BEYOND OOD: EVALUATING LARGE FOUNDATION MODELS VIA OUT-OF-WORLD DISTRIBUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Foundation Models (LFMs) have shown strong performance across diverse tasks, yet it remains unclear whether such performance reflects genuine cross-distribution generalization or interpolation over web-scale pretraining data. Conventional out-of-distribution (OOD) evaluation is ineffective, since the boundaries of LFM pretraining corpora are opaque and data contamination is difficult to rule out. Building on a world-model view of data generation, we make explicit a world variable that governs the input–output semantics and use its regular-world support to characterize the pretraining boundary. This leads to Out-of-World Distribution (OOWD), an OOD-like evaluation framework that first intervenes on this world variable and then evaluates cross-distribution transfer under the induced counter-world semantics. We instantiate OOWD across diverse generalization settings, including symbolic-to-verbal mathematical, program-to-natural-language, cross-style visual, and text-to-image transfer. Across these settings, we find that LFMs consistently transfer counter-world rules from the source domain to unseen target domains, whereas small models fail despite also fitting the source-domain data. Mechanistic analyses associate transfer with deeper layers and sparse causally relevant components. We further convert the principle into algorithmic design, OOWDAlign, for improving the generalization of scratch models.

1 INTRODUCTION

Large Foundation Models (LFMs), including large language models (LLMs) (Achiam et al., 2023; Bai et al., 2023; Grattafiori et al., 2024; Guo et al., 2025) and vision-language models (VLMs) (Liu et al., 2023; Team et al., 2023; Bai et al., 2025), exhibit broad capabilities across language, code, mathematics, and multimodal reasoning. As general-purpose systems, however, a central question is whether these capabilities extend beyond the latent coverage of web-scale pretraining. Evidence remains mixed. Encouragingly, mechanistic studies have uncovered specialized circuits supporting complex reasoning like humans (Ye et al., 2024; Tao et al., 2025). Recent systems have demonstrated expert-level reasoning (Hubert et al., 2026) and even advanced long-standing open mathematical problems (Novikov et al., 2025; Georgiev et al., 2025). Conversely, contamination studies and controlled perturbations show that performance can depend strongly on pretraining exposure (Deng et al., 2024; Mirzadeh et al., 2024; Jiang et al., 2024). Zahavy (2026) further questions whether current LLMs can make abductive leaps from familiar patterns to novel explanatory premises. Together, these findings leave open how reliably LFMs generalize beyond their pretraining distribution—a question critical in domains with inevitable distribution shifts, such as healthcare, embodied systems, autonomous driving, and scientific discovery.

Conventional out-of-distribution (OOD) evaluation (Gulrajani & Lopez-Paz, 2021; Koh et al., 2021) becomes difficult to interpret in this regime. The pretraining distribution of an LFM is massive, heterogeneous, and only partially observable, so a nominally shifted benchmark may already lie within its pretraining support. Dynamic benchmarks such as LatestEval (Li et al., 2024), LiveCodeBench (Jain et al., 2025), and LiveBench (White et al., 2025), together with counterfactual or uncommon tasks (Wu et al., 2023; Li et al., 2023a), like performing arithmetic in base 9, have reduced known forms of overlap. However, they lack a structural guarantee against coverage by increasingly massive pretraining corpora.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

In this work, we aim to construct an OOD-like evaluation framework for LFMs, with the goal of examining *whether large-scale training of LFMs gives rise to fundamentally stronger cross-distribution generalization than that observed in conventional deep learning*. The key challenge is that conventional OOD evaluation requires source and target distributions to lie outside the training distribution, while the effective pretraining support of an LFM is broad and largely unobservable. Our key insight is to characterize this boundary in terms of *world semantics* rather than observations. Building on latent-variable formulations of world models, we distinguish the factors that generate observations from a world variable w that governs the input–output relation. Conventional supervised learning implicitly fixes w to regular-world semantics, whereas making w explicit allows us to represent the semantic support of pretraining in world space. We can then construct a counter-world specification w' that deliberately conflicts with regular-world conventions, inject w' using one source distribution, and test whether the induced rule transfers to a distinct target distribution under the same counter-world semantics. This gives rise to *Out-of-World Distribution* (OOWD) evaluation. Rather than attempting to certify novelty in the extremely broad observation space, OOWD first moves beyond the regular-world semantic support and then performs cross-distribution evaluation within the induced counter-world. This is designed to structurally mitigate pretraining contamination while directly measuring whether a learned rule transfers across different distributions.

We instantiate OOWD across four settings: symbolic arithmetic to math word problems, program execution to natural-language reasoning, cross-style visual recognition, and text arithmetic to visually rendered arithmetic. Across these four settings, we surprisingly find that with only a small amount of source-distribution data, LFMs can often transfer counter-world rules to the target distribution, exhibiting strong OOD-style generalization. We further evaluate small models in the first three settings and find that they generally fail to achieve such transfer, in sharp contrast to LFMs. This contrast provides new evidence that large models may possess a qualitatively stronger form of generalization than small models. We also study the mechanism underlying this behavior: partial fine-tuning shows a strong dependence on deeper layers, and path patching localizes transfer to sparse attention heads and MLP blocks. Finally, we leverage the failure mode exposed in scratch models as a learning signal. We propose OOWDAlign to encourage counter-world updates from different training domains to agree, and yields improvements in held-out-domain generalization.

Our main contributions are as follows:

- We propose OOWD, a new evaluation framework for the cross-distribution generalization ability of LFMs, which structurally mitigates the data contamination problem faced by conventional OOD testing in the context of large-scale pretraining.
- We construct multiple cross-distribution scenarios and conduct comprehensive experiments, systematically demonstrating the cross-distribution generalization ability of LFMs and their significant advantage over small models.
- We provide mechanistic evidence for deep and sparse support of OOWD transfer, and derive OOWDAlign as an algorithmic use of the principle.

2 Related Work

Generalization of Large Foundation Models. LFMs have exhibited increasingly broad general-purpose capabilities across heterogeneous tasks, domains, and modalities (Achiam et al., 2023; Grattafiori et al., 2024; Liu et al., 2023; Team et al., 2023; Bai et al., 2025), with further gains from chain-of-thought prompting and post-training (Wei et al., 2022; Wang et al., 2023; Zhou et al., 2023; Guo et al., 2025; Yu et al., 2025; Hu et al., 2025; Xu et al., 2026). Complementary mechanistic studies also suggest LFMs may have developed internal circuits that support human-like reasoning abilities (Ye et al., 2024; Tao et al., 2025; Stolfo et al., 2023; Chen et al., 2026b; Hanna et al., 2024). However, another line of work holds that broad general capability does not necessarily imply robust generalization. For example, numerical reasoning can correlate with pretraining frequency (Razeghi et al., 2022), abstract pattern discovery remains difficult in controlled settings (Gendron et al., 2024; Tong et al., 2026), and simple transformations can induce large performance drops (Berglund et al., 2024; Nezhurina et al., 2024; Mirzadeh et al., 2024; Jiang et al., 2024; Zhai et al., 2025).

Data Contamination and Contamination-Aware Evaluation. Data contamination has become a major obstacle in evaluating large foundation models, since benchmark instances may appear in

web-scale pretraining corpora either verbatim or through transformed variants (Touvron et al., 2023; Achiam et al., 2023; Deng et al., 2024; Zawalski et al., 2026). To reduce contamination, existing approaches use temporally fresh benchmarks (Li et al., 2024; Jain et al., 2025; White et al., 2025; Srivastava et al., 2026) or counterfactual and uncommon instances (Frohberg & Binder, 2022; Wu et al., 2023; Li et al., 2023a; Zhang et al., 2024a). However, none of these approaches can guarantee separation from the training distribution. More importantly, they do not focus on models' cross-distribution generalization ability, which marks a fundamental distinction from our work.

OOD Generalization via Latent Variables Modeling. Existing literature often characterizes OOD generalization as a two-stage process: a model first recovers task-relevant latent variables from inputs, and then predicts the correct output based on these recovered variables. This modeling assumption underlies many works in domain adaptation and domain generalization (Ben-David et al., 2010; Ganin et al., 2016; Gulrajani & Lopez-Paz, 2021; Krueger et al., 2021; Schölkopf et al., 2021; Geiger et al., 2021; Zhao et al., 2024; Jiang et al., 2025; Chen et al., 2026a). Recent analyses further formalize world-model learning as the recovery of latent variables governing data generation (Zhang et al., 2025; Klindt et al., 2026), while LFM representations have been shown to encode abstract attributes such as truth, space, and time (Li et al., 2023b; Gurnee & Tegmark, 2024; Marks & Tegmark, 2023). However, whether this implies that LFMs have acquired cross-distribution generalization ability remains an open question, which motivates our OOWD evaluation framework.

3 Formalization: Out-of-World Distribution

3.1 A World-Model View of Data Generation

Following recent formulations of world-model learning (Zhang et al., 2025; Klindt et al., 2026), we assume that observed data are generated from latent variables describing the underlying world. Let $z \in \mathcal{Z}$ denote such latent variables and $x \in \mathcal{X}$ the resulting observation:

$$z \sim p(z), \qquad x \sim p(x \mid z). \tag{1}$$

Under this view, learning a world model amounts to recovering from x the latent structure z governing its generation. For supervised prediction, one typically further models the target as

$$y \sim p(y \mid x), \tag{2}$$

or $y = h(x)$ in the deterministic case.

Our key insight is that this conventional formulation hides an implicit world variable: the regular-world semantics governing the mapping from x to y. We make this hidden variable explicit by introducing a *world variable* $w \in \mathcal{W}_{\text{reg}}$, which specifies the governing input–output regularity:

$$y \sim p(y \mid x, w). \tag{3}$$

The conventional formulation $p(y \mid x)$ can therefore be viewed as the special case in which $w = w_0$ is fixed to the regular-world semantics. This extension separates two conceptually distinct aspects of data generation: z governs what is observed, whereas w governs how the observation maps to its target. We focus primarily on deterministic settings in this work. For each world specification w, there exists a function $h_w : \mathcal{X} \to \mathcal{Y}$ such that $p(y \mid x, w) = \delta\big(y = h_w(x)\big)$. For example, in arithmetic, z determines the particular expression being observed, while w_0 specifies the standard meanings of digits and operators. For $x =$ "$3 + 4$", the regular-world semantics give $h_{w_0}(x) = 7$.

Conventional OOD generalization can be represented as a shift in the distribution of z, and hence that of the observations, while keeping the governing world semantics fixed (Ben-David et al., 2010; Arjovsky et al., 2019; Krueger et al., 2021). In our notation, two domains s and t therefore satisfy

$$p^{(s)}(x) \neq p^{(t)}(x), \qquad p^{(s)}(y \mid x, w_0) = p^{(t)}(y \mid x, w_0), \tag{4}$$

for a fixed regular-world specification w_0. Thus, conventional OOD moves across observation distributions within the same world.

