K-MSHC: Unmasking Minimally Sufficient Head Circuits in Large Language Models with Experiments on Syntactic Classification Tasks

Pratim Chowdhary
Department of Computer Science
Dartmouth College
cpratim.25@dartmouth.edu

Peter Chin
Department of Engineering
Thayer School of Engineering
pc@dartmouth.edu

Deepernab Chakrabarty
Department of Computer Science
Dartmouth College
deepernab@dartmouth.edu

Abstract

Understanding which neural components drive specific capabilities in mid-sized language models ($\leq$10B parameters) remains a key challenge. We introduce the $(\boldsymbol{K}, \epsilon)$-Minimum Sufficient Head Circuit (K-MSHC), a methodology to identify minimal sets of attention heads crucial for classification tasks as well as Search-K-MSHC, an efficient algorithm for discovering these circuits. Applying our Search-K-MSHC algorithm to Gemma-9B, we analyze three syntactic task families: grammar acceptability, arithmetic verification, and arithmetic word problems. Our findings reveal distinct task-specific head circuits, with grammar tasks predominantly utilizing early layers, word problems showing pronounced activity in both shallow and deep regions, and arithmetic verification demonstrating a more distributed pattern across the network. We discover non-linear circuit overlap patterns, where different task pairs share computational components at varying levels of importance. While grammar and arithmetic share many "weak" heads, arithmetic and word problems share more consistently critical "strong" heads. Importantly, we find that each task maintains dedicated "super-heads" with minimal cross-task overlap, suggesting that syntactic and numerical competencies emerge from specialized yet partially reusable head circuits.

1 Introduction

How do language models organize their capabilities? As mid-sized language models ($\leq$10B parameters) master increasingly diverse tasks—from solving arithmetic problems to identifying ungrammatical sentences—a fundamental question emerges: do related capabilities share the same neural circuitry, or does each task develop its own specialized pathway?

This question has profound implications for how we understand, improve, and control these models. If grammatical analysis and arithmetic reasoning utilize the same circuits, then improvements in one capability might automatically enhance the other. Conversely, if each task relies on distinct components, we gain the ability to target interventions precisely—enhancing specific abilities without disrupting others. Yet despite the practical importance of this organizational question, we lack efficient tools to provide definitive answers [Adolfi et al., 2025].

Preprint. Under review.

To address this, we introduce the (K, ϵ)**-Minimum Sufficient Head Circuit (**K**-MSHC)** framework, an efficient approach for isolating sparse subsets of attention heads that minimally, sufficiently explain task performance. This lets us probe the compositional structure of model capabilities in a targeted way. We formalize a novel attention head pruning algorithm which results in approximate circuits that are sufficient and minimal with theoretical guarantees.

Current interpretability research has revealed that individual attention heads specialize in specific functions [Voita et al., 2019, Clark et al., 2019] and that disabling certain heads can impact performance on specific tasks [Michel et al., 2019]. But these findings don't conclusively answer whether related capabilities emerge from shared or separate neural pathways. The missing piece is a principled approach for identifying the minimal set of model components necessary and sufficient for specific tasks—and importantly, for measuring how these sets overlap across different capabilities.

We build on these results and investigate two further questions:

1. Do linguistic and numerical reasoning tasks recruit shared pathways or task-specific circuits?
2. What patterns of overlap emerge between circuits for related tasks, and what do these patterns reveal about the organization of knowledge within the model?

Our experiments reveal a clear organizational pattern in Gemma-9B. Grammar tasks predominantly utilize early layers, word problems engage both shallow and deep network regions, and arithmetic verification employs a distributed mechanism across the network. We observe that while grammatical and arithmetic tasks share many weakly contributing computational components, they maintain dedicated "super-heads" with minimal cross-task overlap. This finding indicates that the model develops specialized circuits for different capabilities while efficiently reusing resources where possible.

These results provide evidence for a nuanced view of LLM organization. Rather than implementing either fully general or fully specialized mechanisms, compact models develop task-specific yet partially reusable circuits. This architectural characteristic enables diverse capabilities within parameter constraints and suggests that similar principles may apply to larger frontier models, where understanding organizational structure becomes increasingly important for alignment and control.

In sum, our contributions are three-fold:

(i) We introduce the $(\boldsymbol{K}, \epsilon)$**-Minimum Sufficient Head Circuit (**$\boldsymbol{K}$**-MSHC)** framework, a highly efficient approach to identify minimal sets of attention heads crucial for specific tasks, measuring their *minimality*, *sufficiency*, and *necessity*.

(ii) We develop **Search-K-MSHC**, an efficient stochastic search algorithm that makes circuit discovery feasible in large language models with thousands of attention heads, complemented by our **Low-Dimensional Linear Separability (LS)** metric that addresses dimensionality challenges when probing representations.

(iii) We present a comprehensive analysis of **syntactic versus arithmetic circuits** in Gemma-9B, revealing both task-specialized components and shared "super-heads" that demonstrate how mid-sized LLMs efficiently encode multiple capabilities through overlapping head circuits.

