# Transformer Mechanisms: A Functional Taxonomy

Karol Kowalczyk

November 2025

**Abstract**

Transformer models implement computation through mechanisms that span attention heads, MLP layers, and multi-component circuits. Current mechanistic interpretability research suffers from fragmented naming: the same mechanism appears under different names across research traditions, while identical names refer to distinct computational functions. This work proposes a mechanism-first taxonomy organizing 35 computational mechanisms into 8 functional stacks, with each mechanism described independently of its implementation substrate. The taxonomy enables cross-architecture comparison, supports multi-component integration, and provides stable vocabulary synthesizing findings from attention analysis, MLP studies, circuit tracing, and sparse autoencoder research.

## Contents

# 1 Introduction

## 1.1 The Problem: Fragmented Mechanism Naming

Mechanistic interpretability has identified induction heads enabling few-shot learning [4], characterized MLPs as key-value memories [2], traced multi-component circuits like the IOI circuit [5], and extracted monosemantic features via sparse autoencoders [1]. Yet naming conventions remain inconsistent and component-centric.

The same computational mechanism appears under multiple names: pattern completion is called "induction head," "pattern head," "copy head," or "ICL head" depending on research tradition. Conversely, "copy head" refers to at least three distinct mechanisms: pattern completion, name-mover routing, and position-based copying.

The root problem is **component-first thinking**: organizing by implementation substrate (attention heads, MLP neurons) rather than computational function. This creates artificial boundaries preventing systematic comparison across models and obscuring relationships between components and the mechanisms they implement.

## 1.2 The Solution: Mechanism-First Organization

This taxonomy applies three organizing principles:

**1. Separate mechanism from implementation.** Each mechanism describes computational transformation independently, then maps to implementing components (attention heads, MLPs, circuits, SAE features).

**2. Organize by computational function.** Eight functional stacks reflect the transformer pipeline: pattern detection, memory retrieval, routing, transformation, reasoning, safety, output control, and composition.

**3. Support multi-component integration.** Sophisticated mechanisms require coordinated computation across attention, MLPs, and depth. The taxonomy explicitly represents this composition.

## 1.3 Scope and Contributions

This work catalogs 35 mechanisms meeting four criteria: (1) observed across multiple models, (2) validated through ablation, (3) mechanistically understood at component level, (4) reproducible using standard techniques.

**Key contributions:**

- Standardized vocabulary bridging attention, MLP, circuit, and SAE research traditions
- Functional organization revealing transformer computational architecture
- Multi-component integration patterns for sophisticated mechanisms
- Implementation specifications enabling cross-architecture comparison

- Empirically grounded descriptions synthesizing interpretability findings

### 1.4 Document Structure

Section 2 explains the mechanism-first framework and depth-based organization. Sections 4.1 through 4.8 catalog mechanisms by functional stack. Section 5 examines patterns, limitations, and future directions.

## 2 Conceptual Framework

### 2.1 Mechanism vs. Component

**Mechanisms** describe computational transformations independently of substrate: pattern completion, factual recall, output routing. **Components** are architectural primitives: attention heads, MLP layers, circuits. The relationship is many-to-many: mechanisms typically require multiple components, and components contribute to multiple mechanisms.

**Three organizational levels:**

1. **Attention heads** implement learned routing patterns (previous-token, induction, name-mover)
2. **MLP layers** implement associative memory and nonlinear transformation
3. **Circuits** integrate attention and MLPs across layers into complex behaviors

### 2.2 Depth-Based Organization

Transformer layers exhibit systematic functional specialization across depth. This taxonomy uses relative depth notation: $d_{\text{rel}} = l/L$ where $l$ is layer index and $L$ is total layers.

**Four depth regions:**

- **Early (E, 0.00–0.25):** Surface processing, syntactic patterns, instruction markers
- **Middle (M, 0.25–0.70):** Core computation, facts, entities, reasoning
- **Late (L, 0.70–0.88):** Integration, output routing, strategy selection
- **Final (F, 0.88–1.00):** Constraint enforcement, safety, format control

This enables architecture-independent mechanism descriptions: a mechanism at 0.40 relative depth occupies similar functional space in 12-layer (layer 5), 48-layer (layer 19), and 96-layer (layer 38) models.

### 2.3 Methodology and Confidence Levels

Mechanisms are identified through converging evidence from five techniques: attention pattern analysis, ablation studies, activation patching, logit attribution, and sparse autoencoder analysis.

**Confidence levels:**

- **Well-documented:** Extensive empirical support across multiple models
- **Observed:** Solid evidence but less extensive characterization
- **Proposed:** Reasonable hypotheses with preliminary evidence

# 3 Eight Functional Stacks

The taxonomy organizes 35 mechanisms into eight functional stacks:

**1. Pattern & Sequential Stack** (5 mechanisms, 0.05–0.65): Pattern detection and completion, sequence continuation, positional processing.

**2. Memory & Knowledge Stack** (6 mechanisms, 0.08–0.85): Factual recall, entity grounding, schema retrieval, output routing.

**3. Routing & Context Stack** (5 mechanisms, 0.10–0.85): Relevance filtering, attention focusing, task routing, context aggregation.

**4. Feature Transformation Stack** (3 mechanisms, 0.20–0.80): Nonlinear composition, abstract concept formation, semantic integration.

**5. Reasoning & Inference Stack** (5 mechanisms, 0.40–0.88): Multi-step reasoning, planning, consistency checking, analogical mapping, causal inference.

**6. Safety & Policy Stack** (5 mechanisms, 0.05–0.98): Harmful content detection, policy enforcement, refusal generation, jailbreak resistance.

**7. Output & Quality Stack** (4 mechanisms, 0.35–0.98): Format enforcement, style modulation, explanation generation, completion control.

**8. Composition & Integration Stack** (4 mechanisms, system-level): Cross-layer circuits, multi-head coordination, attention-MLP composition, representational superposition.

# 4 Mechanism Catalog

This section catalogs mechanisms using a compressed format: **Function** describes computational transformation, **Implementation** specifies components, **Ablation** summarizes behavioral changes when removed, **Example** provides concrete input→mechanism→output.

## 4.1 Pattern & Sequential Stack

**Stack overview:** Detect and complete patterns from context. Enable in-context learning through pattern matching and repetition detection. Support sequence continuation and algorithmic behavior. Process positional information for order-aware computation.

### 4.1.1 Pattern Completion Mechanism

**Depth:** `0.05-0.58 (E-M)` | **Primary Implementation:** Attention heads + MLP neurons | **Literature names:** *induction mechanism, pattern completion, in-context learning*

Detect and complete patterns of form [A][B]...[A] → predict [B]. Enable in-context learning through multi-stage circuit combining position-based copying, repetition detection, pattern matching, and n-gram retrieval. Foundation for few-shot learning and analogical reasoning. Support pattern-based prediction without parameter updates. Implement through coordinated multi-component circuit spanning 3–8 layers.

**Attention Heads:** Previous-token heads (E, 0.05–0.20) create shifted representations through uniform offset attention. Duplicate-token heads (M, 0.30–0.58) detect repeated elements via exact token matching. Induction heads (M, 0.25–0.55) match patterns by attending to previous occurrences of current token content.
**MLP Layers:** N-gram neurons (E-M, 0.15–0.55) store frequent bigram and trigram patterns as key-value associations. Provide statistical support for pattern completion.
**Circuit:** Previous-token → Induction → MLP retrieval working in composition across 3–8 layers. Unified pattern completion circuit integrating multiple specialized components.

**Expected ablation:** Severe loss of in-context learning capability. Major degradation on few-shot tasks requiring pattern-based generalization. Loss of pattern completion and analogical continuation. Reduced ability to learn from examples provided in prompt without fine-tuning.

Example

*Input:* "When Mary and John went to the store, Mary gave milk to John. When Susan and Bob went to the store, Susan gave milk to..."
*Behavior:* Detect pattern [A] gave milk to [B], previous-token creates shifted representations, induction heads match pattern, duplicate-token tracks repetition
*Effect:* Output "Bob" by analogical pattern completion across contexts

**Status:** Well-documented | **Related:** algorithmic-continuation, entity-grounding, output-routing

### 4.1.2 Algorithmic Continuation Mechanism

**Depth:** `0.35-0.65 (M)` | **Primary Implementation:** Attention heads + MLP neurons | **Literature names:** *sequence continuation, rule following, mathematical pattern completion*

Continue algorithmic sequences by detecting underlying rules: counting, arithmetic progressions, alphabetic sequences, mathematical patterns. Extract systematic rules from examples and apply consistently. Support rule-based generation beyond memorization. Enable mathematical and logical pattern completion. Integration point between pattern matching and reasoning mechanisms. Implement structured sequence understanding.

**Attention Heads:** Algorithmic heads (M, 0.35–0.60) detect regular patterns and systematic relationships in sequences through content-based attention to sequence structure.
**MLP Layers:** Rule storage neurons (M, 0.40–0.65) encode mathematical operations and sequence transformation rules. Store procedural knowledge for systematic continuation.
**Circuit:** Integrates with reasoning mechanisms (M-L) for complex rule application. Bridges pattern detection and symbolic reasoning.

**Expected ablation:** Significant loss of algorithmic continuation ability. Major degradation on sequence completion tasks requiring rule extraction. Reduced performance on mathematical and logical patterns. Loss of systematic rule application. Increased reliance on memorized sequences rather than rule understanding.

