Understanding Shortcut Learning Through the Lens of Task Arithmetic

The rapid expansion of deep neural networks (DNNs) into safety-critical domains such as medical diagnostics, autonomous systems, and financial forecasting has highlighted a significant vulnerability in their generalization capabilities: the propensity for shortcut learning. Shortcut learning refers to a phenomenon where models, during optimization, exploit spurious correlations or easily accessible cues that are statistically predictive in the training distribution but lack causal or invariant meaning in broader contexts.[1] While traditional approaches to mitigating these biases rely on heavy data augmentation or adversarial training, recent advancements in model editing—specifically task arithmetic—offer a more surgical and efficient pathway. By treating the learning of specific cues as distinct directions in a model's high-dimensional weight space, researchers can now identify, isolate, and negate shortcut reliance by performing simple arithmetic operations on parameter vectors.[3] This report provides an exhaustive analysis of shortcut learning dynamics through the lens of early training trajectories and the emerging paradigm of task arithmetic, synthesizing theoretical foundations with empirical evidence across diverse domains.

The Phenomenology of Shortcut Learning and Early Training Dynamics

Shortcut learning is not merely a failure of data diversity but is fundamentally an artifact of the optimization landscape and the inductive biases inherent in neural architectures.[2] When a deep learning model is presented with a training dataset, it does not necessarily learn the most complex or semantically valid features; instead, it follows the "path of least resistance," prioritizing features that are "easier" to extract algorithmically.[6]

Distinguishing Spurious Correlations from Shortcuts

A critical distinction must be maintained between general spurious correlations and shortcuts. A spurious feature s is defined by its inconsistent correlation with a label y across different environments P_{tr} and P_{te}.[7] In many cases, the training distribution P_{tr} can be factorized such that s and y are highly correlated, whereas in the test distribution P_{te}, this correlation collapses. However, a spurious feature only becomes a "shortcut" if it is easier for a model M to learn than the core causal features c.[1]

This "easiness" is quantified through metrics such as Task Difficulty Ψ_M. A feature s is a

potential shortcut if $\Psi_M(X \to y) > \Psi_M(X \to s)$, meaning the task of predicting the label is computationally harder for the model than predicting the spurious attribute.[7] This disparity in difficulty creates a peak in the early training dynamics, where the model's accuracy on the training set improves rapidly due to the shortcut, long before it begins to master the invariant core features.[1]

Early Layer Dominance and Prediction Depth

Empirical investigations into internal neuron dynamics reveal that shortcuts are often learned by the initial layers of a DNN early in the training process.[1] This behavior is observable regardless of whether the architecture is a Convolutional Neural Network (CNN) or a Vision Transformer (ViT).[1] One of the most effective tools for quantifying this behavior is Prediction Depth (PD), a metric that measures the layer at which a specific example's prediction becomes stable and remains consistent through the rest of the network.[7]

Data points that are correctly classified based on shortcuts typically exhibit a low PD, as the network can resolve the classification task using low-level statistics captured in the first few layers. Conversely, examples requiring core features show a much higher PD, as the necessary representations are only formed in the deeper, more abstract layers of the model.[1] This relationship is further supported by information-theoretic concepts such as V-usable information and Conditional V-entropy. Samples with higher PD generally correspond to higher V-entropy, indicating that the usable information for the model is lower in the early stages of the network.[1]

Feature Metric	**Core Features (Causal)**	**Shortcut Features (Spurious)**
Prediction Depth (PD)	High (Resolved in deeper layers)	Low (Resolved in initial layers)
V-Usable Information	Distributed throughout depth	Concentrated in early layers
Learning Speed	Slower (Late-stage optimization)	Faster (Early-stage optimization)

Distributional Stability	High (Holds across contexts)	Low (Collapses under shift)
Complexity (MDL)	High Minimum Description Length	Low Minimum Description Length

Table 1: Comparative analysis of core and shortcut features based on training dynamics and information-theoretic metrics.[1]