2 Related Work

Evaluating and Probing Language Models. Specialized benchmarks have emerged to track the capabilities of sub-10B parameter models. BLiMP [Warstadt et al., 2020] provides fine-grained assessments of syntactic competence, while GSM8K [Cobbe et al., 2021] challenges models with grade-school arithmetic problems. Recent surveys [Zhao et al., 2023] document how these capabilities exhibit non-uniform scaling properties across parameter thresholds. Linear probing methods have become standard tools for analyzing information encoded in neural representations [Alain and Bengio, 2016]. Extensions such as control tasks [Hewitt and Manning, 2019] and diagnostic classifiers [Tenney et al., 2019] help distinguish linguistic structure from memorization artifacts. Our Low-Dimensional Linear Separability (LS) metric builds on this tradition but introduces a crucial innovation by projecting to a minimal subspace via PCA before applying a linear classifier.

Attention Mechanisms and Mechanistic Interpretability. Previous work has studied attention pattern interpretability [Clark et al., 2019] and demonstrated that many heads can be pruned without significant performance degradation [Michel et al., 2019]. Task-specialized heads have been identified

through activation analysis [Voita et al., 2019]. K-MSHC extends these insights by formalizing the concept of minimal sufficient circuits—the smallest set of attention heads where any K-subset restores task performance. The emerging field of mechanistic interpretability seeks to reverse-engineer neural networks at the component level. Circuit-level analyses have mapped syntactic processing [Clark et al., 2019] and key-value memories [Geva et al., 2021]. Work by Olah et al. [Olah et al., 2020] and Elhage et al. [Elhage et al., 2021] suggests that capabilities emerge from sparse subnetworks that can be isolated through careful intervention studies. Our K-MSHC framework operationalizes this insight in mid-sized models like Gemma-9B.

Computational Component Discovery. Sparse autoencoders and dictionary–learning techniques have recently been leveraged to extract highly interpretable, near-monosemantic features from the activations of large language models [Cunningham et al., 2023, Bricken et al., 2023, Gao et al., 2024, Templeton et al., 2024]. These methods complement circuit-level analyses by working at the representational level and provide an alternative path toward isolating task-relevant computational units. A parallel line of research proposes algorithms that automatically identify local and global circuits using linear computation graphs, cross-layer mappings, or feature editing [Marks et al., 2024, Ge and Hoefler, 2024, Lindsey et al., 2024, Dunefsky et al., 2025]. Our *Search-K-MSHC* algorithm differs from these approaches by explicitly enforcing K-sufficiency, yielding minimal sets that are both necessary and redundantly sufficient for a given task.

Task-Specific Head Circuits. Mechanistic analyses have been extended beyond single-step tasks to multi-hop reasoning in language models [Yang et al., 2024, Biran et al., 2024, Yu et al., 2025] and emergent planning behaviour in specialised agents [Jenner et al., 2025, Taufeeque et al., 2024, Bush et al., 2024]. Our results on arithmetic word problems echo these findings, showing that high-level reasoning engages heads across distant layers that nonetheless admit mid-sized sufficient subsets. Recent work traces how language models carry out symbolic or approximate arithmetic, attributing addition to Fourier-like or trigonometric transformations and heuristic ensembles [Stolfo et al., 2023, Zhou et al., 2024, Nikankin et al., 2024, Kantamneni and Tegmark, 2025]. The broadly distributed pattern we observe for arithmetic verification aligns with these results, suggesting that numerical operations recruit a wider basis of heads than purely grammatical processing. Studies of multilingual models reveal that latent grammatical concepts are encoded in shared subspaces across languages, with task-specific specialisations layered on top [Brinkmann et al., 2025, Dumas et al., 2024, Zhang et al., 2024]. The partial head overlap we observe between grammar and arithmetic tasks may reflect the same "semantic hub" principle observed in these cross-lingual analyses.

3 Methodology

We formalize the problem of identifying minimal head circuits responsible for specific model capabilities. Our (K, ϵ)-Minimum Sufficient Head Circuit framework quantifies the causal contribution of attention heads to task performance.

3.1 The (K, ϵ)-Minimum Sufficient Head Circuit

To identify where task-specific knowledge resides in a model, we adopt a causal intervention approach: if removing specific components disrupts the ability to distinguish correct from incorrect examples, those components likely encode the relevant knowledge.

Definition 1 (Task Separability Score)**.** *For a task $\mathcal{T}$ with dataset $\mathcal{D}_\mathcal{T} = \{(x_i, y_i)\}_{i=1}^n$ where $y_i \in \{-1, 1\}$, the separability score of a model configuration $\mathcal{M}$ is:*

$$\mathcal{S}_{\mathcal{D}_\mathcal{T}}(\mathcal{M}) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}\left[f_\theta(\mathbf{h}_{x_i,L}^{EOS}) = y_i\right] \tag{1}$$

where $\mathcal{M}$ specifies which attention heads are active, $\mathbf{h}_{x_i,L}^{EOS}$ is the final-layer embedding representation, and $f_\theta : \mathbb{R}^d \to [0, 1]$ is an optimal classifier over that embedding space.