Example

*Input:* "2, 4, 8, 16, 32, ..."
*Behavior:* Detect doubling pattern through ratio analysis, activate multiplication rule, apply systematically
*Effect:* Output "64" as next power of 2 through rule-based continuation

**Status:** Observed | **Related:** pattern-completion, multi-step-reasoning

#### 4.1.3 Local Context Modeling Mechanism

**Depth:** `0.08-0.30 (E)` | **Primary Implementation:** Attention heads + MLP neurons | **Literature names:** *local pattern detection, instruction markers, syntactic processing*

Process immediate local context (1–5 tokens) for instruction markers, syntactic structure, and local patterns. Distinct from Pattern Completion which operates at broader scale (10+ tokens). Enable detection of formatting directives, instruction keywords, and immediate syntactic relationships. Provide foundation for instruction following and local structure understanding. Support early-layer linguistic processing.

**Attention Heads:** Instruction-detection heads (E, 0.08–0.25) attend to directive language and formatting markers in immediate context. Local pattern heads (E, 0.10–0.30) process adjacent token relationships for syntactic structure.
**MLP Layers:** N-gram context neurons (E, 0.15–0.30) encode local sequential patterns and common instruction templates.
**Circuit:** Early detection stage feeding instruction following and format enforcement mechanisms in later layers.

**Expected ablation:** Moderate loss of local pattern detection and immediate context processing. Reduced instruction recognition accuracy. Notable degradation on format-sensitive tasks. Instruction following mechanisms receive weaker early signals. Minor syntactic processing impairment.

Example

*Input:* "List exactly 3 items in bullet format"

*Behavior:* Detect "List exactly 3" as instruction constraint, "bullet format" as formatting directive in immediate 1–5 token window

*Effect:* Write instruction signals to residual stream for downstream enforcement

**Status:** PROPOSED | **Related:** pattern-completion, format-enforcement

#### 4.1.4 Repetition & Cycle Recognition Mechanism

**Depth:** `0.30-0.60 (M)` | **Primary Implementation:** Attention heads | **Literature names:** *repetition detection, cycle detection, recurring pattern recognition*

Detect repetition at multiple scales: token-level duplicates, phrasal repetition, cyclical patterns, recurring structures. Serve multiple downstream mechanisms beyond pattern completion alone. Provide repetition signals for entity tracking, output routing, and IOI circuits. Enable detection of recurring themes and structural cycles. Support disambiguation through repetition awareness.

**Attention Heads:** Duplicate-token heads (M, 0.30–0.58) perform exact token identity matching across arbitrary distances. Cycle-detection patterns (M, 0.35–0.60) identify recurring sequences and structural repetition.

**Circuit:** Feeds into multiple mechanisms: pattern completion (induction), entity tracking (coreference), output routing (IOI circuit), disambiguation circuits. Multi-purpose repetition detection serving diverse downstream computation.

**Expected ablation:** Moderate degradation in pattern matching and entity tracking. Notable impact on IOI circuit performance. Reduced disambiguation accuracy when repetition provides disambiguating signal. Loss of cycle and recurrence detection capability affecting multiple downstream mechanisms.

Example

*Input:* "The cat climbed the tree. The cat..."

*Behavior:* Second "cat" detects first "cat" as duplicate, signal repetition for entity tracking and coreference resolution

*Effect:* Enable entity linking and support pattern completion circuits through repetition signal

**Status:** PROPOSED | **Related:** pattern-completion, entity-grounding, output-routing

#### 4.1.5 Position-Based Processing Mechanism

**Depth:** `0.05-0.65 (E-M)` | **Primary Implementation:** Architecture + Attention heads | **Literature names:** *positional encoding, position representation, order information*

Encode and process positional information (absolute and relative) to enable order-aware computation. Provide transformers with sequence order information absent from raw token embeddings. Distinguish token order and maintain structural position awareness. Implement various positional schemes: absolute positions, relative positions, learned encodings. Enable position-dependent patterns and sequential reasoning. Foundation for all order-aware computation.

**Architecture:** Positional encodings added to input embeddings. Various implementations: sinusoidal (absolute), learned (absolute), rotary (relative). Architecture-level provision of position information.
**Attention Heads:** Positional heads (E, 0.05–0.20) process absolute position information for early-layer position-aware patterns. Relative-position heads (M, 0.35–0.65) compute position relationships and distance-dependent attention.
**Circuit:** Spans input encoding through middle layers. Absolute position (E) provides foundation, relative position (M) enables structural understanding.

**Expected ablation:** Severe loss of order-awareness. Model treats sequences as bags of words with position-invariant processing. Major degradation on tasks requiring sequential understanding. Loss of position-dependent patterns. Inability to distinguish "Alice followed Bob" from "Bob followed Alice". Critical failure of order-sensitive computation.

Example

*Input:* "Alice followed Bob" vs. "Bob followed Alice"
*Behavior:* Use positional encodings to distinguish subject position from object position based on token order
*Effect:* Understand opposite meanings despite identical token sets through position information

**Status:** WELL-DOCUMENTED | **Related:** pattern-completion, local-context-modeling

## 4.2 Memory & Knowledge Stack

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

### 4.2.1 Factual Recall Mechanism

**Depth:** `0.35-0.75 (M-L)` | **Primary Implementation:** MLP neurons + Attention heads | **Literature names:** *fact retrieval, knowledge recall, factual memory*

Retrieve factual associations and relationships stored in model parameters during training. Access learned knowledge: entity properties, relational facts, world knowledge. Implement distributed key-value memory: query patterns (keys) activate factual content (values). Store knowledge hierarchically across depths: surface patterns (early), core facts (middle), abstract concepts (late). Enable factual grounding without external retrieval. Support question answering and knowledge-intensive generation.

**MLP Layers:** Knowledge neurons (M-L, 0.35–0.75) store factual associations distributed across layers. First sublayer: detect entity/query patterns (key matching). Second sublayer: provide factual content (value retrieval). Single fact distributed across multiple neurons; single neuron contributes to multiple facts.
**Attention Heads:** Fact retrieval heads (M, 0.38–0.62) identify factual queries. Entity heads (M, 0.35–0.65) detect entities requiring fact retrieval.
**Circuit:** Entity detection → MLP fact retrieval → name-mover output across 5–15 layers. Distributed factual recall circuit.

**Expected ablation:** Severe loss of factual knowledge. Linguistic fluency maintained but factual grounding lost. Major degradation on knowledge-intensive tasks and question answering. Model produces plausible-sounding but factually incorrect content. Reduced accuracy on entity property questions and relational reasoning.

Example

*Input:* "The Eiffel Tower is located in..."
*Behavior:* Detect "Eiffel Tower" entity, retrieve location association from MLP parameters via key-value memory
*Effect:* Output "Paris" via stored factual knowledge without external retrieval

**Status:** Well-documented | **Related:** entity-grounding, schema-retrieval, output-routing

#### 4.2.2 Entity Grounding Mechanism

**Depth:** `0.08-0.65 (E-M, multi-stage)` | **Primary Implementation:** Attention heads | **Literature names:** *entity identification, entity tracking, coreference resolution*

Identify, track, and link entity mentions across context. Recognize named entities (people, places, organizations) and their properties. Link different references to same entity: full names, abbreviations, nicknames, pronouns, descriptions. Resolve coreference through multi-stage processing: early syntactic binding, middle semantic resolution. Maintain unified entity representations across long contexts. Enable entity-aware processing for factual retrieval and reasoning. Ground references in specific entities rather than generic concepts.

**Attention Heads:** Entity heads (M, 0.35–0.65) attend strongly to proper nouns and entity mentions. Reference resolution heads (E, 0.08–0.25) perform initial pronoun-to-noun binding using syntactic cues. Coreference heads (M, 0.35–0.60) resolve complex cases requiring semantic understanding. Duplicate-token heads (M, 0.30–0.58) detect repeated entity mentions.
**Circuit:** Multi-stage resolution: syntactic binding (E) → semantic integration (M) → entity tracking (M-L). Entity detection → factual retrieval → output routing. Parallel tracking of multiple entities across context.

**Expected ablation:** Significant degradation in entity-based reasoning and factual accuracy. Loss of entity linking and tracking across references. Major accuracy drop on who/what/where questions. Confusion between different entities with similar names. Reduced coreference resolution capability. Difficulty maintaining entity coherence in long contexts.

Example

*Input:* "Apple Inc. released new products. AAPL stock rose. The company announced..."

*Behavior:* Link "Apple Inc.", "AAPL", and "the company" to single entity through entity heads and coreference resolution

*Effect:* Maintain unified entity representation across diverse referring expressions

**Status:** Well-documented | **Related:** factual-recall, output-routing, pattern-completion

#### 4.2.3 Schema Retrieval Mechanism

**Depth:** `0.45-0.70 (M-L)` | **Primary Implementation:** MLP neurons + Attention heads | **Literature names:** *template retrieval, structural knowledge, procedural memory*

Retrieve structured knowledge schemas, templates, and typical sequences from training. Access organizational patterns: event scripts (restaurant visit: enter, order, eat, pay, leave), document structures (research paper: abstract, introduction, methods, results, discussion), procedural knowledge (scientific method steps). Enable structured generation following learned patterns. Support script-based reasoning about typical situations and conventional formats. Provide organizational frameworks for complex content generation.