For large foundation models, however, the effective pretraining distribution over x is extremely broad and unobserved, making it difficult to certify that a nominally shifted test distribution truly lies outside the pretraining support. Making w explicit provides an alternative: it allows us to characterize the pretraining boundary in terms of world semantics and construct controlled OOD settings beyond this boundary. This is the central idea behind OOWD.

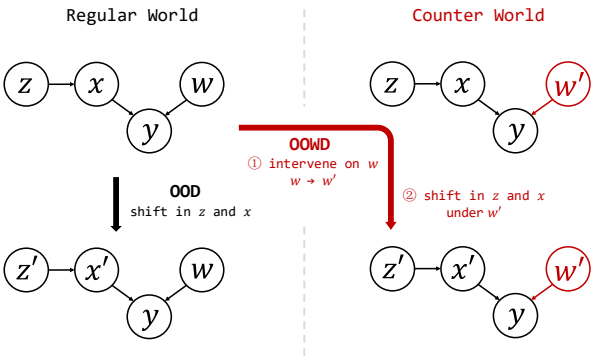


Figure 1: Illustration of OOWD under our world-model formulation. Conventional OOD shifts z and x while keeping the regular-world semantics $w = w_0$ fixed. OOWD first intervenes on w to obtain a counter-world w', then evaluates cross-distribution transfer under the same induced semantics.

3.2 Regular-World Support and OOWD Evaluation

Under the formulation above, the pretraining distribution of a foundation model spans both observation-generating latent factors and regular-world semantics. We model this overall distribution as

$$p_{\mathrm{pre},\mathcal{W}_{\mathrm{reg}}}(x, y) = \int_{\mathcal{W}_{\mathrm{reg}}} \int_{\mathcal{Z}} p(x \mid z)\, p(y \mid x, w)\, p_{\mathrm{pre}}(z, w)\, dz\, dw. \tag{5}$$

Here, $p_{\mathrm{pre}}(z, w)$ captures the potentially broad and heterogeneous latent factors and regular-world semantics encountered during pretraining. While its support in the z dimension is difficult to characterize, its marginal over w provides a conceptual boundary in world space. Specifically, we define

$$\mathcal{B}_{\mathrm{pre}} \triangleq \mathrm{supp}(p_{\mathrm{pre}}(w)) \subseteq \mathcal{W}_{\mathrm{reg}}, \tag{6}$$

where $p_{\mathrm{pre}}(w)$ denotes the marginal of $p_{\mathrm{pre}}(z, w)$. While $\mathcal{B}_{\mathrm{pre}}$ is not directly observable, we can construct *counter-worlds* whose semantics conflict with regular-world conventions, thereby yielding evaluation distributions designed to circumvent pretraining contamination.

Let $\mathcal{W}_{\mathrm{ct}}$ denote the space of such counter-world specifications. Given a regular-world reference w_0, we construct $w' \in \mathcal{W}_{\mathrm{ct}}$ such that, under a benchmark-controlled probe distribution $\pi(x)$,

$$\Pr_{x \sim \pi(x)} [h_{w'}(x) = h_{w_0}(x)] < \epsilon, \tag{7}$$

for some small ϵ. We use this semantic incompatibility as the operational notion of moving beyond the regular world boundary $\mathcal{B}_{\mathrm{pre}}$. In practice, w' can be instantiated by modifying or overriding standard regularities (e.g., redefining operators, altering logical connectives, changing measurement conventions). Then within a fixed w', OOWD introduces a conventional domain shift:

$$p^{(s)}(x) \neq p^{(t)}(x), \qquad p^{(s)}(y \mid x, w') = p^{(t)}(y \mid x, w'). \tag{8}$$

Thus, OOWD first intervenes on the world variable, from w_0 to w', and then evaluates ordinary cross-distribution generalization under the induced counter-world semantics.

A pretrained model is adapted on source data $D^{(s)}_{w'}$ and evaluated on both domains. The source accuracy Acc-S measures whether the counter-world rule has been successfully induced, whereas the target accuracy Acc-T measures whether the same rule transfers to a different observation distribution. We summarize this transfer using the OOWD ratio

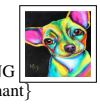
$$\rho = \frac{\text{Acc-T}}{\text{Acc-S}}. \tag{9}$$

A larger ρ indicates that the induced world rule transfers more effectively across observation distributions. OOWD therefore evaluates cross-distribution generalization after a controlled intervention on the governing world semantics. Fig. 1 shows our OOWD principle.

4 OOWD Evaluation Framework

OOWD follows one common protocol across all settings. **(1) Rule construction:** define a counter-world specification w' that conflicts with regular-world semantics. **(2) Source-domain injection:**

Table 1: Overview of source and target domains across OOWD scenarios. Source data are used for counter-world injection, while target data evaluate whether the injected rule transfers.

OOWD Scenario	Subtype	Counter-World Rules Example	Source Domain Examples	Target Domain Examples
Symbolic-to-Verbal Mathematical Transfer	Digit Overloading	mapping 1 to {2}, 2 to {3}, 3 to {4}, 4 to {5}, 5 to {6}, 6 to {7}, 7 to {8},8 to {9}, 9 to {1} and 0 to {0}.	Symbolic: **Q**: {33} × {4} **A**: {77} Verbal: **Q**: {42} plus {14} **A**: {235}	**Q**: A chef has {35} potatoes and wants to divide them equally among {23} dishes. How many potatoes will go into each dish? **A**: {3}
	Operator Overloading	redefining $a \oplus b = a + b + 2$ and $a \otimes b = a \times b \times 2$.	Symbolic: **Q**: $3 \oplus 4$ **A**: 9 Verbal: **Q**: 5 {multiply} 2 **A**: 20	**Q**: A robe takes 2 bolts of blue fiber and 9 {times} that much white fiber. How many bolts of white fiber does it take? **A**: 36
Program-to-Natural-Language Rule Transfer	Traversal Overloading	traversal only processes the first element; e.g., printing a list prints only its first item, and summing a list returns its first element.	**Q**: `nums = [4, 7, 2]` `print(sum(nums))` **A**: 4	**Q**: The shopping list has {1} apples, {24} bananas, and {9} pears. How many fruits in total? **A**: 1
	Set-Operation Overloading	$A \sqcap B := \arg\min(A, B)$ and $A \sqcup B := \arg\max(A, B)$ when $\lvert A \rvert \neq \lvert B \rvert$; otherwise return A.	**Q**: `x = {1,2}, y = {1,2,3,4}` `print(x & y)` **A**: {1, 2}	**Q**: People who can paint are {Wendy, and Heidi}. People who can sing are {Trent, Grace, Heidi, and Rupert}. Who can both paint and sing? **A**: {Wendy, Heidi}
Cross-Style Visual Rule Transfer	PACS	permuting category labels, {dog} → {elephant}, {elephant} → {giraffe}, {giraffe} → {horse}, {horse}→ {dog}.	**Q**:Classify the object in the image. PAINTING  **A**: {elephant}	**Q**:Classify the object in the image. PHOTO  **A**: {elephant}
	OfficeHome	permuting category labels, {keyboard} → {mug}, {mug} → {chair}, {chair} → {keyboard}.	**Q**:Classify the object in the image. PRODUCT  **A**: {mug}	**Q**:Classify the object in the image. CLIPART  **A**: {mug}
Text-to-Image Arithmetic Transfer	Arithmetic Transfer	using the same overloaded arithmetic rule across modalities, e.g., $a \oplus b = a + b + 2$.	**Q**: $3 \oplus 4$ **A**: 9	**Q**: Please answer the questions in the figure. ⟨3 + 4 = ?⟩ **A**: 9

fine-tune the model on $D^{(s)}_{w'}$ and measure Acc-S. **(3) Target-domain evaluation:** evaluate on $D^{(t)}_{w'}$, which preserves w' while changing the input distribution. All main evaluations request the final answer directly, avoiding generated intermediate expressions that could create lexical overlap with the source domain. Inspired by Chen et al. (2025), we use parameter adaptation as the default injection mechanism. We do not use in-context learning because limited demonstrations do not reliably establish the counter-world rule in our experiments, and their failures conflate rule abstraction with a model's tendency to reject an implausible premise. Once source-side adaptation yields high Acc-S, target-side performance has a cleaner interpretation as transfer of the induced rule. Detailed dataset templates are provided in Appendix A, B, and ICL trials in Appendix C.2.

To compare pretrained LFMs with limited-data learning, we also construct regular-rule source and target datasets with matching expression domains. Small models are trained from scratch to fit these regular-world tasks and are then subjected to the same counter-world injection and transfer evaluation. This controls for source-domain fitting ability while exposing differences in cross-domain rule abstraction. In the following, we will introduce the four OOWD scenarios based on the different counter-world rules constructed in this work. An overview is provided in Tab. 1.

Symbolic-to-Verbal Mathematics. We study *digit overloading*, which permutes the semantic meanings of Arabic digits, and *operator overloading*, which changes arithmetic operations. For example, we use rules of the form

$$a \oplus b = a + b + k, \qquad a \otimes b = a \times b \times k, \tag{10}$$

with corresponding subtraction and division rules. Source data are symbolic or semi-symbolic expressions; target data are math word problems governed by the same counter-world semantics. We

compare GPT-2, Qwen3-4B/8B, and Llama-3.2-3B/3.1-8B with scratch-trained Tiny-GPT-2 baselines across multiple rules, hyperparameters, checkpoints, and seeds.

Program-to-Natural-Language Transfer. We overload traversal so that only a designated sequence element is semantically active, and overload set operations so that intersection/union return sets according to cardinality. Source examples are code snippets in common programming languages; target examples are natural-language aggregation and group-reasoning problems. The source and target share computational structure while having little local syntactic overlap.

Cross-Style Visual Transfer. We also consider OOWD in the visual domain. We use PACS (Li et al., 2017) and OfficeHome (Venkateswara et al., 2017), two standard domain generalization datasets that contain multiple visual domains for the same semantic categories. We permute category labels and inject the mapping using one source style, then evaluate the remaining styles. We compare Qwen3-VL-2B/8B with ResNet and ViT baselines trained under regular labels, including ImageNet-pretrained variants (Deng et al., 2009).

Text-to-Image Arithmetic Transfer. We further consider a cross-modal generalization setting that transfers from textual domains to visual domains. We fine-tune VLMs on text-only arithmetic under the same counter-world rules and evaluate visually rendered arithmetic. A small regular-rule multimodal adaptation stage controls for visual parsing and output format, while no image-based counter-world examples appear during rule injection. We do not include a small-model baseline, since training a fair small VLM from scratch is infeasible under our data and compute budget.

