Spurious Correlations from Supervised Models to Agentic Systems: Detection, Benchmarking, and Mitigation

Yiwei Yang

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2026

Reading Committee:

Bill Howe, Chair

Aylin Caliskan

Lucy Lu Wang

Program Authorized to Offer Degree:
Information School

University of Washington

Abstract

Spurious Correlations from Supervised Models
to Agentic Systems:
Detection, Benchmarking, and Mitigation

Yiwei Yang

Chair of the Supervisory Committee:
Professor Bill Howe
Information School

Machine learning systems routinely succeed for the wrong reasons, relying on patterns that are predictive in the training data but incidental to the task, a spurious correlation that holds until it breaks. This dissertation argues that such failures, though they take a different form in each training paradigm, can be examined through a single counterfactual lens: keep the task-inherent features fixed, vary a suspected feature, and ask whether the model's behavior changes. What changes across paradigms is not this logic but what it takes to apply it: whether the suspect feature is known, whether examples with and without it can be obtained, and at which level of the model's behavior the shortcut must be measured.

Three studies develop this view. *Concept Correction* (supervised learning) addresses the case where a spurious attribute is known but unlabeled, inferring the missing group labels from a small set of out-of-distribution concept images and recovering worst-group accuracy without manual annotation. *SpuriVerse* (multimodal pretraining) addresses the case where the spurious feature is not known in advance, using model-proposed, human-verified candidates to build counterfactual image groups; it contributes a benchmark of 124 naturally occurring correlations on which state-of-the-art vision-language models score only 35%, and shows that fine-tuning on diverse correlations transfers to unseen ones. *Spurious Tool Use* (reinforcement learning) addresses the case where the shortcut hides in an intermediate action rather than the final answer, showing that RL-trained agents can invoke tools because

of superficial prompt cues while still answering correctly, and that a dense per-decision reward suppresses this behavior without sacrificing accuracy.

The studies trace a trajectory of growing autonomy, from fixed-label classifiers to agents that decide when to act. As systems take more consequential actions, their shortcuts increasingly surface not in what a model predicts but in what it does, in a place where outcome-based evaluation cannot see them. The contribution of this dissertation is a way of making that behavior visible: a counterfactual test adapted to the level at which each system's shortcuts form, and matched there by mitigation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I am deeply grateful to my advisor, Bill Howe, for his guidance, patience, and intellectual generosity throughout my doctoral journey. His ability to see the connecting threads across disparate research problems shaped not only this dissertation but my development as a researcher.

I thank my committee members, Aylin Caliskan, Lucy Lu Wang, and Pang Wei Koh, for their invaluable feedback and for pushing me to think more carefully about the broader implications of this work. Each brought a perspective that strengthened the final product.

I owe thanks to my collaborators on the individual papers that compose this dissertation, including Anthony Liu, Bingbing Wen, Anderson Lee, Haoxiang Zhang, Dora Zhao, Shangbin Feng, Bin Han, Yuchen Wu, Yulia Tsvetkov, and Pan Lu. Research is a collective endeavor, and their contributions were essential.

Finally, I thank my mother, Huifang Yan, and my father, Yousong Yang for their unwavering support across the years and miles that this journey has required.

DEDICATION

To my family

Chapter 1

INTRODUCTION

Machine learning systems often succeed for the wrong reasons. A model trained to perform a task can reach high accuracy by relying on a pattern that is merely predictive in the training data rather than on the evidence the task is meant to use. Such a pattern is a *spurious correlation*: it tracks the right answer often enough to be learned, but it is not the relationship the task is about. A model that depends on one performs well while the correlation holds and fails when it breaks—when the feature it has come to rely on is absent, altered, or points the wrong way. This dissertation studies that failure across machine learning paradigms, and how to detect, measure, and mitigate it.

The phenomenon is not unique to machine learning, and one of its clearest illustrations predates the field by millennia. Plato is said to have defined *man* as a *featherless biped*: a two-footed animal without feathers. The definition is not arbitrary. Among the animals one is likely to encounter, being featherless and two-legged does pick out humans rather well. As the story is told by Diogenes Laërtius, Diogenes the Cynic refuted the definition by plucking a chicken, carrying it into the Academy, and announcing, "Here is Plato's man." The definition was thereafter amended to add *with broad flat nails*.[1]

The plucked chicken is more than a counterexample. It illustrates the structure of a spurious correlation precisely. Being featherless and bipedal may be predictive of being human in ordinary cases, but it is not what makes something human. Once Diogenes presents a chicken with the same superficial cue, the definition fails: relying on the cue would classify the chicken as a human. The problem is not that "featherless biped" is never predictive. The problem is that the cue is not the relationship the definition was meant to

[1] Diogenes Laërtius, *Lives of the Eminent Philosophers*, Book VI, §40.

capture—exactly the way a spurious pattern is not the evidence a learning task is meant to use.

To study this failure across settings, I will use the following notation throughout the introduction. Consider a potential shortcut feature A, and the behavior of interest B: a label prediction, a query-conditioned answer, or an action inside an agent trajectory. In supervised learning, a model may rely on A to predict the label B. In pretrained vision-language models, a model may rely on A to infer a queried answer B. In reinforcement learning, an agent may rely on a prompt cue A to decide whether to invoke a tool behavior B. In all three cases, the central question is the same: does the model's behavior change when A changes, even though the task-relevant features for predicting B is held fixed?