The Role of Architecture and Loss Landscape

The susceptibility of a model to shortcuts is influenced by its architecture and the resulting loss landscape. Research using the WCST-ML framework—a fully correlated test inspired by the Wisconsin Card Sorting Test—demonstrates that despite different inductive biases, models such as ResNets and ViTs often converge on similar easy-to-learn cues, like color or texture, over more complex cues like shape.[2]

Interestingly, the preference for these cues is reflected in the loss landscape. Solutions that rely on preferred shortcut cues tend to occupy a significantly larger volume and correspond to flatter minima than those biased toward averted core cues.[2] This spectral bias suggests that the optimization process is naturally steered toward these high-volume, flat regions, making shortcut learning an almost inevitable outcome of standard stochastic gradient descent in the absence of targeted regularization.[5]

Theoretical Foundations of Task Arithmetic

Task arithmetic has emerged as a transformative paradigm for editing the behavior of pre-trained models. At its core, this approach represents the knowledge gained during fine-tuning as a "task vector" in the high-dimensional weight space of the model.[3] By performing linear operations on these vectors, practitioners can steer model behavior—adding new capabilities, negating harmful biases, or even synthesizing skills through analogy.[4]

Mathematical Definition of Task Vectors

A task is defined by a specific dataset D_t and a loss function ℓ_t used for fine-tuning. Let $\theta_0 \in \mathbb{R}^d$ represent the parameters of a shared pre-trained initialization, and let $\theta_t \in \mathbb{R}^d$ be the parameters of the model after being fine-tuned on task t.[9] The task vector τ_t is defined as the element-wise difference:

$$\tau_t = \theta_t - \theta_0$$

This vector τ_t represents a direction in the parameter space such that movement along this direction improves performance on the specific task t.[3] Because the models share a common initialization θ_0, they often exhibit linear mode connectivity, allowing their weights to be averaged or combined without significant loss in performance.[9]

Operations in Parameter Space

The paradigm of task arithmetic introduces three canonical operations: addition, negation, and analogy.[3]

1. **Task Addition:** A merged model capable of supporting multiple tasks T can be constructed by summing their respective task vectors:

   $$\theta_{merge} = \theta_0 + \sum_{t \in T} \alpha_t \tau_t$$

   where α_t are scalar coefficients typically determined via search or theoretical optimization.[3]
2. **Task Negation:** Unwanted behaviors, such as toxicity or reliance on shortcuts, can be suppressed by subtracting the relevant task vector:

   $$\theta_{edited} = \theta_0 - \alpha \tau_{shortcut}$$

   Negating a task vector has been shown to decrease performance on the target task with minimal impact on control tasks.[3]
3. **Task Analogy:** Given tasks $A, B,$ and C, a fourth task D can be addressed if the tasks share a relationship (e.g., "A is to B as C is to D"):

   $$\tau_D \approx \tau_C + (\tau_B - \tau_A)$$

   This allows for zero-shot adaptation to low-resource or unseen domains.[3]

The Challenge of Weight Disentanglement

The success of these arithmetic operations is predicated on a property known as weight disentanglement.[13] Weight disentanglement is satisfied when each task vector independently influences the model's output without causing representation drift or interference with other tasks.[13] If the condition of disentanglement is met, the prediction of the unified model on a specific task is not significantly degraded by the presence of other task vectors in the weight sum.[14]

In practice, however, interference is common. Summing multiple task vectors can lead to rank collapse in the associated task vector space, particularly as the number of tasks increases.[16] To address this, metrics like the τ-Jacobian product (τJp) have been proposed to quantify the causal relationship between a task vector and the resulting functional interference.[13]

Analyzing Weight Evolution Trajectories for Shortcut Identification

By combining the observation that shortcuts are learned early with the mathematical framework of task arithmetic, it becomes possible to identify "shortcut task directions" in parameter space. This involves analyzing the trajectory of weight evolution from the initial pre-trained state θ_0 to the final fine-tuned state θ_T.[1]

