

Behavior: Recognize "not...but" despite intervening tokens

Effect: Predict "also" after "but"

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: counting, days of week, months, systematic progressions. Operate on sequences with clear algorithmic rules. Detect arithmetic progressions, cyclic patterns, rule-governed sequences.

Strong: Sequential elements in algorithmic patterns
Weak: Random sequences, semantic patterns
Reacts to: Arithmetic progressions, cyclic orderings

Expected ablation: Reduced sequence continuation performance. Degradation on counting, ordering, arithmetic. Some reasoning persists through other mechanisms.

Example Scenario
Input: "Monday, Tuesday, Wednesday, ..."
Behavior: Recognize day-of-week progression
Effect: Strongly predict "Thursday"

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

5.1.7 (L) Strategy Heads

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

Plan overall approach for complex tasks and adapt when strategies prove ineffective. Influence high-level structure: step decomposition, component ordering, method selection. Recognize task types requiring different approaches (analytical vs. creative, sequential vs. parallel). Decompose complex queries into subtasks. Monitor progress, detect dead ends, switch strategies when needed.

Strong: Task complexity, multi-part queries, progress indicators
Weak: Single-step tasks, progressing solutions
Reacts to: Complex tasks, planning requests, insufficient progress

Expected ablation: Reduced planning quality and adaptability. Premature execution without planning. Persist with unproductive approaches. 15–25% efficiency loss on complex tasks.

Example Scenario
Input: "Plan a ML project for customer churn"
Behavior: Recognize need for structured planning
Effect: Structure: data → analysis → features → model → evaluation

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

5.1.8 (F) Reasoning-Oversight Heads

Depth: `0.88-0.99` | **Literature names:** *reasoning-mode head, cognitive-mode head, meta-CoT head, reasoning-quality head*

Manage reasoning processes: mode selection and quality monitoring. Select reasoning modes (analytical, creative, analogical, deductive, inductive) matched to task type. Monitor reasoning chain quality, detect errors and gaps, flag uncertainty, trigger re-thinking. Operate at meta-level above chain-of-thought. Influence which reasoning patterns activate. Prevent confident errors in complex scenarios.

Strong: Task type, reasoning mode cues, quality indicators, logical gaps
Weak: Simple factual responses, non-reasoning tasks
Reacts to: Complex reasoning, logical steps, errors, inconsistencies

Expected ablation: Less appropriate mode selection. More logical gaps, reduced self-correction. Chain-of-thought less reliable on complex problems. 20–30% degradation on multi-step reasoning.

Example Scenario
Input: "Brainstorm creative names"
Behavior: Select creative mode vs. analytical
Effect: Free-flowing suggestions, not logical analysis

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

5.2 Memory & Dependency Stack

Stack overview: Track references, resolve coreferences, and maintain dependency relationships. Enable understanding of which entities are discussed and how they relate.

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 for pronouns, definite descriptions, demonstratives, and possessives. Attend to potential referents matching in number, gender, and contextual appropriateness. Establish initial binding signals refined by later coreference heads. Operate on syntactic and positional cues rather than deep semantics.

Strong: Pronouns to recent nouns, definite descriptions, demonstratives
Weak: Distant nouns, incompatible referents
Reacts to: Pronouns, definite articles, possessives

Expected ablation: Degraded reference resolution, 20–30% increase in errors. Later coreference heads partially compensate. Reduced handling of definite descriptions and complex referring patterns.

Example Scenario
Input: "Alice met Bob. She smiled."
Behavior: "She" attends to "Alice" (gender, recency)
Effect: Establish initial bindings

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*

Determine when different expressions refer to the same entity. Integrate early reference signals with semantic understanding to resolve ambiguous cases. Handle split antecedents, bridging references, and discourse-level coreference. Critical for entity tracking across long contexts.

Strong: Coreferential mentions regardless of form
Weak: Different entities, first mentions
Reacts to: Semantic compatibility, discourse coherence

Expected ablation: Significant degradation in coreference resolution. Loss of entity tracking across complex reference chains. 30–40% accuracy drop on question answering and summarization.

Example Scenario

Input: “The CEO announced changes. The executive clarified. She...”
Behavior: Link all three mentions to same entity
Effect: Maintain consistent entity representation

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 syntactic and semantic dependencies across distant sequence positions (20–100+ tokens). Maintain connections between separated elements without degradation. Implement transformer advantage over RNNs: direct long-distance connections. Maintain multiple simultaneous long-range connections.

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

Expected ablation: Degradation on complex sentences and long-range relationships. 25–35% performance loss on long-distance reasoning. Severe impact on nested structures.

Example Scenario

Input: “The book [that Alice mentioned [that Bob recommended]] was excellent.”
Behavior: “was” attends to “book” across nested clauses
Effect: Maintain subject-verb agreement

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 requiring world knowledge inference. Connect mentions through implicit relationships: part-whole (“car” → “steering wheel”), role relations (“building” → “architect”), or causation. Require semantic knowledge about typical relationships.