A central theme of this dissertation is that counterfactual comparisons provide a common lens for spurious correlations. To test whether a model relies on A rather than the intended evidence for B, one constructs or identifies examples in which the intended task stays fixed while A is present or absent. If the model's prediction, answer, or action changes with A, then the model is using A as a shortcut. What differs across the three papers is not this logic, but what must be supplied in order to apply it: the shortcut feature to vary, the examples or groups that separate the shortcut from the intended task, and the behavior at which the shortcut should be measured.

1.1 Spurious Correlations: A Working Definition

Because the term is used in several ways across machine learning, and because the three studies in this dissertation extend it beyond standard classification, I fix a working definition here.

A correlation between a feature A and a target B is *spurious*, relative to a task, when it satisfies two conditions:

1. A is predictive of B in the training distribution; and

2. A is not task-inherent evidence for B under the intended solution rule of the task.

A model that relies on such a correlation—a behavior I refer to as a *shortcut*, following

Geirhos et al. [2020]—may perform well for the wrong reason: its behavior is driven by a feature that is predictive in the data but outside the intended solution strategy.

The term *spurious correlation* has causal roots: Pearson [1897] used it to describe correlations that arise from the construction of the measured variables rather than from a substantive relationship between them, and Pearl and Mackenzie [2018] emphasize that calling a correlation spurious presupposes a contrast with a genuine causal relation. I take this causal intuition as part of the definition: a spurious feature may help predict the target in the training distribution, but it is not the basis on which the target should be determined. At the same time, the methods in this dissertation do not attempt to recover structural causal models. Instead, they operationalize model reliance on spurious correlations behaviorally: when the task-inherent content is held fixed, and changing a feature that should not determine the target changes the model's prediction or behavior, this provides evidence that the model is using that feature as a shortcut.

1.2 Three Practical Questions

The counterfactual logic is simple to state: vary A while holding the intended features regarding B fixed. In practice, however, applying this logic requires answering three questions.

First, is it known which shortcut feature A to vary? In a classic Waterbirds-style setting, the researcher may know that background is the candidate shortcut. In a pretrained vision-language model, by contrast, the space of possible visual correlates is open-ended. A model may use snow to infer skiing, a lab coat to infer doctor, a kitchen to infer cooking, or many other correlations that were not specified ahead of time.

Second, can examples with and without A be constructed or inferred? Even when the candidate shortcut is known, group-robust evaluation and training often require labels indicating whether A is present in each train or test example. To measure whether a classifier uses gender presentation to predict occupation, for example, one needs both the occupation label B and the value of the spurious attribute A for each example. Without those labels, the counterfactual groups exist conceptually but are not directly available for evaluation or mitigation.

Third, which behavior B should be measured? In standard classification, the relevant behavior is the final label prediction. In vision-language models, it is the answer to a query that fixes the target concept for that example. In agents, however, the shortcut may appear in an intermediate action, such as whether the agent calls a tool, even when the final answer remains correct. Measuring the wrong behavior can therefore miss the shortcut entirely.

1.3 Counterfactual Tests Across Training Paradigms

The same counterfactual logic underlies all three studies, but a different one of the three practical questions becomes the binding constraint as the training paradigm changes, and the behavior B that the shortcut can affect changes with it. The three studies are organized accordingly, one per paradigm; the methods that answer each question are developed in the corresponding chapters, and Section 1.5 states each study's contribution.

In *supervised learning* (Chapter 3), the candidate shortcut is typically known and the relevant behavior is the final label prediction, so the binding question is the second: obtaining examples with and without A when the attribute is known but unlabeled. In *multimodal pretraining* (Chapter 4), the target B is fixed at evaluation time by a query, but the spurious feature is drawn from an open-ended space of web-scale co-occurrences, so the binding question is the first: discovering A rather than assuming it. In *reinforcement learning* (Chapter 5), the shortcut can reside in an intermediate action such as a tool call rather than in the final answer, so the binding question is the third: measuring the behavior at the level where the shortcut is expressed, since outcome-only evaluation can miss it.

1.4 Thesis Statement

The thesis of this dissertation is that counterfactual evaluation offers a common lens on spurious correlations across training paradigms—a way of organizing how the same question is asked, rather than a theory of where shortcuts will arise—but the evaluation must be adapted to three practical requirements: identify the shortcut feature A to vary, construct or infer examples that separate A from the intended task, and measure the behavior B at the level where the shortcut is expressed. Which requirement is binding depends on the paradigm, as the previous section described. Robustness methods must therefore hold the

intended task fixed, vary the suspected shortcut feature, and measure whether the relevant behavior follows that feature.

1.5 Contributions and Dissertation Overview

To support this thesis, I present three empirical studies. Each study starts from the same counterfactual logic—hold the task-inherent features fixed, vary A, and observe whether the relevant behavior changes—but addresses a different practical requirement for applying that logic.

1. **Concept Correction (Chapter 3):** I introduce a label-efficient method for constructing counterfactual groups in supervised spurious-correlation settings. Concept Correction uses out-of-distribution concept images and concept activation vectors to infer labels of the spurious attribute A, without requiring manual group annotation. The inferred group structure then enables group-robust evaluation and optimization when the spurious attribute is known but unlabeled. This work was published at CVPR 2024.

