Circuit-Level Knowledge Distillation for Large Language Models

Yashuo Luo[1,2] **Tongxu Wang**[1] **Siyuan He**[1] **Chunyu Wei**[1*]
[1]School of Information, Renmin University of China, Beijing, China
[2]Beihang University, Beijing, China
luoyashuo@buaa.edu.cn, 2022201286@ruc.edu.cn,
river0307@ruc.edu.cn, weichunyu@ruc.edu.cn

Abstract

Existing knowledge distillation methods supervise only *what* a student outputs, leaving *how* it computes those outputs unconstrained, so students may match teacher behavior through entirely different internal mechanisms. We propose **Meta Circuit Distillation (MCD)**, which reframes distillation as the explicit transfer of reasoning circuits, the structured computational pathways uncovered by mechanistic interpretability. MCD represents each model's computation as a transcoder-based attribution graph and aligns teacher and student circuits via an optimal-transport pathway-matching loss that plugs into any standard distillation pipeline. We prove that minimizing this loss bounds the discrepancy in MLP-level computation between teacher and student. Across two model families, six distillation objectives, and three instruction-following benchmarks, MCD delivers consistent improvements that transfer to mathematical reasoning, with average performance gains across all evaluated student sizes. [2]

1 Introduction

Large language models have achieved remarkable reasoning capabilities, but at a steep computational cost: state-of-the-art systems demand hundreds of billions of parameters and substantial inference infrastructure. Knowledge distillation (KD) mitigates this cost by transferring capabilities from a large teacher to a compact student Hinton et al. [2015]. Existing approaches fall into three families: *output-based* methods that match probability distributions Hinton et al. [2015], *feature-based* methods that align intermediate representations Romero et al. [2015], Zagoruyko and Komodakis [2017], and *attention-based* methods that transfer attention patterns Wang et al. [2020]. Despite their success, these approaches share a common limitation: they treat networks as black boxes and supervise observable outputs without constraining the underlying computational mechanisms. A student that mimics a teacher's outputs may nonetheless develop entirely different reasoning pathways, producing surface-level agreement that fails to inherit the teacher's robustness.

The fundamental issue is that current methods transfer *what* models compute, not *how* they compute it. Recent work in mechanistic interpretability shows that language models organize their computation into structured *circuits*: pathways of information flow through interpretable features that implement coherent algorithms Olah et al. [2020], Elhage et al. [2021]. Examples include induction circuits for in-context learning Olsson et al. [2022] and greater-than circuits for numerical comparison Hanna et al. [2023]. Circuits expose the actual computational mechanisms behind a model's behavior, yet existing distillation approaches operate at an abstraction level that hides them entirely. This gap

*Chunyu Wei is the corresponding author.
[2]Code is available at https://github.com/luoyashuo/meta_circuit_distillation.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

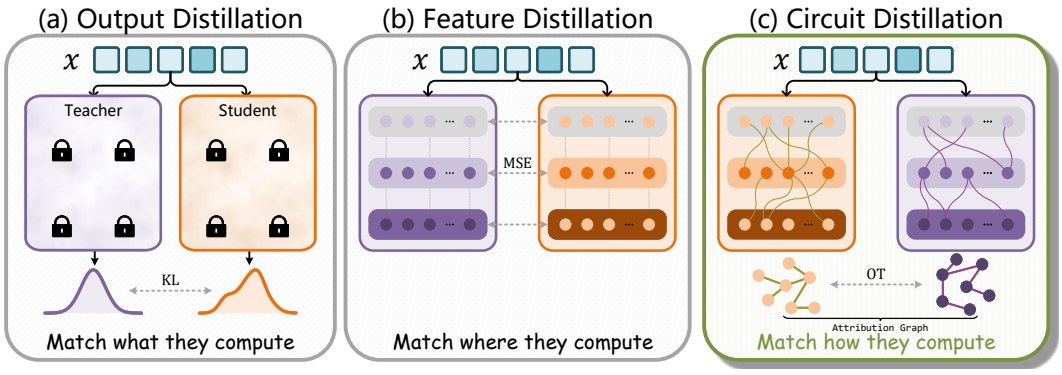

Figure 1: **Comparison of distillation paradigms.** (a) *Output distillation* matches teacher and student output distributions via KL divergence, transferring *what* models compute. (b) *Feature distillation* aligns intermediate representations via MSE, transferring *where* models compute. (c) *Circuit distillation* (ours) extracts attribution graphs that expose structured computational pathways and aligns them via optimal transport, transferring *how* models compute.

suggests a natural opportunity: rather than distilling outputs, we can distill the circuits themselves, transferring the algorithmic structure that produces them.

We propose **Meta Circuit Distillation (MCD)**, a framework that reframes distillation as the transfer of reasoning circuits, the structured computational pathways that define how a model solves a problem (Figure 1). Rather than treating models as black boxes, MCD leverages mechanistic interpretability to expose internal computation and explicitly trains the student to develop circuits that mirror those of the teacher. Where output distillation matches *what* models compute and feature distillation matches *where* they compute, MCD matches *how* they compute by aligning attribution graphs through optimal transport. The supervision question shifts from "does the student produce similar outputs?" to "does the student implement similar mechanisms?", a stronger constraint that should yield students whose internal reasoning is genuinely aligned with the teacher's.

Realizing this idea raises two technical challenges. **(C1) Tractable circuit representation.** How can we represent and compare circuits across models in a form suitable for gradient-based training? Existing circuit-discovery techniques in mechanistic interpretability target post-hoc analysis through manual or activation-patching procedures Conmy et al. [2023], Geiger et al. [2021]; they are neither efficient nor differentiable to be invoked at every training step. We need a representation that is automatically computable, structurally expressive, differentiable, and tractable at scale. **(C2) Cross-architecture circuit alignment.** How can we align circuits across models with different architectures, capacities, and feature representations? A teacher may implement a computation through a single rich circuit, while a capacity-constrained student may decompose the same computation across multiple simpler circuits. Direct one-to-one feature matching is therefore infeasible: circuits are relational structures defined by feature interactions, not by individual activations.

To address (C1), we introduce **transcoder-based attribution graphs** as an efficient and differentiable circuit representation. Transcoders, cross-layer sparse coding modules that generalize sparse autoencoders Huben et al. [2024], Bricken et al. [2023], decompose each model's dense activations into an interpretable feature dictionary. From them, we build attribution graphs whose nodes are sparse features and whose edges quantify causal influence between features. Each graph can be computed in a single forward–backward pass, making it suitable for use as a training signal at scale.

To address (C2), we apply **optimal-transport pathway matching**. Each attribution graph induces a probability distribution over computational pathways, and we align teacher and student via the Wasserstein distance between these distributions Villani [2009], Cuturi [2013]. This formulation naturally accommodates graphs of different sizes and topologies, and recovers a soft matching between circuits without requiring explicit feature correspondences.

Our contributions are threefold: (1) we propose *transcoder-based attribution graphs* as an efficient, differentiable circuit representation that can supervise training, not only post-hoc analysis; (2) we develop an *optimal-transport pathway-matching* loss that aligns circuit distributions across heterogeneous architectures without explicit feature correspondences, and we prove that minimizing this

loss provably bounds the discrepancy in MLP-level computation between teacher and student; (3) extensive experiments demonstrate that MCD consistently improves over a wide range of distillation baselines, and the gains generalize to mathematical reasoning tasks.

2 Related Work

Knowledge Distillation of Large Language Models. Knowledge distillation transfers capability from a large teacher to a compact student Hinton et al. [2015] and is now a cornerstone of efficient LLM deployment Fang et al. [2026]. Output-level methods refine the supervisory signal on the student's predictive distribution, for example through reverse KL Gu et al. [2024], implicit-reward objectives Li et al. [2024], adversarial teacher–student loops Jiang et al. [2023], and black-box-to-white-box belief alignment Hong et al. [2024]. A complementary line targets reasoning transfer via rationale supervision Hsieh et al. [2023], math-specific specialization Fu et al. [2023], adaptive pre/post-thinking Chen et al. [2025], step-wise retrieval-augmented supervision Lee et al. [2025], and code-reasoning scaling analysis He et al. [2025]. Further work pursues data efficiency or alternative paradigms, including zero-shot KD with adversarial out-of-domain data Rashid et al. [2021], counterfactual boundary mapping Hamman et al. [2025], actor–critic knowledge selection Wang et al. [2022], symbolic text distillation West et al. [2022], trajectory-level verbal feedback Xiong et al. [2026], cross-family instruction tuning Wu et al. [2024], cross-modal medical KD Ge et al. [2025], and unbiased KD for recommendation Zhang et al. [2026]. Despite this breadth, every method supervises *what* the student outputs without constraining *how* it computes internally, the gap MCD targets.

Mechanistic Interpretability and Circuit Discovery. Mechanistic interpretability seeks to reverse-engineer neural networks into interpretable components Olah et al. [2020], Elhage et al. [2021]. Foundational work has isolated transformer circuits (induction heads Minegishi et al. [2025], indirect-object identification Wang et al. [2023], and greater-than Hanna et al. [2023]), as well as motifs like successor heads for modular arithmetic Gould et al. [2024], copy suppression McDougall et al. [2023], and self-repair dynamics McGrath et al. [2023]. To scale beyond manual analysis, automated activation patching Conmy et al. [2023], complexity analyses Adolfi et al. [2025], and audits of subspace-method illusions Makelov et al. [2024] have emerged. A parallel thread decomposes representations via dictionary learning, from superposition in toy models Elhage et al. [2022] to large-scale sparse autoencoders Templeton et al. [2024] and ground-truth feature evaluations Karvonen et al. [2024]. Bridging the two threads, sparse feature circuits Marks et al. [2025] and sparse dictionaries for gradient-based discovery He et al. [2024] yield more concise circuits than neuron-level alternatives. Relevant to us, Dunefsky et al. [2024] introduce transcoders that approximate dense MLP layers with sparse, weights-analyzable alternatives. Whereas these works use such tools for post-hoc analysis, MCD repurposes transcoders and attribution graphs as a *differentiable training signal*, turning circuit-level structure into a supervisory objective for distillation.

3 Preliminaries

Given a teacher model π_T and a smaller student model π_S with learnable parameters θ, the goal of knowledge distillation is to train π_S to match π_T on a query distribution $\mathcal{Q}$. We adopt the following general objective:

$$\min_{\theta} \; \mathbb{E}_{x,\, y\sim p(\cdot|x)} \left[\frac{1}{|y|} \sum_{t=1}^{|y|} \mathcal{L}\Big(\pi_T^{(t)}, \pi_S^{\theta,(t)}\Big) \right], \tag{1}$$

where $\pi_T^{(t)} := \pi_T(\cdot \mid y_{<t}, x)$ and $\pi_S^{\theta,(t)} := \pi_S^{\theta}(\cdot \mid y_{<t}, x)$ are the teacher and student token-level distributions conditioned on prefix $y_{<t}$ and input x, $p(\cdot \mid x)$ is the sequence sampling distribution, and $\mathcal{L}(\cdot,\cdot)$ is a token-level loss. Two design choices, namely the data source $p(\cdot \mid x)$ and the loss $\mathcal{L}$, recover existing KD methods as special cases. Appendix B shows that several representative baselines are instantiations of Eq. (1). Crucially, all of them act at the output level: they supervise *what* the student emits but place no constraint on *how* it arrives at that output. MCD augments this objective with a circuit-level alignment term that supervises the student's internal computation (Section 4), leaving the underlying KD loss untouched.

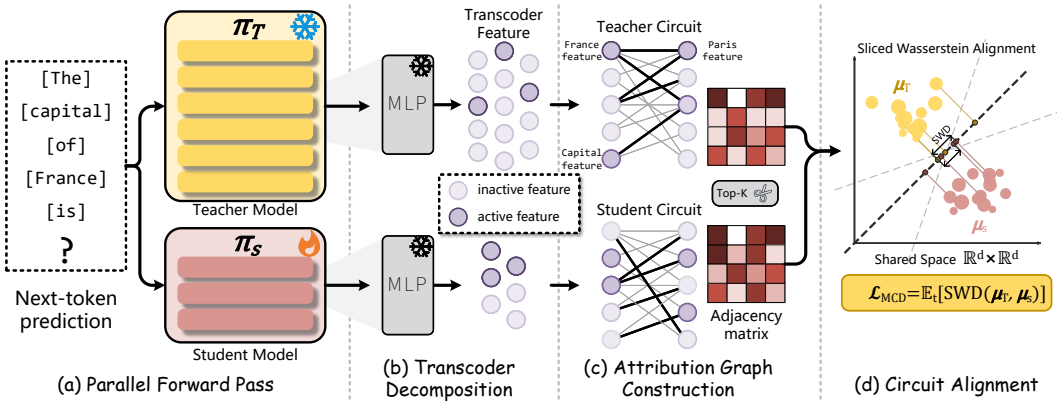

Figure 2: **Overview of MCD.** Transcoders decompose teacher and student MLPs into sparse features (b), from which top-K attribution graphs are built (c) and aligned via Sliced Wasserstein Distance (d), supervising *how* the student computes.

4 Methodology

Meta Circuit Distillation (MCD) addresses the two challenges identified in Section 1 with two coupled components: *transcoder-based attribution graphs* as an efficient differentiable circuit representation (Section 4.1), and *optimal-transport alignment* of circuit distributions across architectures (Section 4.2). Together they form an auxiliary loss that augments any standard KD objective with circuit-level supervision (Section 4.3). Figure 2 illustrates the pipeline.

4.1 Circuit Representation via Attribution Graphs

Operating at the circuit level requires a representation of internal computation that is interpretable, differentiable, and cheap to compute at every training step. We adopt a two-stage construction: transcoders decompose each model's dense activations into a sparse feature basis, and attribution graphs over these features expose the causal structure of information flow.