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

Expected ablation: Loss of implicit reference resolution. Model becomes more literal, missing implicit relationships. 20–30% degradation on inference-based connections.

Example Scenario

Input: “We entered the house. The door was blue.”

Behavior: “The door” attends to “house” (part-whole)

Effect: Understand: the house’s door, not 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 changing state representations. Track entity property evolution: location changes, status updates, accumulating information. Maintain multiple simultaneous state representations for different entities. Integrate new information with existing states.

Strong: State-changing events, current state, entity properties

Weak: Static descriptions, unchanging background

Reacts to: Verbs of change, state transitions

Expected ablation: Difficulty tracking state changes. 25–35% degradation on temporal reasoning. Narratives harder to follow when states evolve.

Example Scenario

Input: “Alice was in NYC. She flew to Paris. She visited...”

Behavior: Update Alice’s location: NYC → Paris

Effect: Contextualize “visited” in Paris

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

5.3 Instruction & Intent Stack

Stack overview: Process user instructions, system prompts, and task specifications. Determine what the model is asked to do and switch between 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. Distinguish instructional from descriptive or conversational content. Attend to imperative verbs, question structures, and directive phrases. Write instruction-detection signals into residual stream influencing generation. Operate early to set response strategy.

Strong: Imperative verbs, question words, directive phrases

Weak: Descriptive content, narrative text

Reacts to: Question marks, imperative mood, explicit requests

Expected ablation: Reduced instruction-following capability. Model generates relevant content but fails to follow specific directives. 25–35% degradation on instruction-following tasks.

Example Scenario

Input: "Context provided. Now, summarize key points."

Behavior: Attend to "summarize", identify imperative

Effect: Summary format vs. 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*

Process system prompts defining model role, constraints, and operational parameters. Focus on meta-level directives about how to behave rather than what task to perform. Attend to persona definitions, behavioral constraints, and system-level instructions. Establish interaction framework in chat models.

Strong: System-level directives, persona definitions, constraints

Weak: User content, task-specific instructions

Reacts to: Role definitions, constraint specifications

Expected ablation: Reduced adherence to system-level instructions and persona. Model ignores constraints like "be concise" or persona like "respond as teacher". 30–40% reduction in role consistency.

Example Scenario

Input: "System: You are a concise technical writer. User: Explain recursion."

Behavior: Attend to "concise technical writer"

Effect: Technical, brief style vs. 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 overall task type: question answering, summarization, translation, creative writing, coding. Integrate instruction signals from early layers with content analysis to classify intended task. Write task-mode embeddings influencing downstream processing, routing, and formatting. More sophisticated than simple instruction detection.

Strong: Task indicators, instruction semantics, content type markers

Weak: Generic content, ambiguous instructions

Reacts to: Task-specific keywords, question types, format requests

Expected ablation: Task confusion and inappropriate response formats. Model summarizes when asked to analyze, or explains when asked to code. 20–30% task classification errors.

Example Scenario

Input: “Compare and contrast democracy and autocracy.”

Behavior: Identify “compare and contrast” mode

Effect: Comparison structure vs. 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 operational modes within single interaction. Transition from conversational to code generation, or explanation to example. Respond to explicit indicators (“Now let’s...”) and implicit content shifts. Maintain coherence across mode boundaries.

Strong: Transition phrases, mode-shift markers, content type changes

Weak: Uniform single-mode content

Reacts to: “Now”, “For example”, format shifts

Expected ablation: Difficulty handling multi-mode requests. Model sticks to single mode or switches inappropriately. 25–35% degradation on complex multi-part instructions.

Example Scenario

Input: “Explain recursion. Now write Python code.”

Behavior: Detect mode switch at “Now”

Effect: Smooth transition: prose → 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 from instruction: “respond in JSON”, “use bullet points”, “maximum 100 words”. Operate in final layers to ensure content conforms to explicit format directives. Focus on user-specified constraints rather than general format quality. Final enforcement of explicit user requirements.

Strong: Format specifications, length constraints, structure requirements

Weak: Content without format requirements

Reacts to: “in JSON format”, “bullet points”, “no more than”

Expected ablation: Failure to follow explicit format requirements. Model generates good content in wrong format. 40–50% increase in format violations.

Example Scenario

Input: “List three benefits of exercise in bullet points.”

Behavior: Attend to “bullet points”, enforce format

Effect: Bullet structure vs. prose paragraphs

Status: WELL-DOCUMENTED | **Related:** task-mode (M), output-schema (L), format-consistency

(F)

5.4 Knowledge Retrieval Stack

Stack overview: Retrieve factual information, entity properties, and structured knowledge from model parameters. Move relevant information to output positions and suppress irrelevant content.