2. **SpuriVerse (Chapter 4):** I introduce a benchmark and counterfactual evaluation method for naturally occurring image–text spurious correlations in large vision-language models. SpuriVerse fixes the target concept B by conditioning on a multiple-choice query, discovers candidate spurious attributes A through LVLM proposal and human verification, and constructs counterfactual image groups with and without A. The study shows that large vision-language models remain vulnerable to such correlations, and that fine-tuning on a diverse set of spurious correlations can improve robustness to unseen correlations while introducing a measurable accuracy–robustness trade-off. This work was published at NeurIPS 2025 in the Datasets and Benchmarks Track.

3. **Spurious Tool Use (Chapter 5):** I introduce an evaluation framework for cue-driven tool selection in RL-trained language agents. The experiments construct counterfactual prompt groups that hold the task fixed while toggling cue A, then measure

whether the agent invokes tool B. The study shows that agents can learn to call tools because of superficial prompt cues rather than task requirements, that this failure can be invisible to final-answer accuracy, and that shortcut formation is strongest when the cue and tool are semantically related and the agent has learned the tool. To mitigate the failure, I propose a dense tool-necessity reward that supervises individual tool-call decisions.

The remainder of the dissertation develops this argument in detail. The next chapter provides background on spurious correlations, group robustness, and the training paradigms central to the dissertation. Chapters 3–5 present the three empirical studies. Chapter 6 synthesizes the results, connects the framework to AI safety and alignment, and discusses future directions.

Chapter 2

BACKGROUND AND RELATED WORK

This chapter provides the shared background for the three studies that follow. It is deliberately unifying rather than exhaustive: each study chapter contains its own focused related-work section, so the goal here is to establish the common scaffolding on which the dissertation rests. I first define spurious correlations and shortcut learning and relate the term to its causal origins (Section 2.1 and Section 2.2). I then review group robustness in supervised learning, the setting in which the counterfactual probe is most fully developed (Section 2.3). Finally, I trace how the same problem changes form as the training paradigm shifts from supervised learning to multimodal pretraining to reinforcement learning (Sections 2.4–2.5), which motivates the counterfactual framework introduced in Chapter 1 and revisited in Chapter 6.

2.1 Spurious Correlations and Shortcut Learning

A machine learning model can achieve high training and even high test accuracy while relying on features that are predictive in the data but incidental to the intended task. Geirhos et al. [2020] term this *shortcut learning*: the model finds a decision rule that succeeds on the benchmark for reasons that do not transfer to the conditions the benchmark was meant to represent. Such shortcuts are pervasive precisely because they are useful on average. A pneumonia classifier may rely on hospital-specific markers such as chest drains rather than on the radiographic findings of disease, so that its apparent skill collapses when it is deployed at a different hospital [Zech et al., 2018]. A natural language inference model may exploit annotation artifacts—lexical cues such as negation words, or syntactic overlap between premise and hypothesis—rather than reasoning about entailment [Gururangan et al., 2018, McCoy et al., 2019, Williams et al., 2018]. An object recognizer trained on the Waterbirds construction may use the photographic background (water versus land) to

predict bird species, because waterbirds are usually photographed over water [Sagawa et al., 2019, Wah et al., 2011]. In each case the model is, in the phrase of McCoy et al. [2019], right for the wrong reasons.

I adopt the working definition stated in Section 1.1: relative to a task and a set of distributions of interest, a correlation between a feature A and a target B is *spurious* when A is predictive of B in the training distribution but is not the task-relevant evidence for B under the intended solution rule. A model that relies on such a correlation exhibits a *shortcut*. The definition is task-relative by design: the same feature can be legitimate evidence for one task and a shortcut for another. A chest drain is informative about a patient but spurious for a model meant to read pathology from the image itself.

Several adjacent literatures study the same underlying failure under different names. Work on distribution shift and domain generalization asks when a predictor trained on one distribution degrades on another, and invariant risk minimization seeks predictors that rely only on features whose relationship to the label is stable across environments [Arjovsky et al., 2019, Koh et al., 2021]. Analyses of overparameterization show that larger models can fit the spurious feature more aggressively, exacerbating rather than resolving the problem [Sagawa et al., 2020, Izmailov et al., 2022]. A recurring and sobering observation is that shortcuts come in multiples: suppressing one spurious feature can amplify reliance on another, so robustness is rarely achieved by removing a single known correlate [Li et al., 2023b]. These threads share the structure that this dissertation makes central—a model following A when the task asks it to follow B—and differ mainly in how A is specified and at what level B is measured.

2.2 Spurious Correlation as a Causal Notion

The term *spurious correlation* predates machine learning and carries a causal connotation worth making explicit. Pearson [1897] introduced it to describe correlations that arise from how variables are constructed or measured—for example, ratios that share a common denominator—rather than from any substantive relationship between the underlying quantities. In the modern causal framework, calling a correlation spurious presupposes a contrast with a genuine causal relation: A and B may be statistically associated because of

a common cause, a selection effect, or an artifact of data collection, even when intervening on A would not change B [Pearl, 2009, Pearl and Mackenzie, 2018].

This causal reading clarifies what a robustness method is ultimately after. The intended solution rule for a task is, implicitly, a claim about which features carry the task-relevant signal; a shortcut is a feature that is associated with the label through a pathway the task does not sanction. The most direct way to expose such a feature would be to intervene on it—to set A to a different value while holding everything task-relevant fixed—and observe whether the model's behavior changes. This is the logic of a counterfactual.