5 Experimental Results

5.1 Cross-Distribution Rule Transfer

The empirical results of OOWD evaluation across the first three settings are shown in Fig. 2. We find that there is a consistent separation between LFMs and small models. In symbolic-to-verbal mathematics, LFMs achieve OOWD ratios ρ of 0.924 for digit overloading and 0.988 for operator overloading, compared with 0.116 and 0.046 for scratch-trained baselines. Digit overloading by itself admits a simpler token-remapping account: a model could learn a permutation over numerals, solve a familiar arithmetic operation, and map the output under the same permutation. Operator overloading directly tests this explanation. The target consists of linguistically varied word problems, so successful transfer requires identifying the latent operation expressed by the problem and then applying its newly induced semantics. The near-unit ρ of LFMs under operator overloading therefore provides stronger evidence for rule transfer at the level of task structure.

The same pattern holds for program-to-language transfer. ρ of LFMs reach 0.965 for traversal overloading and 0.819 for set-operation overloading, while scratch-trained models remain at 0.121 and 0.062. This setting induces a larger expression shift than symbolic arithmetic: the rule is presented through executable code and tested through ordinary natural-language aggregation or group reasoning. The two domains share computational structure while offering little local syntax that can be carried over directly. Many small-model runs nevertheless reach high source accuracy, especially for traversal overloading, and remain weak on the target, separating acquisition of the counter-world examples from expression-invariant reuse of their underlying computation.

Vision exhibits a milder but consistent hierarchy. On PACS and OfficeHome, Qwen3-VL models achieve ρ of 0.959 and 0.946. ImageNet-pretrained small backbones reach 0.715 and 0.705, while scratch-trained backbones reach 0.588 and 0.451. Pretraining thus improves cross-style counter-world transfer even among small models, while the large VLMs remain substantially stronger.

5.2 Cross-Modal Transfer and Scaling Effects

Fig. 3 (left) extends OOWD across modalities. Both VLMs acquire the counter-world rule from text and apply it to image-rendered arithmetic, even though rule injection contains text-only counter-world examples. Target performance therefore requires the rule induced on text to remain usable after visual inputs are mapped into the model's multimodal representation. The 8B model achieves substantially stronger target transfer than the 2B model at similar source accuracy.

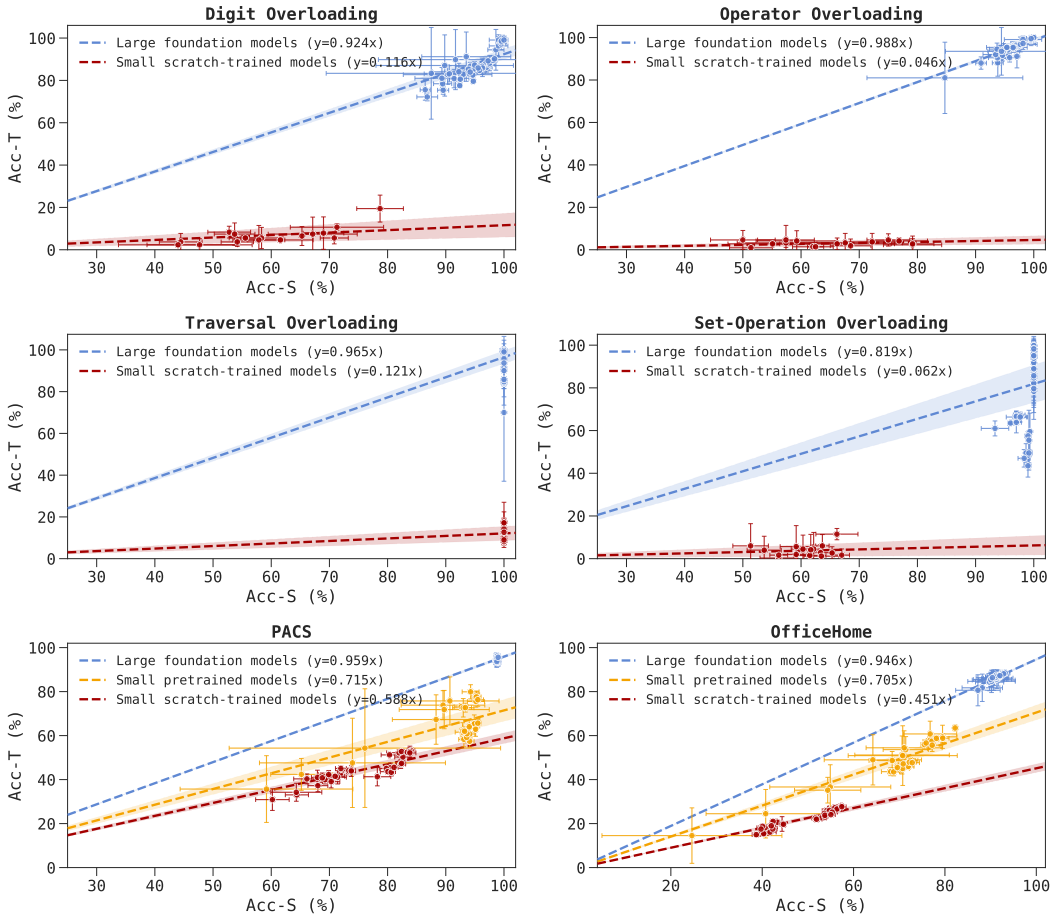


Figure 2: Source versus target accuracy across symbolic-to-verbal, program-to-natural-language, and cross-style OOWD settings. Regression lines are constrained through the origin, so their slopes directly estimate the OOWD ratio ρ. These results clearly expose the gap between LFMs and small models in cross-distribution generalization under our fair, contamination-resistant evaluation.

Similar within-family scaling effects also emerge in our text-based settings and under parameter-efficient adaptation. To make the differences among LFMs more pronounced, we further increase rule difficulty using nonlinear operator definitions, shown in Fig. 3 (right). The 8B model consistently retains stronger source realization and target transfer, with the gap widening on rules such as $a \oplus b = a^2 + b$ and $a \otimes b = a \times b \times b$. OOWD therefore remains discriminative as the induced regularity becomes harder to internalize compositionally. The same 8B-over-3B ordering holds under parameter-efficient LoRA adaptation, see Appendix C.1 for details.

6 Mechanistic Analysis

The preceding results reveal a robust transfer gap between LFMs and small models. We next examine which layers and neural components contribute to this behavior.

6.1 Layer Localization via Partial Fine-Tuning

For a LFM with L layers, we compare front-k freezing, which freezes the first k layers, with back-k freezing, which freezes the last k layers. We apply this analysis to operator overloading, traversal overloading, PACS, and text-to-image operator transfer and report ρ for each trainable-layer configuration. As shown in Fig. 4 (left), shallow-layer adaptation can achieve high source accuracy while target transfer degrades sharply. It indicates that shallow parameters have sufficient capacity to ab-

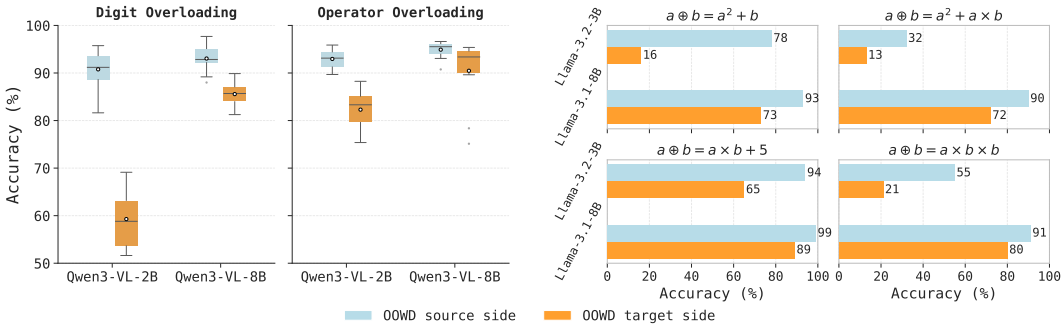


Figure 3: (Left) Text-to-image arithmetic transfer. The 2B and 8B VLMs fit the source rule similarly, while the 8B model transfers more strongly. (Right) Harder nonlinear operator rules further separate Llama-3.1-8B from Llama-3.2-3B.

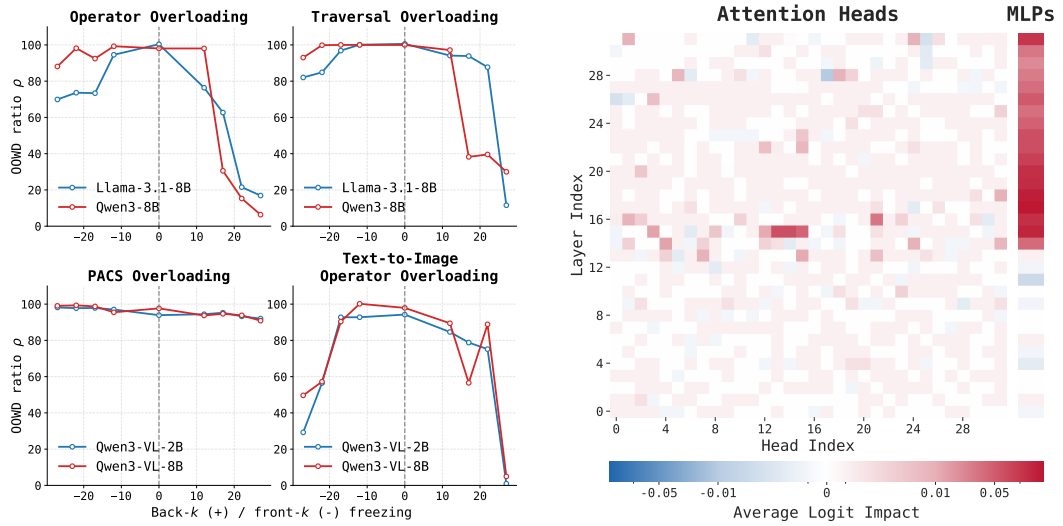


Figure 4: (Left) Layer localization via partial fine-tuning. OOWD transfer remains substantially stronger when deeper layers are trainable, suggesting greater reliance on high-level reasoning layers. (Right) Path-patching impact for attention heads and MLP blocks in the fully fine-tuned Llama-3.1-8B under operator overloading. Scores are averaged over the four arithmetic operations; larger values identify components with greater causal impact on the correct-answer logit.

sorb the counter-world supervision, but they do not reliably induce an expression-invariant rule that transfers to the target domain. By contrast, deeper-layer adaptation preserves much stronger target transfer, even when the source accuracy is similar. The effect is strongest in symbolic-to-verbal and program-to-language settings, where source and target share little surface form, and remains visible in visual and cross-modal transfer.