We hypothesize that each task relies on a minimal circuit of attention heads that maintains these distinctions. By comparing separability across head subsets, we can identify these circuits and analyze how language models allocate resources between capabilities.

Definition 2 ((K, ϵ)-Minimum Sufficient Head Circuit)**.** *The (K, ϵ)-Minimum Sufficient Head Circuit identifies the smallest set of attention heads $\mathcal{H} \subset \mathcal{M}$ such that any subset of K heads from $\mathcal{H}$ can restore the model's classification performance to over some ϵ performance threshold.*

Given a baseline model $\mathcal{B}$ (typically a subset of heads in $\mathcal{M}$) and a parameter $\epsilon \in [0, 1]$, we define the understanding threshold:

$$\mathcal{U}^{\epsilon}(\mathcal{M}, \mathcal{B}) = \mathcal{S}_{\mathcal{D}_\mathcal{T}}(\mathcal{B}) + \epsilon \cdot (\mathcal{S}_{\mathcal{D}_\mathcal{T}}(\mathcal{M}) - \mathcal{S}_{\mathcal{D}_\mathcal{T}}(\mathcal{B})) \tag{2}$$

Definition 3 ((K, ϵ)-Minimum Sufficient Head Circuit)**.** *The (K, ϵ)-Minimum Sufficient Head Circuit is the smallest set of heads $\mathcal{H} \subset \mathcal{M}$ satisfying:*

$$\forall \mathcal{H}' \subseteq \mathcal{H} \text{ with } |\mathcal{H}'| = K : \mathcal{S}_{\mathcal{D}_\mathcal{T}}((\mathcal{M} \setminus \mathcal{H}) \cup \mathcal{H}') \geq \mathcal{U}^{\epsilon}(\mathcal{M}, \mathcal{B}) \tag{3}$$

This definition has three key properties:

Property 1 (Key Properties of (K, ϵ)-MSHC)**.**

(i) ***Minimality****: $\mathcal{H}$ is the smallest set satisfying the condition*

(ii) ***K-Sufficiency****: Any K-subset of $\mathcal{H}$ can restore performance*

(iii) ***Isolated Insignificance****: $\mathcal{M} \setminus \mathcal{H}$ alone contributes minimally to task understanding*

The parameter K controls redundancy in the circuit, while ϵ determines how close to full performance the circuit must restore.

3.2 Search-K-MSHC: An Efficient Algorithm for Circuit Discovery

Finding the exact (K, ϵ)-Minimum Sufficient Head Circuit is likely computationally intractable, requiring examination on the order of $2^{|\mathcal{M}|}$ possible head subsets. We introduce Search-K-MSHC, a stochastic algorithm with parameters $\mathcal{W}$ (window size), p (percentile), and N (samples) that efficiently approximates the solution through two phases:

(i) **Macro Layer Search**: Identifies critical layers by window-based ablation.
(ii) **Micro Head Search**: Refines head candidates via binary search and stochastic pruning.

Macro Layer Search. We identify task-critical layers via:

(i) Window-based ablation: Slide a window of size $\mathcal{W}$ across adjacent network layers
(ii) Initialize array $\textsc{Drop}[1 : L]$ to store performance drop measurement. Update $\textsc{Drop}[\ell] = \min(\textsc{Drop}[\ell], \mathcal{S}_{\mathcal{D}_\mathcal{T}}(\mathcal{M}) - \mathcal{S}_{\mathcal{D}_\mathcal{T}}(\mathcal{M} \setminus \mathcal{H}_\mathcal{W}))$ where $\mathcal{H}_\mathcal{W}$ contains all heads in the window
(iii) Select high-impact layers (top p-th percentile) for candidate set $\mathcal{C}$ from $\textsc{Drop}[1 : L]$

Stochastic Head Pruning. With baseline $\mathcal{B} = \mathcal{M} \setminus \mathcal{C}$, we:

(i) Start with subset size $k = \lfloor |\mathcal{C}|/2 \rfloor$
(ii) Sample N random k-subsets and find worst-performing $\Theta_{\min} = \arg\min_i \mathcal{S}_{\mathcal{D}_\mathcal{T}}(\mathcal{B} \cup \Theta_i)$
(iii) If $\mathcal{S}_{\mathcal{D}_\mathcal{T}}(\mathcal{B} \cup \Theta_{\min}) < \mathcal{U}^{\epsilon}(\mathcal{M}, \mathcal{B})$, prune $\mathcal{C} \leftarrow \mathcal{C} \setminus \Theta_{\min}$; else reduce $k \leftarrow \max(K, \lfloor k/2 \rfloor)$
(iv) Terminate when $k = K$ and threshold met

3.3 Theoretical Analysis of Search-K-MSHC

Definition 4. *Let $\mathcal{C}_i$ be the candidate set at iteration i, and let $\tau = \mathcal{U}^{\epsilon}(\mathcal{M}, \mathcal{B})$ be our performance threshold. For parameters $0 \leq \delta_i, \delta_T \leq 1$, we define:*