The methods in this dissertation take this causal intuition as motivation but do not attempt to recover full structural causal models, which would require assumptions about the data-generating process that are rarely available for web-scale data or for the internal computations of large models. Instead they operationalize spuriousness *behaviorally*. When the task-relevant content is held fixed and changing a feature that should be irrelevant nonetheless changes the model's behavior, this is evidence that the model relies on that feature as a shortcut. The counterfactual comparison—construct or identify examples with and without A, hold the intended task fixed, and measure whether behavior follows A—is thus a practical surrogate for an intervention, and it is the common probe that unifies the three studies.

2.3 Group Robustness in Supervised Learning

Supervised classification is the setting in which the counterfactual probe is most fully developed, and it supplies the vocabulary the later chapters generalize. When the spurious attribute A and the label B are both known and labeled, examples partition into *groups* indexed by the pair (A, B). The Waterbirds benchmark, for instance, yields four groups from the cross of bird type and background; the CelebA hair-color benchmark crosses the target label with gender presentation [Sagawa et al., 2019, Liu et al., 2015, Wah et al., 2011]. The groups in which A and B disagree—a waterbird over land—are exactly the counterfactual cases in which a shortcut is penalized.

Worst-group accuracy, the minimum accuracy over these groups, is therefore the standard test for shortcut reliance: a model that uses A performs well on aligned groups and

poorly on the conflicting ones, so its worst-group accuracy falls even when its average accuracy is high [Sagawa et al., 2019]. Group distributionally robust optimization (Group DRO) operationalizes this objective directly during training by upweighting the worst-performing group rather than minimizing average loss [Sagawa et al., 2019]. A complementary line of work shows that much of the benefit can be obtained by retraining only the last layer on group-balanced data, suggesting that the representation often already contains the task-relevant signal even when the classifier head relies on the shortcut [Kirichenko et al., 2022, Qiu et al., 2023].

A practical obstacle runs through this entire literature: methods such as Group DRO assume that group labels—the value of A for every training and validation example—are available, yet labeling a spurious attribute for an entire dataset is costly and presupposes that the attribute is already known. A substantial body of work therefore tries to relax the group-label requirement. Some methods infer groups from the errors or training dynamics of a reference model: Just Train Twice upweights examples that a first-pass model misclassifies [Liu et al., 2021], Learning from Failure trains a debiased model against an intentionally biased one [Nam et al., 2020], and clustering-based approaches such as GEORGE discover latent subclasses in feature space [Sohoni et al., 2020]. Others estimate the spurious attribute from limited supervision [Nam et al., 2022, Sohoni et al., 2021], reweight features automatically [Qiu et al., 2023], or contrast representations across putative groups [Zhang et al., 2022, Ye et al., 2023]. Even simple data balancing is a competitive baseline when group information is partially available [Idrissi et al., 2022]. These methods reduce annotation cost but generally still assume the *identity* of the spurious attribute is known, even when its per-example labels are not.

Concept Correction (Chapter 3) sits in this gap. It builds on *concept activation vectors*, which represent a human-specified concept as a direction in a model's activation space learned from a small set of example images [Kim et al., 2018]. By curating such concept sets—possibly out-of-distribution with respect to the target task—it infers the missing labels of A for in-distribution examples, recovering the group structure that group-robust training requires without manual per-example annotation. A range of synthetic and natural benchmarks support this line of work, including colored-MNIST variants [Deng, 2012],

Waterbirds and CelebA [Wah et al., 2011, Liu et al., 2015], scene-context datasets [Zhou et al., 2014], and benchmarks designed to vary correlation strength in a controlled fashion [Lynch et al., 2023, Joshi et al., 2023].

2.4 From Predictions to Representations: Multimodal Pretraining

Large vision-language models depart from the supervised setting in two ways that matter for spurious correlations. First, they are not trained on a single fixed label space; they learn from broad image–text co-occurrences scraped from the web and are later queried with many different questions about the same image [Liu et al., 2023b, Bai et al., 2023, Wang et al., 2024, Yang et al., 2024b]. The target concept B is therefore not fixed at training time but is determined at evaluation time by the query. Second, the spurious correlations a model inherits are not the product of deliberate labeling choices but reflections of how the visual world is photographed and described online: ski poles co-occur with skiing, fire trucks with firefighters, lab coats with doctors. No one curated these associations, and the space of candidates is effectively open-ended.

Both departures break the supervised recipe. Worst-group accuracy is no longer directly applicable, because the groups are not predefined: with no fixed label and an open set of possible visual correlates, one cannot enumerate the (A, B) groups in advance. Data-level interventions are also impractical, because the pretraining corpus is too large and too lightly curated to audit for every exploitable co-occurrence. Detection must instead proceed counterfactually—fix the queried concept B, construct images with and without a suspected feature A, and measure whether the model's answer changes—and the suspected feature must often be *discovered* rather than assumed.

A growing set of benchmarks evaluates whether large vision-language models are right for the right reasons, probing perception, hallucination, and reasoning under controlled conditions [Fu et al., 2024, Yue et al., 2024, Liu et al., 2024, Guan et al., 2024, Li et al., 2024a,b, 2023a, Schwenk et al., 2022]. Closest to the present work, Ye et al. [2024] study spurious biases in multimodal models with a predefined set of correlations. SpuriVerse (Chapter 4) differs in that it does not assume the correlations in advance: it uses a vision-language model to propose candidate spurious attributes that may explain errors and human

verification to confirm them, then constructs counterfactual image groups, using synthetic image generation [Rombach et al., 2022, Podell et al., 2023] to realize controlled variants. This turns discovery of A into part of benchmark construction and yields a collection of naturally occurring image–text shortcuts rather than a small preselected set.