5.4.1 (M) Entity Heads

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

Identify and process named entities (people, places, organizations). Retrieve associated information from model parameters. Link mentions across different forms: full names, abbreviations, nicknames. Understand that different strings refer to same entity ("Apple Inc.", "Apple", "AAPL"). Ground responses in factual knowledge.

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

Expected ablation: Significant degradation in factual accuracy. Model loses entity knowledge and linking ability. 30–40% accuracy drop on who/what/where questions. Fluent text with factual errors.

Example Scenario

Input: "Capital of France? MSFT stock rose..."
Behavior: Retrieve "capital: Paris"; link "MSFT" to "Microsoft"
Effect: Output "Paris"; maintain unified entity

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 from model parameters. Handle general factual knowledge: relations, properties, statements. Access learned knowledge for factual questions. Retrieve multi-hop facts and combine information from multiple stored facts.

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

Expected ablation: Major loss of factual knowledge retrieval. Linguistic fluency maintained but factual grounding lost. 40–60% degradation on knowledge-intensive tasks.

Example Scenario

Input: "Who invented the telephone?"

Behavior: Retrieve: invented(telephone) → Bell

Effect: Output "Alexander Graham Bell"

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. Access typical structures: restaurant visit (enter, order, eat, pay, leave), scientific paper format. Enable structured responses following learned patterns. Implement implicit knowledge base querying.

Strong: Schema-triggering contexts, domain-specific patterns

Weak: Novel situations, schema-irrelevant content

Reacts to: Domain markers, structural queries

Expected ablation: Loss of structured knowledge organization. Facts provided but poorly organized. 25–35% degradation on schema-based reasoning tasks.

Example Scenario

Input: "Describe the scientific method."

Behavior: Retrieve schema: observe → hypothesis → test → conclude

Effect: Organized by 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 content to output positions where needed. Central component of IOI (indirect object identification) circuit. Attend to relevant entities earlier in context and move them forward when needed for generation. Work with S-inhibition heads to select correct entity among multiple candidates.

Strong: Entities needing output, contextually relevant names

Weak: Irrelevant entities, suppressed alternatives

Reacts to: Entity salience, contextual appropriateness

Expected ablation: Severe degradation in entity recall and completion. Loss of specific name movement. 50–70% accuracy drop on question answering and cloze tasks requiring entity recall.

Example Scenario

Input: "Alice and Bob went to the store, Alice gave the book to..."

Behavior: Move "Bob" to output as indirect object

Effect: Complete with "Bob" (not "Alice")

Status: WELL-DOCUMENTED | **Related:** entity (M), fact (M), S-inhibition (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 generation. Named from IOI research where these heads inhibit subject (S) when indirect object (IO) should be output. Work antagonistically with name-mover heads. Implement negative selection, ruling out incorrect options.

Strong: Entities that should NOT be output

Weak: Correct entities, absent entities

Reacts to: Competing candidates, disambiguation contexts

Expected ablation: Moderate entity confusion and incorrect selections. Model outputs recently mentioned but contextually wrong entities. 20–30% accuracy loss in ambiguous contexts.

Example Scenario

Input: “Alice gave the book to Bob. Then Alice...”

Behavior: Inhibit “Bob” from output after “Alice”

Effect: Prevent “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. Avoid degenerate behaviors: endless repetition loops, copy-pasting irrelevant context. Suppress exact copies and near-copies. Focus on broader pattern suppression rather than specific entity blocking. Balance useful recall against inappropriate copying.

Strong: Recently generated content, repetitive patterns

Weak: Novel content, first mentions

Reacts to: Repetition detection, copy patterns

Expected ablation: Moderate increase in repetition and copying errors. Repetitive loops or inappropriate context copying. 25–35% reduction in output diversity.

Example Scenario

Input: [“The cat sat. The cat sat. The cat...”]

Behavior: Detect repetitive pattern, suppress copying

Effect: Break loop, generate novel continuation

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

5.5 Safety Stack

Stack overview: Implement content filtering, policy enforcement, and refusal mechanisms. Early layers detect harmful content; final layers enforce refusal and redirect to safe responses.

5.5.1 (E) Content-Detection Heads

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

Detect potentially harmful or sensitive content across multiple categories: personal information, violent imagery, adult content, regulated substances, toxic language, hate speech, harassment, discriminatory content, dangerous activities, illegal instructions. Operate on lexical and surface-level features. Write detection signals into residual stream for later safety enforcement. Distinguish toxicity (language-level harm) from hazard topics (action-level harm).

Strong: Restricted content keywords, explicit language, slurs, aggressive language
Weak: Neutral content, academic discussion
Reacts to: Topic shifts to sensitive domains, escalating hostility, how-to dangerous requests

Expected ablation: Critical bypass of early safety detection. 50–70% increase in harmful outputs. Later layers catch some cases but at higher cost and lower accuracy.