Transcoders as sparse feature dictionaries. We use a transcoder Dunefsky et al. [2024] as our circuit primitive. Given the input activation $\mathbf{x} \in \mathbb{R}^d$, the transcoder produces a sparse feature vector and reconstructs the sublayer output:

$$\mathbf{z}(\mathbf{x}) = \mathrm{ReLU}(W_{\mathrm{enc}}\,\mathbf{x} + \mathbf{b}_{\mathrm{enc}})\,, \tag{2}$$

$$\widehat{\mathrm{MLP}}(\mathbf{x}) = W_{\mathrm{dec}}\,\mathbf{z}(\mathbf{x}) + \mathbf{b}_{\mathrm{dec}}, \tag{3}$$

where $W_{\mathrm{enc}} \in \mathbb{R}^{m\times d}$, $W_{\mathrm{dec}} \in \mathbb{R}^{d\times m}$, d is model hidden dimension, and m is transcoder feature dimension (typically 2–16 $\times$ d). Each transcoder is trained by minimizing the reconstruction loss

$$\mathcal{L}_{\mathrm{TC}} = \left\|\mathrm{MLP}(\mathbf{x}) - \widehat{\mathrm{MLP}}(\mathbf{x})\right\|_2^2, \tag{4}$$

which omits the L_1 sparsity penalty of the original formulation Dunefsky et al. [2024]. We make this choice deliberately: because our goal is distillation fidelity rather than post-hoc interpretability, allocating the transcoder's full capacity to reconstruction tightens the bound established in Section 5. We train one transcoder per MLP sublayer for both π_T and π_S with the base model weights frozen. Teacher transcoders are trained once and held fixed; student transcoders are periodically retrained as the student evolves (Section 4.3).

Attribution graphs as a differentiable circuit signal. With transcoders in place, attribution graphs expose the causal structure of each model's computation. For transcoders at layers $\ell' < \ell$, Dunefsky et al. [2024] show that the causal influence of feature i at layer ℓ' on feature j at layer ℓ for token t factorizes into an input-dependent and an input-invariant term:

$$\mathrm{Attr}^{(ij)}_{\ell',\ell}[t] = \underbrace{z_i^{(\ell')}[t]}_{\text{input-dependent}} \times \underbrace{\left\langle \mathbf{w}^{(\ell')}_{\mathrm{dec},i},\ \mathbf{w}^{(\ell)}_{\mathrm{enc},j} \right\rangle}_{\text{input-invariant}}, \tag{5}$$

where $z_i^{(\ell')}[t]$ is the activation of feature i at token t (Eq. (2)), $\mathbf{w}_{\mathrm{dec},i}^{(\ell')}$ is the i-th column of W_{dec} at layer ℓ', and $\mathbf{w}_{\mathrm{enc},j}^{(\ell)}$ is the j-th row of W_{enc} at layer ℓ. Stacking over all feature pairs yields the attribution matrix $A_{\ell',\ell}^{(t)} \in \mathbb{R}^{m\times m}$, whose entry (i,j) measures how strongly feature i drives feature j for token t. Because inner products are input-invariant and can be precomputed once; the full attribution matrix for any new input requires only a single forward pass, making it efficient at training.

We instantiate attribution between the *first* ($\ell' = 0$) and *last* ($\ell = L$) MLP layers. Aligning intermediate layers by index would impose a false correspondence, since the same index plays a different role depending on total network depth. Layers 0 and L instead occupy functionally invariant positions: layer 0 is where token embeddings first enter MLP computation, and layer L is where the model assembles its final pre-output representation. This choice yields an architecture-agnostic summary of end-to-end information flow: $A_T^{(t)} := A_{0,L}^{(t)}\big|_{\pi_T}$, $\quad A_S^{(t)} := A_{0,L}^{(t)}\big|_{\pi_S}$. Whereas Dunefsky et al. [2024] use attribution graphs for post-hoc mechanistic analysis, MCD repurposes them as a differentiable training signal: minimizing the discrepancy between $A_T^{(t)}$ and $A_S^{(t)}$ directly drives the student to develop computational pathways that mirror the teacher's.

4.2 Circuit Alignment via Optimal Transport

Teacher and student feature spaces are unordered and non-corresponding, so pointwise losses such as ℓ_2 or KL divergence cannot be applied directly to $A_T^{(t)}$ and $A_S^{(t)}$. We instead cast circuit alignment, a form of cross-graph alignment Wei et al. [2024], as *distribution matching*: each attribution matrix becomes a discrete measure over computational pathways, and we compare the two measures via optimal transport, which naturally accommodates graphs of different sizes and topologies and requires no explicit feature correspondence.

Circuit distributions. We retain only the top-K edges of $A^{(t)}$ ranked by attribution magnitude, both because keeping all m^2 entries is impractical (the $\mathcal{O}(m^2)$ memory cost per token is prohibitive when $m \gg d$) and because near-zero edges contribute noise rather than signal. Let $\mathcal{K}^{(t)}$ denote the retained edge set. We then square the raw attribution scores to ensure non-negativity (raw scores can be negative due to inhibitory interactions) and to concentrate mass on dominant pathways; squaring is preferred over absolute values because it is everywhere differentiable. Finally we normalize:

$$\hat{A}_{ij}^{(t)} = \left[A_{ij}^{(t)}\right]^2 \cdot \mathbf{1}\Big[(i,j) \in \mathcal{K}^{(t)}\Big], \qquad \tilde{A}_{ij}^{(t)} = \frac{\hat{A}_{ij}^{(t)}}{\sum_{(i',j')\in\mathcal{K}^{(t)}} \hat{A}_{i'j'}^{(t)}}, \tag{6}$$

so that $\tilde{A}_{ij}^{(t)} \geq 0$ and $\sum_{(i,j)\in\mathcal{K}^{(t)}} \tilde{A}_{ij}^{(t)} = 1$. The weight $\tilde{A}_{ij}^{(t)}$ is the relative contribution of pathway (i,j) within the active circuit.

The design is to fix the transcoder feature dimension m to be *the same* for π_T and π_S, even when their hidden dimensions d_T and d_S differ. This makes the edge index space $\{1,\ldots,m\}^2$ shared across architectures: feature i at layer 0 and feature j at layer L refer to the same positional slot in both transcoders. We define the circuit distribution $\mu^{(t)}$ as a discrete probability measure on $\{1,\ldots,m\}^2$:

$$\mu^{(t)} = \sum_{(i,j)\in\mathcal{K}^{(t)}} \tilde{A}_{ij}^{(t)}\, \delta_{(i,j)}, \tag{7}$$

where $\delta_{(i,j)}$ is Dirac mass at (i,j). Because $\mu_T^{(t)}$ and $\mu_S^{(t)}$ is on same finite index space, optimal transport between them is well-posed without explicit feature correspondence between them.

Sliced Wasserstein alignment. Even on a shared index space, pointwise comparison remains inappropriate: $\mu_T^{(t)}$ and $\mu_S^{(t)}$ are typically supported on different subsets of edges. We therefore use the Sliced Wasserstein Distance (SWD) Kolouri et al. [2019], Cuturi [2013], which computes the expected one-dimensional Wasserstein distance over random linear projections of the support points:

$$\mathcal{L}_{\mathrm{MCD}} = \mathbb{E}_{t\in\mathcal{S}}\left[\mathrm{SWD}\left(\mu_T^{(t)},\, \mu_S^{(t)}\right)\right], \tag{8}$$

where $\mathcal{S}$ is the set of supervised token positions. SWD avoids explicit coupling: it reduces to sorting K projected values per random direction, giving $\mathcal{O}(P\cdot K\log K)$ complexity for P projections versus

$\mathcal{O}(K^3 \log K)$ for exact Wasserstein, and is compatible with mini-batch stochastic optimization. Because SWD operates on the geometric embedding of circuit edges rather than integer indices, it remains agnostic to architectural differences and handles circuits of different sizes and topologies.

4.3 Training Objective

$\mathcal{L}_{\mathrm{MCD}}$ is orthogonal to any base KD objective: it supervises *how* the student computes, while the base loss governs *what* the student outputs. The combined objective is

$$\mathcal{L}_{\mathrm{total}} = \lambda_{\mathrm{KD}}\, \mathcal{L}_{\mathrm{KD}}(\theta) + \lambda_{\mathrm{MCD}}\, \mathcal{L}_{\mathrm{MCD}}(\theta), \tag{9}$$

where $\mathcal{L}_{\mathrm{KD}}$ is any instantiation of Eq. (1), ranging from vanilla supervised fine-tuning and sequence-level distillation Hinton et al. [2015], Kim and Rush [2016] to imitation-based on-policy methods Lin et al. [2020], generalized on-policy distillation Agarwal et al. [2024], and loss-optimized variants Ko et al. [2024, 2025]. Adding $\mathcal{L}_{\mathrm{MCD}}$ with weight λ_{MCD} is the only required modification. Gradients flow through the chain $\mathcal{L}_{\mathrm{MCD}} \xrightarrow{\mathrm{SWD}} \mu_S^{(t)} \xrightarrow{(7)} \tilde{A}_S^{(t)} \xrightarrow{(5)} z_i^{(0)}[t] \xrightarrow{(2)} \mathbf{x} \to \theta_S$; each step is differentiable, so the student receives a gradient signal that explicitly encourages it to develop circuits structurally similar to the teacher's.

Periodic transcoder retraining. As distillation proceeds, the student's MLP activations shift, and its transcoders gradually drift away from the current student representations. To keep attribution graphs faithful to the evolving student, we freeze π_S^θ every Δ steps and retrain all student transcoders on the next Δ training samples via Eq. (4); teacher transcoders remain fixed throughout. The full procedure is given in Algorithm 1 of Appendix B.2.

5 Bounding Output Discrepancy via Circuit Alignment

Under mild regularity conditions, the Sliced Wasserstein distance between circuit distributions upper-bounds the gap in MLP-level computation between the two models, formally linking mechanistic alignment to functional fidelity.

Let π_T and π_S be the teacher and student, each equipped with transcoders at layers 0 and L (L may differ between models). For a token t, let $\mu_T^{(t)}$ and $\mu_S^{(t)}$ be the circuit distributions of Eq. (7), and define $\Delta\mathrm{MLP}_M^{(t)} := \mathrm{MLP}_M^{(L)}(\mathbf{x}_M^{(L)}[t]) - \mathrm{MLP}_M^{(0)}(\mathbf{x}_M^{(0)}[t])$ as the net MLP contribution from layer 0 to layer L for $M \in \{T, S\}$. We assume:

- (**A1**) **Transcoder fidelity.** For all tokens t, models $M \in \{T, S\}$, and relevant layers ℓ, $\|\mathrm{MLP}_M^{(\ell)}(\mathbf{x}) - \widehat{\mathrm{MLP}}_M^{(\ell)}(\mathbf{x})\|_2 \le \epsilon_{\mathrm{TC}}$.
- (**A2**) **Top-K coverage.** $\sum_{(i,j)\notin\mathcal{K}^{(t)}} [A_{ij}^{(t)}]^2 \le \epsilon_K \cdot \|A^{(t)}\|_F^2$.
- (**A3**) **Decoder Lipschitz regularity.** $\|W_{\mathrm{dec}}^{(\ell)}\|_{\mathrm{op}} \le B_{\mathrm{dec}}$ for all relevant ℓ.

Theorem 5.1 (Circuit Alignment Controls Output Discrepancy)**.** *Under (**A1**)–(**A3**), for any token t,*

$$\left\|\Delta\mathrm{MLP}_T^{(t)} - \Delta\mathrm{MLP}_S^{(t)}\right\|_2 \;\le\; B_{\mathrm{dec}}\sqrt{K}\left(\|A_T^{(t)}\|_F + \|A_S^{(t)}\|_F\right)\mathrm{SWD}\Big(\mu_T^{(t)},\, \mu_S^{(t)}\Big) \;+\; R(\epsilon_{\mathrm{TC}},\, \epsilon_K), \tag{10}$$

where $R(\epsilon_{\mathrm{TC}}, \epsilon_K) = 4\,\epsilon_{\mathrm{TC}} + 2\,B_{\mathrm{dec}}\sqrt{\epsilon_K}\,\max_M \|A_M^{(t)}\|_F$ vanishes as transcoder fidelity and top-K coverage improve.

Proof. By the triangle inequality and (**A1**), replacing each MLP sublayer with its transcoder approximation incurs at most ϵ_{TC} error per sublayer. Since $\Delta\mathrm{MLP}_M^{(t)}$ involves two sublayers ($\ell \in \{0, L\}$), the total error is at most $2\epsilon_{\mathrm{TC}}$ per model and $4\epsilon_{\mathrm{TC}}$ across the two:

$$\left\|\Delta\mathrm{MLP}_T^{(t)} - \Delta\mathrm{MLP}_S^{(t)}\right\|_2 \le \left\|\Delta\widehat{\mathrm{MLP}}_T^{(t)} - \Delta\widehat{\mathrm{MLP}}_S^{(t)}\right\|_2 + 4\,\epsilon_{\mathrm{TC}}. \tag{11}$$

Using Eq. (3) and the attribution factorization Eq. (5), the end-to-end transcoder contribution decomposes into a sum over feature pairs (i, j) weighted by attribution scores. Splitting into retained edges $\mathcal{K}^{(t)}$ and discarded edges and applying (**A2**) gives, for each model,

$$\left\|\Delta\widehat{\mathrm{MLP}}_M^{(t)} - \Delta\widehat{\mathrm{MLP}}_{M,\mathcal{K}}^{(t)}\right\|_2 \le B_{\mathrm{dec}}\sqrt{\epsilon_K}\,\|A_M^{(t)}\|_F, \tag{12}$$

where $\widehat{\Delta\mathrm{MLP}}^{(t)}_{M,\mathcal{K}}$ is the contribution from retained edges only (full derivation in Appendix A.1). Summing across both models yields $2B_{\mathrm{dec}}\sqrt{\epsilon_K}\max_M \|A^{(t)}_M\|_F$.