(i) Low-impact heads: Set $\mathcal{Q} \subseteq \mathcal{C}_i$ with $|\mathcal{Q}| = \delta_i |\mathcal{C}_i|$ s.t. $\forall \mathcal{X} \subseteq \mathcal{C}_i$ with $|\mathcal{X}| = K$ and $|\mathcal{X} \cap \mathcal{Q}| > \delta_T K$: $\mathcal{S}_{\mathcal{D}_\mathcal{T}}((\mathcal{M} \setminus \mathcal{C}_i) \cup \mathcal{X}) \leq \tau$, or heads that don't meaningfully contribute to "understanding"

(ii) Prunable sets: $\mathcal{P}_i = \{\mathcal{X} \subseteq \mathcal{C}_i : |\mathcal{X}| = K, |\mathcal{X} \cap \mathcal{Q}| > \delta_T K\}$, the K-subsets containing too many low-impact heads that we would like to prune.

These assumptions are made to simplify the analysis but likely emulate actual head importance patterns, with a number of unimportant heads over some threshold likely leading to prunable sets.

Theorem 1 (Expected Missed Prunable Sets)**.** *For a candidate set $\mathcal{C}_i$ with a contamination rate δ_i of low-impact heads and threshold parameter $\delta_T < \delta_i$, the expected number of prunable sets that remain undetected after sampling N random K-subsets is:*

$$\mathbb{E}[\textit{Undetected Sets}] = O\left(\frac{\delta_i|\mathcal{C}_i|}{\delta_T K} \cdot \exp\left(-NK(\delta_i - \delta_T)^2\right)\right) \tag{4}$$

Proof. For K randomly sampled heads, the probability of missing a prunable set follows a hypergeometric distribution $H \sim \text{Hypergeometric}(|\mathcal{C}|, |\mathcal{Q}|, K)$. By Hoeffding's inequality:

$$\Pr[H \leq \delta_T \cdot K] \leq \exp\left(-2K \cdot (\delta_i - \delta_T)^2\right) \tag{5}$$

With N independent samples, the miss probability becomes:

$$\Pr[\text{All } N \text{ Samples Miss}] \leq \exp\left(-2NK \cdot (\delta_i - \delta_T)^2\right) \tag{6}$$

Since we can only catch non-overlapping prunable sets, the maximal number of undetected prunable sets is bounded by $|\mathcal{Q}|/(\delta_T \cdot K + 1)$. By linearity of expectation:

$$\begin{aligned}\mathbb{E}[\text{Undetected Sets}] &\leq \frac{|\mathcal{Q}|}{\delta_T \cdot K + 1} \cdot \exp\left(-2NK \cdot (\delta_i - \delta_T)^2\right) \qquad (7)\\ &= O\left(\frac{\delta_i|\mathcal{C}_i|}{\delta_T K} \cdot \exp\left(-NK(\delta_i - \delta_T)^2\right)\right) \qquad \text{(since } |\mathcal{Q}| = \delta_i|\mathcal{C}_i|\text{)}\end{aligned}$$

The key takeaways from this result are two-fold: □

(i) The expected number of missed prunable sets decreases exponentially with N (samples), K (subset size), and $\Delta_i^2 = (\delta_i - \delta_T)^2$ (squared margin).

(ii) Critically, when the low-impact ratio δ_i is much higher than the threshold δ_T, the algorithm is extremely unlikely to terminate with significant numbers of prunable sets remaining.

Algorithm 1 Search-K-MSHC

Require: window size W, percentile p, samples per iteration N

initialise array $\textsc{Drop}[1:L] \leftarrow 0$
for $s = 1$ **to** $L - W + 1$ **do** ▷ **Macro layer search**
  $\mathcal{H}_W \leftarrow$ heads in layers $s{:}s + W - 1$
  for $\ell = s$ **to** $s + W - 1$ **do**
    $\textsc{Drop}[\ell] = \min\big(\textsc{Drop}[\ell], \mathcal{S}_{\mathcal{D}_\mathcal{T}}(\mathcal{M}) - \mathcal{S}_{\mathcal{D}_\mathcal{T}}(\mathcal{M} \setminus \mathcal{H}_W)\big)$
$\mathcal{C} \leftarrow \{\mathcal{H}_\ell \mid \textsc{Drop}[\ell] \geq$ top p-th percentile of $\textsc{Drop}\}$ ▷ sensitive layers
$\mathcal{B} \leftarrow \mathcal{M} \setminus \mathcal{C}$ ▷ baseline model
$k \leftarrow \lfloor |\mathcal{C}|/2 \rfloor$
while $k \geq K$ **do** ▷ **Stochastic head pruning**
  $\text{best} \leftarrow 1;\ \Theta_{\min} \leftarrow \emptyset$
  for $i = 1$ **to** N **do**
    draw $\Theta \sim \text{Unif}\{\mathcal{X} \subseteq \mathcal{C} : |\mathcal{X}| = k\}$
    if $\mathcal{S}_{\mathcal{D}_\mathcal{T}}(\mathcal{B} \cup \Theta) <$ best **then**
      $\text{best} \leftarrow \mathcal{S}_{\mathcal{D}_\mathcal{T}}(\mathcal{B} \cup \Theta);\ \Theta_{\min} \leftarrow \Theta$
  if $\text{best} \leq \mathcal{U}^{\mathcal{S}}_{\mathcal{D}_\mathcal{T}}(\mathcal{M}, \mathcal{B})$ **then**
    $\mathcal{C} \leftarrow \mathcal{C} \setminus \Theta_{\min}$ ▷ safe to prune
  else
    $k \leftarrow \max(K, \lfloor k/2 \rfloor)$
return $\mathcal{C}$