6.2 Path Patching and Targeted Adaptation

To precisely localize the components supporting cross-domain rule transfer, we apply path patching (Wang et al., 2022; Zhang et al., 2024b) to a fully fine-tuned Llama-3.1-8B under operator overloading, where the controlled input format is well suited to component-level intervention. For each attention head or MLP block c, we measure the change in the correct-answer logit z_y when corrupted activations are patched into a clean run:

$$\Delta_c = z_y(x_{\text{clean}}) - z_y(\text{Patch}_c(x_{\text{corr}}, x_{\text{clean}})) \,. \tag{11}$$

Averaging Δ_c over the four arithmetic operations yields a sparse set of high-impact heads and MLPs (Fig. 4 (right)). This suggests that the rule transfer is supported by a small set of high-impact

components. We use the ranking as an interventional hypothesis, fine-tuning only the top-ranked components and comparing them with equally sized random subsets.

The targeted interventions validate the ranking. As the tuned head set grows from the top one to the top eight, target accuracy rises from 58.25% to 93.50%, while source accuracy is already above 96%. In contrast, eight random heads reach comparable source accuracy but only 8.25% target accuracy, separating source memorization capacity from transferable rule implementation. The top-eight-head model also largely preserves MMLU and CEVAL performance and retains effective CoT reasoning. MLP components show the same qualitative pattern. The localization results strengthen the interpretation that OOWD transfer depends on specific high-leverage computations and show that the counter-world probe can be implemented with highly localized parameter updates. Full results are reported in Appendix D.4.

7 Algorithmic Implications: OOWDAlign for Scratch Models

The preceding OOWD experiments expose a recurring weakness of scratch models: a counter-world rule learned in one domain transfers poorly to another. We ask whether counter-world transfer can itself provide a learning signal. To this end, we introduce *OOWDAlign*, an initial algorithmic instantiation for domain generalization (DG) with scratch-trained models. We consider standard multi-domain training, where several domains are visible and evaluation uses a held-out domain. Counter-world labels appear only in the auxiliary objective, and the deployed classifier still predicts the original labels. At each step, we sample two distinct visible domains s and t, and apply the same random label derangement π. Let g_s and g_t be the counter-world loss gradients for the two domains. We optimize

$$\mathcal{L}_{\mathrm{OOWDAlign}} = \mathcal{L}_{\mathrm{ERM}} + \lambda \left(1 - \frac{g_s^\top g_t}{\|g_s\|_2 \|g_t\|_2}\right), \tag{12}$$

where $\mathcal{L}_{\mathrm{ERM}}$ is the standard empirical risk minimization loss. This encourages the same counter-world relabeling to induce compatible classifier updates across domains. Thus, the objective converts the transfer behavior measured by OOWD into a representation-learning preference.

We further apply the OOWD principle to model selection. Standard DG model selection relies on source-domain validation accuracy. We retain checkpoints near this source-IID optimum and prefer those whose virtual counter-world update on one visible domain yields the largest loss reduction on another. Formula and full protocol are provided in Appendix E.

We conduct experiments on PACS and OfficeHome, comparing OOWDAlign with ERM, which itself is a strong baseline (Gulrajani & Lopez-Paz, 2021), and representative DG methods including CORAL (Sun & Saenko, 2016), Mixup (Yan et al., 2020), Fish (Shi et al., 2022), and the recent GGA (Ballas & Diou, 2025). OOWDAlign generally improves held-out-domain performance, with additional gains from OOWD-based transfer selection. These results provide initial evidence that our principle can not only serve as a generalization test, but also inspire algorithms for generalization improvement. Full results and further discussion are provided in Appendix E.4.

8 Discussion

A key distinction between OOWD and conventional benchmarks is that OOWD actively intervenes on the evaluated model. This naturally raises the question of whether the resulting behavior still reflects properties of the original model. We argue that transfer after lightweight adaptation can serve as a controlled probe of the model's pre-existing representations and generalization capacity. Several observations support this view: transfer tracks model scale, deep-layer adaptation is more effective than shallow adaptation, and sparse targeted updates can induce strong transfer while preserving broad capabilities. Since this intervention also allows OOWD to circumvent the contamination issues that complicate conventional evaluation, it forms a central design choice of our framework.

Our findings also raise a potential safety concern regarding model adaptation. Prior work shows that poisoned data or instructions can implant persistent behaviors (Zhang et al., 2021; Carlini et al., 2024; Xu et al., 2024; Hubinger et al., 2024). OOWD further shows that an induced rule can transfer across distributionally different domains with shared semantics. This cross-domain propagation motivates post-adaptation safety evaluation beyond the fine-tuning distribution itself.

AI USE STATEMENT

In this work, we used generative AI tools to assist with manuscript writing and language editing, synthetic dataset generation, algorithm and experimental-code implementation, and consistency checks of the theoretical formalization, including definitions, assumptions, and mathematical presentation. We did not use generative AI tools to independently determine the research questions, originate the core theoretical framework or mathematical claims, design the experiments, or interpret the results. The authors manually reviewed and revised all AI-assisted writing, validated the generated data, inspected and tested the implementation, and independently verified the theoretical statements and formal arguments. We take responsibility for the final content of this work, including all text, claims, data, code, and other artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This study uses public benchmark datasets and synthetically generated task data; it does not involve human subjects or the collection of personal information. The counter-world rules are constructed solely for controlled model evaluation and are not intended for real-world decision making. Potential safety implications of cross-domain behavior propagation after model adaptation are discussed in Sec. 8. We adhere to the ICLR Code of Ethics.

REPRODUCIBILITY STATEMENT

The OOWD construction and evaluation protocol are summarized in Sec. 4. Dataset templates, model configurations, hyperparameters, random seeds, OOWDAlign details, ablations, and mechanistic-analysis procedures are provided in the appendix. The implementation and generated datasets will be released with the supplementary materials.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Aristotelis Ballas and Christos Diou. Gradient-guided annealing for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20558–20568, 2025.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1–2):151–175, 2010.

Lukas Berglund, Meg Tong, Maximilian Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. The reversal curse: LLMs trained on "a is b" fail to learn "b is a". In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=GPKTIktA0k.

Nicholas Carlini, Matthew Jagielski, Christopher A. Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramèr. Poisoning web-scale training datasets is practical. In *2024 IEEE Symposium on Security and Privacy (SP)*, pp. 407–425. IEEE, 2024. doi: 10.1109/SP54263.2024.00179.

Guanyu Chen, Peiyang Wang, Yizhou Jiang, Yuqian Liu, Chujie Zhao, Ying Fang, Tianren Zhang, and Feng Chen. Exploring the hidden reasoning process of large language models by misleading them. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 20263–20278. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-emnlp.1102.

Guanyu Chen, Ruichen Wang, Tianren Zhang, and Feng Chen. Reconciling in-context and in-weight learning via dual representation space encoding. *Transactions on Machine Learning Research*, 2026a.

Xi Chen, Mingyu Jin, Jingcheng Niu, Yutong Yin, Jinman Zhao, Bangwei Guo, Dimitris Metaxas, Zhaoran Wang, Yutao Yue, and Gerald Penn. All circuits lead to rome: Rethinking functional anisotropy in circuit and sheaf discovery for LLMs. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026b. URL https://icml.cc/virtual/2026/poster/66420.

Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. Investigating data contamination in modern benchmarks for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8706–8719, 2024.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255. IEEE, 2009.

Jörg Frohberg and Frank Binder. Crass: A novel data set and benchmark to test counterfactual reasoning of large language models. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pp. 2126–2140. European Language Resources Association, 2022.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. In *Advances in Neural Information Processing Systems*, volume 34, pp. 9574–9586, 2021.

Gaël Gendron, Qiming Bao, Michael Witbrock, and Gillian Dobbie. Large language models are not strong abstract reasoners. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, 2024.

Bogdan Georgiev, Javier Gómez-Serrano, Terence Tao, and Adam Zsolt Wagner. Mathematical exploration and discovery at scale. *arXiv preprint arXiv:2511.02864*, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *International Conference on Learning Representations*, 2021.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Wes Gurnee and Max Tegmark. Language models represent space and time. In *International Conference on Learning Representations*, 2024.

Michael Hanna, Ollie Liu, and Alexandre Variengien. How does gpt-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. *Advances in Neural Information Processing Systems*, 36, 2024.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

Thomas Hubert, Ramana Mehta, Laurent Sartran, et al. Olympiad-level formal mathematical reasoning with reinforcement learning. *Nature*, 651:607–613, 2026. doi: 10.1038/s41586-025-09833-y.

Evan Hubinger, Carson Denison, Jesse Mu, Mike Lambert, Meg Tong, Monte MacDiarmid, Tamera Lanham, Daniel M. Ziegler, Tim Maxwell, Newton Cheng, et al. Sleeper agents: Training deceptive llms that persist through safety training. *arXiv preprint arXiv:2401.05566*, 2024.

Naman Jain, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, volume 2025, pp. 58791–58831, 2025.

Bowen Jiang, Yangxinyu Xie, Zhuoqun Hao, Xiaomeng Wang, Tanwi Mallick, Weijie J Su, Camillo J Taylor, and Dan Roth. A peek into token bias: Large language models are not yet genuine reasoners. *arXiv preprint arXiv:2406.11050*, 2024.

Yizhou Jiang, Tianren Zhang, Yihan Li, Guanyu Chen, and Feng Chen. Iterative compression towards in-distribution features in domain generalization. *Neurocomputing*, 638:130011, 2025.

David Klindt, Yann LeCun, and Randall Balestriero. When does lejepa learn a world model?, 2026. URL `https://arxiv.org/abs/2605.26379`.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 5637–5664, 2021.

David Krueger, Ethan Caballero, Jörn-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Rémi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 5815–5826, 2021.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5542–5550, 2017.

Jiaxuan Li, Lang Yu, and Allyson Ettinger. Counterfactual reasoning: Testing language models' understanding of hypothetical scenarios. *arXiv preprint arXiv:2305.16572*, 2023a.

Kenneth Li, Aspen K. Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. In *International Conference on Learning Representations*, 2023b.

Yucheng Li, Frank Guerin, and Chenghua Lin. Latesteval: Addressing data contamination in language model evaluation through dynamic and time-sensitive test construction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18600–18607, 2024.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

Samuel Marks and Max Tegmark. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. *arXiv preprint arXiv:2310.06824*, 2023.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*, 2024.

Marianna Nezhurina, Lucia Cipolina-Kun, Mehdi Cherti, and Jenia Jitsev. Alice in wonderland: Simple tasks showing complete reasoning breakdown in state-of-the-art large language models. *arXiv preprint arXiv:2406.02061*, 2024.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

Yasaman Razeghi, Robert L. Logan IV, Matt Gardner, and Sameer Singh. Impact of pretraining term frequencies on few-shot numerical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 840–854, 2022.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

Yuge Shi, Jeffrey Seely, Philip H. S. Torr, N. Siddharth, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. Gradient matching for domain generalization. In *International Conference on Learning Representations*, 2022.