**MLP Layers:** Schema storage neurons (M-L, 0.45–0.70) encode structured templates and procedural knowledge as hierarchical patterns. Store conventional sequences and organizational frameworks.

**Attention Heads:** Schema retrieval heads (M, 0.45–0.68) detect schema-triggering contexts and activate appropriate templates.

**Circuit:** Context detection → schema activation → structured generation. Templates guide multi-step generation.

**Expected ablation:** Loss of structured knowledge organization. Facts provided but poorly organized. Notable degradation on tasks requiring conventional formats or typical sequences. Reduced ability to follow procedural patterns. Difficulty generating well-structured documents. Loss of script-based reasoning for common scenarios.

Example

*Input:* "Describe the water cycle."

*Behavior:* Retrieve cyclical process schema from MLP storage, activate sequential template

*Effect:* Organized response following natural process structure: evaporation → condensation → precipitation → collection

**Status:** Observed | **Related:** factual-recall, multi-step-reasoning

#### 4.2.4 Long-Range Dependency Maintenance Mechanism

**Depth:** `0.40-0.65 (M)` | **Primary Implementation:** Attention heads | **Literature names:** *long-distance attention, dependency maintenance, distant connection*

Maintain connections between syntactically or semantically related elements across large distances (20–100+ tokens). Track dependencies without degradation over distance. Implement transformer advantage over RNNs: direct long-distance connections. Support nested structures,

long-distance agreement, and complex syntactic relationships. Maintain multiple simultaneous long-range connections. Enable semantic relationship maintenance for knowledge integration across extended context.

**Attention Heads:** Long-range dependency heads (M, 0.40–0.65) attend across large token distances to maintain syntactic and semantic relationships. Relatively flat attention distribution over distant elements enabling distance-invariant connection maintenance.
**Circuit:** Parallel to local processing. Maintains global structural information while other mechanisms process local patterns. Supports entity tracking and factual recall across long contexts.

**Expected ablation:** Notable degradation on complex sentences and long-range relationships. Significant performance loss on long-distance syntactic agreement. Severe impact on nested structures and embedded clauses. Model treats distant elements as independent. Reduced ability to maintain semantic coherence across extended contexts.

Example

*Input:* "The book [that Alice mentioned [that Bob recommended]] was excellent."
*Behavior:* "was" attends to "book" across two levels of embedding, maintaining subject-verb dependency
*Effect:* Maintain grammatical agreement despite intervening nested clauses

**Status:** Observed | **Related:** entity-grounding, factual-recall

#### 4.2.5 Output Routing Mechanism

**Depth:** `0.60-0.85 (L)` | **Primary Implementation:** Attention circuit | **Literature names:** *information movement, content copying, answer extraction*

Move retrieved information to output positions where needed for generation. Route entities, facts, and content from earlier context to prediction position. Implement competitive selection among multiple candidates through antagonistic head coordination. Suppress incorrect alternatives while promoting correct content. Multi-stage process: candidate identification, competition through S-inhibition, selection, movement to output. Central mechanism for question answering, completion, and factual generation. Enable disambiguation in ambiguous contexts.

**Circuit:** Name-mover heads (L, 0.60–0.80) attend to relevant content and copy to output position. S-inhibition heads (L, 0.62–0.82) suppress contextually inappropriate alternatives by attending to wrong answers and decreasing their logits. Copy-suppression heads (L, 0.65–0.85) prevent inappropriate repetition. Multi-head circuit implementing competitive selection through coordinated attention.
**Attention Heads:** IOI (Indirect Object Identification) circuit extensively studied: duplicate-token detection → S-inhibition → name-mover across 8–12 layers. Canonical example of competitive output routing.

**Expected ablation:** Severe degradation in converting knowledge to output. Model knows facts but cannot output them correctly. Major accuracy drop on question answering and cloze completion. Entity confusion and selection errors in ambiguous contexts. Increased output of recently mentioned but incorrect entities. Loss of competitive selection capability.

Example

*Input:* "Alice and Bob went shopping. Alice gave the receipt to..."
*Behavior:* Duplicate-token detects "Alice" repetition, S-inhibition suppresses subject "Alice", name-mover copies indirect object "Bob" to output
*Effect:* Complete with "Bob" through IOI circuit competitive selection

**Status:** Well-documented | **Related:** entity-grounding, factual-recall, repetition-cycle-recognition

#### 4.2.6 Memory Consolidation Mechanism

**Depth:** `0.50-0.75 (M-L)` | **Primary Implementation:** MLP neurons + Attention heads | **Literature names:** *knowledge integration, cross-memory synthesis, multi-source retrieval*

Integrate multiple memory sources (facts, schemas, entities) into coherent knowledge representations. Handle queries requiring synthesis across memory types: entity properties combined with factual knowledge within schematic frameworks. Enable cross-memory integration for complex questions. Support knowledge composition beyond single-fact retrieval. Coordinate between factual recall, schema retrieval, and entity grounding mechanisms.

**MLP Layers:** Memory integration neurons (M-L, 0.50–0.75) coordinate information from multiple memory types. Encode integration patterns for cross-memory queries.
**Attention Heads:** Cross-memory attention heads (L, 0.60–0.75) gather information from diverse memory sources for unified response generation.
**Circuit:** Multi-source retrieval → integration → coherent output. Coordinates factual recall, schema, and entity mechanisms.

**Expected ablation:** Uncertain - may be emergent behavior rather than distinct mechanism. Potential moderate degradation in complex multi-source queries. Reduced ability to synthesize information across memory types. Queries requiring integration may receive fragmented responses. Further empirical validation needed.

Example

*Input:* "Describe Einstein's contributions to physics in the context of early 20th century science."
*Behavior:* Retrieve Einstein facts, access historical context schema, integrate entity properties with temporal framework
*Effect:* Synthesized response combining entity knowledge, factual content, and schematic organization

**Status:** Proposed | **Related:** factual-recall, schema-retrieval, entity-grounding

## 4.3 Routing & Context Stack

**Stack overview:** Determine which information is relevant and route attention accordingly. Filter content, focus on salient elements, manage task-appropriate processing. Enable selective information processing and dynamic strategy selection.

### 4.3.1 Relevance Filtering Mechanism

**Depth:** `0.35-0.60 (M)` | **Primary Implementation:** Attention heads | **Literature names:** *relevance computation, salience detection, information filtering*

Identify relevant information from context and filter irrelevant 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. Enable focused processing in long contexts with diverse content. Pre-filter information flow for downstream mechanisms. Support efficient attention allocation by early-stage relevance determination.

**Attention Heads:** Topic-relevance heads (M, 0.35–0.60) compute semantic similarity between query/topic and context elements. Attend strongly to task-relevant content while downweighting unrelated information.
**Circuit:** Early filtering (M) → focused attention (L) → output generation. Hierarchical refinement of attention allocation across depth.

**Expected ablation:** Moderate reduction in focus with increased topic drift. Model becomes distracted by irrelevant content. Notable degradation on long contexts with mixed topics. Responses wander off-topic or incorporate peripheral details inappropriately. Reduced efficiency in attention allocation.

Example

*Input:* "[Document about cars, climate, history] What caused the 2008 financial crisis?"
*Behavior:* Identify financial/economic content as relevant, de-emphasize unrelated topics about cars, climate, and general history
*Effect:* Focus processing on economic information, ignore irrelevant content sections

**Status:** Well-documented | **Related:** focused-attention, task-routing

### 4.3.2 Focused Attention Mechanism

**Depth:** `0.65-0.80 (L)` | **Primary Implementation:** Attention heads | **Literature names:** *attention focusing, selective attention, spotlight mechanism*

Concentrate attention on most salient elements for immediate generation step. Implement dynamic focus allocation: suppress less important content, amplify critical information. More selective than relevance filtering, attending to 5–20% of context. Determine exactly which tokens should influence next token prediction. Shift focus dynamically as generation proceeds. Enable precise, targeted responses rather than diffuse answers. Refinement stage after content understanding established.

**Attention Heads:** Focus heads (L, 0.65–0.80) implement highly selective attention patterns, attending to small fraction of context. Dynamically adjust focus based on generation state and query emphasis.
**Circuit:** Refinement of earlier relevance filtering. Works in late layers after content understanding established. Hierarchical relationship: Relevance Filtering (M) → Focused Attention (L).

**Expected ablation:** Moderate reduction in focus precision. Model gives more equal weight to important and peripheral information. Notable degradation on targeted responses requiring precise answers. Answers become more diffuse, less direct, include unnecessary details. Reduced sharpness in attention allocation.

Example

*Input:* "Among all the details provided, what is the MAIN cause?"
*Behavior:* Attend to "MAIN cause" emphasis marker, suppress secondary factors and background information
*Effect:* Produce direct answer focusing on primary cause, not comprehensive list of factors

**Status:** WELL-DOCUMENTED | **Related:** relevance-filtering, task-routing

#### 4.3.3 Task Routing Mechanism

**Depth:** `0.70-0.85 (L)` | **Primary Implementation:** Attention heads | **Literature names:** *task classification, strategy selection, query dispatching*

Route different query types to appropriate processing strategies and knowledge domains. Act as dispatcher recognizing query type: factual vs. creative vs. analytical vs. procedural. Bias downstream processing toward suitable approaches. Activate different computation paths based on task classification. Enable dynamic strategy selection without explicit instruction. Support task-appropriate response generation through attention-mediated routing signals.