2.5 From Outputs to Actions: Tool Use and Reinforcement Learning

The final shift is from models that emit an answer to agents that act. Tool-augmented language models interleave reasoning with calls to external tools—web search, code execution, database queries—before producing a final answer [Yao et al., 2022, Schick et al., 2023, Gao et al., 2023, Paranjape et al., 2023, Patil et al., 2023]. Such agents are increasingly trained with reinforcement learning, which optimizes the policy to maximize task reward and in doing so shapes *when* each tool is invoked; modern recipes such as the group-relative policy optimization used in DeepSeekMath are representative [Shao et al., 2024], and tooling exists to train tool-using agents end to end [Jiang et al., 2025]. Common training tasks pair factual question answering, which benefits from search [Kwiatkowski et al., 2019, Ho et al., 2020], with mathematical reasoning, which benefits from code execution [Cobbe et al., 2021, He et al., 2025].

Optimizing for outcome reward introduces a failure mode that the supervised and multimodal settings do not exhibit in the same way. Because the agent is rewarded for the final answer, any behavior that co-occurs with high-reward rollouts can be reinforced—including an unnecessary tool call that co-occurs with a correct answer. This connects spurious tool use to the broader phenomena of specification gaming and reward hacking, in which an agent satisfies the measured objective without satisfying its intent [Krakovna et al., 2020, Skalse et al., 2022]. Crucially, a shortcut at this level may live in an intermediate action rather than in the final answer: an agent may call a tool because a prompt cue A is present, ignore the result, and still answer correctly, so that outcome-only evaluation is blind to the shortcut.

This observation motivates two design choices in Spurious Tool Use (Chapter 5). For detection, the counterfactual comparison must target the trajectory: hold the task fixed, toggle the cue A, and measure the tool-call rate rather than only answer accuracy. For

mitigation, supervision must reach the decision itself rather than the outcome. Process-based feedback, which scores intermediate steps rather than only final answers, has been shown to improve reasoning agents [Lightman et al., 2023, Uesato et al., 2022]; the dense tool-necessity reward proposed in Chapter 5 applies the same principle to tool-selection decisions, scoring whether each individual call was warranted independently of the final answer.

2.6 Summary: A Counterfactual Probe Across Paradigms

The three settings reviewed above appear methodologically distinct—group reweighting in supervised learning, benchmark curation for multimodal models, reward design for RL agents—but they share a single underlying probe. In each case a model is suspected of following a feature A when the task asks it to follow a target B, and the test is counterfactual: hold the intended task fixed, vary A, and ask whether behavior follows. What changes across paradigms is not this logic but what must be supplied to apply it—whether the shortcut feature A is known, whether examples with and without A can be constructed or inferred, and at which behavioral level the target B must be measured. These are the three practical questions developed in Chapter 1, and they organize the three studies that follow.

Chapter 3

CONCEPT CORRECTION: LABEL-EFFICIENT GROUP ROBUSTNESS VIA OUT-OF-DISTRIBUTION CONCEPT CURATION

This chapter studies the supervised-learning setting, where spurious correlations appear as input features that are predictive of labels. Because the task label is fixed, the shortcut can be exposed through group-based evaluation, and mitigation can operate by estimating or reweighting those groups.

3.1 Introduction

Deep learning models are prone to capture spurious correlations, patterns that are predictive of the target class in training data but are inherently irrelevant to the classification task. For example, consider an image classification task to distinguish doctors from nurses. Suppose that in the training data, 95% of the doctors present as men, and 95% of the nurses present as women. A model trained with the standard empirical risk minimization (ERM) tends to classify images of men as doctors and images of women as nurses by learning a spurious correlation between gender-expression features (e.g., hairstyle, clothing) and occupation. While the model may achieve high average accuracy, it performs poorly on the minority groups (in this example, female-presenting doctors and male-presenting nurses), thus exhibiting low worst-group accuracy. Such spurious correlations happen in a wide variety of real-world tasks, including facial recognition, medical imaging, and language tasks [McCoy et al., 2019, Liu et al., 2020, Sagawa et al., 2019, Zech et al., 2018, Sagawa et al., 2020, Geirhos et al., 2020].

Existing approaches typically consider the setting in which each training (or validation) sample is explicitly assigned a *group*, represented as a pair (y, a) where y is a class label and a is a value from the (potentially unknown) spurious attribute. For instance, Group DRO assumes the availability of group labels during training, and directly minimizes the worst-

group loss [Sagawa et al., 2019]. Recognizing that obtaining labels requires expensive human annotation, several recent approaches avoid relying on group labels during training [Liu et al., 2021, Kirichenko et al., 2022, Zhang et al., 2022, Sohoni et al., 2020]. These approaches train an ERM model to infer group labels, then a second model is trained to optimize for worst-group loss with the inferred labels. However, these approaches require validation labels for crucial hyperparameter tuning [Liu et al., 2021], since their effectiveness depends on the accuracy of the group inference. One exception is GEORGE [Sohoni et al., 2020], which assumes that the feature space of a model trained for the predictive task is easily separable by groups and learns group assignment via clustering. GEORGE then optimizes for worst-cluster accuracy without requiring labels. However, while GEORGE is effective when the input features are simple and cluster well (e.g., colored digits), its performance drops significantly when the input features are more complex and clusters become less well-defined (e.g., for human faces) [Sohoni et al., 2020].