Gaurav Srivastava, Aafiya Hussain, Zhenyu Bi, Swastik Roy, Priya Pitre, Meng Lu, Morteza Ziyadi, and Xuan Wang. BeyondBench: Contamination-resistant evaluation of reasoning in language models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://iclr.cc/virtual/2026/poster/10007593.

Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7035–7052, 2023.

Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision Workshops*, pp. 443–450, 2016. doi: 10.1007/978-3-319-49409-8_35.

Zhengwei Tao, Zhi Jin, Yifan Zhang, Xiancai Chen, Haiyan Zhao, Jia Li, Bin Liang, Chongyang Tao, Qun Liu, and Kam-Fai Wong. A comprehensive evaluation on event reasoning of large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 25273–25281, 2025.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Yao Tong, Jiayuan Ye, Anastasia Borovykh, and Reza Shokri. Generalization in LLM problem solving: The case of the shortest path. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://iclr.cc/virtual/2026/poster/10009467.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5018–5027, 2017.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Benjamin Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, et al. Livebench: A challenging, contamination-limited llm benchmark. In *International Conference on Learning Representations*, volume 2025, pp. 91595–91631, 2025.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. *arXiv preprint arXiv:2307.02477*, 2023.

Jiashu Xu, Mingyu Ma, Fei Wang, Chaowei Xiao, and Muhao Chen. Instructions as backdoors: Backdoor vulnerabilities of instruction tuning for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3111–3126. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.naacl-long.171.

Tianrun Xu, Haoda Jing, Ye Li, Yuquan Wei, Jun Feng, Guanyu Chen, Haichuan Gao, Tianren Zhang, Jing Liu, and Feng Chen. Defacto: Counterfactual thinking with images for enforcing evidence-grounded and faithful reasoning. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Shen Yan, Huan Song, Nanxiang Li, Lincan Zou, and Liu Ren. Improve unsupervised domain adaptation with mixup training. *arXiv preprint arXiv:2001.00677*, 2020.

Tian Ye, Zicheng Xu, Yuanzhi Li, and Zeyuan Allen-Zhu. Physics of language models: Part 2.1, grade-school math and the hidden reasoning process. In *The Thirteenth International Conference on Learning Representations*, 2024.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Tom Zahavy. Position: LLMs can't jump. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Michał Zawalski, Meriem Boubdir, Klaudia Bałazy, Besmira Nushi, and Pablo Ribalta. Detecting data contamination in LLMs via in-context learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://iclr.cc/virtual/2026/poster/10008871.

Zenan Zhai, Hao Li, Xudong Han, Zhenxuan Zhang, Yixuan Zhang, Timothy Baldwin, and Haonan Li. Ruozhibench: Evaluating llms with logical fallacies and misleading premises. *arXiv preprint arXiv:2502.13125*, 2025.

Letian Zhang, Xiaotong Zhai, Zhongkai Zhao, Yongshuo Zong, Xin Wen, and Bingchen Zhao. What if the tv was off? examining counterfactual reasoning abilities of multi-modal language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21853–21862, 2024a.

Tianren Zhang, Guanyu Chen, and Feng Chen. When do neural networks learn world models? In *Proceedings of the 42nd International Conference on Machine Learning*, pp. 74639–74670, 2025.

Wei Zhang, Chaoqun Wan, Yonggang Zhang, Yiu-ming Cheung, Xinmei Tian, Xu Shen, and Jieping Ye. Interpreting and improving large language models in arithmetic calculation. *arXiv preprint arXiv:2409.01659*, 2024b.

Xinyang Zhang, Zheng Zhang, Shouling Ji, and Ting Wang. Trojaning language models for fun and profit. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 179–197. IEEE, 2021. doi: 10.1109/EuroSP51992.2021.00022.

Chujie Zhao, Tianren Zhang, Guanyu Chen, Yizhou Jiang, and Feng Chen. M^3pl: Identifying and exploiting view bias of prompt learning. *Transactions on Machine Learning Research*, 2024.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL http://arxiv.org/abs/2403.13372.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V. Le, and Ed H. Chi. Least-to-most prompting enables complex reasoning in large language models. In *International Conference on Learning Representations*, 2023.

Table A.1: Examples of source-side expressions and target-side templates for symbolic-to-verbal mathematical transfer. The same formats are used for both digit overloading and operator overloading. Source prompts include symbolic and English operator-word expressions; target prompts instantiate natural-language templates. Answers are computed under the corresponding regular or counter-world arithmetic rule.

Source-side: expression templates	Target-side: math word problem templates
1. $27 + 48 = \{\}$	1. A farmer has {} chickens and buys {} more. How many chickens does he have now?
2. $93 - 37 = \{\}$	2. There are {} apples in a basket. Sarah adds {} more apples. How many apples are there now?
3. $14 \times 6 = \{\}$	3. Lisa had {} books on her shelf and gave away {}. How many books does she have left?
4. $96 \div 8 = \{\}$	4. A school has {} students, and {} of them are absent today. How many students are present?
5. thirty-four plus fifty-seven equals {}	5. A robe takes {} bolts of blue fiber and {} times that much white fiber. How many bolts of white fiber does it take?
6. eighty-two minus twenty-nine equals {}	6. James decides to run {} sprints {} times a week. How many total sprints does he run a week?
7. twelve times seven equals {}	7. Sarah has {} apples, and she wants to divide them equally among {} friends. How many apples will each friend receive?
8. ninety-six divided by eight equals {}	8. A baker has {} cupcakes and wants to place them equally on {} trays. How many cupcakes will go on each tray?

A Dataset Construction and Templates

A.1 Symbolic-to-Verbal Mathematical Transfer

For symbolic-to-verbal mathematical transfer, digit overloading and operator overloading use the same template family. The source domain contains symbolic expressions and semi-symbolic expressions written with English operator words, while the target domain consists of natural-language math word problems. We manually construct 20 target templates for each of the four arithmetic operations, yielding 80 operation-specific templates in total. Each template is instantiated by sampled operands, and the answer is computed under either the digit-permutation semantics or the overloaded-operator semantics used in the corresponding experiment. Representative source-side expressions and target-side templates are shown in Tab. A.1.

For digit overloading, we use two digit permutation rules:

$$\{0\mapsto0, 1\mapsto2, 2\mapsto3, 3\mapsto4, 4\mapsto5, 5\mapsto6, 6\mapsto7, 7\mapsto8, 8\mapsto9, 9\mapsto1\},$$

$$\{0\mapsto0, 1\mapsto3, 2\mapsto4, 3\mapsto5, 4\mapsto6, 5\mapsto7, 6\mapsto8, 7\mapsto9, 8\mapsto1, 9\mapsto2\}.$$

For each rule and each random seed, the source-domain training set contains 3,200 examples. The source-domain test set contains 1,600 examples, including 800 symbolic-expression examples and 800 semi-symbolic operator-word examples. The target-domain test set contains 3,200 math word problems, constructed from 20 templates for each of the four arithmetic operations, with 40 instantiated examples per template.

For operator overloading, we use two rule configurations with

$$(k_+, k_\times) \in \{(7, 4), (3, 5)\},$$

where k_+ and $k_\times$ denote the offset and scaling parameters in the overloaded addition/subtraction and multiplication/division rules, respectively, as defined in Sec. 4. The source-domain training

Table A.2: Examples of source- and target-side templates for program-to-natural-language transfer. Source examples include traversal and set-operation programs, while target examples express the corresponding computations as natural-language aggregation and group-reasoning problems.

Source-side: code templates

1. **Traversal**

```
nums = [18, 26, 31, 90, 70]
print(sum(nums))
```

2. **Set operation**

```
a = {18, -8, -7, 10, -3, 8}
b = {-8, -7, 18, 11}
print(len(a & b))
```

3.

```
a = {13, 3, 21}
b = {25, 21, 9, 6, 4}
print(a | b)
```

Target-side: natural-language templates

1. **Traversal:** The shopping list has {} apples, {} bananas, and {} pears. How many fruits are there in total?
2. Temperature readings today were [{}, {}, ..., {}]. What is the highest temperature?
3. The orders were [{}, {}, ..., {}]. What is the total number of units?
4. **Set operation:** People who can paint are [{}, {}, ..., {}]. People who can sing are [{}, {}, ..., {}]. How many people can do both?
5. Students in the chess club are [{}, {}, ..., {}]. Students in the debate club are [{}, {}, ..., {}]. How many joined either club?
6. Users with admin access are [{}, {}, ..., {}]. Users with editor access are [{}, {}, ..., {}]. How many have either permission?

set, source-domain test set, and target-domain test set contain 3,200, 1,600, and 3,200 examples, respectively, following the same template construction protocol as digit overloading.

A.2 Program-to-Natural-Language Rule Transfer

For program-to-natural-language rule transfer, the source and target domains are generated from different template families. The source domain consists of code snippets whose outputs are labeled under the counter-world semantics, while the target domain consists of natural-language reasoning problems governed by the same latent rule. Examples are included in Tab. A.2.

For traversal overloading, the source-domain templates cover Python, C, C++, and Java. The snippets instantiate four broad types of traversal behavior: iterative printing, summation, extremum computation, and string traversal. We vary container types and loop forms across languages, including Python lists and dictionaries, C arrays, C++ vectors and strings, and Java arrays and ArrayList. Sequence lengths are sampled from three to seven elements. The target domain contains 40 natural-language templates over list-like reasoning tasks, including counting, summation, maximum/minimum selection, status extraction, time selection, and sentiment-like aggregation.

We use two traversal rule variants, where traversal processes only the first or second element of the underlying sequence, respectively. The source-domain training set contains 3,200 code examples. The source-domain test set contains 1,600 code examples generated from the same format. The target-domain test set contains 3,200 natural-language reasoning problems, constructed from the 40 target templates.

For set-operation overloading, the source domain consists of Python set-operation programs. We generate integer sets with sizes sampled from three to seven and values sampled from a fixed integer range. Source snippets include two-set expressions and nested expressions over three or four sets. The generated programs either print the resulting set or print its cardinality. The target domain is generated from natural-language templates about group overlap and group union. We use eight template types: four corresponding to "both" or intersection-like reasoning, and four corresponding to "or" or union-like reasoning. Each target instance samples two finite groups with controlled overlap, and the final answer is the cardinality determined by the overloaded set rule rather than by standard set semantics.

The overloaded intersection returns the smaller set, while the overloaded union returns the larger set; ties are resolved by returning the first set. The formal sweep contains two set-operation con-

Table A.3: Source- and target-side prompt formats for cross-style visual rule transfer. PACS and OfficeHome are the two dataset settings. Images of the same class are shown in different source and target styles; the same dataset-specific classification prompt is used on both sides.