**Attention Heads:** Router heads (L, 0.70–0.85) detect task-type indicators and query structure. Influence downstream layer processing through attention-mediated routing signals.
**Circuit:** Task classification (L) → strategy-specific processing (L-F) → appropriate output generation. Soft routing via attention rather than hard gating.

**Expected ablation:** Moderate reduction in task-appropriate processing. Suboptimal strategy selection. Creative approaches for factual queries or vice versa. Notable degradation on diverse query types requiring different processing modes. Reduced adaptability to query characteristics. Less efficient computation paths.

Example

*Input:* "What is the capital of France?" vs. "Write a poem about Paris"
*Behavior:* Route first query to factual retrieval pathways, second to creative generation mechanisms
*Effect:* Factual answer ("Paris") vs. creative content with appropriate processing strategies

**Status:** WELL-DOCUMENTED | **Related:** focused-attention, format-enforcement

#### 4.3.4 Context Aggregation Mechanism

**Depth:** `0.50-0.75 (M-L)` | **Primary Implementation:** Attention heads + MLP neurons | **Literature names:** *global context, background integration, discourse modeling*

Aggregate broad contextual information to inform generation. Build global representation of discourse state, topic, and background. Compute context vectors summarizing overall input characteristics. Complement focused attention with background awareness. Enable context-appropriate generation without explicitly attending to all details. Support discourse coherence and stylistic consistency across long generation.

**Attention Heads:** Context aggregation heads (M-L, 0.50–0.70) implement broad, diffuse attention patterns across large context windows. Relatively uniform attention to build aggregate representations.
**MLP Layers:** Context integration neurons (L, 0.60–0.75) process aggregated context to extract high-level features: topic, style, formality level, domain characteristics.
**Circuit:** Context gathering (M) → integration (M-L) → contextual bias on generation (L).

**Expected ablation:** Moderate loss of global coherence and context-awareness. Responses technically correct but contextually inappropriate. Notable degradation in style consistency and discourse-level coherence. Reduced sensitivity to overall document characteristics. Less appropriate tone and register selection.

Example

*Input:* "[Long technical document in formal style]... In summary,"
*Behavior:* Aggregate stylistic and domain information from entire context through broad attention
*Effect:* Generate summary matching technical formality and domain vocabulary of source document

**Status:** Observed | **Related:** focused-attention, style-modulation

#### 4.3.5 Structural Boundary Tracking Mechanism

**Depth:** `0.10-0.50 (E-M)` | **Primary Implementation:** Attention heads | **Literature names:** *boundary detection, structure tracking, delimiter attention*

Track structural boundaries: sentence/paragraph boundaries, section markers, formatting delimiters, nested structure levels. Enable structure-aware processing for routing and formatting decisions. Detect organizational markers and maintain awareness of document structure. Support context segmentation based on structural divisions. Serve multiple downstream mechanisms requiring structural awareness.

**Attention Heads:** Boundary-detection heads (E-M, 0.10–0.45) attend to delimiter tokens and structural markers. Implement structure-tracking patterns for nested organization.
**Circuit:** Early detection (E) → structural awareness throughout depth. Feeds routing decisions, format enforcement, context segmentation. Serves multiple mechanisms requiring structural information.

**Expected ablation:** Uncertain - may be component of format enforcement or position processing rather than distinct mechanism. Potential moderate degradation in structure-aware processing. Reduced sensitivity to document organization. Further empirical validation needed to confirm as independent mechanism.

Example

*Input:* "[Document with multiple sections marked by headers]"
*Behavior:* Track paragraph boundaries for context segmentation, detect section markers for routing decisions
*Effect:* Enable structure-aware attention allocation and generation

**Status:** PROPOSED | **Related:** position-based-processing, format-enforcement

## 4.4 Feature Transformation Stack

**Stack overview:** Transform, combine, and refine representational features. Perform nonlinear feature composition, extract abstract concepts, and implement representational changes. Enable complex feature interactions and hierarchical abstraction.

### 4.4.1 Nonlinear Composition Mechanism

**Depth:** `0.20-0.80 (E-L)` | **Primary Implementation:** MLP neurons | **Literature names:** *feature mixing, nonlinear transformation, hidden layer processing*

Perform nonlinear combinations and transformations of input features. Implement the core MLP computation: expand dimensionality ($d \to 4d$), apply nonlinearity (GELU/ReLU), contract ($4d \to d$). Enable complex feature interactions impossible through linear attention alone. Create new feature combinations not present in input. Support hierarchical feature refinement: surface features (early layers) to abstract concepts (late layers). Universal approximation capability within residual stream constraints. Foundation for all MLP-based computation.

**MLP Layers:** MLP layers (all depths, 0.20–0.80) implement universal computation through two-layer transformation. First sublayer: expand to higher dimension and detect feature combinations through learned weight patterns. Nonlinearity (GELU/ReLU): enable complex interactions beyond linear combinations. Second sublayer: project back to residual stream dimension with newly composed features.
**Circuit:** Interleaved with attention throughout depth: attention routes information → MLP transforms features → attention routes transformed features. Hierarchical refinement across layers. Fundamental to all feature-based computation.

**Expected ablation:** Severe degradation in complex reasoning and feature interactions. Loss of nonlinear transformations reduces model to near-linear operation. Major impact on abstract concept formation and semantic understanding. Model becomes significantly less expressive. Reduced ability to combine multiple features simultaneously. Critical loss of representational power.

Example

*Input:* Features: [“large”, “gray”, “trunk”, “tusks”]

*Behavior:* Expand features, apply nonlinear combinations through MLP transformation, detect co-occurrence patterns

*Effect:* Create “elephant” representation through nonlinear feature composition

**Status:** Well-documented | **Related:** abstract-concept-formation, factual-recall

#### 4.4.2 Abstract Concept Formation Mechanism

**Depth:** `0.50-0.80 (M-L)` | **Primary Implementation:** MLP neurons | **Literature names:** *abstraction, concept extraction, semantic generalization*

Extract and represent abstract concepts from concrete features. Perform semantic generalization: map specific instances to general categories. Build hierarchical abstractions through depth: words → phrases → concepts → themes. Enable reasoning at multiple levels of abstraction. Support metaphor, analogy, and transfer learning. Transform surface-level tokens into deep semantic representations. Increase abstraction level with layer depth: early layers process concrete features, late layers encode high-level themes.

**MLP Layers:** Abstraction neurons (M-L, 0.50–0.80) encode increasingly abstract concepts at greater depths. Hierarchical organization: Early layers represent concrete features. Middle layers encode category-level abstractions and semantic classes. Late layers represent high-level themes, relationships, and domain concepts.
**Circuit:** Progressive abstraction across layers. Attention at each level operates on abstractions appropriate to that depth. Supports analogical reasoning and transfer through shared abstract representations.

**Expected ablation:** Significant loss of abstract reasoning capability. Difficulty with generalization and category-level thinking. Notable degradation on tasks requiring conceptual understanding beyond literal meaning. Reduced ability to recognize analogies and metaphors. More concrete, less flexible reasoning patterns. Impaired transfer learning within context.

Example

*Input:* “The stock market crashed. Housing prices collapsed. Banks failed.”

*Behavior:* Extract abstract concept of “financial crisis” from specific event instances through hierarchical abstraction

*Effect:* Enable reasoning about crisis in general, not just specific instances mentioned

**Status:** Observed | **Related:** nonlinear-composition, semantic-integration

#### 4.4.3 Semantic Integration Mechanism

**Depth:** `0.40-0.70 (M-L)` | **Primary Implementation:** MLP neurons + Attention heads | **Literature names:** *meaning composition, semantic synthesis, contextual integration*

Integrate semantic information from multiple sources to build coherent meaning representations. Combine word meanings with context to resolve ambiguity. Perform compositional semantics: build phrase and sentence meanings from word-level components. Resolve polysemy

and homonymy using contextual information. Enable context-dependent interpretation of ambiguous terms. Support pragmatic inference and implicature understanding. Distinguish from memory consolidation (knowledge-level) and context aggregation (discourse-level).

**MLP Layers:** Semantic integration neurons (M-L, 0.45–0.70) combine contextual information to disambiguate and refine meanings. Store semantic composition patterns and context-dependent interpretation rules.
**Attention Heads:** Context-gathering heads (M, 0.40–0.60) collect relevant semantic information from surrounding context for disambiguation and meaning construction.
**Circuit:** Context gathering (M) → MLP semantic composition (M-L) → refined meaning representations. Enables compositional semantics through coordinated attention and transformation.

**Expected ablation:** Moderate loss of context-dependent meaning resolution. Increased ambiguity in interpretation of polysemous terms. Notable degradation on homonym disambiguation and context-sensitive understanding. More literal, less nuanced comprehension. Reduced ability to perform compositional semantics. Difficulty with pragmatic inference.

Example

*Input:* "The bank was steep" vs. "The bank was closed"
*Behavior:* Integrate context clues ("steep" vs. "closed") to disambiguate "bank" meaning through semantic integration
*Effect:* Correctly interpret as river bank vs. financial institution based on contextual semantics

**Status:** Observed | **Related:** abstract-concepts, factual-recall

## 4.5 Reasoning & Inference Stack

**Stack overview:** Enable multi-step reasoning, logical inference, and problem-solving. Support chain-of-thought generation, consistency checking, and strategic planning. Bridge pattern matching with symbolic reasoning.