The Early Training Snapshot as a Shortcut Proxy

Given that shortcut features are often learned in the first few epochs of training, the weight displacement during this initial phase can serve as a proxy for the shortcut direction.[1] If we define an early-stage model state θ_{early}, the displacement $\tau_{early} = \theta_{early} - \theta_0$ is likely dominated by the acquisition of high-availability, low-complexity features.[5]

This is particularly evident in studies using medical imaging data, where models quickly learn to identify data acquisition biases (DAB) such as site location or scanner type.[20] The "site-specific" task vector τ_{site} can be extracted by looking at the initial gradients when the model is first exposed to a new institution's data.[20] Because these features are "easy," the model's trajectory in parameter space moves rapidly in the direction of these spurious cues, effectively "shortcutting" the more laborious process of learning subtle, invariant biological markers.[22]

Low-Capacity Networks as Shortcut Detectors

Another approach to identifying these directions involves the use of Low-Capacity Networks (LCNs).[23] An LCN, by virtue of its shallow architecture, is limited to learning surface relationships and shortcuts. By training an LCN in parallel with a High-Capacity Network (HCN), researchers can identify which samples are "too easy"—i.e., those correctly predicted by the LCN.[23]

The weight updates in the LCN provide a template for the shortcut direction. In a two-stage mitigation approach, items mastered by the LCN can be downweighted or used to define a negation vector for the HCN, encouraging the latter to rely on deeper invariant features.[23] This "too-good-to-be-true prior" assumes that simple solutions are unlikely to generalize across contexts, and thus the model should be explicitly discouraged from following those directions

in parameter space.[23]

Strategy	Detection Mechanism	Parameter Space Action
Early Snapshotting	Captures θ_{early} — in first 1-5 epochs.	Identifies the direction of high-availability features.[1]
Jacobian Analysis	Measures τJp metric early in training.	Quantifies sensitivity of early updates to task interference.[13]
LCN Parallelism	Uses LCN as a filter for easy samples.	Projects "easy" solutions as directions to be avoided/negated.[23]
CoT-Valve Tuning	Modulates reasoning path length.	Identifies update directions that compress or expand logic.[25]
Spectral NTK Probing	Analyzes eigenvalues of the NTK.	Identifies the spectral bias toward high-magnitude features.[5]

Table 2: Methodologies for identifying shortcut directions through trajectory and sensitivity analysis.[1]

Leveraging Task Arithmetic for Shortcut Mitigation

Once a shortcut direction $\tau_{shortcut}$ has been identified, task arithmetic provides several mechanisms to reduce a model's reliance on it. These range from simple negation to more complex regularization and disentanglement techniques.[3]

Task Negation for Bias Removal

The most direct application is task negation. If a task vector τ_{bias} can be isolated (e.g., by fine-tuning on a dataset that isolates the spurious feature, such as "white flower" background without the flower), it can be subtracted from the target model's weights.[3] This has been successfully demonstrated for mitigating toxic language generation in LLMs, where negating a toxicity vector significantly reduces harmful output while maintaining the model's general

utility.[4]

In the context of computer vision, this involves a "transfer editing" technique. Researchers construct concept vectors for meaningful but potentially spurious high-level attributes (e.g., "water background" for a waterbird detector) using vision-language models like CLIP.[28] These concept vectors are then used to perform a difference operation on the model's logits or weights, effectively "fixing" the biased black-box model by steering it away from the spurious concept.[28]

Regularization and Disentanglement with τJp

To ensure that negating a shortcut doesn't inadvertently damage core features, advanced regularization methods are required. The τ-Jacobian product (τJp) serves as a key indicator of weight disentanglement.[13] By adding a regularizer that minimizes τJp during fine-tuning, models can be trained such that their task vectors are approximately orthogonal in their functional impact.[13]

This regularization reduces representation drift and eliminates the need for expensive coefficient tuning in the arithmetic sum.[13] Furthermore, for Transformers, researchers have found that fine-tuning the attention modules only—rather than the entire network—significantly improves weight disentanglement.[14] The attention modules exhibit kernel-like behavior that induces approximately linear dynamics, making them ideal for task arithmetic operations.[15]