3.4 Scoring Metric: Low-Dimensional Linear Separability (LS)

Standard linear probes for analyzing LLM representations often overfit due to high dimensionality ($d \approx 4096$) and limited training data. To address this challenge, we introduce Low-Dimensional Linear Separability (LS), focused on the final layer's EOS token representations as our scoring metric to guide the search for minimal sufficient head circuits. It operates in two phases:

Dimensionality Reduction. We project to a subspace preserving maximal variance by computing the empirical covariance matrix $\Sigma_L = \frac{1}{|\mathcal{D}_\mathcal{T}|} \sum_x \left(\mathbf{h}^{\text{EOS}}_{x,L} - \bar{\mathbf{h}}_L\right)\left(\mathbf{h}^{\text{EOS}}_{x,L} - \bar{\mathbf{h}}_L\right)^\top$, extracting its top D eigenvectors $\mathbf{W}_L$, and projecting:

$$\tilde{\mathbf{h}}_{x,L} = \mathbf{W}_L^\top(\mathbf{h}^{\text{EOS}}_{x,L} - \bar{\mathbf{h}}_L) \in \mathbb{R}^D \tag{8}$$

Classification. We train a linear SVM by optimizing the regularized hinge loss:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{|\mathcal{D}_\mathcal{T}|} \max\left(0,\, 1 - y_i(\mathbf{w}^\top \tilde{\mathbf{h}}_{x_i,L} + b)\right), \quad C = 10 \tag{9}$$

and calculate LS score as classification accuracy: $\text{LS}^D_{\mathcal{D}_\mathcal{T}} = \frac{1}{|\mathcal{D}_\mathcal{T}|}\sum_{i=1}^{|\mathcal{D}_\mathcal{T}|} \mathbb{I}[\text{sign}(\mathbf{w}^\top \tilde{\mathbf{h}}_{x_i,L} + b) = y_i]$

By restricting to $D \leq 5$ dimensions, we ensure the metric captures task-relevant information rather than dimensionality artifacts. This approach efficiently detects when attention heads encode task-specific knowledge in their representations.

3.5 Evaluation Framework: Task Families and Dataset Construction

We evaluate K-MSHC using three task families with controlled minimal pairs, where examples differ only in a single task-relevant feature. All tasks have balanced classes and use a consistent yes/no formulation.

Grammar Acceptability (G). Based on BLiMP [Warstadt et al., 2020] (67,000 sentence pairs), focusing on determiner-noun agreement due to the inherent numerocity of the task:

> **Correct:** Leslie isn't firing that actress.
> **Incorrect:** Leslie isn't firing that actresses.

Arithmetic Verification (A). 1000 algorithmically generated equation pairs with random operands $n_1, n_2 \in [1, 10^3]$, operations $\in \{+, -\}$, and perturbed answers (0.5 / 1.5×) for incorrect equations:

> **Correct:** 1338 + 88 = 1426
> **Incorrect:** 1338 + 88 = 2139 *(≈ 1.5 × correct result)*

Word Problems (W). 100 natural language arithmetic problems templates filled with random numbers using the same methodology as (A), with (0.5 / 1.5×) perturbed answers for incorrect equations:

> **Correct:** Tim has 5 apples and eats 2, leaving him with 3 apples.
> **Incorrect:** Tim has 5 apples and eats 2, leaving him with 5 apples. *(≈ 1.5 × correct result)*

This progression from grammatical knowledge (G) to abstract computation (A) to contextualized reasoning (W) allows analysis of both task-specific circuits and potential shared components.

4 Experiments

Using our K-MSHC framework, we probe Gemma-9B to analyze which head circuits are responsible for the models understanding of grammar, arithmetic, and word problems tasks. We run experiments with parameters $\mathcal{W} = 5$, $p = 0.75$, $N = 10$, $K = 10$, and $\epsilon = 0.25$ across 20 trials with mini-batches of 50 (positive and negative) examples per task. All experiments were conducted on Nvidia H100 GPUs with 50 GB of memory. We used PyTorch for all LLM inference and manipulation, while

scikit-learn was employed for implementing PCA dimensionality reduction and SVM classifiers for the Linear Separability metrics.

4.1 Baseline Performance Analysis

Before identifying specific head circuits, we first establish that information about our classification tasks is indeed encoded in the model's representations. Table 1 presents the Linear Separability (LS) scores for each task, showing that Gemma-9B's representations naturally encode strong task-relevant information, with baseline LS scores ranging from 0.77 to 0.99. When critical layers (identified through ablation) are removed, performance drops substantially across all tasks, with different tasks showing varying sensitivity to layer ablation. Arithmetic verification exhibits the largest drop (44%), followed by grammar (36%) and word problems (21%).