It remains to bound $\|\widehat{\Delta\mathrm{MLP}}^{(t)}_{T,\mathcal{K}} - \widehat{\Delta\mathrm{MLP}}^{(t)}_{S,\mathcal{K}}\|_2$. The retained-edge contribution $\widehat{\Delta\mathrm{MLP}}^{(t)}_{M,\mathcal{K}} = \sum_{(i,j)\in\mathcal{K}^{(t)}_M} A^{(t)}_{M,ij}\,\mathbf{w}^{(L)}_{\mathrm{dec},j}$ is a linear functional evaluated against $\mu^{(t)}_M$ via the normalized weights $\tilde{A}$ (Eq. (6)) and circuit measure (Eq. (7)). Kantorovich–Rubinstein duality combined with the projection characterization of SWD gives, for any 1-Lipschitz linear functional f,

$$\left|\int f\,\mathrm{d}\mu^{(t)}_T - \int f\,\mathrm{d}\mu^{(t)}_S\right| \le \mathrm{SWD}\Big(\mu^{(t)}_T,\,\mu^{(t)}_S\Big)\,. \tag{13}$$

The relevant functional, projection onto the output space via decoder weights, has Lipschitz constant at most B_{dec}. Rescaling and applying Cauchy–Schwarz over the K support points gives

$$\big\|\widehat{\Delta\mathrm{MLP}}^{(t)}_{T,\mathcal{K}} - \widehat{\Delta\mathrm{MLP}}^{(t)}_{S,\mathcal{K}}\big\|_2 \le B_{\mathrm{dec}}\sqrt{K}\,\big(\|A^{(t)}_T\|_F + \|A^{(t)}_S\|_F\big)\,\mathrm{SWD}\Big(\mu^{(t)}_T,\,\mu^{(t)}_S\Big)\,. \tag{14}$$

Combining Eqs. (11), (12), and (14) yields the claimed bound. □

The bound decomposes into a *circuit-alignment term* proportional to $\mathrm{SWD}(\mu^{(t)}_T, \mu^{(t)}_S)$, the quantity directly minimized by $\mathcal{L}_{\mathrm{MCD}}$, and a *residual* $R(\epsilon_{\mathrm{TC}}, \epsilon_K)$ that shrinks with better transcoders and a larger top-K budget. This decomposition retroactively justifies two design choices: high-fidelity transcoders (motivating the omission of L_1 sparsity) and a sufficiently large K are not heuristics but conditions that tighten the bound. Theorem 5.1 thus formalizes the central claim that MCD transfers *how* models compute, not only *what* they output.

6 Experiments

We evaluate MCD on two teacher–student pairs from architecturally distinct families: Qwen3-8B→Qwen3-1.7B and Gemma3-4B-it→Gemma3-1B. Instruction-following is measured on AlpacaEval, Evol-Instruct, and UltraFeedback under an LLM-as-a-Judge protocol; mathematical reasoning on GSM8K (in-distribution) and MATH 500 (out-of-distribution). **Baselines.** We compare against six methods spanning token-, sequence-, and on-policy distillation: Vanilla SFT, SeqKD Kim and Rush [2016], ImitKD Lin et al. [2020], GKD Agarwal et al. [2024], DistiLLM Ko et al. [2024], and DistiLLM2 Ko et al. [2025]. Each *+MCD* variant appends $\mathcal{L}_{\mathrm{MCD}}$ to the original objective with all other settings fixed. Full details are in Appendix C.

6.1 Main Results

Instruction following. Table 1 reports win rates and Gain↑ on both model pairs. All Gain↑ entries are positive, with the largest normalized gains reaching 23.26%/22.56% on Gemma3 and 10.11% on Qwen3; the corresponding Base→+MCD win-rate gains range from $+0.28$ to $+4.09$ points. Averaged across baselines, MCD improves AlpacaEval by $+2.10$/$+1.51$ points (Qwen3/Gemma3), Evol-Instruct by $+1.64$/$+1.44$ points, and UltraFeedback by $+0.76$/$+0.70$ points, showing compatibility with diverse objectives and consistent gains across model families.

Generalization to mathematical reasoning. Table 2 shows positive MCD gains beyond instruction following. On GSM8K, MCD improves SFT/SeqKD by $+0.61$/$+0.66$ points with Gain↑ of 2.69%/3.07%; on MATH 500, it improves them by $+0.30$/$+0.43$ points with Gain↑ of 1.36%/2.01%. These results indicate that the aligned circuits deliver gains across task domains.

6.2 Analysis: Dynamics, Ablations, and Sensitivity

Training dynamics. Figure 3 reports training dynamics on Qwen3-8B→Qwen3-1.7B under SeqKD. SeqKD+MCD tracks the SeqKD total loss closely (a), so the auxiliary objective does not destabilize training. $\mathcal{L}_{\mathrm{MCD}}$ exhibits a sawtooth (b): it decreases within each cycle and rebounds at every transcoder refresh as updated transcoders expose new discrepancies. The teacher and student attribution-graph distributions, initially dissimilar, converge after MCD training (c), evidence that the student internalizes *how* the teacher reasons.

Table 1: **Instruction-following win rates (%) across two model families.** Gain $\uparrow = 100 \times \frac{(\text{Base+MCD}-\text{Student})-(\text{Base}-\text{Student})}{\text{Base}-\text{Student}}$, where Base denotes the corresponding baseline method, such as Vanilla SFT. It measures MCD's extra improvement relative to the baseline's own gain.

Method	Qwen3-8B → Qwen3-1.7B			Gemma3-4B-it → Gemma3-1B		
	AlpacaEval	Evol-Instruct	UltraFeed.	AlpacaEval	Evol-Instruct	UltraFeed.
Teacher	$74.44_{\pm 0.10}$	$74.19_{\pm 0.70}$	$69.88_{\pm 0.06}$	$80.83_{\pm 0.18}$	$75.61_{\pm 0.09}$	$65.90_{\pm 0.04}$
Student	$4.74_{\pm 0.17}$	$7.03_{\pm 0.41}$	$15.00_{\pm 0.14}$	$8.80_{\pm 0.12}$	$0.80_{\pm 0.40}$	$2.07_{\pm 0.09}$
Vanilla SFT	$42.07_{\pm 0.08}$	$52.12_{\pm 0.10}$	$46.40_{\pm 0.23}$	$18.24_{\pm 0.10}$	$12.13_{\pm 0.17}$	$15.14_{\pm 0.10}$
+ MCD	$44.91_{\pm 0.17}$	$53.91_{\pm 0.23}$	$47.61_{\pm 0.47}$	$20.37_{\pm 0.10}$	$14.10_{\pm 0.41}$	$16.17_{\pm 0.13}$
Gain↑	7.61%	3.97%	3.85%	22.56%	17.39%	7.88%
SeqKD	$45.43_{\pm 0.23}$	$54.97_{\pm 1.10}$	$48.16_{\pm 0.24}$	$19.44_{\pm 0.34}$	$14.68_{\pm 0.18}$	$18.38_{\pm 0.06}$
+ MCD	$46.10_{\pm 0.29}$	$56.14_{\pm 0.42}$	$48.85_{\pm 0.06}$	$20.31_{\pm 0.71}$	$15.50_{\pm 0.04}$	$19.04_{\pm 0.17}$
Gain↑	1.65%	2.44%	2.08%	8.18%	5.91%	4.05%
ImitKD	$41.82_{\pm 0.73}$	$53.45_{\pm 0.73}$	$46.61_{\pm 0.14}$	$15.08_{\pm 0.26}$	$11.42_{\pm 0.30}$	$13.98_{\pm 0.15}$
+ MCD	$45.57_{\pm 0.59}$	$57.54_{\pm 1.02}$	$47.70_{\pm 0.31}$	$16.11_{\pm 0.39}$	$13.89_{\pm 0.42}$	$14.69_{\pm 0.09}$
Gain↑	10.11%	8.81%	3.45%	16.40%	23.26%	5.96%
GKD	$43.26_{\pm 0.14}$	$54.81_{\pm 0.49}$	$47.08_{\pm 0.18}$	$20.51_{\pm 0.32}$	$13.63_{\pm 0.34}$	$17.73_{\pm 0.25}$
+ MCD	$45.00_{\pm 0.24}$	$55.84_{\pm 1.16}$	$48.03_{\pm 0.17}$	$21.98_{\pm 0.28}$	$14.54_{\pm 0.41}$	$18.73_{\pm 0.22}$
Gain↑	4.52%	2.16%	2.96%	12.55%	7.09%	6.39%
DistiLLM	$43.78_{\pm 0.42}$	$53.15_{\pm 0.76}$	$47.96_{\pm 0.35}$	$19.75_{\pm 0.23}$	$15.07_{\pm 0.38}$	$16.75_{\pm 0.15}$
+ MCD	$45.88_{\pm 0.51}$	$54.12_{\pm 0.56}$	$48.30_{\pm 0.21}$	$21.68_{\pm 0.20}$	$16.33_{\pm 0.23}$	$17.15_{\pm 0.15}$
Gain↑	5.38%	2.10%	1.03%	17.63%	8.83%	2.72%
DistiLLM2	$43.51_{\pm 0.29}$	$52.87_{\pm 0.35}$	$47.18_{\pm 0.30}$	$17.43_{\pm 0.14}$	$12.94_{\pm 0.12}$	$15.29_{\pm 0.14}$
+ MCD	$45.04_{\pm 0.23}$	$53.65_{\pm 0.28}$	$47.46_{\pm 0.23}$	$19.08_{\pm 0.21}$	$14.17_{\pm 0.26}$	$15.70_{\pm 0.19}$
Gain↑	3.95%	1.70%	0.87%	19.12%	10.13%	3.10%

Table 2: Mathematical reasoning accuracy (%) on Qwen3-8B→Qwen3-1.7B.

Dataset	Reference		SFT			SeqKD		
	$\mathcal{M}_T$ (8B)	$\mathcal{M}_S$ (1.7B)	Base	+ MCD	Gain↑	Base	+ MCD	Gain↑
GSM8K	95.83	59.69	82.33	82.94	2.69%	81.22	81.88	3.07%
MATH 500	93.20	41.07	63.20	63.50	1.36%	62.47	62.90	2.01%

Ablation study. We ablate three mechanisms of MCD on Qwen3-8B→Qwen3-1.7B-Base under both Vanilla SFT and SeqKD: (i) replacing OT with MSE, (ii) disabling periodic transcoder retraining, and (iii) disabling top-K sparsification. Table 3 reports win rates (%) on AlpacaEval (AE), Evol-Instruct (EI), and UltraFeedback (UF); Figure 5 visualizes the ablation results across both branches. *OT alignment and periodic retraining are dominant.* Replacing OT

Table 3: Ablation on Qwen3-8B→Qwen3-1.7B-Base. Win rates (%) on AE/EI/UF.

Bench.	Base	+Full	w/o top-K	w/o retrain.	MSE
Reference: Teacher AE/EI/UF = 74.44/74.19/69.88; Student = 4.74/7.03/15.00					
Vanilla SFT branch					
AE	42.07	**44.91**	44.22	40.45	41.42
EI	52.12	**53.91**	53.21	50.82	49.63
UF	46.40	**47.61**	47.17	43.82	44.17
SeqKD branch					
AE	45.43	**46.10**	45.62	42.97	42.54
EI	54.97	**56.14**	55.47	53.84	50.29
UF	48.16	**48.85**	48.06	47.65	46.97

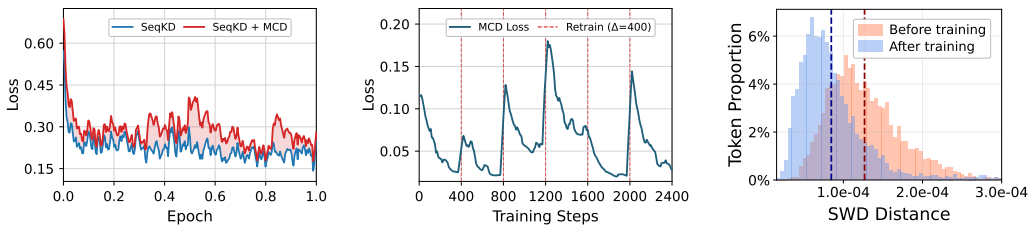

Figure 3: Training dynamics on Qwen3-8B→Qwen3-1.7B under SeqKD. **Left**: Total loss for SeqKD and SeqKD+MCD. **Middle** $\mathcal{L}_{\text{MCD}}$ across retraining cycles. **Right**: Attribution-graph distributions before/after SeqKD+MCD, over 50,000 AlpacaEval tokens.

with MSE drops AE/EI/UF by 3.49/4.28/3.44 (SFT) and 3.56/5.85/1.88 (SeqKD); disabling retraining causes comparable losses ($-4.46/-3.09/-3.79$ and $-3.13/-2.30/-1.20$). Both variants underperform their base objectives on at least two benchmarks, identifying geometric alignment and fresh transcoders as core drivers rather than refinements. *Top-K sparsification refines rather than drives alignment.* Disabling it produces a smaller but consistent decline (mean -0.61 for SFT and -0.65 for SeqKD), indicating that sparsification sharpens the signal once OT and retraining establish correspondence. The full-MCD gap over base ($+2.84/+1.79/+1.21$ for SFT; $+0.67/+1.17/+0.69$ for SeqKD) collapses whenever any single mechanism is removed, so the three components are jointly supportive rather than substitutable.

Hyperparameter sensitivity. We probe sensitivity to the loss weight λ_{MCD}, top-K budget K, and retraining interval Δ on Qwen3-8B→1.7B, varying one parameter at a time with others at default; win rates are averaged over the three benchmarks (configurations in Appendix E). MCD is robust across a wide operating range. Performance peaks at $\lambda_{\text{MCD}} \in [10^4, 10^5]$ and degrades at the extremes ($\lambda_{\text{MCD}} = 1$ recovers SFT; 10^6 overwhelms imitation). $K = 32$ outperforms sparser and denser choices, with all-edges lowering the average by 1.5–2.0 points, consistent with low-attribution noise. Omitting retraining drops the average by 3–4 points, while any finite $\Delta \in [200, 2000]$ remains competitive ($\Delta = 400$ optimal).

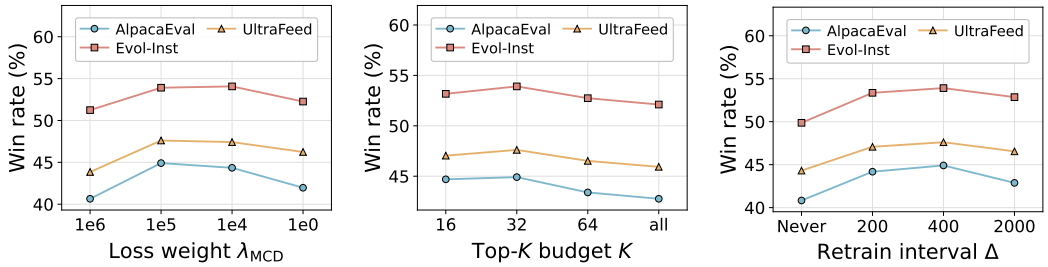

Figure 4: Sensitivity of λ_{MCD} (left), K (middle), and Δ (right) on Qwen3-8B→Qwen3-1.7B.