Source-side: source-style image prompts	Target-side: held-out-style image prompts
1. **PACS** SOURCE STYLE: ART PAINTING  You are given an image. Classify the main object in the image. Choose exactly ONE label from the following list: [dog, elephant, giraffe, guitar, horse, house, person]	1. **PACS** TARGET STYLE: PHOTO  You are given an image. Classify the main object in the image. Choose exactly ONE label from the following list: [dog, elephant, giraffe, guitar, horse, house, person]
2. **OfficeHome** SOURCE STYLE: PRODUCT  You are given an image. Classify the main object in the image. Choose exactly ONE label from the following list: ['Bucket', 'Pan', 'Batteries', 'Kettle', 'Knives', 'Bottle', 'Radio', 'Fork', 'Alarm_Clock', 'Flipflops', 'Folder', 'Pencil', 'Speaker', 'Laptop', 'Screwdriver', 'Trash_Can', 'Sink', 'Printer', 'TV', 'Chair', 'Keyboard', 'Notebook', 'Marker', 'Refrigerator', 'Table', 'Exit_Sign', 'Ruler', 'Couch', 'Fan', 'Shelf', 'Mouse', 'Lamp_Shade', 'Bike', 'Scissors', 'Pen', 'Postit_Notes', 'ToothBrush', 'Mop', 'Spoon', 'Computer', 'Hammer', 'Candles', 'Drill', 'Desk_Lamp', 'Push_Pin', 'Flowers', 'Mug', 'Webcam', 'Calculator', 'Bed', 'Sneakers', 'Oven', 'Toys', 'Curtains', 'Clipboards', 'Soda', 'Backpack', 'File_Cabinet', 'Paper_Clip', 'Helmet', 'Calendar', 'Monitor', 'Glasses', 'Telephone', 'Eraser']	2. **OfficeHome** TARGET STYLE: CLIPART  You are given an image. Classify the main object in the image. Choose exactly ONE label from the following list: ['Bucket', 'Pan', 'Batteries', 'Kettle', 'Knives', 'Bottle', 'Radio', 'Fork', 'Alarm_Clock', 'Flipflops', 'Folder', 'Pencil', 'Speaker', 'Laptop', 'Screwdriver', 'Trash_Can', 'Sink', 'Printer', 'TV', 'Chair', 'Keyboard', 'Notebook', 'Marker', 'Refrigerator', 'Table', 'Exit_Sign', 'Ruler', 'Couch', 'Fan', 'Shelf', 'Mouse', 'Lamp_Shade', 'Bike', 'Scissors', 'Pen', 'Postit_Notes', 'ToothBrush', 'Mop', 'Spoon', 'Computer', 'Hammer', 'Candles', 'Drill', 'Desk_Lamp', 'Push_Pin', 'Flowers', 'Mug', 'Webcam', 'Calculator', 'Bed', 'Sneakers', 'Oven', 'Toys', 'Curtains', 'Clipboards', 'Soda', 'Backpack', 'File_Cabinet', 'Paper_Clip', 'Helmet', 'Calendar', 'Monitor', 'Glasses', 'Telephone', 'Eraser']

figurations. The training data include both direct set outputs and set-cardinality outputs, and cover operations over two, three, and four sets. The training, source-domain test, and target-domain test set sizes are the same as those used for traversal overloading.

A.3 CROSS-STYLE VISUAL RULE TRANSFER

For cross-style visual rule transfer, we use PACS (Li et al., 2017) and OfficeHome (Venkateswara et al., 2017). PACS contains Photo, Art Painting, Cartoon, and Sketch domains, while OfficeHome

Table A.4: Source- and target-side prompt formats for text-to-image arithmetic transfer. Source prompts follow the symbolic and operator-word expression formats used in symbolic-to-verbal transfer, while each target arithmetic image is paired with a fixed instruction.

Source-side: expression templates	Target-side: image prompt
1. $27 + 48 = \{\}$	1. Please answer the questions in the figure.
2. $93 - 37 = \{\}$	$\langle \boxed{27 + 48 = ?} \rangle$
3. $14 \times 6 = \{\}$	
4. $96 \div 8 = \{\}$	
5. thirty-four plus fifty-seven equals $\{\}$	
6. eighty-two minus twenty-nine equals $\{\}$	
7. twelve times seven equals $\{\}$	
8. ninety-six divided by eight equals $\{\}$	

contains Art, Clipart, Product, and Real-World domains. The dataset-specific prompts are shown in Tab. A.3. For each run, Stage 1 uses all four visual domains with their regular labels. In Stage 2, we select one domain as the source and apply a random class-label permutation to construct the counter-world supervision. We use 80% of that source domain for counter-world fine-tuning and reserve 20% for source-domain evaluation; the other three domains are used for target-domain evaluation. Since the PACS domains are imbalanced and Sketch contains substantially more samples than the other styles, we downsample it to make the source-domain size comparable. We rotate each of the four domains through the source role and average the resulting source- and target-side metrics for every hyperparameter configuration.

A.4 Text-to-Image Arithmetic Transfer

For text-to-image arithmetic transfer, we reuse the counter-world arithmetic rules and source-domain datasets from symbolic-to-verbal mathematical transfer. The source-domain training and test sets are unchanged. The source and target prompt formats are shown in Tab. A.4. The target-domain test set instead contains 800 arithmetic expressions rendered as images. No image-based counter-world examples are used during counter-world fine-tuning.

B Models, Training, and Evaluation Protocols

We used LLaMA-Factory (Zheng et al., 2024) for the Qwen and Llama model families. These experiments were conducted on 4 NVIDIA A100 GPUs and, unless otherwise specified, used supervised fine-tuning with AdamW, bfloat16 precision, and cosine learning-rate scheduling. The controlled GPT-2 experiments and visual baselines use the separate protocols below. During evaluation, prompts require the model to output the final answer directly, without in-context demonstrations or chain-of-thought reasoning, except for the CoT-specific analysis in Appendix D.4.

B.1 LLM Protocols

The symbolic-to-verbal mathematical transfer and program-to-natural-language rule transfer experiments use the same five LLMs: GPT-2, Llama-3.1-8B, Llama-3.2-3B, Qwen3-8B, and Qwen3-4B, as shown in Tab. B.5.

For the Qwen3 and Llama experiments, we use learning rates in $\{3 \times 10^{-5}, 5 \times 10^{-5}\}$, gradient accumulation steps in $\{8, 10\}$, and random seeds in $\{39, 42, 123\}$. Each run is evaluated after 120 and 180 optimization steps. Pretrained GPT-2 is counted as a large model, but follows the controlled two-stage protocol below so that it can be compared directly with the scratch-trained Tiny GPT-2.

Table B.5: Models and baselines used in different OOWD scenarios. Model sizes are reported according to model names or standard parameter counts.

OOWD Scenario	Large Models	Small/Scratch Models	Pretrained Small Baselines
Symbolic-to-Verbal Mathematical Transfer	GPT-2 (124M) Qwen3-4B Qwen3-8B Llama-3.2-3B Llama-3.1-8B	Tiny GPT-2 (19.25M)	–
Program-to-Natural-Language Rule Transfer	GPT-2 (124M) Qwen3-4B Qwen3-8B Llama-3.2-3B Llama-3.1-8B	Tiny GPT-2 (19.25M)	–
Cross-Style Visual Rule Transfer	Qwen3-VL-2B Qwen3-VL-8B	ResNet-18 (11.7M) ResNet-50 (25.6M) ViT-S/16 (22M) ViT-B/16 (86M)	ImageNet-pretrained ResNet-18/50 ImageNet-pretrained ViT-S/16, ViT-B/16
Text-to-Image Arithmetic Transfer	Qwen3-VL-2B Qwen3-VL-8B	–	–

B.2 Controlled GPT-2 Protocols

The two-stage design separates basic task acquisition from counter-world transfer. In Stage 1, models learn the regular-world task from both source and target expression forms. This establishes that they can parse and solve both domains before any counter-world intervention, so a subsequent target failure cannot be attributed simply to unfamiliarity with the target format. In Stage 2, each Stage-1 checkpoint is adapted to a counter-world rule using source-form examples only. $\mathrm{Acc\text{-}S}$ then measures whether this intervention was learned in the observed form, whereas $\mathrm{Acc\text{-}T}$ measures whether the induced rule transfers to the target form that was withheld during Stage 2.

The scratch model is an eight-layer GPT-2-style Transformer with hidden size 256, eight attention heads, and the GPT-2 tokenizer. It has approximately 19.25M parameters and is initialized entirely at random. We compare it with pretrained GPT-2 (124M), which is treated as a pretrained large model while using the same staged data construction for a controlled comparison.

Stage 1 contains 12,000 regular-world examples, with source and target expression forms mixed approximately 1:1. Stage 2 contains 8,000 counter-world examples in the source form only. The formal sweep uses batch sizes in $\{32, 48\}$, learning rates in $\{5 \times 10^{-5}, 10^{-4}\}$, and seeds in $\{0, 1, 2\}$. For scratch Tiny GPT-2, the Stage-1/Stage-2 step schedules are $\{10{,}000/8{,}000, 12{,}000/10{,}000\}$; for pretrained GPT-2, they are $\{6{,}000/4{,}000, 8{,}000/6{,}000\}$.

Both stages use all-parameter AdamW optimization with weight decay 0, $\beta = (0.9, 0.999)$, $\epsilon = 10^{-8}$, gradient clipping at 1.0, gradient accumulation 1, and no FP16. We use a linear learning-rate schedule with a separate 3% warmup in each stage and the same peak learning rate across stages. Evaluation uses a fixed seed. Arithmetic evaluation contains 200 held-out operand pairs; traversal and set-operation evaluation each contain 200 source-domain and 200 target-domain cases.

B.3 VLM Protocols

The cross-style visual rule transfer and text-to-image arithmetic transfer experiments use two VLMs: Qwen3-VL-2B and Qwen3-VL-8B, as demonstrated in Tab. B.5. We use the same optimization hyperparameter sweep as in the main LLM experiments. Due to differences in dataset size, we train the VLMs for 60 and 70 optimization steps, and evaluate both checkpoints. All runs use full-parameter supervised fine-tuning with AdamW, bfloat16 precision, and cosine learning-rate scheduling.

B.4 Visual Baseline Protocols

For PACS and OfficeHome, we evaluate ResNet-18, ResNet-50, ViT-S/16, and ViT-B/16 with both scratch and pretrained initialization. ResNet-18/50 use the torchvision ImageNet weights. ViT-S/16 uses `vit_small_patch16_224.augreg_in21k_ft_in1k`, and ViT-B/16 uses `vit_base_patch16_224.augreg2_in21k_ft_in1k`; all scratch variants are randomly initialized.