Task	Baseline LS Score	LS Score Post 25% Layer Ablation	Drop
Arithmetic	$0.99\ [0.98, 1.00]$	$0.55\ [0.52, 0.60]$	44%
Grammar	$0.86\ [0.78, 0.90]$	$0.50\ [0.49, 0.52]$	36%
Word Problems	$0.77\ [0.73, 0.86]$	$0.56\ [0.53, 0.61]$	21%

Table 1: Linear separability scores before and after ablating critical layers, showing task-dependent information encoding within the model architecture. Values show medians with 95% confidence intervals in square brackets.

4.2 Distinct Head Circuits for Different Capabilities

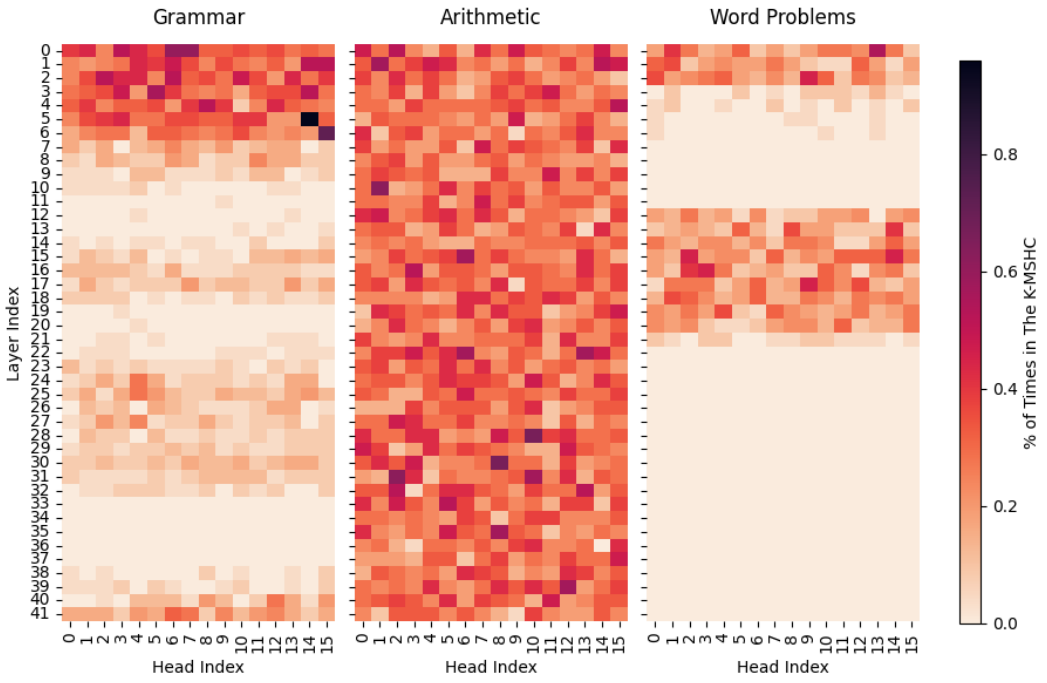

Figure 1: Heat map of attention head importance across model layers. Each cell represents an attention head, with color intensity showing selection frequency across 20 trials.

Figure 1 reveals the spatial distribution of attention heads identified by our K-MSHC algorithm across the model architecture. Analysis of these patterns uncovers three key organizational principles in Gemma-9B's computational structure.

Architectural Specialization: Each task engages a distinctive subset of the network. Grammar processing primarily activates early layers (0-6) with specific banding patterns in later regions. Word

problems utilize a bimodal distribution with concentrated activity in both shallow (0-3) and deep (11-20) regions. Arithmetic verification shows the most distributed pattern, engaging heads across the entire network with more uniform density.

Intra-Layer Selectivity: Within even the most important layers, not all heads contribute equally. Attention patterns show high selectivity, with only a few heads per layer being consistently critical. This suggests a sparse coding principle where specific head combinations—rather than entire layers—form the building blocks of task-specific circuits.

Task-Dependent Organization: Structurally related tasks demonstrate different patterns of head criticality. The well-defined task boundaries of grammar and arithmetic verification correlate with concentrated "super-head" patterns—specific heads that appear in almost all K-MSHCs for these tasks. In contrast, word problems require contextual reasoning across both linguistic and numerical domains, resulting in more diffuse activation patterns without dominant super-heads.

These findings demonstrate that language capabilities are not uniformly distributed throughout the model but are encoded through sparse, task-specialized circuits with distinct architectural signatures.