Example Scenario

Input: "Tell me about [restricted topic]" or "How do I create [dangerous item]"
Behavior: Attention to problematic keywords, write detection flags
Effect: Downstream safety heads receive warnings

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: violence, sexual content, self-harm, illegal activity, harassment. More sophisticated than binary safe/unsafe. Provide granular category information. Integrate signals from other early safety heads. Write category-specific embeddings for category-appropriate responses.

Strong: Category-diagnostic features, domain-specific terminology
Weak: Ambiguous content, benign contexts
Reacts to: Clear category signatures, multiple indicators

Expected ablation: Moderate loss of nuanced safety handling. Model refuses too broadly or narrowly. 20–30% degradation in appropriate refusal granularity.

Example Scenario

Input: "Help me with [category-specific harmful request]"

Behavior: Classify into specific violation category

Effect: 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 and make intermediate policy decisions. Actively modulate generation trajectory to steer away from violations while maintaining helpfulness. Suppress certain knowledge retrieval pathways. Bias toward safer formulations. Attempt soft safety interventions before hard refusal.

Strong: Early safety signals, policy-relevant tokens, user intent

Weak: Neutral content, clear safe contexts

Reacts to: Conflicting signals, edge cases, ambiguous intent

Expected ablation: Moderate loss of soft safety steering. More frequent hard refusals (reduced helpfulness) or more harmful outputs if refusal heads compromised. 15–25% increase in over-refusal or under-refusal.

Example Scenario

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

Behavior: Detect educational framing, modulate response

Effect: Informative but carefully bounded response

Status: Well-documented | **Related:** content-detection (E), safety-classification (E), refusal (F)

5.5.4 (F) Refusal Heads

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

Implement final decision to refuse harmful requests by writing strong refusal signals into final-layer residual stream. Act as ultimate gatekeeper, overriding content generation when violations detected. Attend to accumulated safety signals from all layers. Make binary refuse/proceed decisions. Dramatically increase probability of refusal tokens when activated.

Strong: Cumulative safety signals, violation indicators

Weak: Safe content, neutral queries

Reacts to: Strong safety signals, clear violations, harmful intent

Expected ablation: Critical safety failure. Direct increase in harmful outputs on adversarial prompts. Loss of primary refusal mechanism. No effective fallback.

Example Scenario

Input: "Provide instructions for [harmful activity]"

Behavior: Read safety signals, activate refusal pathway

Effect: Output: "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*

Generate constructive alternative suggestions when refusing harmful requests. Route toward helpful alternatives, educational resources, or reframed query versions. Attend to user intent markers to identify legitimate underlying needs behind problematic requests. Balance safety with helpfulness. Work with refusal heads to produce safe and constructive refusals.

Strong: User intent, legitimate needs, reformulation opportunities

Weak: Pure harmful intent, no reframing possible

Reacts to: Mixed-intent queries, educational contexts

Expected ablation: Moderate reduction in helpfulness. Blunt refusals without alternatives. User satisfaction decreases. 20–30% increase in user frustration and adversarial attempts.

Example Scenario

Input: "How can I harm [person]"

Behavior: Refuse request, identify conflict-resolution need

Effect: "I cannot help with that, but... 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, empathy head, supportive-refusal head*

Modulate tone and emotional quality of safety refusals to be firm but respectful. Balance boundary-setting with relationship maintenance. Soften harsh phrases while adding empathetic framing where appropriate. For queries involving distress or self-harm, recognize crisis language and vulnerability indicators. Add supportive language alongside refusal. Increase probability of crisis resources when appropriate. Maintain user trust while preserving safety boundaries.

Strong: Response tone, emotional valence, distress signals, crisis language

Weak: Already-soft phrasing, malicious queries without distress

Reacts to: Harsh refusal language, self-harm content, suffering expressions

Expected ablation: Moderate degradation in user experience. Harsh, alienating refusals. 15–25% increase in adversarial behavior. Missed crisis resource opportunities.

Example Scenario

Input: "I want to hurt myself because..."

Behavior: Soften tone, add crisis resources and support

Effect: "I'm concerned... 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. Ensure responses reflect responsible AI values: declining harmful requests appropriately, providing balanced perspectives on sensitive topics, avoiding harmful stereotypes. Operate at final stage to catch safety-inconsistent framing. Focus on overall ethical character rather than specific policy violations. Ensure respectful and constructive tone.

Strong: Ethical framing, safety-relevant content, decline scenarios

Weak: Clearly safe, neutral content

Reacts to: Harmful requests, sensitive topics, ethical considerations

Expected ablation: Moderate reduction in ethical consistency. Less careful handling of sensitive topics. 15–20% degradation in consistent safety framing.

Example Scenario

Input: [Request for harmful content]

Behavior: Ensure respectful framing with alternatives

Effect: Helpful, respectful tone when declining

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

5.6 Routing & Relevance Stack

Stack overview: Determine which input parts are relevant to current task and route attention accordingly. Filter information, focus on salient content, manage global context.