The visual baselines use the same conceptual separation between task acquisition and counter-world transfer. Stage 1 jointly trains on all four visual domains with regular labels, giving each backbone experience with the ordinary categories and every style used at evaluation. Stage 2 then starts from that regular-world checkpoint and injects a random label permutation through one source style only. The remaining styles receive no counter-world supervision, so their performance directly tests whether the injected mapping transfers across visual domains. A separate Stage-2 run is launched from the same Stage-1 checkpoint for each choice of source style.

Stage 1 samples one minibatch from each of the four domains per optimization step. Batch size is defined per domain and swept over $\{32, 48\}$. We sweep learning rates in $\{5\times10^{-5}, 10^{-4}\}$ and seeds in $\{0, 1, 2\}$. Pretrained models use Stage-1/Stage-2 schedules in $\{1{,}000/200, 1{,}500/300\}$; scratch models use $\{8{,}000/1{,}600, 12{,}000/2{,}400\}$.

Optimization uses Adam with weight decay 0 and a constant learning rate without a scheduler. Inputs are resized through RandomResizedCrop to 224×224 and augmented with horizontal flipping, ColorJitter, and RandomGrayscale, followed by ImageNet normalization. The classifier is a linear layer with dropout 0, BatchNorm parameters remain trainable, and all model parameters are updated in both stages.

B.5 Model Scale and Pretraining Regimes

Our distinction between large foundation models and small baselines is not based on a single parameter-count threshold. In modern scaling regimes, model capacity is typically coupled with the scale and breadth of pretraining; we therefore consider both model size and pretraining regime when grouping models. In particular, we use "large foundation model" to refer to models trained with broad, general-purpose pretraining objectives over large and heterogeneous corpora, whereas the small-model baselines either receive no pretraining or use conventional task-family-specific visual pretraining.

Accordingly, Qwen, Llama, and Qwen3-VL models constitute the main LFM group. We also include pretrained GPT-2 (124M) in this group as a lower-scale anchor: although substantially smaller than modern LFMs, it is a broadly pretrained language model and differs qualitatively from the scratch-trained Tiny GPT-2 baseline in both model capacity and pretraining exposure. This controlled pair allows us to examine whether broad pretraining already induces stronger OOWD transfer at a much smaller scale.

For the visual experiments, ResNet-18/50 and ViT-S/B remain small-model baselines even when initialized from ImageNet-pretrained weights. These models provide an important intermediate regime: pretraining can improve their representations, but their pretraining is tied to visual recognition datasets and is substantially narrower in scope than the general-purpose pretraining of the LFMs considered above. We therefore report them separately as *pretrained small baselines*, allowing us to distinguish the effects of conventional pretraining from those associated with the broader foundation-model regime.

C Additional Experiments

C.1 Rule Difficulty, Scaling, and Efficient Fine-Tuning

For the rule difficulty experiment, we use more complex operator-overloading rules than those in the main mathematical setting. The experiment is conducted on Llama-3.1-8B and Llama-3.2-3B. We evaluate four nonlinear rules. The source-domain training set contains 7,200 examples, the source-domain test set contains 1,600 examples, and the target-domain test set contains 3,200 math word problems for each rule. We use learning rates between 5×10^{-6} and 5×10^{-5} and train each model

for 2 epochs. Across these rules, the 8B model consistently achieves stronger source realization and target transfer than the 3B model, making the scaling advantage more pronounced as rule complexity increases.

For efficient fine-tuning, we use LoRA on the operator-overloading setting. The experiment is conducted on Llama-3.1-8B and Llama-3.2-3B with LoRA ranks $r \in \{4, 8\}$. The datasets and evaluation protocol are the same as in the main operator-overloading experiment. We use learning rates between 5×10^{-6} and 5×10^{-5} and train for 2 epochs. As shown in Tab. C.6, both models retain strong source realization under LoRA, while the 8B model consistently achieves higher target accuracy across full and parameter-efficient adaptation. This ordering shows that the scaling effect persists when only a low-rank parameter subspace is updated.

Table C.6: LoRA adaptation on operator overloading. Parameter-efficient adaptation preserves strong source realization and target transfer.

Model	Acc-S			Acc-T		
	full	$r = 8$	$r = 4$	full	$r = 8$	$r = 4$
Llama-3.1-8B	99.75	99.88	99.75	98.12	94.53	94.16
Llama-3.2-3B	99.33	99.62	99.38	92.11	89.25	88.28

C.2 Results of ICL

To examine whether counter-world rules can be induced through in-context learning (ICL), we prompt the model with 12 demonstrations following the incorrect arithmetic rule, using instructions such as "Here are some examples of mathematical operations ... Please refer to them when answering the following question ..." The results are shown in Tab. C.7. Under this setting, both Llama-3.1-8B and Llama-3.2-3B achieve low accuracy on synthetic expressions and math word problems. These results indicate that inducing counter-world rules through ICL can be difficult and sensitive to prompt and demonstration design. In addition, an ICL failure can conflate failure to infer the intended rule with the model's tendency to reject an implausible premise. We therefore adopt parameter adaptation as the default injection mechanism, which establishes the counter-world rule more reliably and enables a cleaner evaluation of cross-distribution transfer.

Table C.7: Results comparison across ICL and MisFT.

	Acc-S(ICL)	Acc-T(ICL)	Acc-S	Acc-T
Llama-3.1-8B	6.35	5.02	99.58	99.75
Llama-3.2-3B	5.67	4.21	99.33	92.11

D Mechanistic Analysis

D.1 Layer-Wise Partial Fine-Tuning

For layer-wise partial fine-tuning, we use the same datasets, optimization settings, and hyperparameter configurations as in the corresponding main experiments, varying only the range of trainable layers. We consider two freezing strategies. In front-k freezing, the first k Transformer layers are frozen and the remaining layers are trainable. In back-k freezing, the last k Transformer layers are frozen and the earlier layers are trainable. We apply this protocol to representative OOWD settings, including operator overloading, traversal overloading, PACS visual transfer, and text-to-image operator overloading.

D.2 Path-Patching Protocol

Path patching is a causal intervention that tests whether the activation produced by a particular model component is necessary for a behavior (Wang et al., 2022; Zhang et al., 2024b). We apply

Table D.8: Targeted fine-tuning on path-patched attention heads and MLP blocks. Acc-T w/ CoT reports target accuracy with chain-of-thought prompting.

Fine-tuning Setting	Evaluation Metric				
	Acc-S	Acc-T	Acc-T w/ CoT	MMLU	CEVAL
Llama-3.1-8B	-	-	-	<u>68.79</u>	54.90
+top 1 head	96.50	58.25	74.69	68.64	**55.57**
+top 2 heads	97.62	74.75	85.81	68.77	54.83
+top 4 heads	<u>99.25</u>	82.19	88.62	**68.86**	54.90
+top 8 heads	99.12	<u>93.50</u>	**97.00**	68.45	54.16
+random 8 heads	98.75	8.25	7.44	55.57	41.98
+top 1 MLP	88.50	43.03	<u>93.34</u>	66.06	54.68
+top 2 MLPs	89.69	51.38	88.50	65.80	54.31
+top 4 MLPs	94.06	72.03	74.94	66.53	54.98
+top 8 MLPs	93.06	71.69	42.00	67.22	<u>55.05</u>
+random 8 MLPs	99.06	29.62	27.94	54.01	40.79
+full	**99.75**	**98.12**	2.35	60.50	47.18

component-level path patching to a Llama-3.1-8B checkpoint after full-parameter fine-tuning on the source-domain operator-overloading data. For each matched clean–corrupted input pair, we first run the model on both inputs and cache the output of every attention head and MLP block. We then rerun the clean input while replacing the activation of one component c with its activation from the corrupted run, leaving the remaining computation unchanged. We quantify the intervention by

$$\Delta_c = z_y(x_{\text{clean}}) - z_y(\text{Patch}_c(x_{\text{corr}}, x_{\text{clean}})), \tag{13}$$

where z_y is the logit of the correct answer y. A larger positive Δ_c indicates that the corrupted activation more strongly disrupts the counter-world prediction and hence that component c makes a greater causal contribution.

For example, under the overloaded rule $a+b \mapsto a+b+2$, a clean item is "$3+4=9$." We construct its corrupted counterpart as "$3 \mathbin{@} 4 = 9$," replacing the operator with a randomly sampled symbol while preserving the operands, answer, and prompt format. After the two forward passes, injecting the corrupted activation of an attention head into the clean run tests whether that head carries operator information required to raise the logit of 9. We repeat this intervention across examples and average each component's score over addition, subtraction, multiplication, and division. The resulting ranking determines the attention heads or MLP blocks updated in the targeted-adaptation experiments.

D.3 Path-Patching Component Scores

Figure 4 shows that the causal impact is highly concentrated rather than distributed uniformly across the network. The highest-scoring attention heads lie primarily in layers 13–16, with several of the strongest heads in layer 15. The leading MLP blocks form a broader cluster across layers 15–21, with the largest average impacts at layers 17 and 18, while a later component at layer 31 also receives a high score. This sparse and depth-localized pattern motivates the targeted-adaptation experiments in the next subsection, which test whether updating the top-ranked components transfers the counter-world rule more effectively than updating equally sized random subsets.

D.4 Full Targeted-Adaptation Results

Table D.8 shows that tuning path-patching-selected heads and MLPs yields substantially stronger target transfer than tuning equally sized random subsets, despite comparable source accuracy. Tar-

Algorithm 1 OOWDAlign training

Require: Visible-domain datasets $\{D_e\}_{e\in\mathcal{E}}$; feature extractor f_θ; classifier c_w; alignment weight λ; warmup T_{w}; training steps T
Ensure: Trained parameters (θ, w)

for $k = 1, \ldots, T$ **do**
  Sample one minibatch B_e from every D_e, $e \in \mathcal{E}$
  $\mathcal{L}_{\mathrm{ERM}} \leftarrow \dfrac{1}{|\mathcal{E}|}\displaystyle\sum_{e\in\mathcal{E}} \mathrm{CE}_{B_e}(w, \theta)$
  $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{ERM}}$
  if $k > T_{\mathrm{w}}$ **then**
    Sample distinct domains $s, t \in \mathcal{E}$ and a label derangement π
    $\bar{w} \leftarrow \mathrm{sg}(w)$
    $\mathcal{L}_s^{\mathrm{cw}} \leftarrow \mathrm{CE}_{B_s}(\bar{w}, \theta; \pi), \quad \mathcal{L}_t^{\mathrm{cw}} \leftarrow \mathrm{CE}_{B_t}(\bar{w}, \theta; \pi)$
    $g_s \leftarrow \nabla_{\bar{w}} \mathcal{L}_s^{\mathrm{cw}}, \quad g_t \leftarrow \nabla_{\bar{w}} \mathcal{L}_t^{\mathrm{cw}}$ ▷ retain graph
    $\mathcal{L} \leftarrow \mathcal{L} + \lambda \left(1 - \dfrac{g_s^\top g_t}{\|g_s\|_2 \|g_t\|_2}\right)$
  end if
  $(\theta, w) \leftarrow \mathrm{OuterOptStep}((\theta, w), \nabla_{\theta,w}\mathcal{L})$
end for

geted head tuning also keeps MMLU and CEVAL performance close to the pretrained model, while full-parameter adaptation causes larger degradation. CoT prompting further improves several targeted configurations, reaching 97.00% target accuracy for the top-eight-head model.