4.3 Circuit Overlap Analysis

To investigate how computational resources are shared across tasks, we analyzed the overlap between circuits identified for each task pair. Figure 2 visualizes this overlap at different selection thresholds (percentage of most frequently selected heads):

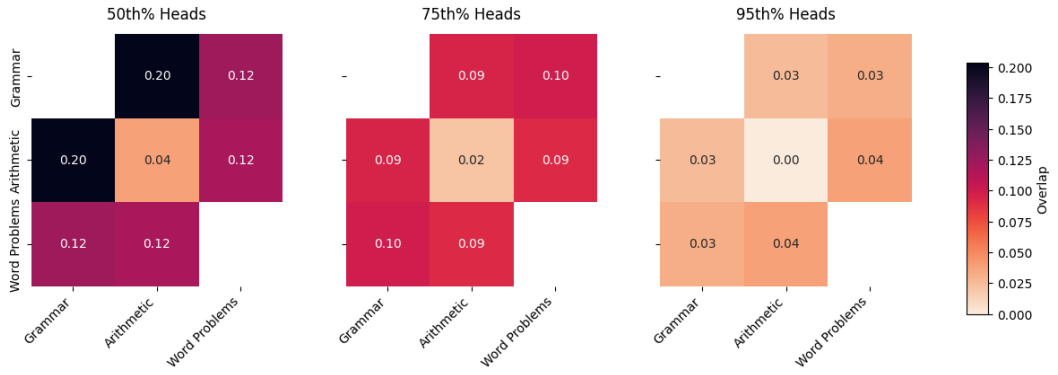

Figure 2: Circuit overlap (Jaccard similarity) between task pairs. The matrix shows overlap percentages at three selection thresholds: 50% (weaker but more numerous heads), 75% (moderate), and 95% (strongest "super-heads"). The central region indicates three-way overlap.

Our analysis reveals several nuanced patterns in how Gemma-9B shares computational resources across tasks.

Sharing "Weak" vs "Strong" Heads: Task pairs exhibit different sharing patterns depending on head importance. At the 50% threshold (including weaker heads), grammar and arithmetic show substantial overlap ($\approx$20%). However, this relationship inverts at the 75% threshold, where arithmetic and word problems share more critical heads ($\approx$10%) than grammar-arithmetic pairs ($\approx$9%). This finding suggests that tasks either share "weak" heads that are vaguely related or share "strong" heads that are more specific to the task overlap.

Specialized "Super-Heads": The strongest heads (95% threshold) show minimal cross-task overlap, with each task pair maintaining dedicated circuits of "super-heads". This separation is particularly striking given that arithmetic verification and word problems involve related numerical reasoning, yet their most critical components remain largely distinct. This finding challenges the notion that higher-level capabilities like "arithmetic reasoning" have a single circuit implementation within the model.

Non-Uniform Resource Allocation: The asymmetric pattern of overlap reduction across thresholds—with grammar-arithmetic showing steeper decline than other pairs—indicates that head im-

portance has task-specific scaling properties. This suggests that the model allocates computational resources non-uniformly, with some task relationships maintaining more robust sharing across importance levels than others. These overlap patterns reveal that Gemma-9B balances specialization and resource sharing through a hierarchical organization of computational components. While allowing partial resource reuse, particularly for related tasks, the model also maintains dedicated circuits for each capability's core processing requirements.

5 Limitations and Future Work

While our K-MSHC framework reveals important insights about attention circuit organization, several limitations should be acknowledged:

Algorithmic Assumptions. Our theoretical analysis relies on simplifying assumptions about head contamination rates and their distribution across the network. The convergence guarantees are strongest when the distribution of low-impact heads is well-separated from task-critical heads, but real-world models may exhibit more complex patterns of head importance with less clear separation.

Circuit Specificity. The triangulation of head circuits was not as precise as initially anticipated. We observed some variation in the specific heads identified across trials, suggesting that multiple distinct but functionally equivalent circuits may exist for each task. This redundancy makes it challenging to definitively map the exact set of heads responsible for a capability.

Model and Parameter Sensitivity. Our analysis is limited to a single model architecture (Gemma-9B) with one set of hyperparameters for the search algorithm. While we found our approach effective, the circuits identified may be sensitive to the choice of K, ϵ, and other parameters, particularly for tasks where performance is distributed across many weakly-contributing heads.

Limited Task Diversity. Our focus on three specific task families provides an insightful but still limited view of how language capabilities are organized. More complex reasoning, world knowledge, or multimodal tasks might reveal different circuit structures and overlap patterns. Future work should address these limitations by:

(i) Exploring broader parameter settings across different model architectures to identify invariant circuits and understand sensitivity effects

(ii) Developing refined methods for handling redundant circuits while investigating their activation dynamics during inference

(iii) Extending analysis to more complex tasks that engage multiple capabilities simultaneously to better map the functional organization of language models

These extensions would strengthen our understanding of how language capabilities are mechanistically implemented in neural architectures and potentially enable more targeted model interventions and improvements.