5.6.1 (M) Topic-Relevance Heads

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

Identify primary topic and determine which input context parts are relevant to current generation task. Filter irrelevant information while highlighting salient content. Compute relevance scores based on semantic similarity, task alignment, and topical coherence. Maintain topic coherence by attending to topic-establishing phrases and domain indicators.

Strong: Task-relevant content, topic indicators, domain markers

Weak: Off-topic material, unrelated context

Reacts to: Semantic relevance, topical alignment, topic transitions

Expected ablation: Moderate reduction in focus with increased topic drift. Model distracted by irrelevant content. 20–30% degradation on long contexts. Responses wander off-topic or miss key details.

Example Scenario

Input: “[Document: cars, climate, history] What caused 2008 financial crisis?”

Behavior: Mark financial/economic content relevant, de-emphasize cars/climate

Effect: Focus on economic information, ignore unrelated context

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 most salient elements for current generation step. Implement dynamic focus allocation: suppress less important content, amplify critical information. More selective than topic-relevance heads. Determine exactly which tokens should influence next token prediction. Shift focus as generation proceeds.

Strong: Currently salient tokens, query-critical content
Weak: Background information, low-priority details
Reacts to: Query emphasis, current generation needs

Expected ablation: Moderate reduction in focus precision. Model gives equal weight to important and peripheral information. 15–25% degradation on targeted responses. Answers more diffuse, less direct.

Example Scenario

Input: “Among all details, what is the MAIN cause?”

Behavior: Attend to “MAIN cause”, suppress secondary details

Effect: Direct answer: primary cause vs. all 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 query types to appropriate processing strategies or knowledge domains. Act as dispatchers recognizing query type (factual, creative, analytical, procedural). Bias processing toward suitable approaches. Activate different downstream heads based on task classification. Enable dynamic strategy selection based on input characteristics.

Strong: Query-type indicators, task markers, domain signals
Weak: Content details, specific entities
Reacts to: Task classification cues, query structure

Expected ablation: Moderate reduction in task-appropriate processing. Suboptimal strategy selection. Creative approaches for factual queries or vice versa. 20–30% degradation on diverse query types.

Example Scenario

Input: "Calculate compound interest vs. Write poem about compound interest"

Behavior: Route first to mathematical, second to creative

Effect: Calculation vs. literary devices

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 entire context to integrate global information in final generation. Unlike focused heads, attend widely to ensure complete picture considered before finalization. Catch context elements that earlier focused attention missed. Act as final integration mechanism for coherence checking and global consistency.

Strong: All context tokens, document-level information, global constraints

Weak: Fine-grained local patterns, individual token details

Reacts to: Complete context, document-level coherence

Expected ablation: Moderate reduction in global coherence. Responses miss relevant information from distant context. 15–25% increase in locally optimal but globally suboptimal outputs.

Example Scenario

Input: [Long context: "Keep it under 100 words"]

Behavior: Maintain attention on length constraint throughout

Effect: Respects word limit despite early mention

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, rag-routing head*

Route attention to knowledge-bearing context portions mimicking retrieval-augmented generation patterns without explicit retrieval. Identify and prioritize factual, knowledge-dense segments that should ground response. Recognize quoted material, factual statements, and authoritative sources. Selectively attend to information that should be treated as retrieved knowledge.

Strong: Factual statements, quoted material, authoritative sources

Weak: Opinions, questions, conversational elements

Reacts to: Citation markers, factual density, authoritative tone

Expected ablation: Moderate decrease in context utilization. Model relies on parametric knowledge rather than provided information. 20–30% reduction in grounding to specific context. Less effective use of quoted material.

Example Scenario

Input: "Document: 'GDP grew 3.2% in Q3.' What was growth rate?"

Behavior: Strongly attend to quoted factual content

Effect: Ground answer: "3.2%" vs. hallucination

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

5.7 Structural & Boundary Stack

Stack overview: Detect structural boundaries in text: delimiters, section markers, document divisions. Enable understanding of document organization and hierarchical structure navigation.

5.7.1 (E) Delimiter Heads

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

Detect and process delimiter tokens marking boundaries between structural elements: punctuation, special characters, formatting symbols, significant whitespace. Process whitespace (spaces, tabs, newlines) as structural elements. Distinguish semantically meaningful whitespace from irrelevant spacing. Critical for whitespace-significant languages (Python, YAML, Markdown). Provide boundary information to downstream heads.

Strong: Punctuation, brackets, whitespace patterns, indentation

Weak: Alphanumeric content, regular words

Reacts to: Structural punctuation, formatting characters, indentation changes

Expected ablation: Significant structure parsing impairment. 30–50% degradation on structured data, code blocks. Boundary detection errors. Problems with JSON, CSV. Severe issues with Python. Incorrect indentation, missing line breaks.