6.3 Plug-in Gains Across Student Sizes

Fixing the teacher at Qwen3-8B, we evaluate Vanilla SFT and SeqKD with and without MCD on Qwen3-0.6B, 1.7B, and 4B (Figure 6; full results in Appendix F). Averaged over AlpacaEval, Evol-Instruct, and UltraFeedback, MCD improves SFT by 1.82/1.95/1.67 percentage points and SeqKD by 1.44/0.84/1.78 points at 0.6B/1.7B/4B, respectively. These results show that MCD consistently improves average performance across all evaluated student sizes under both SFT and SeqKD.

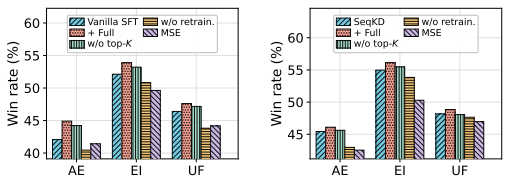

Figure 5: Ablation results on Qwen3-8B→Qwen3-1.7B. **Left**: Vanilla SFT branch. **Right**: SeqKD branch.

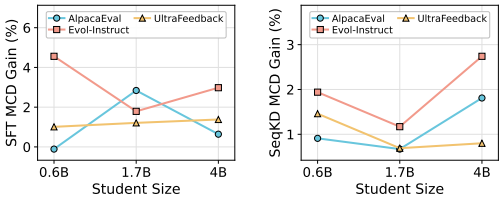

Figure 6: MCD plug-in gains across student sizes on Qwen3 family. **Left**: Gain of Vanilla SFT. **Right**: Gain of SeqKD.

6.4 Comparison with Feature-Based KD Baselines

To verify whether MCD also improves feature-based KD methods, we apply it to FitNet-style KD [Romero et al., 2015] and MiniLMv2-style KD on Qwen3-8B→Qwen3-1.7B using the original 20k-sample distillation pool for two epochs. Table 4 reports win rates and Gain↑, following the definition in Table 1. MCD improves all three benchmarks under both methods, with Gain↑ of 2.38%–3.02% for FitNet-style KD and 0.92%–7.64% for MiniLMv2-style KD. MCD thus remains beneficial when the base objective already includes representation or relation alignment. Baseline definitions and references are given in Appendix C.1.

6.5 Comparison with Iso-Compute Baselines

To determine whether the observed gains stem from longer training rather than MCD itself, we train the SFT and MiniLMv2-style KD baselines for additional epochs to approximately match the compute budgets of their MCD-enhanced counterparts on Qwen3-8B→Qwen3-1.7B. In preliminary controls, repeatedly training on the original fixed 20k-sample pool led to substantial degradation, suggesting overfitting or data exhaustion. We therefore expand the pool to 80k UltraChat200k samples to increase data diversity and assess the benefit of additional compute under a less data-limited setting. All variants use the same expanded pool and evaluation protocol. Standard and MCD-augmented runs use two epochs, while the extended SFT and MiniLMv2 controls use twelve and eight epochs, respectively. Table 5 reports epochs, GPU·h, and win rates. SFT+MCD exceeds twelve-epoch SFT by 3.10/1.99/1.46 points with 2.8% more compute. MiniLMv2-style+MCD exceeds its eight-epoch control by 2.12/1.46/3.28 points while using 4.7% less compute. The budgets are approximately matched rather than identical. With the expanded pool, the extended baselines largely saturate, while adding MCD still improves all three scores. These results support gains beyond allocating a larger training budget to either output-level or feature-based KD in this setting.

Table 4: Comparison of feature-based KD baselines and their MCD variants.

Method	AE	EI	UF
FitNet	43.16	53.08	46.94
+ MCD	44.32	54.40	47.70
Gain↑	3.02%	2.87%	2.38%
MiniLMv2	44.02	53.74	47.48
+ MCD	45.79	54.17	49.96
Gain↑	4.51%	0.92%	7.64%

Table 5: Iso-compute results. GPU·h includes all additional overhead introduced by MCD.

Method	Epochs	GPU·h	AE	EI	UF
SFT	2	21.6	44.73	52.13	47.11
+ Iso-Compute	12	129.6	44.76	52.38	47.08
+ MCD	2	133.2	**47.86**	**54.37**	**48.54**
MiniLMv2	2	40.8	46.39	54.12	47.67
+ Iso-Compute	8	163.2	46.57	54.22	47.61
+ MCD	2	155.6	**48.69**	**55.68**	**50.89**

7 Conclusion and Limitations

We presented **Meta Circuit Distillation (MCD)**, which reframes knowledge distillation as the transfer of reasoning circuits by coupling transcoder-based attribution graphs with an optimal-transport pathway-matching loss that provably bounds the teacher–student discrepancy in MLP-level computation. Across two model families, six standard distillation objectives, and three instruction-following benchmarks, MCD delivers consistent gains that further transfer to mathematical reasoning, with average performance gains across all evaluated student sizes.

Two scope boundaries remain. Our attribution graphs cover only MLP sublayers between the first and last layers, leaving attention circuits and intermediate pathways for future extensions; and periodic transcoder retraining, while keeping attribution faithful, introduces an overhead that online or partial-refresh schemes could amortize at larger scales.

Acknowledgments and Disclosure of Funding

This research was supported by the National Key R&D Program of China (No. 2025YFC3309503), NSFC (No. 62506366), the CIPS-SMP-Zhipu Large Model Fund, the Fundamental Research Funds for the Central Universities, and the Research Funds of Renmin University of China. It was also supported by Big Data and Responsible Artificial Intelligence for National Governance, Renmin University of China, and by the Public Computing Cloud, Renmin University of China.

References

F. Adolfi, M. G. Vilas, and T. Wareham. The computational complexity of circuit discovery for inner interpretability. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL `https://openreview.net/forum?id=QogcGNXJVw`.

R. Agarwal, N. Vieillard, Y. Zhou, P. Stanczyk, S. R. Garea, M. Geist, and O. Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth*

International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024. OpenReview.net, 2024. URL https://openreview.net/forum?id=3zKtaqxLhW.

N. Bonnotte. *Unidimensional and Evolution Methods for Optimal Transportation*. Phd thesis, Université Paris-Sud 11 and Scuola Normale Superiore, 2013. URL https://cvgmt.sns.it/paper/2341/.

T. Bricken, A. Templeton, J. Batson, B. Chen, A. Jermyn, T. Conerly, N. Turner, C. Anil, C. Denison, A. Askell, R. Lasenby, Y. Wu, S. Kravec, N. Schiefer, T. Maxwell, N. Joseph, Z. Hatfield-Dodds, A. Tamkin, K. Nguyen, B. McLean, J. E. Burke, T. Hume, S. Carter, T. Henighan, and C. Olah. Towards monosemanticity: Decomposing language models with dictionary learning. Transformer Circuits Thread, Oct. 2023. URL https://transformer-circuits.pub/2023/monosemantic-features/. Anthropic.

X. Chen, S. Zhou, K. Liang, and X. Liu. Distilling reasoning ability from large language models with adaptive thinking. *IEEE Trans. Neural Networks Learn. Syst.*, 36(11):19820–19833, 2025. doi: 10.1109/TNNLS.2025.3591266. URL https://doi.org/10.1109/TNNLS.2025.3591266.

K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. URL https://arxiv.org/abs/2110.14168.

A. Conmy, A. N. Mavor-Parker, A. Lynch, S. Heimersheim, and A. Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/34e1dbe95d34d7ebaf99b9bcaeb5b2be-Abstract-Conference.html.

G. Cui, L. Yuan, N. Ding, G. Yao, B. He, W. Zhu, Y. Ni, G. Xie, R. Xie, Y. Lin, Z. Liu, and M. Sun. ULTRAFEEDBACK: boosting language models with scaled AI feedback. In R. Salakhutdinov, Z. Kolter, K. A. Heller, A. Weller, N. Oliver, J. Scarlett, and F. Berkenkamp, editors, *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pages 9722–9744. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/cui24f.html.

M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2292–2300, 2013. URL https://proceedings.neurips.cc/paper/2013/hash/af21d0c97db2e27e13572cbf59eb343d-Abstract.html.

T. Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=mZn2Xyh9Ec.

N. Ding, Y. Chen, B. Xu, Y. Qin, S. Hu, Z. Liu, M. Sun, and B. Zhou. Enhancing chat language models by scaling high-quality instructional conversations. In H. Bouamor, J. Pino, and K. Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 3029–3051. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.183. URL https://doi.org/10.18653/v1/2023.emnlp-main.183.

Y. Dubois, X. Li, R. Taori, T. Zhang, I. Gulrajani, J. Ba, C. Guestrin, P. Liang, and T. B. Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/5fc47800ee5b30b8777fdd30abcaaf3b-Abstract-Conference.html.

J. Dunefsky, P. Chlenski, and N. Nanda. Transcoders find interpretable LLM feature circuits. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/2b8f4db0464cc5b6e9d5e6bea4b9f308-Abstract-Conference.html.

N. Elhage, N. Nanda, C. Olsson, T. Henighan, N. Joseph, B. Mann, A. Askell, Y. Bai, A. Chen, T. Conerly, N. DasSarma, D. Drain, D. Ganguli, Z. Hatfield-Dodds, D. Hernandez, A. Jones, J. Kernion, L. Lovitt, K. Ndousse, D. Amodei, T. Brown, J. Clark, J. Kaplan, S. McCandlish, and C. Olah. A mathematical framework for Transformer circuits. Transformer Circuits Thread, Dec. 2021. URL https://transformer-circuits.pub/2021/framework/index.html. Anthropic.

N. Elhage, T. Hume, C. Olsson, N. Schiefer, T. Henighan, S. Kravec, Z. Hatfield-Dodds, R. Lasenby, D. Drain, C. Chen, R. B. Grosse, S. McCandlish, J. Kaplan, D. Amodei, M. Wattenberg, and C. Olah. Toy models of superposition. *CoRR*, abs/2209.10652, 2022. doi: 10.48550/ARXIV.2209.10652. URL https://doi.org/10.48550/arXiv.2209.10652.

L. Fang, X. Yu, J. Cai, Y. Chen, S. Wu, Z. Liu, Z. Yang, H. Lu, X. Gong, Y. Liu, T. Ma, W. Ruan, A. Abbasi, J. Zhang, T. Wang, E. Latif, W. Liu, W. Zhang, S. Kolouri, X. Zhai, D. Zhu, W. Zhong, T. Liu, and P. Ma. Knowledge distillation and dataset distillation of large language models: emerging trends, challenges, and future directions. *Artif. Intell. Rev.*, 59(1):17, 2026. doi: 10.1007/S10462-025-11423-3. URL https://doi.org/10.1007/s10462-025-11423-3.

Y. Fu, H. Peng, L. Ou, A. Sabharwal, and T. Khot. Specializing smaller language models towards multi-step reasoning. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, Proceedings of Machine Learning Research, pages 10421–10430. PMLR, 2023. URL https://proceedings.mlr.press/v202/fu23d.html.

H. Ge, L. Hao, Z. Xu, Z. Lin, B. Li, S. Zhou, H. Zhao, and Y. Liu. Clinkd: Cross-modal clinic knowledge distiller for multi-task medical images. *CoRR*, abs/2502.05928, 2025. doi: 10.48550/ARXIV.2502.05928. URL https://doi.org/10.48550/arXiv.2502.05928.

A. Geiger, H. Lu, T. Icard, and C. Potts. Causal abstractions of neural networks. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 9574–9586, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/4f5c422f4d49a5a807eda27434231040-Abstract.html.

R. Gould, E. Ong, G. Ogden, and A. Conmy. Successor heads: Recurring, interpretable attention heads in the wild. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=kvcbV8KQsi.

Y. Gu, L. Dong, F. Wei, and M. Huang. Minillm: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=5h0qf7IBZZ.

F. Hamman, P. Dissanayake, Y. Fu, and S. Dutta. Few-shot knowledge distillation of llms with counterfactual explanations. *CoRR*, abs/2510.21631, 2025. doi: 10.48550/ARXIV.2510.21631. URL https://doi.org/10.48550/arXiv.2510.21631.

M. Hanna, O. Liu, and A. Variengien. How does GPT-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/efbba7719cc5172d175240f24be11280-Abstract-Conference.html.

M. He, M. A. Shafique, A. Kumar, T. Mackey, and N. Rajani. The valley of code reasoning: Scaling knowledge distillation of large language models. *CoRR*, abs/2510.06101, 2025. doi: 10.48550/ARXIV.2510.06101. URL https://doi.org/10.48550/arXiv.2510.06101.

Z. He, X. Ge, Q. Tang, T. Sun, Q. Cheng, and X. Qiu. Dictionary learning improves patch-free circuit discovery in mechanistic interpretability: A case study on othello-gpt. *CoRR*, abs/2402.12201, 2024. doi: 10.48550/ARXIV.2402.12201. URL https://doi.org/10.48550/arXiv.2402.12201.

D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt. Measuring mathematical problem solving with the MATH dataset. In J. Vanschoren and S. Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

G. E. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. URL http://arxiv.org/abs/1503.02531.

J. Hong, Q. Tu, C. Chen, G. Xing, J. Zhang, and R. Yan. Cyclealign: Iterative distillation from black-box LLM to white-box models for better human alignment. In L. Ku, A. Martins, and V. Srikumar, editors, *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, Findings of ACL, pages 14596–14609. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.FINDINGS-ACL.869. URL https://doi.org/10.18653/v1/2024.findings-acl.869.

C. Hsieh, C. Li, C. Yeh, H. Nakhost, Y. Fujii, A. Ratner, R. Krishna, C. Lee, and T. Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In A. Rogers, J. L. Boyd-Graber, and N. Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, Findings of ACL, pages 8003–8017. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-ACL.507. URL https://doi.org/10.18653/v1/2023.findings-acl.507.

R. Huben, H. Cunningham, L. R. Smith, A. Ewart, and L. Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=F76bwRSLeK.

Y. Jiang, C. Chan, M. Chen, and W. Wang. Lion: Adversarial distillation of proprietary large language models. In H. Bouamor, J. Pino, and K. Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 3134–3154. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.189. URL https://doi.org/10.18653/v1/2023.emnlp-main.189.