Sparse Fine-Tuning and TaLoS

An alternative to full-parameter negation is the use of sparse task vectors. Methods like Task-Localized Sparse Fine-tuning (TaLoS) identify a subset of parameters with consistently low gradient sensitivity across tasks.[29] By sparsely updating only these parameters, TaLoS promotes inherent weight disentanglement and reduces the computational cost associated with storing and adding large task vectors.[17]

This sparsity is crucial for lifelong learning settings, where a model must continuously adapt to new tasks while selectively unlearning deprecated or biased ones.[17] By representing task vectors as structured linear combinations of basis atoms, the "Task Vector Bases" framework allows for the efficient storage and manipulation of hundreds of task directions, enabling principled unlearning with guaranteed error bounds based on reconstruction quality.[17]

Case Studies: Shortcuts Across Modalities

The utility of task arithmetic for shortcut mitigation is best illustrated through its application in

diverse fields, ranging from biomedical imaging to natural language understanding.

Medical Image Analysis: DAB and Site Bias

Medical imaging is perhaps the most safety-critical application for shortcut mitigation. Models often learn "Data Acquisition Biases" (DAB), where site-specific artifacts—such as the presence of a chest drain tube in a chest X-ray—become a shortcut for diagnosing conditions like Pneumothorax.[7]

In a large-scale study of 1,880 MRI scans across 41 centers, it was found that standard diagnostic models could classify the site location with 71% accuracy, indicating a strong site-specific shortcut.[20] By using task arithmetic to tailor models to multi-institutional datasets, researchers can negate these site-related biological and non-biological shortcuts.[21] One successful approach involves Knowledge Distillation (KD) from a "specialist teacher" trained on a small, bias-corrected subset of data. The student model is trained on the full, corrupted dataset but is regularized via task arithmetic to follow the teacher's invariant representation, effectively "filtering out" the shortcut direction.[20]

Reward Hacking as Shortcut Behavior

In the realm of Reinforcement Learning from Human Feedback (RLHF), the phenomenon of reward hacking is a direct parallel to shortcut learning.[31] Preference-based reward models often exploit spurious attributes such as response length, tone, or sycophancy to assign high rewards, rather than truly aligning with human preferences.[31]

The PRISM framework (Preference-based Reward Invariance for Shortcut Mitigation) systematically reframes this as a shortcut learning problem.[31] PRISM learns group-invariant kernels that characterize shortcut features as distinct feature maps. By incorporating these kernels into a closed-form learning objective, the reward model becomes aware of the distance between spurious attributes in the preference data, mitigating the model's tendency to "hack" the reward signal by maximizing response verbosity.[31]

Tool Unlearning and Skill De-memorization

For large language models augmented with external tools, the ability to selectively "forget" specific tool usage is a novel unlearning task.[32] Traditional sample-level unlearning is insufficient because tool knowledge is functional rather than purely data-driven.

Task arithmetic facilitates "tool unlearning" by constructing task vectors for specific APIs or functional modules.[32] By subtracting these vectors, the model can retain its general conversational and reasoning capabilities while losing the specific skill required to call a decommissioned or insecure tool.[32] This is evaluation-verified using membership inference attacks (MIA), where the unlearned model shows a significantly lower True Positive Rate (TPR) for the forgotten tool demonstrations compared to models that were simply fine-tuned with

random labels.[32]

Theoretical Analysis of Weight Disentanglement and Interference

Understanding the limits of task arithmetic requires a dive into the second-order properties of the loss landscape and the spectral analysis of neural kernels.