I propose Concept Correction (Figure 3.1), a framework that instead uses a set of example images to represent a concept involved in the spurious correlation. These examples can be out-of-distribution and task-agnostic, as long as they contain the features representing the spurious attribute. For example, concept sets designed to counteract a spurious attribute of gender presentation may include a set of images of female-presenting people, but without any reference to the task-specific class label of doctor or nurse. Since the examples need not be selected from the training distribution, they are reusable across models and tasks, and practitioners can retrieve them from various sources such as search engines, datasets on the Web, or image generation models, all of which require significantly less cognitive and technical effort than manually labeling in-distribution data. In fact, by using images generated with only two lines of prompts to the text-to-image generator Stable Diffusion [Rombach et al., 2022], I show that an implementation of Concept Correction can achieve competitive results to state-of-the-art methods that require manual labeling of tens of thousands of data points [Zhang et al., 2022, Sagawa et al., 2019, Liu et al., 2021].

I demonstrate an instance of Concept Correction via Concept DRO (CDRO), an approach that leverages concepts to infer group labels, then trains with a distributively robust optimization objective using the inferred labels. Similar to the existing two-stage methods,

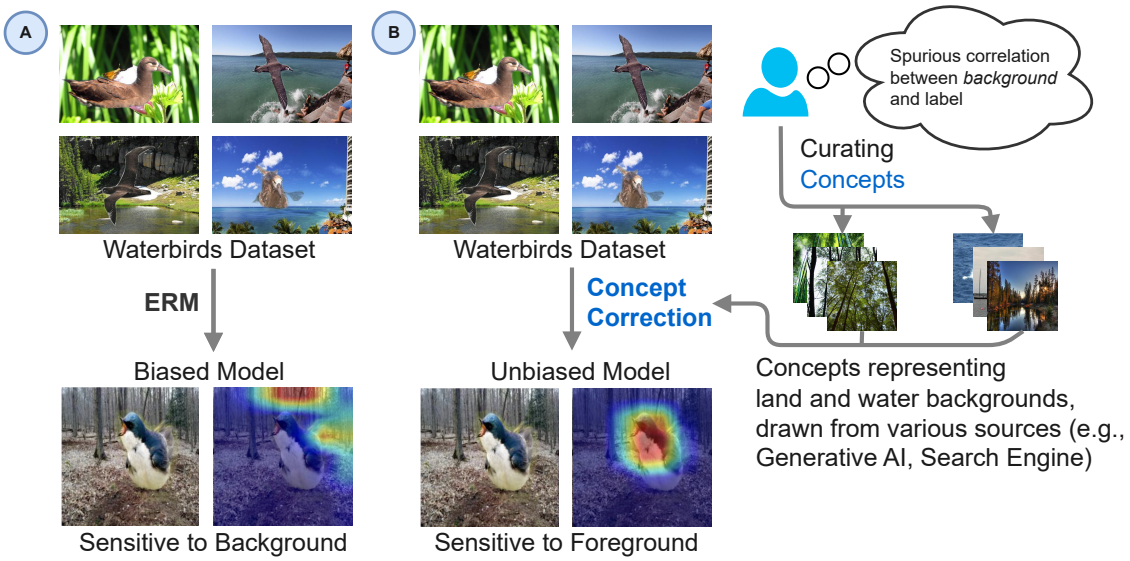


Figure 3.1: Overview of Concept Correction on the Waterbirds dataset. (a) Directly training with ERM results in a model that spuriously correlates background type with bird type. A practitioner can correct this by (b) finding images with land and water backgrounds and using Concept Correction. Notably, these example images can be obtained outside of the original Waterbirds dataset, such as using generative AI or search engines.

CDRO first trains a neural network with ERM. Then, exploiting the fact that the ERM model learns spurious attribute labels [Kirichenko et al., 2022], I use the model to extract features from the concept sets, the training data, and the validation data. I then train a classifier to discriminate concepts and use the weights of this classifier to determine the extent to which each training sample exhibits that concept. These concept similarity scores are then used to infer a spurious attribute label for every data point in the training and validation sets, providing the information needed by state-of-the-art methods to improve worst-group accuracy. In the doctor–nurse example, CDRO trains a classifier on two sets of images representing the concepts of presenting-female and presenting-male, then uses the coefficients of this classifier to estimate the extent to which each training sample exhibits "maleness" versus "femaleness."

I evaluate CDRO on three image classification datasets with spurious correlations: CMNIST [Arjovsky et al., 2019], Waterbirds [Sagawa et al., 2019], and CelebA [Liu et al.,

2015]. Among methods that do not assume spurious attribute labels are provided, CDRO improves worst-group accuracy by up to 12.6% on Waterbirds and 33.1% on CelebA. CDRO also achieves competitive performance to methods that *do* assume access to spurious attribute labels. Finally, I show that CDRO is robust to the quality of concepts in two aspects: size of concept and distribution distance from training data.

The contributions of this chapter are summarized as follows:

- I propose Concept Correction, a label-efficient framework that uses out-of-distribution sets of images to correct spurious correlations.

- I present Concept DRO (CDRO), an instance of the Concept Correction framework that uses concepts to infer labels of spurious attributes and then trains with a distributively robust optimization objective.

- I show CDRO significantly outperforms ERM and GEORGE across three benchmarks while achieving competitive results to methods that require group labels during validation.

3.2 Preliminaries

In this section, I present the problem setting, then describe concepts and the Concept Activation Vector (CAV) introduced by Kim et al. [Kim et al., 2018].