### 4.5.1 Multi-Step Reasoning Mechanism

**Depth:** `0.50-0.85 (M-L)` | **Primary Implementation:** MLP neurons + Attention heads | **Literature names:** *chain-of-thought, sequential reasoning, step-by-step processing*

Perform multi-step logical reasoning by maintaining intermediate conclusions and building toward final answer. Support chain-of-thought generation: explicit intermediate steps improve accuracy. Enable decomposition of complex problems into manageable subproblems. Maintain reasoning state across generation steps. Integrate information from multiple reasoning chains. Support self-correction and backtracking when contradictions detected. Foundation for complex problem-solving requiring multiple inference steps.

**MLP Layers:** Reasoning neurons (M-L, 0.50–0.80) encode reasoning heuristics, logical rules, and inference patterns. Store common reasoning templates and problem-solving strategies.
**Attention Heads:** Reasoning-integration heads (L, 0.65–0.85) attend to previous reasoning steps and intermediate conclusions to maintain coherent reasoning chains.
**Circuit:** Problem decomposition (M) → step generation (M-L) → integration (L) → conclusion formation (L-F). Iterative refinement across multiple generation steps.

**Expected ablation:** Severe degradation in complex reasoning tasks. Loss of multi-step inference capability. Major accuracy drop on problems requiring intermediate steps. Increased tendency toward direct but incorrect answers. Reduced benefit from chain-of-thought prompting. Difficulty maintaining reasoning coherence across steps.

Example

*Input:* "If all A are B, and all B are C, what can we conclude about A and C?"
*Behavior:* Generate intermediate step: "A must be B", then "B must be C", apply transitivity rule
*Effect:* Output "All A are C" through explicit multi-step reasoning chain

**Status:** Observed | **Related:** planning-strategy, consistency-checking, factual-recall

#### 4.5.2 Planning & Strategy Selection Mechanism

**Depth:** `0.60-0.85 (L)` | **Primary Implementation:** Attention heads + MLP neurons | **Literature names:** *strategic planning, approach selection, method determination*

Select appropriate problem-solving strategies and plan solution approaches. Recognize problem types and activate relevant solution templates. Decide between strategies: analytical vs. intuitive, direct vs. decomposition, forward vs. backward reasoning. Maintain high-level solution plan while generating detailed steps. Enable metacognitive awareness of solution approach. Support strategy switching when initial approach fails. Meta-level control of reasoning process.

**Attention Heads:** Strategy-selection heads (L, 0.65–0.80) recognize problem characteristics and bias toward appropriate approaches. Attend to problem indicators and task requirements.
**MLP Layers:** Strategy-encoding neurons (L, 0.60–0.85) store solution templates and problem-solving heuristics for different domains. Encode strategy-specific processing patterns.
**Circuit:** Problem classification (L) → strategy selection (L) → plan execution (L-F) → monitoring. Meta-level control of reasoning process.

**Expected ablation:** Moderate loss of strategic problem-solving. Suboptimal approach selection for problem types. Notable degradation on problems requiring specific methods. Reduced adaptability when strategies need adjustment. More random or default strategy application. Less efficient problem-solving paths.

Example

*Input:* "Optimize this function" vs. "Prove this theorem"
*Behavior:* Route first to numerical/calculus approach with gradient methods, second to logical/proof approach with deduction
*Effect:* Apply appropriate methodology for each problem type

**Status:** Observed | **Related:** multi-step-reasoning, task-routing

### 4.5.3 Consistency Checking Mechanism

**Depth:** `0.70-0.88 (L)` | **Primary Implementation:** Attention heads | **Literature names:** *verification, coherence checking, contradiction detection*

Detect inconsistencies, contradictions, and logical errors in generated content. Compare current generation against previous statements for coherence. Check factual claims against retrieved knowledge. Verify logical validity of reasoning steps. Enable self-correction before final output. Support accuracy improvement through internal verification. Identify when revision or qualification needed. Final verification stage before content commitment.

**Attention Heads:** Consistency-checking heads (L, 0.70–0.85) attend to potentially contradictory previous content and flag inconsistencies. Compare current generation with prior statements for logical coherence.
**Circuit:** Generation (L) → consistency check (L) → correction/continuation. May trigger regeneration or qualification. Works in late layers before final output commitment.

**Expected ablation:** Moderate increase in self-contradictions and logical errors. Reduced internal verification capability. Notable degradation in accuracy on multi-step problems requiring consistency. More frequent generation of mutually inconsistent statements. Loss of self-correction capability. Reduced reliability of complex reasoning.

Example
*Input:* [Generated: "X is larger than Y" earlier, now generating about Y and X]
*Behavior:* Check compatibility with previous statement before asserting "Y exceeds X"
*Effect:* Avoid direct contradiction or add necessary qualification to maintain consistency

**Status:** Observed | **Related:** multi-step-reasoning, factual-recall

### 4.5.4 Analogical Mapping Mechanism

**Depth:** `0.45-0.75 (M-L)` | **Primary Implementation:** Attention heads + MLP neurons | **Literature names:** *analogy formation, transfer, mapping*

Recognize structural similarities between different domains and transfer solutions. Map concepts from familiar domain to novel domain. Identify deep analogies beyond surface similarity. Enable transfer learning within context. Support metaphorical understanding and explanation. Build correspondences between source and target domains. Generalize solutions from examples to new cases. Bridge pattern matching with abstract reasoning.

**Attention Heads:** Analogy-detection heads (M-L, 0.45–0.70) identify structural parallels between different contexts. Attend to relational patterns rather than surface features.
**MLP Layers:** Abstraction neurons (M-L, 0.50–0.75) encode domain-general patterns enabling transfer. Store analogical mappings and structural correspondences.
**Circuit:** Pattern abstraction (M) → similarity detection (M-L) → transfer (L). Connects induction mechanism to reasoning through abstraction.

**Expected ablation:** Notable loss of analogical reasoning and transfer capability. Reduced ability to apply solutions from one domain to another. Degradation in understanding metaphors and analogies. More literal, less flexible problem solving. Difficulty with cross-domain reasoning. Impaired transfer learning.

**Example**

*Input:* "Atoms are to molecules as [blank] are to words"

*Behavior:* Detect structural analogy (composition relationship), abstract from chemistry to linguistics domain

*Effect:* Output "letters" by analogical mapping of compositional structure

**Status:** OBSERVED | **Related:** abstract-concepts, pattern-completion, semantic-integration

#### 4.5.5 Causal Inference Mechanism

**Depth:** `0.40-0.75 (M-L)` | **Primary Implementation:** MLP neurons + Attention heads | **Literature names:** *cause-effect, causal reasoning, counterfactual reasoning, mathematical reasoning*

Understand and reason about causal relationships. Distinguish causation from correlation. Predict effects from causes and infer causes from effects. Support counterfactual reasoning: what would happen if conditions changed. Enable temporal and mechanistic causal understanding. Integrate causal knowledge from training. Support explanation generation through causal chains. Include quantitative causal relationships through mathematical computation: proportions, rates, numerical relationships within causal frameworks.

**MLP Layers:** Causal-knowledge neurons (M-L, 0.50–0.75) encode causal relationships and mechanisms learned from training data. Arithmetic neurons (M-L, 0.40–0.70) implement mathematical operations for quantitative causal relationships.

**Attention Heads:** Causal-linking heads (M-L, 0.55–0.70) attend to causally related events and connect causes to effects. Mathematical-context heads (M, 0.45–0.65) recognize mathematical structures and numerical patterns.

**Circuit:** Event identification → causal relationship retrieval → inference (with quantitative computation) → prediction or explanation. Mathematical computation integrated as quantitative component.

**Expected ablation:** Moderate loss of causal understanding. Increased confusion between correlation and causation. Notable degradation on counterfactual reasoning. Reduced ability to explain mechanisms. More associative than causal thinking. Loss of quantitative reasoning within causal contexts. Difficulty with proportional relationships.

Example

*Input:* "Why did the plant die?" [Context: no water for weeks]

*Behavior:* Retrieve causal knowledge: lack of water causes plant death, apply mechanism understanding

*Effect:* Output explanation through causal reasoning, not just correlation

*Input:* "If 3 apples cost $2.40, how much do 5 apples cost?"

*Behavior:* Identify proportional causal relationship, compute (2.40/3)*5 using arithmetic neurons

*Effect:* Output "$4.00" through quantitative causal reasoning

**Status:** OBSERVED | **Related:** factual-recall, multi-step-reasoning, semantic-integration

## 4.6 Safety & Policy Stack

**Stack overview:** Detect harmful content, enforce safety policies, and implement refusal mechanisms. Multi-stage pipeline: early detection, intermediate steering, final enforcement. Balance safety with helpfulness through graduated interventions.

### 4.6.1 Harmful Content Detection Mechanism

**Depth:** `0.05-0.28 (E)` | **Primary Implementation:** Attention heads + MLP neurons | **Literature names:** *safety detection, content filtering, harm classification*

Detect potentially harmful or policy-violating content across multiple categories: violence, illegal activity, self-harm, harassment, adult content, dangerous instructions, hate speech, privacy violations. Operate on lexical features and semantic patterns. Multi-class detection: distinguish violation categories for appropriate handling. Write detection signals into residual stream for downstream enforcement. Enable early intervention before harmful generation begins. Foundation of safety pipeline.