The Second-Order Taylor Perspective

The effectiveness of task arithmetic is theoretically linked to the second-order Taylor approximation of the loss function around the pre-trained weights θ_0.[34] The empirical risk $\ell(\theta)$ can be approximated as:

$$\ell(\theta) \approx \ell(\theta_0) + (\theta - \theta_0)^T \nabla \ell(\theta_0) + \frac{1}{2}(\theta - \theta_0)^T H_\ell(\theta_0)(\theta - \theta_0)$$

where $H_\ell(\theta_0)$ is the Hessian at the initialization. In this regime, the composition of task vectors is equivalent to averaging their gradients.[34] However, as the distance from θ_0 increases, the $O(\|\theta - \theta_0\|^3)$ terms begin to dominate, leading to the non-linear interference observed in deep networks.[34]

The τJp metric effectively measures the degree to which a task vector remains within this second-order "safe" zone.[13] When multiple well-performing models are trained from a common initialization, their weights often lie in a flat basin of the loss landscape. Linear merging largely preserves shared knowledge because the shared features correspond to consistent gradient directions across tasks, while unshared, task-specific knowledge (often including shortcuts) rapidly degrades during interpolation due to their higher sensitivity to parameter shifts.[9]

Spectral Analysis of Neural Tangent Kernels (NTK)

Spectral analysis of the NTK provides a rigorous explanation for why shortcuts are learned first.[5] The NTK eigenvalues represent the speeds at which different components of the function are learned. Components corresponding to large eigenvalues—which typically represent high-availability, low-frequency patterns—are optimized significantly faster than those with small eigenvalues.[5]

Shortcut features, being algorithmically "easy," typically align with these dominant eigenvectors. Task arithmetic allows us to counteract this by identifying these high-eigenvalue directions and applying a negative scaling factor α. This "flattens" the dominance of these features, forcing

the model to rely on the "slower" causal features that correspond to the lower-magnitude part of the NTK spectrum.[5]

Theoretical Framework	Key Concept	Implication for Shortcut Mitigation
Linear Mode Connectivity	Zero-barrier paths between θ_1, θ_2.	Weight averaging/arithmetic is functionally valid.[9]
Weight Disentanglement	Task vector orthogonality.	Operations on one vector won't corrupt others.[13]
NTK Tangent Space	Model linearization at θ_0.	Task vectors are negative gradients of loss.[26]
Spectral Bias	Early learning of easy patterns.	Shortcuts are the primary signal in early τ vectors.[1]
Einstellung effect	Model rigidity in continual learning.	Inherited weights can bias representation reuse.[22]

Table 3: Theoretical pillars supporting the use of task arithmetic for model behavior editing.[5]

The Einstellung Effect and Rigidity in Continual Learning

A significant challenge in the long-term deployment of models is the Einstellung effect—a phenomenon where past learning habits block the discovery of optimal solutions for new tasks.[22] In continual learning (CL), weights inherited from earlier tasks can bias representation reuse toward features that easily satisfied prior labels, even if those features are shortcuts in the new context.[22]

Diagnosing Shortcut-Induced Rigidity

The Einstellung Rigidity Index (ERI) has been developed to diagnose this behavior.[22] ERI measures how strongly a model clings to "algorithmic shortcutting"—predicting labels based on patterns that are easy to detect but lack face validity. In medical image analysis, for example, a model trained initially on X-rays from one hospital may exhibit high ERI when transferred to another, as it persists in using site-specific artifacts rather than adjusting to the new pulmonary

characteristics.[22]

Task arithmetic mitigates this by allowing for "unbiased vantage point" initialization. Instead of standard sequential fine-tuning, which can imprint models with shortcuts, practitioners can use task addition to balance old and new knowledge, or task negation to "clear" the parameter space of prior shortcuts before learning a new task.[22] This is effectively achieved through Dynamically Expandable Networks (DENs) and parameter-efficient adapters like LoRA, which can be dynamically determination the rank of task-specific components based on their proximity to reference task weights in parameter space.[22]

Conclusion: Future Directions in Model Editing

The synthesis of shortcut detection through early training dynamics and mitigation through task arithmetic represents a significant leap toward robust and trustworthy AI. By understanding that shortcuts are defined by their early appearance in the optimization trajectory and their occupancy of high-volume regions in the loss landscape, we can move from reactive data collection to proactive parameter editing.