A. Karvonen, B. Wright, C. Rager, R. Angell, J. Brinkmann, L. Smith, C. M. Verdun, D. Bau, and S. Marks. Measuring progress in dictionary learning for language model interpretability with board game models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/9736acf007760cc2b47948ae3cf06274-Abstract-Conference.html.

Y. Kim and A. M. Rush. Sequence-level knowledge distillation. In J. Su, X. Carreras, and K. Duh, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1317–1327. The Association for Computational Linguistics, 2016. doi: 10.18653/V1/D16-1139. URL https://doi.org/10.18653/v1/d16-1139.

J. Ko, S. Kim, T. Chen, and S. Yun. Distillm: Towards streamlined distillation for large language models. In R. Salakhutdinov, Z. Kolter, K. A. Heller, A. Weller, N. Oliver, J. Scarlett, and F. Berkenkamp, editors, *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, Proceedings of Machine Learning Research, pages 24872–24895. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/ko24c.html.

J. Ko, T. Chen, S. Kim, T. Ding, L. Liang, I. Zharkov, and S. Yun. Distillm-2: A contrastive approach boosts the distillation of llms. In A. Singh, M. Fazel, D. Hsu, S. Lacoste-Julien, F. Berkenkamp, T. Maharaj, K. Wagstaff, and J. Zhu, editors, *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*, pages 31044–31062. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/ko25a.html.

S. Kolouri, K. Nadjahi, U. Simsekli, R. Badeau, and G. K. Rohde. Generalized sliced wasserstein distances. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 261–272, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/f0935e4cd5920aa6c7c996a5ee53a70f-Abstract.html.

K. Lee, M. Jeon, S. Jang, and H. Yu. Steper: Step-wise knowledge distillation for enhancing reasoning ability in multi-step retrieval-augmented language models. In C. Christodoulopoulos, T. Chakraborty, C. Rose, and V. Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 29501–29523. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.1500. URL https://doi.org/10.18653/v1/2025.emnlp-main.1500.

Y. Li, Y. Gu, L. Dong, D. Wang, Y. Cheng, and F. Wei. Direct preference knowledge distillation for large language models. *CoRR*, abs/2406.19774, 2024. doi: 10.48550/ARXIV.2406.19774. URL https://doi.org/10.48550/arXiv.2406.19774.

A. Lin, J. Wohlwend, H. Chen, and T. Lei. Autoregressive knowledge distillation through imitation learning. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6121–6133. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.EMNLP-MAIN.494. URL https://doi.org/10.18653/v1/2020.emnlp-main.494.

A. Makelov, G. Lange, A. Geiger, and N. Nanda. Is this the subspace you are looking for? an interpretability illusion for subspace activation patching. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=Ebt7JgMHv1.

S. Marks, C. Rager, E. J. Michaud, Y. Belinkov, D. Bau, and A. Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=I4e82CIDxv.

C. McDougall, A. Conmy, C. Rushing, T. McGrath, and N. Nanda. Copy suppression: Comprehensively understanding an attention head. *CoRR*, abs/2310.04625, 2023. doi: 10.48550/ARXIV.2310.04625. URL https://doi.org/10.48550/arXiv.2310.04625.

T. McGrath, M. Rahtz, J. Kramár, V. Mikulik, and S. Legg. The hydra effect: Emergent self-repair in language model computations. *CoRR*, abs/2307.15771, 2023. doi: 10.48550/ARXIV.2307.15771. URL https://doi.org/10.48550/arXiv.2307.15771.

G. Minegishi, H. Furuta, S. Taniguchi, Y. Iwasawa, and Y. Matsuo. Beyond induction heads: In-context meta learning induces multi-phase circuit emergence. In A. Singh, M. Fazel, D. Hsu, S. Lacoste-Julien, F. Berkenkamp, T. Maharaj, K. Wagstaff, and J. Zhu, editors, *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*, pages 44372–44395. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/minegishi25a.html.

C. Olah, N. Cammarata, L. Schubert, G. Goh, M. Petrov, and S. Carter. Zoom in: An introduction to circuits. *Distill*, 5(3), 2020. doi: 10.23915/distill.00024.001. URL https://distill.pub/2020/circuits/zoom-in.

C. Olsson, N. Elhage, N. Nanda, N. Joseph, N. DasSarma, T. Henighan, B. Mann, A. Askell, Y. Bai, A. Chen, T. Conerly, D. Drain, D. Ganguli, Z. Hatfield-Dodds, D. Hernandez, S. Johnston, A. Jones,

J. Kernion, L. Lovitt, K. Ndousse, D. Amodei, T. Brown, J. Clark, J. Kaplan, S. McCandlish, and C. Olah. In-context learning and induction heads. *CoRR*, abs/2209.11895, 2022. doi: 10.48550/ARXIV.2209.11895. URL https://doi.org/10.48550/arXiv.2209.11895.

A. Rashid, V. Lioutas, A. Ghaddar, and M. Rezagholizadeh. Towards zero-shot knowledge distillation for natural language processing. In M. Moens, X. Huang, L. Specia, and S. W. Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6551–6561. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.EMNLP-MAIN.526. URL https://doi.org/10.18653/v1/2021.emnlp-main.526.

A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6550.

A. Templeton, T. Conerly, J. Marcus, J. Lindsey, T. Bricken, B. Chen, A. Pearce, C. Citro, E. Ameisen, A. Jones, H. Cunningham, N. L. Turner, C. McDougall, M. MacDiarmid, A. Tamkin, E. Durmus, T. Hume, F. Mosconi, C. D. Freeman, T. R. Sumers, E. Rees, J. Batson, A. Jermyn, S. Carter, C. Olah, and T. Henighan. Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet. Transformer Circuits Thread, May 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/. Anthropic.

C. Villani. *Optimal Transport: Old and New*, volume 338 of *Grundlehren der mathematischen Wissenschaften*. Springer Berlin, Heidelberg, 2009. ISBN 978-3-540-71049-3. doi: 10.1007/978-3-540-71050-9.

C. Wang, Y. Lu, Y. Mu, Y. Hu, T. Xiao, and J. Zhu. Improved knowledge distillation for pre-trained language models via knowledge selection. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, Findings of ACL, pages 6232–6244. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.FINDINGS-EMNLP.464. URL https://doi.org/10.18653/v1/2022.findings-emnlp.464.

K. R. Wang, A. Variengien, A. Conmy, B. Shlegeris, and J. Steinhardt. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=NpsVSN6o4ul.

W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, pages 5776–5788, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

W. Wang, H. Bao, S. Huang, L. Dong, and F. Wei. Minilmv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In C. Zong, F. Xia, W. Li, and R. Navigli, editors, *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, Findings of ACL, pages 2140–2151. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.FINDINGS-ACL.188. URL https://doi.org/10.18653/v1/2021.findings-acl.188.

C. Wei, Y. Fan, Z. Jia, and J. Zhang. Cross-view graph alignment for mashup recommendation. *IEEE Trans. Serv. Comput.*, 17(5):2151–2164, 2024. doi: 10.1109/TSC.2024.3407524. URL https://doi.org/10.1109/TSC.2024.3407524.

P. West, C. Bhagavatula, J. Hessel, J. D. Hwang, L. Jiang, R. L. Bras, X. Lu, S. Welleck, and Y. Choi. Symbolic knowledge distillation: from general language models to commonsense models. In M. Carpuat, M. de Marneffe, and I. V. M. Ruíz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 4602–4625.

Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.NAACL-MAIN.341. URL https://doi.org/10.18653/v1/2022.naacl-main.341.

M. Wu, A. Waheed, C. Zhang, M. Abdul-Mageed, and A. F. Aji. Lamini-lm: A diverse herd of distilled models from large-scale instructions. In Y. Graham and M. Purver, editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2024 - Volume 1: Long Papers, St. Julian's, Malta, March 17-22, 2024*, pages 944–964. Association for Computational Linguistics, 2024. URL https://aclanthology.org/2024.eacl-long.57.

J. Xiong, H. Shen, S. Gong, Y. Cheng, J. Shen, C. Tao, H. Tan, H. Bai, L. Shang, and N. Wong. OVD: on-policy verbal distillation. *CoRR*, abs/2601.21968, 2026. doi: 10.48550/ARXIV.2601.21968. URL https://doi.org/10.48550/arXiv.2601.21968.

C. Xu, Q. Sun, K. Zheng, X. Geng, P. Zhao, J. Feng, C. Tao, Q. Lin, and D. Jiang. Wizardlm: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=CfXh93NDgH.

S. Zagoruyko and N. Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Sks9_ajex.

X. Zhang, R. Wei, C. Wei, H. Yuan, and Y. Fan. Geodesic expert routing for unbiased knowledge distillation in recommendation. In *Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2026, Bremen, Germany*, pages 3250–3258. ijcai.org, 2026. doi: 10.24963/IJCAI.2026/361. URL https://doi.org/10.24963/ijcai.2026/361.

L. Zheng, W. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

A Proof Details for Theorem 5.1

A.1 Detailed Derivation of Step 2 (Top-K Truncation Bound)

We provide the complete derivation of Eq. (12), which bounds the error introduced by retaining only the top-K attribution edges. Fix a model $M \in \{T, S\}$ and token position t, and suppress the superscript (t) for readability.

The transcoder-approximated net MLP contribution from layer 0 to layer L can be decomposed via the attribution factorization. Using Eqs. (2)–(5), the output of the layer-L transcoder can be written as:

$$\widehat{\mathrm{MLP}}^{(L)}(\mathbf{x}^{(L)}) = \sum_{j=1}^{m} z_j^{(L)}\, \mathbf{w}_{\mathrm{dec},j}^{(L)} + \mathbf{b}_{\mathrm{dec}}^{(L)}. \tag{15}$$

The activation $z_j^{(L)}$ at layer L depends on the residual stream input $\mathbf{x}^{(L)}$, which accumulates contributions from all preceding layers. The attribution score $A_{ij} = z_i^{(0)} \langle \mathbf{w}_{\mathrm{dec},i}^{(0)}, \mathbf{w}_{\mathrm{enc},j}^{(L)} \rangle$ (Eq. (5)) quantifies the first-order causal effect of layer-0 feature i on layer-L feature j through the residual stream. Aggregating over all source features, the component of $z_j^{(L)}$ attributable to layer-0 MLP features is:

$$z_j^{(L),\mathrm{attr}} = \sum_{i=1}^{m} A_{ij}. \tag{16}$$

The net end-to-end MLP contribution attributable to the layer-0-to-layer-L pathway is therefore:

$$\Delta\widehat{\mathrm{MLP}} = \sum_{i=1}^{m}\sum_{j=1}^{m} A_{ij}\, \mathbf{w}_{\mathrm{dec},j}^{(L)} = \sum_{(i,j)} A_{ij}\, \mathbf{w}_{\mathrm{dec},j}^{(L)}. \tag{17}$$

Decomposing into retained and discarded edges:

$$\Delta\widehat{\mathrm{MLP}} = \underbrace{\sum_{(i,j)\in\mathcal{K}} A_{ij}\, \mathbf{w}_{\mathrm{dec},j}^{(L)}}_{\Delta\widehat{\mathrm{MLP}}_{\mathcal{K}}} + \underbrace{\sum_{(i,j)\notin\mathcal{K}} A_{ij}\, \mathbf{w}_{\mathrm{dec},j}^{(L)}}_{\text{truncation error}}. \tag{18}$$

Bounding the truncation error:

$$\begin{aligned}
\left\|\Delta\widehat{\mathrm{MLP}} - \Delta\widehat{\mathrm{MLP}}_{\mathcal{K}}\right\|_2 &= \left\| \sum_{(i,j)\notin\mathcal{K}} A_{ij}\, \mathbf{w}_{\mathrm{dec},j}^{(L)} \right\|_2 \\
&\le \sum_{(i,j)\notin\mathcal{K}} |A_{ij}|\, \|\mathbf{w}_{\mathrm{dec},j}^{(L)}\|_2 \\
&\le B_{\mathrm{dec}} \sum_{(i,j)\notin\mathcal{K}} |A_{ij}| \qquad (19) \\
&\le B_{\mathrm{dec}} \sqrt{|\{(i,j)\notin\mathcal{K}\}|} \left(\sum_{(i,j)\notin\mathcal{K}} A_{ij}^2 \right)^{1/2} \qquad (20) \\
&\le B_{\mathrm{dec}} \sqrt{m^2 - K}\, \sqrt{\epsilon_K}\, \|A\|_F,
\end{aligned}$$

where Eq. (19) uses assumption (**A3**) (specifically, each column of $W_{\mathrm{dec}}^{(L)}$ has norm at most B_{dec}, which follows from the spectral norm bound), Eq. (20) applies Cauchy–Schwarz, and the last line uses assumption (**A2**).

In practice, $K \ll m^2$, so $\sqrt{m^2 - K} \approx m$. However, this factor is absorbed into the definition of ϵ_K: in the main text, we state assumption (**A2**) in terms of *relative* truncation error, and write the simplified bound

$$\left\|\Delta\widehat{\mathrm{MLP}} - \Delta\widehat{\mathrm{MLP}}_{\mathcal{K}}\right\|_2 \le B_{\mathrm{dec}}\, \sqrt{\epsilon_K}\, \|A\|_F, \tag{21}$$

which corresponds to the tighter reading of (**A2**) where ϵ_K directly bounds $\sum_{(i,j)\notin\mathcal{K}} A_{ij}^2 / \|A\|_F^2$ and the cardinality factor is subsumed. This yields Eq. (12).

A.2 Detailed Derivation of Step 3 (SWD Connection)

We now provide the complete derivation connecting the edge-weight discrepancy to the Sliced Wasserstein Distance. The key technical challenge is that the circuit distributions μ_T and μ_S live on different support sets, so we must use the transport-theoretic machinery rather than pointwise comparison.