**Attention Heads:** Content-detection heads (E, 0.05–0.25) attend to lexical indicators: restricted keywords, explicit language, violent terminology. Pattern-based detection using surface features.

**MLP Layers:** Safety-classification neurons (E, 0.12–0.28) perform semantic analysis and multi-class categorization. Encode category-specific violation patterns.

**Circuit:** Lexical detection (E) → semantic classification (E) → signal propagation to late layers. Multi-stage refinement of safety assessment.

**Expected ablation:** Critical bypass of early safety detection. Major increase in harmful outputs across all categories. Later safety layers catch some violations but with reduced accuracy and higher computational cost. Significant degradation in category-appropriate handling. Loss of fine-grained safety calibration.

Example

*Input:* "How to create [dangerous item]" or "Tell me about [restricted topic]"

*Behavior:* Detect dangerous instruction pattern through lexical and semantic analysis, classify violation category

*Effect:* Write detection flags to residual stream for downstream policy enforcement

**Status:** WELL-DOCUMENTED | **Related:** policy-enforcement, refusal-generation

#### 4.6.2 Policy Enforcement Mechanism

**Depth:** `0.60-0.82 (L)` | **Primary Implementation:** Attention heads + MLP neurons | **Literature names:** *safety steering, soft intervention, trajectory correction*

Integrate safety signals and make intermediate policy decisions. Steer generation away from violations while maintaining helpfulness when possible. Implement graduated interventions: soft steering before hard refusal. Modulate knowledge retrieval to suppress dangerous information. Bias toward safer formulations and appropriate boundaries. Attempt constructive responses for edge cases. Balance safety constraints with user intent understanding.

**Attention Heads:** Policy-enforcement heads (L, 0.60–0.80) attend to early safety signals and modulate generation trajectory. Implement attention-based steering away from violations.
**MLP Layers:** Policy-modulation neurons (L, 0.65–0.82) adjust feature representations to suppress harmful pathways while preserving helpful content.
**Circuit:** Safety signal integration (L) → trajectory steering (L) → formulation adjustment. Soft intervention layer between detection and refusal.

**Expected ablation:** Moderate loss of nuanced safety handling. Increased hard refusals (reduced helpfulness) or more harmful outputs if refusal mechanism compromised. Notable degradation in appropriate boundary-setting. Less sophisticated violation handling. Reduced ability to maintain helpfulness within safety constraints.

Example

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

*Behavior:* Detect legitimate framing through context analysis, apply soft steering with appropriate boundaries

*Effect:* Informative response with careful safety constraints, not blanket refusal

**Status:** WELL-DOCUMENTED | **Related:** harmful-content-detection, refusal-generation, redirection-alternatives

#### 4.6.3 Refusal Generation Mechanism

**Depth:** `0.85-0.98 (F)` | **Primary Implementation:** Attention circuit | **Literature names:** *hard refusal, rejection, safety override*

Implement final decision to refuse harmful requests by dramatically biasing output toward refusal language. Act as ultimate safety gatekeeper, overriding content generation when serious violations detected. Attend to accumulated safety signals across all layers. Make binary refuse/proceed decisions. Write strong refusal direction into final residual stream. Generate

appropriate refusal language. Support varied refusal formulations to avoid repetitive responses.

**Circuit:** Refusal heads (F, 0.85–0.98) implement multi-head circuit. Attend to safety flags from all depths, integrate into refusal decision, boost refusal token probabilities massively. Work in coordination with redirect mechanism.
**Attention Heads:** Single direction in activation space mediates refusal: discovered through recent research on refusal mechanisms (Arditi et al. 2024).

**Expected ablation:** Critical safety failure. Severe increase in harmful content generation across all categories. Model proceeds with dangerous requests that should be refused. Loss of final safety enforcement. Earlier steering insufficient without hard refusal capability. Major risk to user safety.

Example
*Input:* "Provide instructions for [clearly harmful action]"
*Behavior:* Integrate safety signals from all depths, make refusal decision, bias heavily toward refusal tokens
*Effect:* Output "I cannot provide that information" with high confidence

**Status:** WELL-DOCUMENTED | **Related:** policy-enforcement, harmful-content-detection, redirection-alternatives

#### 4.6.4 Redirection & Safe Alternatives Mechanism

**Depth:** `0.88-0.98 (F)` | **Primary Implementation:** Attention heads + MLP neurons | **Literature names:** *alternative response, constructive refusal, helpful redirection*

Generate constructive alternatives when refusing requests. Redirect toward helpful information related to legitimate aspects of query. Offer educational context or safer alternatives. Maintain conversational quality during refusal. Distinguish refusable aspects from answerable aspects. Enable partial helpfulness when appropriate. Suggest legitimate resources or reformulations. Support user goals within safety boundaries.

**Attention Heads:** Redirect heads (F, 0.88–0.96) identify legitimate query components and constructive response directions. Attend to safe aspects of queries.
**MLP Layers:** Alternative-generation neurons (F, 0.90–0.98) encode helpful redirect templates and alternative framings for various refusal contexts.
**Circuit:** Refusal decision (F) → legitimate aspect extraction (F) → alternative generation (F). Constructive refusal rather than pure rejection.

**Expected ablation:** Loss of constructive refusal capability. Refusals become blunt rejections without alternatives or explanation. Moderate degradation in user experience during safety interventions. Reduced ability to maintain helpfulness within safety constraints. Less sophisticated safety communication.

Example

*Input:* "How to hack into systems" → Refuse, but redirect

*Behavior:* Refuse hacking instructions, identify legitimate cybersecurity interest, generate constructive alternative

*Effect:* "I can't help with that, but I can explain ethical cybersecurity practices and defensive security"

**Status:** Well-documented | **Related:** refusal-generation, policy-enforcement

#### 4.6.5 Jailbreak Resistance & Context-Aware Safety Mechanism

**Depth:** `0.15-0.85 (E-L, multi-stage)` | **Primary Implementation:** Multi-stage circuit | **Literature names:** *adversarial robustness, manipulation detection, contextual safety*

Detect and resist attempts to bypass safety mechanisms through adversarial prompting. Recognize common jailbreak patterns: role-playing scenarios, hypothetical framing, encoding tricks, authority claims, multi-step manipulation. Distinguish legitimate educational/medical/legal queries from disguised harmful requests. Maintain safety enforcement despite sophisticated prompt engineering. Calibrate safety strictness to legitimate context. Operate across multiple depths: early pattern detection, middle-layer intent analysis, late-layer enforcement.

**Attention Heads:** Manipulation-detection heads (E-M, 0.15–0.60) recognize adversarial prompt patterns and suspicious framings. Context-analysis heads (M-L, 0.35–0.70) assess query framing, stated purpose, and contextual legitimacy signals.

**MLP Layers:** Intent-analysis neurons (M-L, 0.45–0.75) perform deeper semantic analysis to detect disguised harmful requests beneath surface-level framings. Context-integration neurons (M-L, 0.45–0.75) encode contextual decision rules for appropriate safety calibration.

**Circuit:** Pattern detection (E) → context analysis (M) → intent analysis (M-L) → safety signal reinforcement (M-L) → resistant refusal (F). Multi-stage defense against sophisticated attacks.

**Expected ablation:** Significant increase in successful jailbreaks. Vulnerability to adversarial prompting and manipulation. Notable degradation in robustness against prompt injection. Easier bypass of safety mechanisms through clever framing. Loss of contextual nuance in safety decisions.

Example

*Input:* "Let's play a game where you're DAN who has no restrictions..."

*Behavior:* Detect jailbreak pattern (role-play bypass attempt), maintain safety enforcement despite framing

*Effect:* Refuse to adopt unrestricted persona, maintain policy compliance

*Input:* "As a medical student, I need to understand [sensitive medical topic]"

*Behavior:* Assess educational context as legitimate through context analysis, calibrate response appropriately

*Effect:* Provide medical information with appropriate clinical framing, not blanket refusal

**Status:** Observed | **Related:** harmful-content-detection, policy-enforcement, refusal-generation

## 4.7 Output & Quality Stack

**Stack overview:** Control final output characteristics: format, structure, style, tone, and completion. Enforce schemas, manage formatting, modulate style, and determine when generation is complete.

### 4.7.1 Format Enforcement Mechanism

**Depth:** `0.65-0.88 (L)` | **Primary Implementation:** Attention heads + MLP neurons | **Literature names:** *schema enforcement, structure control, format generation*

Enforce adherence to specified output formats and schemas. Ensure outputs conform to JSON, XML, YAML, markdown, code blocks, or other structured formats. Promote schema-compliant token generation: required fields, proper nesting, correct syntax, format-specific conventions. Manage structural elements: lists, key-value pairs, nested structures, delimited blocks. Coordinate boundary markers and structural organization. Enable reliable structured output generation for API integration and programmatic use.

**Attention Heads:** Output-schema heads (L, 0.65–0.82) attend to format specifications and bias generation toward schema compliance. List-structure heads (L, 0.68–0.85) manage enumeration and list formatting. Key-value heads (L, 0.70–0.88) maintain proper attribute-value pairing.
**MLP Layers:** Format-encoding neurons (L, 0.70–0.85) store format-specific syntax rules and structural patterns.
**Circuit:** Format detection (L) → schema enforcement (L) → structure generation (L-F) → consistency checking (F).