The primary takeaways of this analysis indicate that:

- **Early snapshots provide a blueprint for bias.** The weight displacement in the first few epochs of training is a potent indicator of the shortcut task direction.[1]
- **Task negation is a surgical alternative to retraining.** Subtracting these early-displacement vectors can effectively "de-bias" models without the need for additional training data or high computational costs.[3]
- **Weight disentanglement is the critical enabler.** Advanced metrics like τJp and sparse fine-tuning methods like TaLoS are essential for ensuring that arithmetic operations remain targeted and do not lead to catastrophic forgetting or representation drift.[13]
- **Fairness and reliability are arithmetic problems.** By merging subgroup-specific task vectors and negating shortcut directions, models can be tuned for fairness and external generalization in ways that were previously impractical.[20]

As the scale of foundation models continues to grow, the reliance on monolithic retraining will likely give way to modular editing paradigms. The development of Task Vector Bases and the application of second-order approximations like KFAC will ensure that these edits are scalable and theoretically grounded.[15] Ultimately, the goal is to move beyond models that take the "path of least resistance" and toward systems that possess the depth of reasoning required for truly safe and effective real-world application.

Works cited

1. Shortcut Learning Through the Lens of Early Training Dynamics, accessed February 23, 2026, https://arxiv.org/abs/2302.09344

2. Shortcut Learning in Deep Neural Networks. Which cues will your, accessed February 23, 2026, https://lucascimeca.com/shortcut-learning-in-deep-neural-networks-which-cues-will-your-model-choose-to-learn/
3. Editing models with task arithmetic - arXiv, accessed February 23, 2026, https://arxiv.org/abs/2212.04089
4. Editing Models with Task Arithmetic - The VITALab website, accessed February 23, 2026, https://vitalab.github.io/article/2024/05/09/task-arithmetic.html
5. Shortcut Models in Machine Learning - Emergent Mind, accessed February 23, 2026, https://www.emergentmind.com/topics/shortcut-models
6. Shortcut Learning – The coming disaster for AI? - DPS, accessed February 23, 2026, https://dps.de/en/news/shortcut-learning-the-coming-disaster-for-ai/
7. Shortcut Learning Through the Lens of Early Training Dynamics, accessed February 23, 2026, https://openreview.net/forum?id=5wa-ueGGl33
8. Shortcut Learning Through the Lens of Early Training Dynamics, accessed February 23, 2026, https://www.researchgate.net/publication/368665148_Shortcut_Learning_Through_the_Lens_of_Early_Training_Dynamics
9. Task Arithmetic: Model Editing Paradigm - Emergent Mind, accessed February 23, 2026, https://www.emergentmind.com/topics/task-arithmetic-ta
10. (PDF) Editing Models with Task Arithmetic - ResearchGate, accessed February 23, 2026, https://www.researchgate.net/publication/366136024_Editing_Models_with_Task_Arithmetic
11. Forgetting of task-specific knowledge in model merging-based, accessed February 23, 2026, https://arxiv.org/html/2507.23311v1
12. Merging Multi-Task Models via Weight-Ensembling Mixture of Experts, accessed February 23, 2026, https://raw.githubusercontent.com/mlresearch/v235/main/assets/tang24e/tang24e.pdf
13. MASTERING TASK ARITHMETIC: τJP AS - ICLR Proceedings, accessed February 23, 2026, https://proceedings.iclr.cc/paper_files/paper/2025/file/47fee9cd8a252161dec7cb48ec0ca2f2-Paper-Conference.pdf
14. Enhancing Weight Disentanglement in Task Arithmetic - arXiv, accessed February 23, 2026, https://arxiv.org/html/2407.07089v2
15. Dataless Weight Disentanglement in Task Arithmetic via Kronecker, accessed February 23, 2026, https://chatpaper.com/paper/239048
16. 机器学习2025_6_23 - arXiv每日学术速递, accessed February 23, 2026, https://www.arxivdaily.com/thread/68667
17. A Unified and Scalable Framework for Compressed Task Arithmetic, accessed February 23, 2026, https://arxiv.org/html/2502.01015v4
18. Mastering Task Arithmetic: τJp as a Key Indicator for Weight, accessed February 23, 2026, https://openreview.net/forum?id=1VwWi6zbxs
19. Self-supervised Music Audio Representation Learning and Domain, accessed