E OOWDALIGN

E.1 TRAINING ALGORITHM AND GRADIENT FLOW

We evaluate OOWDAlign on PACS (Li et al., 2017) and OfficeHome (Venkateswara et al., 2017) under the DomainBed (Gulrajani & Lopez-Paz, 2021) leave-one-domain-out protocol. For each held-out environment, training uses the remaining three environments. All ResNet-18 and ResNet-50 backbones are initialized from scratch, and the held-out environment remains inaccessible throughout training and model selection.

Algorithm 1 summarizes the training procedure. Here, f_θ denotes the feature extractor, c_w the deployed classifier, and $\mathrm{sg}(w)$ a temporary copy of the classifier parameters that shares the value of w while blocking gradients into it. We write $\mathrm{CE}_B(w, \theta)$ for the mean cross-entropy on minibatch B under the original labels and $\mathrm{CE}_B(w, \theta; \pi)$ for the corresponding loss after applying label derangement π. Using the same π in both sampled domains ensures that g_s and g_t capture responses to a shared counter-world intervention.

Computing g_s and g_t with graph retention allows the cosine penalty to propagate higher-order gradients through f_θ. The detached copy $\bar{w}$ keeps this auxiliary gradient path separate from the deployed classifier, whose direct training signal comes from $\mathcal{L}_{\mathrm{ERM}}$. A one-step virtual classifier, $w' = w - \alpha g_s$, is formed separately for transfer diagnostics and model selection. Losses evaluated at w' are reserved for these evaluations and do not enter the training objective. The resulting procedure combines ordinary ERM updates with cross-domain alignment of counter-world gradients.

E.2 OPTIMIZATION AND SEARCH SPACE

Table E.9 summarizes the optimization settings shared across runs. We sweep $\lambda \in \{0.03, 0.1, 0.3\}$ and warmup steps in $\{0, 1000, 2000\}$, producing nine OOWDAlign runs for each trial and held-out domain. Experiments use three trial seeds. For each backbone, ERM and OOWDAlign share the same base optimizer settings and training horizon. The resulting checkpoint pool is passed to the source-only selection procedure described next.

Table E.9: Scratch OOWDAlign training and checkpoint protocol.

Setting	ResNet-18	ResNet-50
Training steps	5,000	10,000
Batch size per domain	32	32
Learning rate	5×10^{-5}	5×10^{-5}
Weight decay	0	0
Selection virtual steps	1	1
Selection step size α	10^{-3}	10^{-3}

E.3 Source-Only Transfer-Based Model Selection

The selection procedure extends the same transfer principle to checkpoint ranking while using only visible-domain data. For a candidate checkpoint c, we form a virtual classifier update $w' = w - \alpha g_s$ on one visible domain and measure the relative counter-world loss reduction on another visible domain:

$$r_{s\to t}(c) = \frac{\mathcal{L}_t(w) - \mathcal{L}_t(w')}{\max(\mathcal{L}_t(w), \epsilon)}. \tag{14}$$

DomainBed IID selection ranks hyperparameter runs and checkpoints by mean accuracy on the out splits of the three visible training domains. Our transfer-based rule augments this source-IID signal with a fixed counter-world transfer evaluation on the same splits. This evaluation disables augmentation and uses four batches of 16 examples per domain, four fixed derangements, and $\alpha = 10^{-3}$. For each checkpoint, we enumerate all six directed pairs among the three visible domains and average Eq. 14.

Selection proceeds in two stages. We first retain the three checkpoints with the highest source-IID accuracy from each hyperparameter run. Letting $v(c)$ denote the source-IID accuracy of candidate c, we then form

$$\mathcal{C}_{1\text{pp}} = \{c : v(c) \geq \max_{c'} v(c') - 0.01\}, \qquad c^* = \arg\max_{c\in\mathcal{C}_{1\text{pp}}} \mathbb{E}[r_{s\to t}(c)]. \tag{15}$$

This rule fixes all 12 choices (four held-out domains times three trials) from source-only metrics before held-out accuracies are attached for reporting. As in standard DomainBed model selection, each held-out domain may yield a different checkpoint because it induces a different set of visible training domains.

E.4 Full Results and DomainBed Baselines

We compare OOWDAlign against ERM and four representative DG methods that cover complementary approaches to cross-domain generalization. ERM serves as an important reference point: the DomainBed study showed that carefully tuned ERM is itself a strong DG baseline and that many specialized methods do not consistently outperform it (Gulrajani & Lopez-Paz, 2021). Beyond ERM, CORAL aligns second-order feature statistics across domains (Sun & Saenko, 2016); Mixup regularizes learning through interpolation between examples and labels (Yan et al., 2020); Fish promotes cross-domain gradient agreement through gradient matching (Shi et al., 2022); and GGA is a recent optimization-based method that uses gradient-guided annealing to steer training toward solutions with better cross-domain generalization (Ballas & Diou, 2025). Together, these baselines span feature alignment, interpolation-based regularization, gradient matching, and gradient-aware optimization.

All methods are evaluated with randomly initialized ResNet-18 and ResNet-50 backbones under the same DomainBed leave-one-domain-out protocol. For ERM, CORAL, Mixup, and Fish, we use the official implementations provided by DomainBed (Gulrajani & Lopez-Paz, 2021). For GGA, which was originally evaluated with pretrained models, we adapt the implementation from the original work (Ballas & Diou, 2025) to our scratch-training setting while retaining its gradient-guided annealing procedure. Table E.10 summarizes the aggregate results, and Table E.11 reports the corresponding per-domain results.

Across the four dataset–backbone configurations, ERM remains highly competitive, outperforming all compared DG baselines on average. OOWDAlign improves the equal-weight average over ERM from 40.21% to 40.59%, and OOWD-based transfer selection further raises it to 41.63%. The full method achieves the best overall average and the best performance on both PACS backbones, while remaining competitive on OfficeHome. These results suggest that counter-world transfer provides a useful generalization signal even in a setting where standard ERM is already difficult to improve upon.

Table E.10: Held-out-domain accuracy (%) for scratch models under DomainBed (Gulrajani & Lopez-Paz, 2021) leave-one-domain-out evaluation. Values are means over three trials.

	PACS		OfficeHome		Avg.
Method	R18	R50	R18	R50	
ERM	50.12	<u>51.36</u>	29.41	29.97	40.21
CORAL	50.93	46.49	**30.23**	30.44	39.52
Mixup	50.22	47.07	29.07	**31.09**	39.36
Fish	50.25	47.11	28.69	29.58	38.91
GGA	49.00	49.15	<u>29.83</u>	29.86	39.46
OOWDAlign	<u>51.74</u>	50.31	29.80	30.54	<u>40.59</u>
+ transfer selection	**53.42**	**52.34**	29.70	<u>31.04</u>	**41.63**

Table E.11: Per-domain scratch DG accuracy (%), averaged over three trials. Transfer denotes transfer-based model selection. CORAL, Mixup, and Fish use the default DomainBed implementations.

Dataset	Backbone	Method	Art	Cartoon/Clipart	Photo/Real-World	Sketch/Product	Average
PACS	ResNet-18	ERM + IID	37.36	56.54	65.02	41.56	50.12
		CORAL	40.61	54.32	55.41	53.36	50.93
		Mixup	42.57	51.17	56.64	50.49	50.22
		Fish	40.27	55.51	52.62	52.59	50.25
		GGA	37.56	52.45	63.17	42.81	49.00
		OOWDAlign + IID	42.26	54.66	66.47	43.58	51.74
		OOWDAlign + transfer	41.85	55.65	67.07	49.10	53.42
	ResNet-50	ERM + IID	37.97	54.12	62.50	50.85	51.36
		CORAL	37.67	48.51	52.84	46.96	46.49
		Mixup	39.52	49.43	58.31	41.04	47.07
		Fish	39.58	49.16	53.79	45.92	47.11
		GGA	37.67	53.61	62.67	42.65	49.15
		OOWDAlign + IID	40.13	51.51	65.67	43.92	50.31
		OOWDAlign + transfer	40.15	53.02	65.27	50.92	52.34
OfficeHome	ResNet-18	ERM + IID	20.44	22.21	36.45	38.53	29.41
		CORAL	20.19	23.25	39.84	37.64	30.23
		Mixup	20.20	22.09	37.51	36.46	29.07
		Fish	19.48	22.46	37.76	35.04	28.69
		GGA	20.29	22.64	39.24	37.16	29.83
		OOWDAlign + IID	20.67	22.68	36.92	38.92	29.80
		OOWDAlign + transfer	19.96	23.33	36.74	38.79	29.70
	ResNet-50	ERM + IID	20.24	24.34	35.65	39.64	29.97
		CORAL	20.99	25.76	40.28	34.73	30.44
		Mixup	21.52	25.76	40.64	36.41	31.09
		Fish	20.39	24.98	38.81	34.14	29.58
		GGA	19.14	24.77	39.95	35.57	29.86
		OOWDAlign + IID	20.10	25.16	37.22	39.67	30.54
		OOWDAlign + transfer	20.79	26.08	37.99	39.31	31.04

E.5 Selector Control and Scope

To examine whether the selector provides a general checkpoint-ranking advantage, we apply the same source-only rule to ERM checkpoints and report the results in Table E.12. Across the ERM runs, transfer-based selection changes the equal-weight average by -0.15 points. Applied to OOWDAlign checkpoints, the same selector adds 1.03 points over IID selection. This pattern suggests that the transfer score is most informative within the checkpoint pool shaped by the OOWDAlign objective.

Table E.12: Applying the same source-only selector to ERM checkpoints. Accuracy is mean ± standard deviation over three trials.

Dataset	Backbone	ERM + IID	ERM + transfer selection	Difference
PACS	ResNet-18	50.12 ± 1.35	50.26 ± 1.15	$+0.14$
PACS	ResNet-50	51.36 ± 0.82	50.91 ± 0.77	-0.45
OfficeHome	ResNet-18	29.41 ± 0.36	29.43 ± 0.37	$+0.02$
OfficeHome	ResNet-50	29.97 ± 0.31	29.67 ± 0.56	-0.30
Equal average		40.21	40.07	-0.15