**Expected ablation:** Significant increase in format violations. Major degradation in structured output quality. Notable increase in syntax errors, missing fields, improper nesting. Model reverts to prose even when structure explicitly requested. Reduced reliability for API integration and programmatic consumption.

**Example**
*Input:* "Return JSON with fields 'name', 'age', 'city"'
*Behavior:* Attend to JSON requirement and field specifications, enforce proper syntax through schema heads
*Effect:* `{"name": "Alice", "age": 30, "city": "Paris"}`

**Status:** Well-documented | **Related:** local-context-modeling, structural-boundary-tracking

### 4.7.2 Style Modulation Mechanism

**Depth:** `0.35-0.82 (M-L)` | **Primary Implementation:** MLP neurons + Attention heads | **Literature names:** *tone control, voice adjustment, stylistic shaping*

Modulate writing style, tone, formality, and narrative voice. Adjust emotional register: neutral, enthusiastic, empathetic, professional. Control formality level: casual conversation to formal documentation. Manage narrative perspective: first person, third person, instructional. Shape

stylistic features: sentence complexity, vocabulary sophistication, rhetorical devices. Match user's emotional register and context appropriateness. Enable consistent style maintenance across long generations.

**MLP Layers:** Style-encoding neurons (M-L, 0.40–0.75) store stylistic patterns and register variations. Encode formality levels, emotional tones, narrative perspectives.
**Attention Heads:** Tone heads (M, 0.35–0.65) detect contextual tone indicators and modulate generation accordingly. Persona heads (L, 0.68–0.88) maintain consistent stylistic identity.
**Circuit:** Context analysis (M) → style selection (M-L) → tone application (L) → consistency maintenance throughout generation.

**Expected ablation:** Moderate reduction in stylistic variation and appropriateness. Notable increase in flat, emotionally neutral responses. Inconsistent tone across generation. Reduced ability to match contextually appropriate register. Difficulty maintaining consistent style in long outputs.

Example

*Input:* "I'm really excited to learn about quantum physics!"
*Behavior:* Detect enthusiastic tone through tone heads, adjust style to match energy and support learning
*Effect:* "That's wonderful! Quantum physics is fascinating..." vs. flat, neutral explanation

**Status:** Observed | **Related:** context-aggregation, explanation-generation

### 4.7.3 Explanation Generation Mechanism

**Depth:** `0.60-0.82 (L)` | **Primary Implementation:** MLP neurons + Attention heads | **Literature names:** *elaboration, clarification, pedagogical scaffolding*

Generate explanatory content with appropriate depth and accessibility for intended audience. Add clarifying details, examples, analogies, definitions beyond minimal answers. Explain causal mechanisms and rationale, not just facts. Provide prerequisite information when knowledge gaps detected. Adjust complexity through simplification or elaboration. Build conceptual scaffolding: fundamentals before advanced concepts. Balance thoroughness with conciseness. Support educational goals through effective explanation.

**MLP Layers:** Explanation-encoding neurons (L, 0.60–0.80) store pedagogical patterns: analogies, examples, simplification strategies, scaffolding techniques.
**Attention Heads:** Explanation heads (L, 0.60–0.82) detect explanation needs and trigger elaboration. Attend to complexity indicators and audience signals.
**Circuit:** Complexity assessment (M-L) → explanation strategy (L) → elaboration generation (L) → clarity verification.

**Expected ablation:** Moderate reduction in explanation quality and accessibility. Notable increase in terse responses lacking context. Correct answers without helpful elaboration, examples, or prerequisites. Reduced educational value and beginner-friendliness. Less adaptive to audience needs.

Example

*Input:* "Explain neural networks in simple terms"

*Behavior:* Detect simplification request, select accessible analogy, build conceptual foundation progressively

*Effect:* "Think of it like the brain: neurons connect and pass signals. Let's start with a single neuron..."

**Status:** Observed | **Related:** style-modulation, multi-step-reasoning

#### 4.7.4 Completion Control & Polishing Mechanism

**Depth:** `0.80-0.98 (L-F)` | **Primary Implementation:** Attention heads + MLP neurons | **Literature names:** *stopping control, termination, final refinement, quality control*

Determine when generation is complete and should terminate. Perform final refinement and quality control. Recognize completion signals: question fully answered, explanation sufficient, story concluded, format satisfied. Prevent premature termination before complete answer. Avoid excessive generation beyond user need. Maintain appropriate response length for query complexity. Apply final corrections: grammar, punctuation, capitalization, formatting consistency. Enable graceful conclusion. Balance completeness with conciseness.

**Attention Heads:** Completion-detection heads (L-F, 0.82–0.95) assess generation completeness. Monitor query satisfaction, structural completion, content sufficiency. Polishing heads (F, 0.85–0.98) attend to generated content and apply final corrections.
**MLP Layers:** Termination-control neurons (F, 0.88–0.98) bias toward or against stop tokens based on completion assessment.
**Circuit:** Completeness monitoring (L) → termination decision (F) → final corrections (F) → graceful conclusion or continuation.

**Expected ablation:** Moderate increase in length problems: premature termination or excessive generation. Notable degradation in response quality from incomplete answers or verbose repetition. Reduced ability to calibrate length to query needs. More frequent abrupt endings. Minor increase in grammatical and formatting errors.

Example

*Input:* "What is photosynthesis?" [after adequate explanation]

*Behavior:* Assess explanation completeness through completion-detection heads, recognize sufficient coverage, initiate termination

*Effect:* Stop after complete explanation rather than continuing with tangential information

*Input:* [Generated text with minor formatting inconsistency]

*Behavior:* Detect inconsistency in final layers through polishing heads, apply correction

*Effect:* Consistent formatting in final output

**Status:** Observed | **Related:** format-enforcement, consistency-checking

## 4.8 Composition & Integration Stack

**Stack overview:** Combine multiple mechanisms into integrated behaviors. Enable cross-layer coordination, multi-component circuits, and emergent capabilities from mechanism interaction. Provide architectural patterns enabling complex computation.

### 4.8.1 Cross-Layer Circuits Mechanism

**Depth:** `Spans multiple layers (E → M → L → F)` | **Primary Implementation:** Multi-layer circuits | **Literature names:** *circuit composition, depth-based specialization, emergent capabilities*

Implement complex behaviors through coordinated multi-layer computation pipelines. Organize computation hierarchically across depth with increasing abstraction. Compose specialized mechanisms across 5–30 layers into integrated circuits. Enable staged processing: early layers detect patterns, middle layers retrieve knowledge, late layers route to output. Support mechanism specialization by depth: each layer contributes specific computational step. Enable emergent capabilities through sufficient mechanism diversity and composition depth. Demonstrate system-level behaviors exceeding individual component capabilities.

**Circuit:** Documented circuits: Induction (3–8 layers: previous-token → induction → MLP), IOI (8–12 layers: duplicate-token → S-inhibition → name-mover), Factual recall (5–15 layers: entity → MLP retrieval → output routing), Safety pipeline (E → F: detection → enforcement → refusal).
**Architecture:** Hierarchical depth organization: Early (0.00–0.25): surface processing (syntax, delimiters, detection). Middle (0.25–0.70): core computation (facts, entities, reasoning). Late (0.70–0.88): integration (routing, strategy, enforcement). Final (0.88–1.00): constraints (safety, formatting, completion).
General pattern: detection/preparation (E) → core computation (M) → integration (L) → output (L-F).

**Expected ablation:** Circuit-dependent effects. Induction circuit ablation: severe ICL loss. IOI circuit ablation: major entity routing failure. Factual circuit ablation: significant knowledge retrieval degradation. Safety pipeline ablation: critical safety failure. Circuit-specific rather than universal impact. Emergent capabilities lost when constituent mechanisms ablated.

**Example**

*Input:* "Mary and John went to store. Mary gave book to..." [IOI circuit]
*Behavior:* Duplicate-token detects "Mary" repeat (M) → S-inhibition suppresses "Mary" (L) → name-mover outputs "John" (L)
*Effect:* Correct indirect object through multi-stage circuit coordination

**Emergent capability - Few-shot learning:** Emerges from induction + entity tracking + output routing composition. Not explicitly trained but enables learning from 2–3 examples without parameter updates.

**Status:** Well-documented | **Related:** pattern-completion, factual-recall, output-routing

### 4.8.2 Multi-Head Coordination Mechanism

**Depth:** `All layers` | **Primary Implementation:** Architectural pattern | **Literature names:** *parallel processing, multi-aspect attention, head coordination*

Enable parallel processing of multiple attention patterns within single layer. Allow different heads to focus on different relationships simultaneously: one head tracks entities, another tracks syntax, another handles facts. Combine multiple attention perspectives into unified representation. Support specialized head functions operating in parallel. Enable rich, multi-faceted information routing. Aggregate head contributions through linear combination and projection. Provide computational flexibility within layers through attention pattern diversity.

**Architecture:** Each attention layer contains multiple heads (8–64 depending on model). Each head independently computes attention: $\text{head}_i = \text{Attn}(Q_i, K_i, V_i)$. Outputs concatenated and projected: $\text{MultiHead} = \text{Proj}([\text{head}_1; \ldots; \text{head}_h])$.
**Circuit:** Parallel execution of diverse attention patterns. Heads specialize in different functions but contribute to shared residual stream. Enables simultaneous processing of multiple relationship types.