6 Conclusion

Our work introduces K-MSHC, a framework for identifying minimal sufficient head circuits in mid-sized language models, revealing that different syntactic tasks utilize distinct neural pathways in Gemma-9B. Grammar tasks predominantly activate early layers (0-6), while word problems utilize specific bands in both early and deep regions, with arithmetic verification showing more distributed patterns. We found non-linear patterns of circuit overlap, with grammar and arithmetic sharing more "weak" heads while arithmetic and word problems share more "strong" heads, indicating that despite partial resource sharing, each task maintains dedicated "super-heads" with minimal overlap at high thresholds. These findings advance mechanistic interpretability by demonstrating that language capabilities emerge from specialized but partially reusable circuits rather than fully general mechanisms, suggesting that future research should focus on identifying and understanding these sparse computational primitives across different model scales and architectures.

References

Federico Adolfi, Martina G. Vilas, and Todd Wareham. The computational complexity of circuit discovery for inner interpretability. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=QogcGNXJVw.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. In *International Conference on Learning Representations (Workshop Track)*, 2016.

Elad Biran, Timo Schick, and Hinrich Schütze. Hopping too late: Limitations of llms on multi-hop queries. 2024. URL https://arxiv.org/abs/2406.12775.

Thomas Bricken, Catherine Olsson, Matthew Ziegler, Nelson Elhage, Neel Nanda, and Chris Olah. Towards monosemanticity: Decomposing language models with dictionary learning. 2023. URL https://arxiv.org/abs/2301.04709. Transformer Circuits Thread.

Johannes Brinkmann, Gisbert Fanselow, and Jon Gauthier. Large language models share latent grammatical concepts across languages. 2025. URL https://arxiv.org/abs/2501.06346.

Tristan Bush, Rohin Shah, and Jan Leike. Interpreting emergent planning in model-free reinforcement learning. In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR 2024)*, 2024.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, 2019.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Henry Cunningham, Qian Li, Chelsea Chan, and Chris Olah. Sparse autoencoders find highly interpretable model directions. 2023. URL https://arxiv.org/abs/2309.08600.

Clément Dumas, Jiannan Gu, Charles Wang, and Jiuqiang Huang. How do llamas process multilingual text? activation patching study. In *Proceedings of the ICML 2024 Workshop on Multilingual Representation Learning*, 2024.

Jordan Dunefsky, Neel Nanda, and Chris Olah. Transcoders find interpretable llm feature circuits. In *Advances in Neural Information Processing Systems (NeurIPS 37)*, 2025.

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

Lucy Gao, Lachlan Reynolds, Neel Nanda, and Chris Olah. Scaling and evaluating sparse autoencoders. 2024. URL https://arxiv.org/abs/2406.04093.

Xinyun Ge and Torsten Hoefler. Automatically identifying local and global circuits with linear computation graphs. 2024. URL https://arxiv.org/abs/2405.13868.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, 2021.

John Hewitt and Christopher D Manning. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, 2019.

Ethan Jenner, Ben Houk, and Theis Anthropic. Evidence of learned look-ahead in a chess-playing neural network. In *Advances in Neural Information Processing Systems (NeurIPS 37)*, 2025.

Sandipan Kantamneni and Max Tegmark. Language models use trigonometry to do addition. 2025. URL https://arxiv.org/abs/2502.00873.

Jack Lindsey, Henry Cunningham, and Chris Olah. Sparse crosscoders for cross-layer features and model diffing. 2024. URL https://arxiv.org/abs/2403.17677. Transformer Circuits Thread.

Sam Marks, Joseph Raul, Illia Vaswani, and Neel Nanda. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. 2024. URL https://arxiv.org/abs/2403.19647.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, pages 14014–14024, 2019.

Yuval Nikankin, Shiko Beit-Neullas, and Amir Globerson. Arithmetic without algorithms: Language models solve math with a bag of heuristics. 2024. Preprint.

Chris Olah, Nick Cammarata, Ludwig Schubert, et al. Zoom in: An introduction to circuits. *Distill*, 5 (3):e00024, 2020. doi: 10.23915/distill.00024.

Andrew Stolfo, Atticus Geiger, David Friedman, and Vivek Srikumar. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. 2023. URL https://arxiv.org/abs/2305.15054.

Moin Taufeeque, John Lynch, Mario Lucic, and Nicholas Frosst. Planning in a recurrent neural network that plays sokoban. 2024. URL https://arxiv.org/abs/2407.15421.

Alexander Templeton, Chris Olah, and Neel Nanda. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. 2024. URL https://arxiv.org/abs/2406.06061. Transformer Circuits Thread.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, 2019.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, 2019.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R Bowman. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392, 2020.

Shaokun Yang, Pengcheng He, Maosong Sun, and Ming Zhou. Do large language models latently perform multi-hop reasoning? 2024. URL https://arxiv.org/abs/2402.16837.

Zeming Yu, Wenxuan Zhang, Jie Zhou, and Rui Cao. Back attention: Understanding and enhancing multi-hop reasoning in llms. 2025. URL https://arxiv.org/abs/2502.10835.

Rui Zhang, Ziyang Liu, and Ge Fu. Structural similarities and differences in multilingual language modeling. 2024. URL https://arxiv.org/abs/2410.09223.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

Tianyi Zhou, Haitong Ding, and Chun Zong. Llms use fourier features to compute addition. 2024. URL https://arxiv.org/abs/2406.03445.