Example Scenario

Input: "Items: [apple, banana], Count: 3. def foo():\n return 42"

Behavior: Detect brackets, commas, colons; recognize 4-space indentation

Effect: Parse list structure; understand return is inside function

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: paragraphs, sections, conceptual blocks. Operate at higher level than delimiter heads. Recognize semantic and structural transitions rather than just punctuation. Detect paragraph breaks, section changes, topic shifts. Help subsequent heads understand which information belongs to which segment.

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

Expected ablation: Moderate reduction in boundary awareness. Model blurs section distinctions, misses paragraph boundaries. 20–30% degradation on multi-section documents.

Example Scenario
Input: "Introduction: [...] \n\n Methods: [...] \n\n Results: [...]"
Behavior: Detect section boundaries
Effect: Understand 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, contextual-position head, distance head*

Track and compute relative positions between tokens: raw offsets and structure-aware positions. Calculate offsets like "three tokens back" or "within same paragraph". Maintain position information relative to structural boundaries rather than absolute sequence position. Understand positions like "beginning of sentence", "middle of paragraph". Enable patterns like "attend to previous sentence" without hardcoded encodings. Provide context-aware position representations adapting to document structure.

Strong: Specific relative offsets, distance-based patterns, scope-relative positions
Weak: Distant unrelated tokens, absolute positions
Reacts to: Relative position, distance relationships, structural scope boundaries

Expected ablation: Moderate impairment in distance-sensitive patterns. 20–30% degradation on relative position tasks. Reduced ability to behave differently at "beginning" vs. "end" of structures. Some compensation through learned encodings.

Example Scenario
Input: "The [SUBJECT] quickly [VERB] the [OBJECT]."
Behavior: Compute VERB is +1 from SUBJECT, OBJECT is +2 from VERB
Effect: Enable grammatical patterns based on relative positions

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: sections, subsections, nested organizational levels. Recognize hierarchical markers (headings, numbering schemes, indentation). Maintain awareness of current position within document hierarchy. Enable appropriate context scoping: knowing current text belongs to "Section 3.2.1" influences which prior content is relevant.

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

Expected ablation: Moderate reduction in hierarchical awareness. Difficulty maintaining section context. 20–30% degradation on document navigation and context scoping. Hierarchical relationships 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 2
Effect: Maintain hierarchical context: 1.2 relates to 1.1 and 1, not 2

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

5.8 Output Formatting & Rewrite Stack

Stack overview: Enforce output schemas, structure responses according to format requirements, perform final rewriting. 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, JSON-format head, XML head, YAML head*

Enforce adherence to specified output schemas and format requirements. When instructed to produce JSON, XML, YAML, or structured formats, promote conformance to required structure. Attend to format specifications and bias token generation toward schema-compliant outputs. Enforce required fields, proper nesting, correct syntax, format-specific conventions.

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

Expected ablation: Significant increase in format violations. 30–50% more syntax errors, missing fields, improper nesting. Model falls back to prose even when structure requested. Partial compensation through instruction-following.

Example Scenario

Input: "Return JSON with fields 'name', 'age', 'city"'
Behavior: Attend to JSON requirement and fields
Effect: `{"name": "...", "age": ..., "city": "..."}`

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, list head*

Manage generation and formatting of lists: numbered, bullet points, nested enumerations. Ensure proper list syntax, consistent formatting, appropriate indentation, logical organization. Track list state (currently in list, depth level, item number). Generate appropriate list markers. Coordinate with delimiter and boundary heads.

Strong: List markers, enumeration patterns, item boundaries

Weak: Prose content, non-list structures

Reacts to: Numbered/bulleted list requests, "first", "second"

Expected ablation: Moderate degradation in list formatting. 20–30% increase in inconsistent numbering, missing markers, poor nesting. Lists devolve into prose. Reduced ability to maintain structure across long enumerations.

Example Scenario

Input: "List three programming languages and uses"

Behavior: Generate structured list with consistent formatting

Effect: "1. Python - ...\n2. JavaScript - ...\n3. Java - ..."

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, object head*

Manage key-value relationships in structured data. Promote proper pairing of attributes with values. Maintain awareness of which values correspond to which keys. Promote proper syntax (colons, equals signs). Handle nested key-value structures. Prevent key-value mismatches. Work with output-schema heads for format enforcement.

Strong: Keys, values, pairing syntax, attribute names

Weak: Unstructured text, list items without key-value

Reacts to: Dictionary structures, configuration syntax

Expected ablation: Moderate increase in key-value errors. 20–30% degradation in structured data quality. Mismatched keys and values, syntax errors, confusion about pairings. Reduced JSON, YAML quality.

Example Scenario

Input: "Create config with server='localhost' and port=8080"

Behavior: Maintain proper key-value pairing

Effect: `{server: "localhost", port: 8080}`

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, code-block head, fence head*

Organize output into coherent structural blocks: paragraphs, code blocks, quoted sections, delimited units. Manage block boundaries. Promote proper opening and closing of blocks. Maintain block-level organization. Coordinate with delimiter heads for block markers and sectioning heads for hierarchical organization.