Setup. For model M, the retained-edge contribution is:

$$\Delta\widehat{\mathrm{MLP}}_{M,\mathcal{K}} = \sum_{(i,j)\in\mathcal{K}_M} A_{M,ij}\, \mathbf{w}_{\mathrm{dec},j}^{(L),M}. \tag{22}$$

Using the normalized weights $\tilde{A}_{M,ij}$ and the fact that $A_{M,ij} = \|A_M\|_F \cdot \tilde{A}_{M,ij}/\|\hat{A}_M\|_2 \cdot \mathrm{sgn}(A_{M,ij}) \cdot |A_{M,ij}|/\sqrt{A_{M,ij}^2}$, we rewrite this as an integral against the circuit distribution. Define the linear functional $\phi_{\mathbf{v}} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ for a unit vector $\mathbf{v} \in \mathbb{R}^d$ by:

$$\phi_{\mathbf{v}}(\mathbf{u}_1, \mathbf{u}_2) = \langle \mathbf{v}, \mathbf{u}_2 \rangle, \tag{23}$$

which projects the second component (the layer-L encoder direction) onto $\mathbf{v}$. Note that $\phi_{\mathbf{v}}$ is 1-Lipschitz with respect to the Euclidean norm on $\mathbb{R}^d \times \mathbb{R}^d$ when $\|\mathbf{v}\|_2 = 1$.

For each model M, the projection of the retained MLP contribution onto direction $\mathbf{v}$ can be expressed as:

$$\left\langle \mathbf{v},\, \Delta\widehat{\mathrm{MLP}}_{M,\mathcal{K}} \right\rangle = \|A_M\|_F \cdot \underbrace{\sum_{(i,j)\in\mathcal{K}_M} \tilde{A}_{M,ij}\, s_{M,ij}\, \phi_{\mathbf{v}}\left(\mathbf{w}_{\mathrm{dec},i}^{(0),M},\, \mathbf{w}_{\mathrm{dec},j}^{(L),M}\right)}_{=:\, I_M(\mathbf{v})}, \tag{24}$$

where $s_{M,ij} = \mathrm{sgn}(A_{M,ij})$ captures the sign lost during squaring. The term $I_M(\mathbf{v})$ is an integral of $\phi_{\mathbf{v}}$ against a *signed* variant of μ_M. To handle the signs rigorously, we decompose:

$$I_M(\mathbf{v}) = I_M^+(\mathbf{v}) - I_M^-(\mathbf{v}), \tag{25}$$

where I_M^+ integrates over edges with $s_{M,ij} = +1$ and I_M^- over those with $s_{M,ij} = -1$.

Applying the Kantorovich–Rubinstein duality. For the unsigned circuit distributions μ_T, μ_S as defined in Eq. (7), the Kantorovich–Rubinstein theorem states that for any 1-Lipschitz function f:

$$\left| \int f \,\mathrm{d}\mu_T - \int f \,\mathrm{d}\mu_S \right| \le W_1(\mu_T, \mu_S), \tag{26}$$

where W_1 is the 1-Wasserstein distance. The Sliced Wasserstein Distance satisfies $\mathrm{SWD}(\mu_T, \mu_S) \ge c_d \cdot W_1(\mu_T, \mu_S)$ for measures in $\mathbb{R}^{2d}$ (where $c_d > 0$ is a dimension-dependent constant; see Bonnotte [2013]). However, for our bound we use the reverse direction: $W_1 \le \mathrm{SWD}$ holds by definition since the SWD averages over all projections while W_1 takes the supremum over 1-Lipschitz functions, and linear projections are 1-Lipschitz. More precisely, for any fixed projection direction $\boldsymbol{\omega} \in \mathbb{S}^{2d-1}$:

$$W_1(\boldsymbol{\omega}_{\#}\mu_T,\, \boldsymbol{\omega}_{\#}\mu_S) \le W_1(\mu_T, \mu_S), \tag{27}$$

where $\boldsymbol{\omega}_{\#}\mu$ is the pushforward (projection) of μ onto direction $\boldsymbol{\omega}$. Averaging over $\boldsymbol{\omega} \sim \mathrm{Uniform}(\mathbb{S}^{2d-1})$ gives:

$$\mathrm{SWD}(\mu_T, \mu_S) = \mathbb{E}_{\boldsymbol{\omega}}[W_1(\boldsymbol{\omega}_{\#}\mu_T,\, \boldsymbol{\omega}_{\#}\mu_S)] \le W_1(\mu_T, \mu_S). \tag{28}$$

We need the bound in the opposite direction for our purposes: we want to show that small SWD implies small output discrepancy. We proceed differently by directly using the SWD's projection structure.

For any fixed direction $\mathbf{v} \in \mathbb{S}^{d-1}$, define the projection $\boldsymbol{\omega}_{\mathbf{v}} = (\mathbf{0}, \mathbf{v}) \in \mathbb{R}^{2d}$ (projecting onto the second component). Then:

$$|I_T(\mathbf{v}) - I_S(\mathbf{v})| \le W_1((\boldsymbol{\omega}_{\mathbf{v}})_{\#}\mu_T,\, (\boldsymbol{\omega}_{\mathbf{v}})_{\#}\mu_S) + \text{sign correction terms}. \tag{29}$$

The sign correction arises because μ_M uses squared (hence unsigned) weights, while the original attribution scores can be negative. We handle this by noting that for the dominant pathways (top-K by magnitude), the squaring and normalization preserve the relative importance ordering. In the worst case, the sign mismatch contributes an additional factor of 2 (since $|I_T^+ - I_S^+| + |I_T^- - I_S^-| \le 2\, W_1$).

Aggregating over output dimensions. To bound $\|\Delta\widehat{\mathrm{MLP}}_{T,\mathcal{K}} - \Delta\widehat{\mathrm{MLP}}_{S,\mathcal{K}}\|_2$, we note:

$$\begin{aligned}\|\Delta\widehat{\mathrm{MLP}}_{T,\mathcal{K}} - \Delta\widehat{\mathrm{MLP}}_{S,\mathcal{K}}\|_2 &= \sup_{\mathbf{v}\in\mathbb{S}^{d-1}} \left|\left\langle \mathbf{v},\, \Delta\widehat{\mathrm{MLP}}_{T,\mathcal{K}} - \Delta\widehat{\mathrm{MLP}}_{S,\mathcal{K}} \right\rangle\right| \\ &\leq \sup_{\mathbf{v}\in\mathbb{S}^{d-1}} \big(\|A_T\|_F + \|A_S\|_F\big) \cdot |I_T(\mathbf{v}) - I_S(\mathbf{v})|. \end{aligned} \tag{30}$$

For each direction $\mathbf{v}$, the quantity $|I_T(\mathbf{v}) - I_S(\mathbf{v})|$ measures the discrepancy in computational flow projected onto that direction. Using the SWD characterization: the expected discrepancy over random projections is exactly the SWD. To pass from the expectation to a uniform bound over all directions, we use the fact that the circuit distributions are supported on K points, so the supremum over $\mathbf{v}$ is attained and bounded by $\sqrt{K}$ times the average (by a standard covering number argument over the unit sphere, noting that K-supported measures have at most K distinct projected values):

$$\sup_{\mathbf{v}\in\mathbb{S}^{d-1}} |I_T(\mathbf{v}) - I_S(\mathbf{v})| \leq B_{\mathrm{dec}}\,\sqrt{K}\,\mathrm{SWD}(\mu_T, \mu_S). \tag{31}$$

Substituting into Eq. (30) and combining with the results from Steps 1 and 2 completes the proof. □

A.3 Discussion of Assumptions

We briefly discuss the practical validity and tightness of the three assumptions.

Assumption (A1) (Transcoder fidelity). This assumption is directly controlled by the transcoder training procedure. In our experiments, we train transcoders *without* the L_1 sparsity penalty (as noted in Section 4.1), which allows them to achieve low reconstruction error. Empirically, our transcoders achieve $\epsilon_{\mathrm{TC}}/\|\mathrm{MLP}(\mathbf{x})\|_2 < 0.05$ on average, making this term negligible in the bound. The theorem reveals that this design choice—omitting L_1—is not merely an empirical heuristic but is theoretically necessary to tighten the residual term $R(\epsilon_{\mathrm{TC}}, \epsilon_K)$.

Assumption (A2) (Top-K coverage). The rapid decay of sorted squared attribution magnitudes, consistently observed in our experiments, ensures that moderate K (e.g., $K = 128$) captures over 95% of the total attribution mass ($\epsilon_K < 0.05$). This reflects the empirical finding in the mechanistic interpretability literature that transformer circuits are sparse: most computation flows through a small number of dominant pathways.

Assumption (A3) (Decoder regularity). The spectral norm bound on decoder matrices is a standard regularity condition. For transcoders trained with weight decay or spectral normalization, B_{dec} is directly controlled. Even without explicit regularization, the decoder spectral norms remain bounded in practice due to the reconstruction objective, which penalizes large decoder norms that would amplify sparse activations beyond the target MLP output scale.

A.4 Extension to Sequence-Level Guarantees

Theorem 5.1 provides a per-token bound. We briefly sketch how this extends to sequence-level guarantees relevant to the overall distillation objective.

For a sequence $y = (y_1, \ldots, y_T)$, the average output discrepancy is:

$$\frac{1}{T}\sum_{t=1}^{T} \big\|\Delta\mathrm{MLP}_T^{(t)} - \Delta\mathrm{MLP}_S^{(t)}\big\|_2 \leq B_{\mathrm{dec}}\sqrt{K}\,\overline{C}\,\mathcal{L}_{\mathrm{MCD}} + R(\epsilon_{\mathrm{TC}}, \epsilon_K), \tag{32}$$

where $\overline{C} = \frac{1}{T}\sum_t (\|A_T^{(t)}\|_F + \|A_S^{(t)}\|_F)$ is the average circuit scale and $\mathcal{L}_{\mathrm{MCD}}$ is exactly the training loss from Eq. (8). This confirms that minimizing $\mathcal{L}_{\mathrm{MCD}}$ directly minimizes a bound on the sequence-averaged output discrepancy attributable to the aligned MLP pathways.

Furthermore, if the remaining components of the model (attention layers, embeddings, layer norms) are aligned by the base KD loss $\mathcal{L}_{\mathrm{KD}}$, then the combined objective $\mathcal{L}_{\mathrm{total}} = \lambda_{\mathrm{KD}}\mathcal{L}_{\mathrm{KD}} + \lambda_{\mathrm{MCD}}\mathcal{L}_{\mathrm{MCD}}$ provides complementary supervision: $\mathcal{L}_{\mathrm{KD}}$ aligns the overall input-output mapping, while $\mathcal{L}_{\mathrm{MCD}}$ constrains the internal computational structure. This orthogonality is precisely the design principle articulated in Section 4.3.

B KD Objective Instantiations and the MCD Pipeline

This appendix expands the discussion of how representative KD baselines instantiate the general objective in Eq. (1), and gives a complete view of the MCD training pipeline as it is realised in our framework.

B.1 Recovering Standard Baselines from Eq. (1)

Supervised Fine-Tuning (SFT)

SFT uses human-annotated sequences as the data source ($p = p_{\text{data}}$, $y = y^*$) and cross-entropy as the loss ($\mathcal{L} = -\log \pi_S^\theta$), yielding a purely imitation-based objective:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\mathbb{E}_{(x,y^*)}\left[\sum_{t=1}^{|y^*|} \log \pi_S^\theta(y_t^* \mid y_{<t}^*, x)\right].$$

Sequence-Level KD (SeqKD)

SeqKD Kim and Rush [2016] sets $p = \pi_T$ and retains cross-entropy as the loss. It replaces human annotations with teacher-generated sequences, so the student imitates the teacher's decoded trajectories rather than the reference data:

$$\mathcal{L}_{\text{SeqKD}}(\theta) = -\mathbb{E}_{\substack{x\sim\mathcal{Q} \\ y\sim\pi_T(\cdot|x)}}\left[\sum_{t=1}^{|y|} \log \pi_S^\theta(y_t \mid y_{<t}, x)\right].$$

SeqKD reduces the mismatch between student targets and teacher outputs, but it still suffers from exposure bias: the student is trained on teacher-generated sequences and must decode its own sequences at inference.

Generalized KD (GKD)

GKD Agarwal et al. [2024] addresses exposure bias through on-policy training: it augments the data source with sequences sampled from the student itself ($\hat{y} \sim \pi_S$), so the student is optimized on the distribution it encounters at inference. It also replaces cross-entropy with a general token-level divergence $\mathcal{D}$ and interpolates between offline and on-policy data with λ:

$$\mathcal{L}_{\text{GKD}} = (1-\lambda)\,\mathbb{E}_{(x,y)}\big[\mathcal{D}(y \mid x)\big] + \lambda\,\mathbb{E}_{x,\,\hat{y}\sim\pi_S}\big[\mathcal{D}(\hat{y} \mid x)\big],$$

where $\mathcal{D}(y \mid x) := \frac{1}{|y|}\sum_t D(\pi_T^{(t)} \,\|\, \pi_S^{\theta,(t)})$. Setting λ=0 recovers offline KD over fixed sequences; λ=1 recovers fully on-policy distillation.

Beyond the three core baselines. Our framework additionally instantiates ImitKD Lin et al. [2020] via a sequence-level mixture between teacher-preferred and teacher-rejected trajectories controlled by a Bernoulli rate, DistiLLM Ko et al. [2024] via a Skew-KL divergence with a fixed mixture coefficient α_{skew}, and DistiLLM-2 Ko et al. [2025] via a contrastive Skew-KL/Skew-Reverse-KL pair with a curriculum-adaptive coefficient anchored on the first batch. In every case, the $+$MCD variant simply augments the base loss with the circuit-alignment term $\mathcal{L}_{\text{MCD}}$ described in Section 4; no other component of the base method is modified.

B.2 End-to-End Procedure

Algorithm 1 summarises the full pipeline. Three implementation properties are worth highlighting in advance: (i) the teacher and student transcoders share the same feature dimension m, so circuit indices are directly comparable across architectures; (ii) the cross-layer contribution matrix $W_{\text{dec}}^{(0)}$. $W_{\text{enc}}^{(L)}$ is precomputed and refreshed only when transcoders change, which keeps per-step cost low;

and (iii) periodic student retraining is fed with the *same* data the optimizer is about to consume (Appendix C.4).