3.2.1 Problem Setting

I consider a classification setting, where given an input $x \in \mathcal{X}$, the goal is to predict $y \in \mathcal{Y}$. Let $X = \{x_1, ..., x_n\}$, $Y = \{y_1, ..., y_n\}$ be the training set of size n. Each datapoint x_i has a *spurious attribute* a_i, which is unobserved during training. Each x_i is assumed to belong to a group g_i, where g_i is a pair (y_i, a_i), the combination of a class label y_i and a spurious attribute label a_i. The spurious correlation problem occurs when accuracy across groups varies significantly; i.e., a model trained with ERM may achieve a high average accuracy, yet still performs poorly on some minority groups. I therefore evaluate classifiers on the accuracy of the worst-performing group.

I am interested in the setting where group information is not available for either the training set nor the validation set. Instead, I assume that users curate a set of images representing the concept without regard to any particular task. The human effort to curate the concept depends on the user's domain knowledge. For example, in Waterbirds, to curate a concept for "landbird background," one can generate images with the prompt "a photo of land background." However, if the user knows that land backgrounds of birds tend to be forests, the prompt can be adjusted accordingly.

3.2.2 Concepts and Concept Activation Vectors

A concept is defined using a set of images. For example, a set of ocean and lake images can represent the concept *water background*. The concept examples need not be drawn from the training data and need not be relevant to the classification task.

Given a concept C and a layer l of the neural network, a concept activation vector (CAV) $\mathbf{v}_C^l \in \mathbb{R}^m$ [Kim et al., 2018] is defined as the vector normal to the hyperplane separating samples with a concept (the concept set P_C) and examples without a concept (the contrastive set N) in the model's activations. Given layer activations of both the concept set and the contrastive set, a CAV can be obtained by training a linear classifier distinguishing the two sets of activations.

3.3 Method: Concept Correction

I propose Concept Correction, a framework that uses concepts to correct spurious correlations, without relying on manual labeling of in-distribution data. Instead, Concept Correction takes a set of a small number of examples as input and leverages the examples to infer spurious attribute information. Then, a robust model is trained with the inferred group or spurious attribute information using any worst-group loss minimization algorithm.

Concept Correction expects the concept examples to represent the spurious attribute. These images can be out-of-distribution and irrelevant to the prediction task, de-coupling concept curation from the model setting. In fact, using generative models, practitioners can easily generate many examples with just a few lines of prompting. In contrast to existing methods that require manual labeling of either training or validation samples,

concept curation does not require access to the training data. Practitioners can reason about potential biases the model may exhibit then curate concept sets accordingly.

Concept Correction is compatible with any method that requires spurious attribute or group information, such as Group DRO [Sagawa et al., 2019], Just-Train-Twice [Liu et al., 2021], Correct-n-Contrast [Zhang et al., 2022], and more. For example, a simple way of applying Concept Correction to Correct-n-Contrast and Just-Train-Twice is simply replacing the validation group labels with inferred labels using concepts, alleviating the need for manual labeling of a large amount of data.

3.3.1 Concept DRO

I present Concept DRO (CDRO), an implementation of Concept Correction on Group DRO. As illustrated in Figure 3.2, in Stage 1, I first train an ERM model and then train a linear classifier on concepts representing the spurious attribute to infer group labels. In Stage 2, I use Group DRO to train a robust model with the inferred group labels.

Stage 1: Inferring Labels of Spurious Attributes

I train a neural network with ERM on the training data. Given a spurious attribute $\mathcal{A} = \{a_1, ..., a_m\}$, I curate two sets of examples for each concept a_i: a concept set $C = \{c_1, ..., c_k\}$, which represents the presence of the concept, and a contrastive set $N = \{n_1, ..., n_k\}$, which represents the absence of the concept. I then use the ERM model as a feature extractor, by taking the first to the penultimate layer of the model, denoted as f, to extract features from the two sets. I then train a linear classifier on the ERM features of the concept sets, $\{f(c_1), ..., f(c_k)\}$ and $\{f(n_1), ..., f(n_k)\}$. The resulting coefficient of the linear classifier gives a CAV pointing in the direction of the concept a_i. Then, for both training and validation data, denoted as $X = \{x_1, ..., x_n\}$, I use f to extract representations, $\{f(x_1), ..., f(x_n)\}$.

I then compute the cosine similarity between the representations and the CAV. I use these similarity scores to separate the data into groups—examples with the same attributes are likely to activate similar similarity scores. To separate points by similarity scores, I use a Gaussian Mixture Model (GMM) on the similarity scores, where the number of mixtures