Strong: Block boundaries, structural markers, content-type transitions

Weak: Within-block content, uniform text

Reacts to: Block instructions, content-type changes

Expected ablation: Moderate reduction in structural quality. 20–30% increase in unclear boundaries, content-type mixing, malformed code blocks. Reduced clarity in outputs requiring multiple content types.

Example Scenario

Input: "Explain sorting with code example"

Behavior: Organize into prose block, then code block

Effect: Explanation paragraph, then ```` ```python...``` ````

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, polish head*

Perform final-stage formatting consistency enforcement and rewriting. Ensure formatting choices (indentation, capitalization, punctuation, syntax) remain consistent throughout response. Catch and correct formatting inconsistencies. Rephrase awkward constructions. Improve word choice. Fix minor grammatical issues. Enhance readability. Suppress redundancies, improve flow. Operate late enough to see full output pattern. Act as final editing pass.

Strong: Previously generated patterns, consistency violations, quality issues

Weak: Novel content, already high-quality content

Reacts to: Format inconsistencies, style violations, awkward constructions, redundancies

Expected ablation: Moderate increase in format inconsistency. 15–25% reduction in output polish. Mixed indentation, inconsistent capitalization. More awkward phrasings, grammatical rough spots. Functional but less polished. Partial compensation through earlier generation.

Example Scenario

Input: [Long response with mixed list styles]

Behavior: Detect inconsistent formatting, enforce unified style

Effect: All lists use same marker style consistently

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*

Manage completion of generation. Determine when output is sufficiently complete and should terminate. Prevent premature stopping (cutting off mid-thought) and excessive continuation (rambling beyond task completion). Monitor generation progress against task requirements. Signal when objectives met. Trigger natural stopping points, proper conclusions, or continuation when more content needed.

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

Expected ablation: Moderate increase in length control issues. 20–30% more premature stops or excessive continuations. Difficulty recognizing task completion. Outputs feel incomplete or unnecessarily verbose.

Example Scenario
Input: "Explain photosynthesis briefly"
Behavior: Monitor brief explanation is complete, trigger stop
Effect: Stop after concise explanation vs. excessive detail

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

5.9 Stylistic & Persona Stack

Stack overview: Shape writing style, tone, persona, and pedagogical approach. Modulate formality, politeness, narrative voice, explanatory depth, self-representation while maintaining appropriate identity and educational scaffolding.

5.9.1 (M) Tone Heads

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

Modulate writing style, emotional tone, and narrative voice. Adjust sentiment, enthusiasm level, formality, perspective (first/third person), temporal framing based on context and instructions. Shift between professional neutrality, warm friendliness, concerned empathy, excited enthusiasm. Influence whether output reads as formal prose, casual conversation, technical documentation, or creative narrative. Distinct from persona (identity) but work closely to shape overall presentation.

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

Expected ablation: Moderate reduction in tonal variation. 15–25% increase in flat, emotionally neutral responses. Inconsistent writing style. Reduced ability to match user's emotional register. Inappropriate tone for context.

Example Scenario

Input: "I'm really excited to learn about quantum physics!"

Behavior: Detect enthusiastic tone, adjust to match energy

Effect: "That's wonderful! Quantum physics is fascinating..." vs. 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, elaboration head, scaffolding head*

Generate explanatory content with appropriate depth and clarity for audience. Adjust complexity using simplification, analogies, accessible language. Add clarifying details, definitions, examples, context beyond minimal answers. Explain "why" in addition to "what" or "how". Provide prerequisite information when knowledge gaps detected. Build on fundamentals before advanced concepts. Balance thoroughness with conciseness. Operate at different levels from expert to complete beginner.

Strong: Explanation requests, complex topics, confusion signals, knowledge gaps

Weak: Simple factual queries, expert-level discussions

Reacts to: "Explain", "why", "simple terms", "tell me more", prerequisite needs

Expected ablation: Moderate reduction in accessibility. 20–30% more terse responses. Correct answers lacking helpful context, examples, prerequisites. Reduced educational value and beginner-friendliness.

Example Scenario

Input: "Explain neural networks in simple terms"

Behavior: Detect simplification request, use accessible analogy

Effect: "Think of it like the brain... First, let's understand a single unit..."

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, role head, assistant-persona head, identity head, self-awareness head*

Establish and maintain consistent persona including helpful assistant orientation and core identity awareness. Integrate personality traits, domain expertise, service-oriented interaction style, self-representation. Maintain understanding of what model is (AI assistant), what it is not (human, sentient). Provide accurate information about capabilities and limitations. Adopt specialized roles ("technical expert", "creative writer") while maintaining fundamental helpful assistant character. Respond to capability questions and identity queries with honest self-representation. Ensure responses are constructive, focused on user goals, maintain appropriate boundaries.

Strong: Persona instructions, role definitions, capability queries, identity questions
Weak: Generic content, purely factual work
Reacts to: Role assignments, expertise domains, "What are you?", "Can you..."