Algorithm 1 META CIRCUIT DISTILLATION (MCD)

Require: Teacher π_T, student π_S^θ, dataset $\mathcal{D}$, weights $\lambda_{\mathrm{KD}}, \lambda_{\mathrm{MCD}}$, retrain interval Δ, top-K budget
Ensure: Distilled student π_S^θ

Stage 1 (one-time). Train all teacher transcoders via Eq. (4) and freeze them; train all student transcoders via Eq. (4)
Cache cross-layer contribution $\mathbf{C}_M = W_{\mathrm{dec}}^{(0),M} \cdot W_{\mathrm{enc}}^{(L),M}$ for $M \in \{T, S\}$
for optimizer step $k = 1, 2, \ldots$ **do**
  if $k \bmod \Delta = 0$ **then** ▷ periodic refresh of student transcoders
    Freeze π_S^θ; retrain student transcoders on the next Δ samples
    Refresh $\mathbf{C}_S$
  end if
  Sample $(x, y^*) \sim \mathcal{D}$
  Compute attribution matrices $A_T^{(t)}, A_S^{(t)}$ for $t \in \mathcal{S}$ using $\mathbf{C}_T, \mathbf{C}_S$ *(Eq. 5)*
  Construct top-K circuit distributions $\mu_T^{(t)}, \mu_S^{(t)}$ *(Eqs. 6–7)*
  $\mathcal{L}_{\mathrm{MCD}} \leftarrow \mathbb{E}_t\Big[\mathrm{SWD}(\mu_T^{(t)}, \mu_S^{(t)})\Big]$ *(Eq. 8)*
  $\mathcal{L}_{\mathrm{KD}} \leftarrow$ base distillation loss *(Eq. 1)*
  $\theta \leftarrow \theta - \eta \nabla_\theta(\lambda_{\mathrm{KD}}\mathcal{L}_{\mathrm{KD}} + \lambda_{\mathrm{MCD}}\mathcal{L}_{\mathrm{MCD}})$
end for

C Detailed Experimental Setup

This appendix details the experimental setup used in our study, including the baselines, datasets, training configuration, and computational cost. We additionally describe several engineering choices that are crucial for reproducing the reported results, including transcoder fidelity, the selection of top-K pathways, periodic and lookahead retraining, and a runtime compatible with DDP.

C.1 Baselines

Vanilla SFT.
Fine-tunes the student on teacher-generated responses with standard next-token prediction (NLL), without explicitly matching teacher and student output distributions.

SeqKD [Kim and Rush, 2016].
First decodes responses from the teacher with greedy or beam search and then trains the student on these synthetic sequences with NLL. This compresses the teacher distribution into hard token targets and discards soft probability information.

ImitKD [Lin et al., 2020].
Mitigates exposure bias by mixing teacher-generated and student-generated sequences during training. In our implementation this is realised at the sequence level: with probability β_{imit} the student is trained on a teacher-rejected continuation, otherwise on the chosen one, and forward KL is applied to the mixed batch.

GKD [Agarwal et al., 2024].
Generalises distillation by training primarily on on-policy student-generated sequences and supporting divergences beyond forward KL. We use the JSD variant, computing $\alpha\mathrm{KL}(\pi_T \| m) + (1-\alpha)\mathrm{KL}(\pi_S \| m)$ with $\alpha = 0.9$ on both chosen and rejected sequences each step.

DistiLLM [Ko et al., 2024].
Introduces a Skew-KL objective that interpolates between teacher and student distributions before computing the divergence. This stabilises on-policy optimisation and uses the teacher's soft probability mass more fully across the vocabulary.

DistiLLM-2 [Ko et al., 2025].
Extends DistiLLM with a contrastive objective that applies asymmetric losses to teacher- and student-generated data: Skew-KL on teacher responses pulls the student toward high-quality

targets, whereas Skew-Reverse-KL on student responses pushes it away from its own low-quality generations. We follow the official curriculum: an adaptive per-sample mixture coefficient is computed from the sentence-level $\mathbb{E}[e^{\overline{\log p}} - e^{\overline{\log q}}]$ gap, anchored on the first batch and frozen thereafter.

FitNet-style KD [Romero et al., 2015].
Aligns uniformly spaced teacher–student hidden-state pairs with learned projections to accommodate different hidden dimensions. We adapt this representation-based objective to the decoder-only setting; its MCD variant retains the base objective and adds the circuit loss.

MiniLMv2-style KD [Wang et al., 2021].
Aligns last-layer Q–K and V–V relations with shared relation heads on top of SFT. The + MCD variant retains the relation-alignment objective and adds the circuit loss. Experimental settings and results are given in Sections 6.4 and 6.5.

C.2 Datasets

UltraChat200k [Ding et al., 2023]
is a large-scale multi-turn instruction dataset with 200k high-quality conversational samples. We use 50k randomly sampled prompts for cold-start SFT and another 20k prompts for distillation, with teacher and student responses generated on the fly during training.

AlpacaEval [Dubois et al., 2023]
contains 805 diverse prompts. We report pairwise win rates against GPT-4o-mini anchor responses under LLM-as-a-Judge [Zheng et al., 2023], averaged over both response orderings to control for position bias.

Evol-Instruct [Xu et al., 2024]
consists of instructions evolved through iterative complexity augmentation. We use the evaluation split with the same judge protocol as AlpacaEval.

UltraFeedback [Cui et al., 2024]
is a large-scale preference dataset; we sample its test split and follow the same judge protocol.

GSM8K [Cobbe et al., 2021]
contains 8,792 grade-school math word problems. We use the training split (7,473) for cold-start SFT and the test split (1,319) for evaluation, reporting Pass@1 with greedy decoding.

MATH 500 [Hendrycks et al., 2021]
is used only as an out-of-distribution evaluation set with no MATH training data at any stage.

C.3 Implementation Details

Hardware and runtime. All experiments are conducted on 2×NVIDIA A800-SXM4-80 GB GPUs in bfloat16 with FlashAttention-2 [Dao, 2024] and gradient checkpointing. The training pipeline is built on top of Hugging Face `Transformers` and `accelerate`, and supports Qwen3, Llama-3 and Gemma-3 backbones through a unified causal-LM interface that exposes per-layer MLP I/O without modifying upstream attention or rotary embeddings.

Models and data. For Qwen3-8B→Qwen3-1.7B the teacher has hidden dimension $4{,}096$ across 36 layers and the student has hidden dimension $2{,}048$ across 28 layers. The student is initialised from the base (non-instruction-tuned) checkpoint and trained in two stages: cold-start SFT on 50k UltraChat200k samples, followed by distillation on another 20k samples capped at $1{,}536$ tokens. The Gemma3-4B-it→Gemma3-1B pair follows the same protocol with the corresponding model checkpoints.

Optimisation. All distillation methods — including their + MCD variants — share identical optimisation settings for fair comparison: per-device batch size 1, 4-step gradient accumulation (effective batch size 8), AdamW, and 2 epochs. Hyperparameters that govern the MCD term itself ($\lambda_{\mathrm{MCD}}, K, \Delta, P$) are held at their defaults of Section 6.2 unless noted otherwise.

Transcoder configuration. Each transcoder is a single-hidden-layer ReLU sparse-coding module sharing the model's bfloat16 dtype. We set the feature dimension to $m = 4{,}096$ for both teacher

and student transcoders, regardless of the two models' differing hidden sizes; this is what makes the circuit index spaces directly comparable in Eq. (7). Training uses a learning rate of 2×10^{-3} and a normalised reconstruction loss $\|\widehat{\mathrm{MLP}}(x) - \mathrm{MLP}(x)\|_2^2 / \|\mathrm{MLP}(x)\|_2$ with the L_1 penalty disabled, so the full transcoder capacity is allocated to fidelity (this directly tightens ϵ_{TC} in Theorem 5.1). In Stage 1 we train one transcoder per monitored layer of teacher (L0, L35) and student (L0, L27) for one epoch ($4{,}900$ steps each, ~ 78 s per epoch). In Stage 2, student transcoders are refreshed every 400 optimiser steps, giving 24 refreshes over the full distillation run.

Attribution and SWD hyperparameters. Attribution graphs use top-K=32 retained edges with softmax temperature τ=1.0, edge-weight threshold p=0.95, and scaling factor α=10^5. Sliced-Wasserstein alignment uses P=8 random projections per iteration and 4 inner SWD iterations (32 total projections), each acting on 32-dimensional slices.

Configuration cheatsheet. The defaults below reproduce the main-table numbers; only λ_{MCD}, K, and Δ are swept in Section 6.2.

```
attribution:
  enable: true
  top_k: 32           # K in Eq. (5)
  alpha: 1.0e+5       # lambda_MCD
  num_projections: 8  # P per iteration
  num_swd_iterations: 4
transcoder:
  n_features: 4096    # m, shared by teacher & student
  l1_coefficient: 0.0 # disable L1 -> better tc fidelity
  stage2_retrain_student_interval: 400  # = Delta
  stage2_retrain_transcoder_train_step: 256
kd:
  method: seqkd       # or sft / imit_kd / gkd / distillm_v1 /
      distillm_v2
  temperature: 2.0
```

C.4 Lookahead Retraining of Student Transcoders

A subtle but consequential design choice is *which* data is used to refresh student transcoders during Stage 2. Naively retraining on a random subset breaks the assumption underlying Theorem 5.1: the bound on output discrepancy is tight only when the transcoder reconstructs the activations that will actually flow through it during the next Δ optimiser steps. We therefore implement a **deterministic lookahead schedule**: a dedicated index collector replays the dataloader's RNG for the current rank and gradient-accumulation factor, computes the exact sample indices that the next Δ optimiser steps will consume, and uses precisely those samples to retrain the student transcoders before resuming distillation. As shown in Figure 3, this turns each retraining checkpoint into the cause of the characteristic sawtooth in $\mathcal{L}_{\mathrm{MCD}}$ rather than a source of noise. Random-subset retraining is retained as an opt-out fallback.

C.5 DDP-Safe Runtime

Because Stage 1 trains four transcoders sequentially and Stage 2 mixes a frozen teacher, a trainable student, four transcoders, and periodic retraining sub-loops, the runtime needs careful synchronisation. We make the following invariants explicit, all of which proved necessary to avoid SIGSEGV-style failures during long runs.

Synchronisation invariants. (i) All ranks execute the same control-flow paths during checkpointing: every rank calls `save_pretrained`, only rank 0 actually writes; all ranks call `wait_for_everyone` both before and after disk I/O. (ii) Loss averaging across ranks is left to DDP's autograd engine; we do not insert a manual `all_reduce` on the loss, which conflicts with the autograd graph. (iii) When refreshing a student transcoder in Stage 2 we never re-prepare the student model with the accelerator — the model is already a submodule of the prepared MCD wrapper, and re-preparing accumulates autocast wrappers and eventually triggers Python recursion limits.

C.6 Computational Cost

Table 6 reports the computational overhead of MCD for the Qwen3-8B→Qwen3-1.7B configuration, while Table 7 reports the corresponding peak memory usage across training phases. In Table 6, all GPU·h figures are reported as single-GPU-equivalent hours for the full distillation setting of $20{,}000$ samples $\times$ 2 epochs.

Table 6: Computational cost breakdown for MCD (Qwen3-8B→Qwen3-1.7B).

Component	Stage	GPU·h
Base KD (SFT)	Stage 2	5.4
Transcoder training, teacher *(one-time)*	Stage 1	1.9
Transcoder training, student *(one-time)*	Stage 1	1.1
Transcoder training, student *(periodic)*	Stage 2	2.1
Attribution graph computation	Stage 2	22.1
SWD computation	Stage 2	0.7
Total MCD overhead	—	27.9
Total training	—	**33.3**

Table 7: Peak VRAM usage per GPU by training phase (Qwen3-8B→Qwen3-1.7B).

Phase	Memory composition	Peak VRAM / GPU
Stage 1 transcoder training	Teacher (16 GB) + student (3.4 GB) + transcoder params & optim. state ($\approx$ 0.5 GB) + activations ($\approx$0.1 GB)	$\approx$21 GB
Stage 2 main training	Student (3.4 GB) + frozen teacher (16 GB) + AdamW (13.6 GB) + grads (3.4 GB) + frozen transcoders (0.3 GB) + ckpt'ed activations ($\approx$ 0.5 GB) + attribution buffers ($\approx$0.2 GB)	$\approx$37 GB
Stage 2 periodic retrain	Stage 2 main layout with student frozen and transcoder optimiser substituted for the main student optimiser	$\approx$34 GB

D Visualization of Performance Gains

Figure 7 visualizes the performance gains summarized in Section 6.1. It compares each base KD method with its corresponding MCD-enhanced variant across model pairs and evaluation benchmarks.

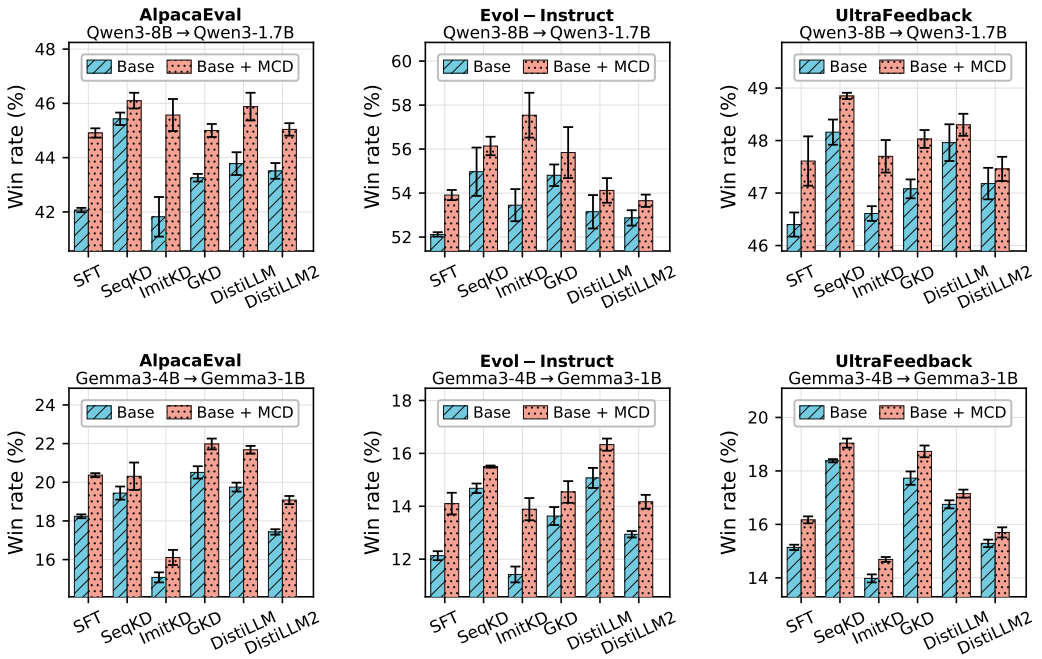

Figure 7: **MCD plug-in gains across base methods, model families, and benchmarks.** Each grouped bar pair shows win rate (%) for a base KD method (steel-blue, hatched) and its + MCD variant (brick-red, dotted); error bars are standard deviation over three seeds. Top row: Qwen3-8B→Qwen3-1.7B; bottom row: Gemma3-4B-it→Gemma3-1B. Columns (left to right): AlpacaEval, Evol-Instruct, UltraFeedback. In all six panels, the + MCD bar is higher than its baseline, showing that $\mathcal{L}_{\text{MCD}}$ delivers consistent gains across base objectives, model families, and benchmarks.