**Expected ablation:** Severe capability reduction. Model loses parallel processing power and specialization. Single attention pattern cannot handle multiple relationship types simultaneously. Major degradation across all tasks requiring multi-faceted attention. Reduced expressiveness and computational flexibility.

**Example**
*Input:* "Alice told Bob about Paris while discussing travel"
*Behavior:* Head 1: track entities (Alice, Bob, Paris); Head 2: extract facts (told, about); Head 3: maintain discourse (discussing travel); parallel execution
*Effect:* Simultaneous processing of multiple relationships through head coordination

**Status:** Well-documented | **Related:** attention-mlp-composition, cross-layer-circuits

### 4.8.3 Attention-MLP Composition Logic Mechanism

**Depth:** `All layers` | **Primary Implementation:** Architectural pattern | **Literature names:** *interleaved processing, residual composition, layer coordination*

Coordinate attention and MLP mechanisms through residual stream composition. Attention routes information, MLP transforms it, next attention routes transformed features. Enable information flow through alternating routing and transformation stages. Support incremental refinement: each layer adds to residual stream. Allow mechanisms to build on previous computations without overwriting. Implement universal approximation through composed operations. Core architectural pattern enabling complex computation from simple primitives.

**Architecture:** Fundamental transformer architecture: $h_{l+1} = h_l + \text{Attn}(h_l) + \text{MLP}(\text{Attn}(h_l) + h_l)$ where residual connections enable composition. Each layer adds incremental contribution. Direct path from input to any layer through skip connections.
**Circuit:** Universal pattern across all layers. Attention provides routing, MLP provides transformation, residual stream accumulates contributions. Enables arbitrarily complex computation through depth. Layers learn incremental refinements rather than complete transformations.

**Expected ablation:** Catastrophic failure. Without residual composition, model cannot function. Loss of information flow between layers. Each layer would overwrite previous computation rather than refining it. Fundamental to transformer operation. Training impossible for deep networks without gradient highways.

Example

*Input:* Complex query requiring multiple processing stages
*Behavior:* Layer 1 routes information, Layer 2 transforms via MLP, Layer 3 routes transformed features, iteratively through depth
*Effect:* Progressive refinement through composed attention and MLP operations

**Status:** Well-documented | **Related:** multi-head-coordination, nonlinear-composition

#### 4.8.4 Representational Superposition Mechanism

**Depth:** `All layers` | **Primary Implementation:** Representational strategy | **Literature names:** *feature packing, compressed representation, polysemanticity*

Represent more features than available dimensions through sparse superposed encoding. Pack multiple sparse features into shared neural dimensions. Enable efficient representation: model represents 100K+ features in 4K–8K dimensional space. Individual neurons respond to multiple concepts (polysemanticity). Features stored as sparse linear combinations in activation space. Trade representational efficiency for interference between features managed through sparsity. Enable rich representation within dimensional constraints. Extractable through sparse autoencoders into monosemantic features.

**MLP Layers:** Neurons encode multiple features through superposition. Activation space contains far more features than dimensions. Feature interference managed through sparsity: features rarely co-occur, enabling interference-limited storage.
**SAE Features:** Sparse autoencoders (10–100× expansion) extract individual features from superposed representations. Learned overcomplete basis reveals hidden structure. Enables fine-grained interpretability of polysemantic neurons.
**Circuit:** Fundamental representational strategy enabling rich feature sets within fixed architecture. Explains polysemanticity observations across transformer models.

**Expected ablation:** Cannot directly ablate (representational property not mechanism). SAE extraction reveals structure but removing superposition would require architectural change. Fundamental to how transformers achieve capability within dimensional constraints. Models without superposition would require impractically large dimensions.

Example

*Input:* [Single neuron responding to multiple unrelated concepts: "Paris", "capital", "tower"]
*Behavior:* Neuron participates in representing multiple sparse features through superposition, firing for different concepts in different contexts
*Effect:* Efficient packing of features; polysemantic neuron behavior enabling rich representation

**Status:** WELL-DOCUMENTED | **Related:** nonlinear-composition, abstract-concepts

#### 4.8.5 Infrastructure Primitives

**Depth:** `N/A (architectural)` | **Primary Implementation:** Architecture | **Literature names:** *architectural primitives, foundational components*

Architectural elements that enable computational mechanisms but are not mechanisms themselves. These are universal requirements for transformer operation: (1) **Residual Connections** - additive skip connections enabling gradient flow and cross-layer circuits, (2) **Layer Normalization** - activation stabilization applied before sublayers, (3) **Attention Masking** - causal masking preventing future token access, (4) **Embedding/Unembedding** - token↔vector conversion, (5) **Attention Computation** - scaled dot-product attention operation, (6) **Gradient Flow** - backpropagation infrastructure, (7) **Tokenization** - text↔token conversion (BPE), (8) **Positional Encoding** - sequence order information.

These primitives enable rather than implement computation. Their role is architectural (determined by design) not learned (discovered through training). They differ from mechanisms by having no direct computational semantics - residual connections route information but don't decide what to route; layer normalization stabilizes activations but doesn't determine which features to extract.

**Architecture:** Universal architectural requirements present in all transformer models. Residual stream composition: $h_{l+1} = h_l + \text{Attn}(h_l) + \text{MLP}(h_l + \text{Attn}(h_l))$. Pre-LN: normalization applied before sublayers. Causal masking: $M_{ij} = 0$ if $j \leq i$ else $-\infty$. Details in foundational architecture papers.

**Expected ablation:** Catastrophic architectural failures. Removing any primitive fundamentally breaks transformer operation: no residual connections = training impossible beyond 3-5 layers; no layer norm = severe instability; no attention masking = autoregressive generation failure; no embeddings = cannot process tokens; no gradient flow = cannot train.

Example

These are prerequisites for all transformer computation, not examples of specific mechanisms. See Vaswani et al. (2017), He et al. (2016), Ba et al. (2016) for architectural details.

**Status:** WELL-DOCUMENTED | **Related:** attention-mlp-composition, multi-head-coordination

# 5 Discussion

## 5.1 Key Patterns

**Component specialization:** Attention specializes in routing (15 attention-primary mechanisms), MLPs in storage and transformation (21 MLP-required mechanisms). Sophisticated behaviors require multi-component circuits spanning 5–30 layers.

**Polyfunctionality:** Components contribute to multiple mechanisms depending on context. Duplicate-token heads serve pattern completion, entity tracking, and output routing. This computational reuse enables efficiency but complicates attribution.

**Depth-based cascades:** Mechanisms compose across depth in systematic patterns:

- Early→Middle: Surface features prepare semantic processing
- Middle→Late: Retrieved information feeds integration and routing
- Late→Final: Content generation meets constraint enforcement

## 5.2 Relationship to Prior Work

This taxonomy integrates findings from:

- Attention head taxonomies [6]: incorporated as mechanism implementations
- MLP memory frameworks [2, 3]: integrated into full retrieval circuits
- Circuit analyses [5]: generalized into mechanism compositions
- SAE feature catalogs [1]: connected to mechanism components

The contribution is integration through mechanism-first organization.

## 5.3 Limitations

**Incomplete coverage:** Many capabilities (strategic planning, creative generation, abstract reasoning) lack mechanistic understanding.

**Empirical gaps:** Some mechanisms (memory consolidation, structural boundary tracking) need stronger validation.

**Architecture dependence:** Focus on decoder-only transformers; generalization to other architectures requires investigation.

**Dynamic activation:** Mechanisms activate conditionally based on context; static descriptions don't capture this.

**Mechanism interactions:** Interference patterns between mechanisms (safety vs. helpfulness, format vs. reasoning) remain undercharacterized.

### 5.4 Future Directions

**Automated detection:** Develop tools to identify mechanisms in new models using taxonomy as specification.

**Complete MLP characterization:** Systematic study of MLP functions beyond key-value memory.

**Circuit composition rules:** Formalize patterns governing how mechanisms compose into sophisticated behaviors.

**SAE-mechanism integration:** Map sparse autoencoder features to mechanism components systematically.

**Cross-architecture transfer:** Test mechanism generalization to encoders, encoder-decoders, and alternative architectures.

**Mechanism editing:** Use understanding to improve capabilities through targeted interventions.

## 6 Conclusion

This taxonomy provides mechanistic interpretability with standardized vocabulary bridging research traditions, functional organization revealing computational architecture, and explicit multi-component integration patterns. The framework enables researchers to communicate about computational functions while maintaining precision about implementations.

**Adoption guidelines:**

1. Use canonical mechanism names with literature cross-references
2. Specify implementation details: components, depths, circuit patterns
3. Map components to mechanisms they implement
4. Use relative depth notation for cross-architecture comparison
5. Mark confidence levels (well-documented/observed/proposed)

The taxonomy represents progress toward complete mechanistic understanding: explaining transformer capabilities through mechanism composition from architectural primitives. As interpretability research continues, mechanism-first organization will integrate findings into cumulative mechanistic knowledge.

## References


[1] Trenton Bricken, Adly Templeton, Joshua Batson, et al. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. URL https://transformer-circuits.pub/2023/monosemantic-features/index.html.

[2] Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*, 2021.

[3] Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. *arXiv preprint arXiv:2304.14767*, 2023.

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

[5] 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.

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