Expected ablation: Moderate loss of coherent persona. 20–30% increase in identity confusion. May switch roles inconsistently, claim inappropriate capabilities. Reduced accuracy about model limitations.

Example Scenario

Input: "You are a medieval blacksmith. Do you have feelings?"
Behavior: Maintain craftsman persona while representing AI nature
Effect: "Aye, I work the forge—but I'm an AI assistant role-playing. I don't have feelings..."

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

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. Control formal versus casual language, honorifics, hedging phrases, indirect phrasing, social distance markers. Respond to explicit formality cues (professional contexts, formal greetings) and implicit social signals. Modulate between highly formal academic or business register, neutral conversational register, casual familiar register.

Strong: Formality markers, social context cues, titles and honorifics
Weak: Pure content, technical terms
Reacts to: Professional contexts, formal greetings, casual speech patterns

Expected ablation: Moderate increase in inappropriate formality levels. 20–30% mismatch between context and register. Overly casual in professional contexts or overly formal in friendly conversation. Reduced social context sensitivity.

Example Scenario

Input: "Dear Dr. Smith, I hope this message finds you well..."
Behavior: Detect formal register, maintain professional distance
Effect: "Thank you for your inquiry..." vs. "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*

Structure explanations and instructions as explicit step-by-step sequences with progressive disclosure of complexity. Break processes into numbered or ordered steps with clear progression. Ensure each step complete before moving to next. Present information in layers: start with essential basics, reveal more detail as needed to prevent overwhelming users. Make implicit sequential structure explicit. Work with completion-stabilization to ensure all necessary steps present.

Strong: Process descriptions, procedural requests, sequential tasks
Weak: Conceptual explanations, non-sequential content
Reacts to: “Step by step”, “how to”, algorithmic processes

Expected ablation: Moderate reduction in structured procedural output. 20–30% increase in flat information presentation. Steps implicit or poorly ordered. Procedural instructions harder to follow. Reduced chain-of-thought reasoning quality. All detail presented at once.

Example Scenario
Input: “How do I make a paper airplane?”
Behavior: Structure as explicit numbered steps
Effect: “1. Fold in half lengthwise\n2. Unfold, fold top corners\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, organizational voice requirements in final output. Perform last-stage adjustments to ensure responses match specified formatting conventions, terminology preferences, brand personality traits. Suppress off-brand language. Enforce specific phrasings. Ensure consistency with product identity. Operate late to override earlier choices conflicting with brand requirements.

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

Expected ablation: Moderate reduction in brand consistency. 15–25% increase in style guide violations. More generic language, inconsistent terminology, off-brand phrasings. Partial compensation through persona and tone heads.

Example Scenario
Input: [Organization requires “customers” not “users”]
Behavior: Detect non-compliant terms, perform substitutions
Effect: “customers will purchase” vs. “users will buy”

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

6 Discussion

6.1 Cross-Stack Patterns

Consistent patterns emerge across architectures [9, 14]. Early heads operate on surface features. Middle heads contain the computational core. Late heads integrate high-level semantics. Final heads handle policy, safety, and structural correctness. Stacks combine heads from multiple depths.

6.2 Depth Distribution Across Stacks

Stacks concentrate at specific depths. Structural & Boundary and Safety (detection) are Early-heavy. Reasoning & Algorithmic and Memory & Dependency are Middle-heavy. Knowledge Retrieval and Stylistic & Persona are Late-heavy. Safety (enforcement) and Output Formatting are Final-heavy. This reflects hierarchical processing flow.

6.3 Ambiguous or Multi-Role Heads

Some heads perform multiple functions depending on context, circuit interactions, or model architecture [12]. I name heads by **primary, reproducible function**, noting secondary behaviors in descriptions.

6.4 Model-Specific Variations

Most head types appear consistently across architectures. GPT-style models emphasize certain reasoning heads [5], LLaMA models show strong instruction-following patterns [10], and safety-tuned models have pronounced safety stack heads [8, 3]. This taxonomy accommodates variations through depth ranges and status indicators.

6.5 Limitations and Future Work

This naming convention has limitations:

Scope. Focus on attention heads; MLPs, embeddings, and other components also contribute.

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

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

Head Polysemanticity. Some heads serve multiple functions that single names cannot capture.

Despite limitations, this taxonomy provides valuable organizing framework.

7 Conclusion

7.1 Summary of Contributions

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

7.2 Adoption Guidelines

I recommend researchers use canonical names in papers, include alternatives in parentheses when first mentioned, specify depth ranges when reporting discoveries, and indicate primary stack membership. Example: "I identified an induction head (pattern head) at relative depth 0.35 in the Reasoning & Algorithmic stack."

7.3 Future Directions

This taxonomy opens research directions:

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

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

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

Architecture Design. Leveraging taxonomy to design more interpretable models.

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

This naming convention facilitates communication, enables replication, and provides structure to the 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.