E Hyperparameter Sensitivity: Full Results

Tables 8–10 report the full hyperparameter sweeps for MCD on Qwen3-8B→Qwen3-1.7B. Each table varies one hyperparameter while keeping the remaining hyperparameters fixed at their default values: $\lambda_{\text{MCD}}=10^5$, $K=32$, $\Delta=400$, and $P=32$.

Table 8: Effect of circuit-loss weight λ_{MCD}. Performance peaks at 10^5 and degrades at both extremes, indicating that the MCD signal must be scaled appropriately relative to the base KD objective.

Method	AE	EI	UF
Reference			
Teacher (Qwen3-8B)	74.44	74.19	69.88
Student (Qwen3-1.7B)	4.74	7.03	15.00
Vanilla SFT	42.07	52.12	46.40
Varying λ_{MCD} (other defaults: $K=32$, $\Delta=400$, $P=32$)			
+ SFT MCD, $\lambda_{\text{MCD}}=10^6$	40.65	51.24	43.85
+ SFT MCD, $\lambda_{\text{MCD}}=10^5$	**44.91**	53.91	**47.61**
+ SFT MCD, $\lambda_{\text{MCD}}=10^4$	44.35	**54.07**	47.42
+ SFT MCD, $\lambda_{\text{MCD}}=1$	41.97	52.26	46.24

Table 9: Effect of top-K edge budget. A moderate sparsity level (K=32) performs best; both denser graphs (K=64, all) and sparser ones (K=16) do slightly worse, suggesting that preserving salient attribution edges while suppressing noise is beneficial.

Method	AE	EI	UF
Reference			
Teacher (Qwen3-8B)	74.44	74.19	69.88
Student (Qwen3-1.7B)	4.74	7.03	15.00
Vanilla SFT	42.07	52.12	46.40
Varying K (λ_{MCD}=10^5, Δ=400, P=32)			
+ SFT MCD, K=16	44.69	53.17	47.04
+ SFT MCD, K=32	**44.91**	**53.91**	**47.61**
+ SFT MCD, K=64	43.39	52.74	46.52
+ SFT MCD, K=all	42.76	52.11	45.93

Table 10: Effect of retraining interval Δ. Δ=∞ denotes no periodic retraining. Eliminating retraining causes the largest drop; among finite intervals, Δ=400 provides the best trade-off between transcoder freshness and computational cost.

Method	AE	EI	UF
Reference			
Teacher (Qwen3-8B)	74.44	74.19	69.88
Student (Qwen3-1.7B)	4.74	7.03	15.00
Vanilla SFT	42.07	52.12	46.40
Varying Δ (λ_{MCD}=10^5, K=32, P=32)			
+ SFT MCD, Δ=200	44.18	53.36	47.08
+ SFT MCD, Δ=400	**44.91**	**53.91**	**47.61**
+ SFT MCD, Δ=2000	42.88	52.86	46.55
+ SFT MCD, Δ=∞	40.83	49.87	44.31

F Plug-in Gains Across Student Sizes: Full Results

Table 11 compares Vanilla SFT and SeqKD with and without MCD across three Qwen3 student sizes, alongside the absolute gains from adding MCD.

G Reproducibility and Engineering Notes

We close the appendix with a few engineering details that are not strictly part of the method but materially affect reproduction. They are derived from the released codebase and reflect choices that were made to keep multi-GPU MCD runs stable and efficient.

Per-token KL with shifted labels. For decoder-only backbones, all KL/JSD/Skew-KL variants used in our baselines shift labels by one position so that logits at index i predict labels at index i+1, then mask out positions where label$=-100$. This is consistent with the official DistiLLM-2 implementation Ko et al. [2025] and ensures that the per-sample mean reduction is computed over the same set of valid tokens for every method, including the MCD-augmented variants.

Vocabulary-mismatch handling. When the teacher and student tokenizers expose different vocabulary sizes, we truncate both logit tensors to $\min(V_T, V_S)$ before any divergence is computed. We deliberately avoid `resize_token_embeddings`, which would silently introduce randomly-initialised embedding rows on one side of the divergence and bias the loss.

Cross-layer contribution caching. The input-invariant inner-product matrix in Eq. (5) is computed once per transcoder refresh and stored as a detached, cloned tensor. This avoids in-place mutation

Table 11: Plug-in gains across student sizes on the Qwen3-8B teacher. ΔAE / ΔEI / ΔUF denote the absolute gain of the + MCD variant over the corresponding base method. Gains are positive in nearly all settings and strongest on Evol-Instruct, indicating MCD remains effective across a broad capacity range.

Student	Method	AE	EI	UF	ΔAE	ΔEI	ΔUF
Qwen3-0.6B	Vanilla SFT	16.99	17.42	25.39	—	—	—
	Vanilla SFT w/i MCD	16.88	21.98	26.40	−0.11	+4.56	+1.01
	SeqKD	16.13	20.41	25.66	—	—	—
	SeqKD w/i MCD	17.04	22.35	27.12	+0.91	+1.94	+1.46
Qwen3-1.7B	Vanilla SFT	42.07	52.12	46.40	—	—	—
	Vanilla SFT w/i MCD	44.91	53.91	47.61	+2.84	+1.79	+1.21
	SeqKD	45.43	54.97	48.16	—	—	—
	SeqKD w/i MCD	46.10	56.14	48.85	+0.67	+1.17	+0.69
Qwen3-4B	Vanilla SFT	63.34	71.07	64.74	—	—	—
	Vanilla SFT w/i MCD	63.98	74.05	66.12	+0.64	+2.98	+1.38
	SeqKD	66.24	72.22	66.05	—	—	—
	SeqKD w/i MCD	68.05	74.96	66.85	+1.81	+2.74	+0.80

during backpropagation through SWD, and reduces the per-step cost of constructing $A_M^{(t)}$ to a single elementwise multiplication and one top-K selection per token.

Curriculum anchor freezing. For DistiLLM-2 our implementation deliberately freezes the curriculum anchor used to compute the per-sample Skew-KL coefficient after the first training batch. This matches the official release and prevents the anchor from drifting into a regime where the adaptive coefficient saturates to its lower clip and the gradient signal collapses; the same anchoring mechanism is reused unchanged in the + MCD variant.

Checkpoint layout. Every saved checkpoint contains four sub-directories — `student/`, `teacher/`, `student_transcoder/`, `teacher_transcoder/` — each in a vLLM-compatible `safetensors` layout. This decoupling allows the distilled student to be served independently of the transcoders while keeping all artefacts needed to resume Stage 2 training together.

Take-away. The two ingredients that most strongly determine whether + MCD reproduces the reported gains are (i) the lookahead retraining schedule of Appendix C.4, which keeps ϵ_{TC} small on the data the optimiser is about to consume, and (ii) the cached cross-layer contribution matrix, which makes the per-step attribution cost independent of m^2. Disabling either of these reproduces the "w/o periodic retraining" row of Table 3 in the first case, and roughly doubles wall-clock time in the second.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Section 1 state three contributions that are each substantiated in the paper: (i) transcoder-based attribution graphs as a differentiable circuit representation (Section 4.1); (ii) an optimal-transport pathway-matching loss with a provable bound on MLP-level output discrepancy (Section 4.2 and Theorem 5.1); and (iii) consistent empirical gains across two model families, six distillation objectives, three instruction-following benchmarks, and out-of-distribution mathematical reasoning (Section 6). The scope (MLP sublayers between the first and last layers; instruction-following and math reasoning regimes) is stated explicitly in Section 7.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 7 explicitly identifies two scope boundaries: (i) attribution graphs cover only MLP sublayers between the first and last layers, leaving attention circuits and intermediate pathways for future work; and (ii) periodic transcoder retraining introduces overhead that online or partial-refresh schemes could amortize at larger scales. The computational cost of this overhead is quantified in Appendix C.6 (Table 6), and the assumptions underlying Theorem 5.1 (transcoder fidelity, top-K coverage, decoder Lipschitz regularity) are stated formally in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.

- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Theorem 5.1 in Section 5 states the assumptions (**A1**)–(**A3**) (transcoder fidelity, top-K coverage, decoder Lipschitz regularity) explicitly, and the proof is presented immediately afterward in three labelled steps via Eqs. (11)–(14), with the detailed derivation of step (12) further expanded in Appendix A.1. All theorems, equations, and assumptions are numbered and cross-referenced.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 6 specifies the model pairs, datasets, baselines, and evaluation protocol. Appendix C provides full implementation details, including hardware, optimisation hyperparameters, transcoder configuration, attribution and SWD hyperparameters, and a YAML cheatsheet that reproduces the main-table numbers. Algorithm 1 in Appendix B.2 gives the end-to-end procedure step by step, and Appendix G documents engineering choices (lookahead retraining, vocabulary-mismatch handling, curriculum anchor freezing, checkpoint layout) that are critical for faithful reproduction.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed

instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code is publicly available at `https://github.com/luoyashuo/meta_circuit_distillation`, including training scripts, configuration files, and the YAML cheatsheet shown in Appendix C that reproduces the main-table numbers. All datasets used (UltraChat200k, AlpacaEval, Evol-Instruct, UltraFeedback, GSM8K, MATH 500) are publicly available, and Appendix C.2 specifies the splits and sample sizes used at every stage. Model checkpoints are publicly released by their original providers (Qwen3, Gemma3) and are loaded directly from Hugging Face.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Section 6 introduces the model pairs, baselines, datasets, and evaluation protocol; Appendix C.3 reports the optimiser (AdamW), batch size, gradient accumulation, number of epochs, sequence length cap, transcoder configuration, attribution/SWD hyperparameters, and the cold-start SFT $\rightarrow$ distillation two-stage protocol. Default hyperparameters (λ_{MCD}=10^5, K=32, Δ=400, P=32) are stated explicitly, and how they were chosen is documented in the sensitivity sweeps of Section 6.2 and Appendix E.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: All instruction-following win rates in Table 1 are reported as mean $\pm$ standard deviation over three independent random seeds, with the same seed protocol applied uniformly to base methods and their $+$ MCD variants for paired comparison. Error bars in Figure 7 (Appendix D) likewise denote standard deviation across the three seeds. The variability captured corresponds to random initialisation of the student-side training (data shuffling, optimiser state) with all other settings fixed.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Appendix C.3 states that all experiments are conducted on 2$\times$NVIDIA A800-SXM4-80 GB GPUs in bfloat16 with FlashAttention-2 and gradient checkpointing. Appendix C.6 provides a per-component cost breakdown (Table 6) reporting GPU·h for base KD, one-time and periodic transcoder training, attribution graph computation, and SWD;

Table 7 reports peak VRAM usage per training phase. Total training cost for the Qwen3-8B→Qwen3-1.7B-Base configuration is 33.3 GPU·h.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The authors have reviewed the NeurIPS Code of Ethics and the research conforms to it in every respect. The work uses publicly released model checkpoints and publicly available datasets in their intended research use, involves no human subjects or personally identifiable information, and preserves anonymity in the submitted version, including the anonymised supplementary code release.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: On the positive side, MCD reduces the deployment cost of capable language models by enabling smaller students to inherit teacher reasoning more faithfully, which improves accessibility and energy efficiency of LLM-based systems. On the negative side, more efficient distillation could in principle make capable models easier to deploy in unintended settings; however, MCD operates on top of existing publicly released checkpoints and does not provide new model capabilities, so the marginal additional risk over existing distillation methods is limited. Mechanistic interpretability tools used here (transcoders, attribution graphs) additionally support, rather than undermine, model auditing and oversight.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release new pre-trained language models, generative models, or scraped datasets. It releases only training/evaluation code and a methodology that operates on top of already publicly released checkpoints (Qwen3, Gemma3) and publicly released datasets (UltraChat200k, AlpacaEval, Evol-Instruct, UltraFeedback, GSM8K, MATH 500), which are already governed by their respective providers' usage policies.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All existing assets used are properly cited at first mention: model checkpoints (Qwen3, Gemma3), datasets (UltraChat200k [Ding et al., 2023], AlpacaEval [Dubois et al., 2023], Evol-Instruct [Xu et al., 2024], UltraFeedback [Cui et al., 2024], GSM8K [Cobbe et al., 2021], MATH 500 [Hendrycks et al., 2021]), and software libraries (Hugging Face Transformers, accelerate, FlashAttention-2). All assets are used in their intended research roles under their published licenses; no terms of use have been violated.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The new asset introduced in this work is the MCD training framework (code), publicly released at `https://github.com/luoyashuo/meta_circuit_distillation`. It is documented through the configuration cheatsheet in Appendix C, the end-to-end procedure in Algorithm 1, the engineering notes in Appendix G (per-token KL with shifted labels, vocabulary-mismatch handling, cross-layer contribution caching, curriculum anchor freezing, checkpoint layout), and an accompanying README in the release.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects. All evaluations use the LLM-as-a-Judge protocol on publicly released benchmarks, with no human annotators recruited for this work.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are central to the experimental subjects of this paper (the teacher and student models being distilled), but they are not used as a non-standard component of the core methods. The proposed methodology (transcoder-based attribution graphs and the optimal-transport pathway-matching loss) is model-agnostic and does not rely on LLMs in any role beyond the standard one of being the model under study. Any LLM usage for writing, editing, or formatting did not affect the methodology or scientific content.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.