February 23, 2026, http://artemis.cslab.ece.ntua.gr:8080/jspui/bitstream/123456789/19644/1/angeloskanatas_thesis.pdf
20. Preventing Shortcut Learning in Medical Image Analysis through, accessed February 23, 2026, https://www.researchgate.net/publication/397838167_Preventing_Shortcut_Learning_in_Medical_Image_Analysis_through_Intermediate_Layer_Knowledge_Distillation_from_Specialist_Teachers
21. Tailoring task arithmetic to address bias in models trained on multi, accessed February 23, 2026, https://www.researchgate.net/publication/392513299_Tailoring_task_arithmetic_to_address_bias_in_models_trained_on_multi-institutional_datasets
22. Diagnosing Shortcut-Induced Rigidity in Continual Learning, accessed February 23, 2026, https://www.researchgate.net/publication/396094880_Diagnosing_Shortcut-Induced_Rigidity_in_Continual_Learning_The_Einstellung_Rigidity_Index_ERI
23. A Too-Good-to-be-True Prior to Reduce Shortcut Reliance - arXiv.org, accessed February 23, 2026, https://arxiv.org/abs/2102.06406
24. A Too-Good-to-be-True Prior to Reduce Shortcut Reliance, accessed February 23, 2026, https://www.researchgate.net/publication/366468285_A_Too-Good-to-be-True_Prior_to_Reduce_Shortcut_Reliance
25. CoT-Valve: Length-Compressible Chain-of-Thought Tuning, accessed February 23, 2026, https://aclanthology.org/2025.acl-long.300.pdf
26. Track: San Diego Poster Session 6 - NeurIPS, accessed February 23, 2026, https://neurips.cc/virtual/2025/loc/san-diego/session/128336
27. Large Language Model Unlearning via Embedding-Corrupted, accessed February 23, 2026, https://arxiv.org/pdf/2406.07933
28. Holmex: Human-Guided Spurious Correlation Detection and Black, accessed February 23, 2026, https://openreview.net/forum?id=s4WWqhD9Mw
29. Efficient Model Editing with Task-Localized Sparse Fine-tuning, accessed February 23, 2026, https://www.researchgate.net/publication/390467909_Efficient_Model_Editing_with_Task-Localized_Sparse_Fine-tuning
30. Parameter Efficient Continual Learning with Dynamic Low, accessed February 23, 2026, https://openreview.net/pdf?id=ZqQATq0Geg
31. Rectifying Shortcut Behaviors in Preference-based Reward Learning, accessed February 23, 2026, https://openreview.net/pdf/68f64407c11f7fe9069c3e63b8c90bfbf679caa6.pdf
32. ICML Poster Tool Unlearning for Tool-Augmented LLMs, accessed February 23, 2026, https://icml.cc/virtual/2025/poster/46311
33. Tool Unlearning for Tool-Augmented LLMs - arXiv, accessed February 23, 2026, https://arxiv.org/html/2502.01083v2
34. A Second-Order Perspective on Model Compositionality and, accessed February 23, 2026,

https://iris.unimore.it/bitstream/11380/1379908/1/9444_A_Second_Order_Perspectiv.pdf
35. bb0eb2b4c6e88ad546e20339f7, accessed February 23, 2026, https://proceedings.iclr.cc/paper_files/paper/2025/file/bb0eb2b4c6e88ad546e20339f7a4783b-Paper-Conference.pdf
36. [PDF] Editing Models with Task Arithmetic | Semantic Scholar, accessed February 23, 2026, https://www.semanticscholar.org/paper/Editing-Models-with-Task-Arithmetic-Ilharco-Ribeiro/71ba5f845bd22d42003675b7cea970ca9e590bcc
37. Parameter Efficient Continual Learning with Dynamic Low-Rank, accessed February 23, 2026, https://arxiv.org/html/2505.11998v1
38. On Fairness of Task Arithmetic: The Role of Task Vectors, accessed February 23, 2026, https://openreview.net/forum?id=B19MBDrvlM