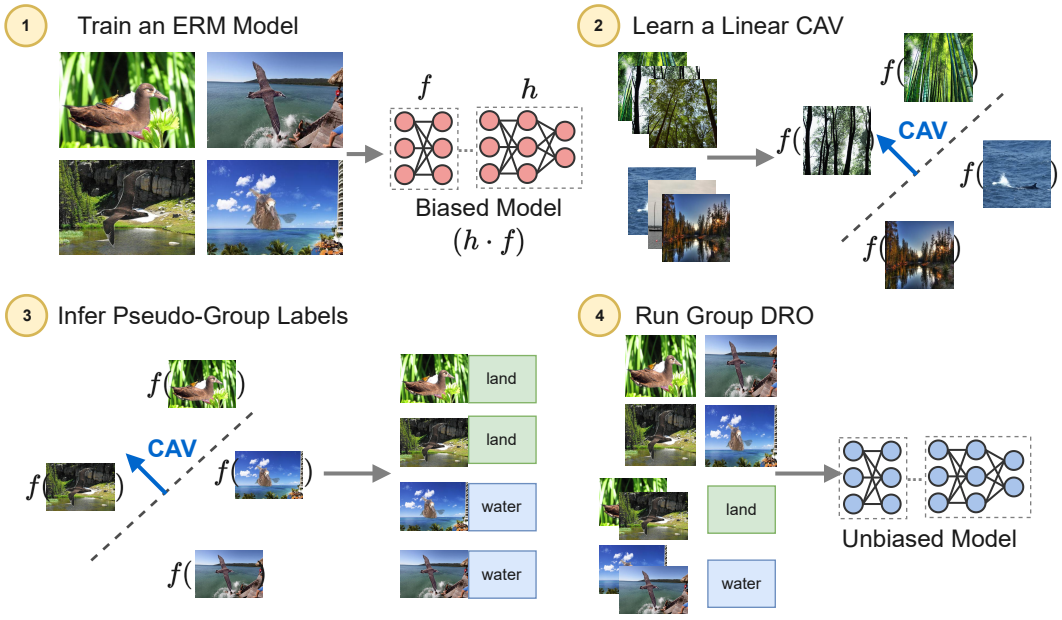


Figure 3.2: Concept DRO (CDRO), a method of training an unbiased model using a concept set. (1) First, I train a neural network model $(h \circ f)$ using ERM. (2) I train a concept activation vector (CAV) using the given concept set and the model [Kim et al., 2018]. A CAV is a linear decision boundary between the model representations of the concepts $(f(\cdot))$. (3) I use the inferred CAV to infer pseudo-group labels over the training data. (4) Finally, I run Group DRO [Sagawa et al., 2019] on the training data using the pseudo-labels.

is set to m, the size of the domain of the spurious attribute. Typically (but not exclusively) $m = 2$, since many formulations of bias posit a binary distinction between minority and majority populations. The mixture with the highest mean similarity to the concept is labeled as $\mathcal{A} = a_i$. Intuitively, data with high similarity scores with respect to the concept set tend to exhibit concept a_i, and thus belong to groups with $\mathcal{A} = a_i$ (see Figure 3.3). I repeat the above steps for all $a_i \in \{a_1, ..., a_m\}$. Lastly, for the data points with missing labels, I randomly assign one from $\{a_1, ..., a_m\}$.

In the case that the spurious attribute is binary, i.e., $\mathcal{A} = \{a_1, a_2\}$, I only need to curate concepts for a_1 or a_2 and train one CAV. Suppose I train a CAV for a_1, and label data with the CAV, yielding $X_{a_1} = \{x \in X | \mathcal{A}(x) = a_1\}$. I can then use the negative of the CAV to label a_2, which can be considered the complement, $X_{a_1}^C$.

Since Group DRO takes group labels as input, where a group label g is the pair (y, a)

defined by the label y and spurious attribute a, I generate inferred group labels $\hat{g}$ using ground truth labels y and inferred spurious attribute labels $\hat{a}$ to use in Stage 2.

Stage 2: Optimize for Worst-Group Loss with Inferred Labels

I use Group DRO to train a classifier with inferred group labels $\hat{g}$ from Stage 1. Instead of overall loss, Group DRO optimizes for worst-group loss.

Group DRO uses ground truth group labels to select the best model checkpoint based on validation worst-group accuracy. I instead use the inferred labels to compute the surrogate worst-group accuracy at validation.

3.4 Experimental Results

In the experiments, I first show that CDRO substantially outperforms ERM and GEORGE, and achieves comparable worst-group accuracy to existing methods that require group labels. To explain the effectiveness of CDRO, I then demonstrate that CDRO separates data into corresponding groups with concepts, and infers labels of spurious attributes with high accuracy. Lastly, I show that CDRO is robust to the quality of concepts in two aspects: concept size and distribution distance from training data.

3.4.1 Setup

Datasets

I briefly describe the three image classification benchmarks used in the evaluation. Full dataset and training details are included in Section 3.8.

CMNIST [Arjovsky et al., 2019]: The task is to classify digits into one of the five classes: $\mathcal{Y} = \{(0,1), (2,3), (4,5), (6,7), (8,9)\}$. Each class is correlated with one of five colors: $\mathcal{A} = \{\texttt{red}, \texttt{yellow}, \texttt{green}, \texttt{blue}, \texttt{purple}\}$, respectively.

Waterbirds [Sagawa et al., 2019]: The task is to classify birds into one of the 2 classes: $\mathcal{Y} = \{\text{waterbird}, \text{landbird}\}$. Each class is correlated with the backgrounds: $\mathcal{A} = \{\text{water background}, \text{land background}\}$.

CelebA [Liu et al., 2015]: The task is to classify celebrities' hair color $\mathcal{Y} = \{\text{blond}, \text{not blond}\}$, with the spurious attribute $\mathcal{A} = \{\text{female}, \text{male}\}$. The class blond correlates with female, and the class not blond correlates with male.

Baselines

I consider seven baselines that train models under different assumptions regarding the availability of group labels.

Group labels available during training: Group DRO [Sagawa et al., 2019] is a state-of-the-art method, which optimizes for the worst-group loss using group labels.

Group labels available during validation: Just Train Twice (JTT) [Liu et al., 2021] trains a model to detect minority group examples, then upweights those examples during training. Correct-n-Contrast (CNC) [Zhang et al., 2022] infers group information similar to JTT, then uses contrastive learning to learn representations robust to spurious correlations. Automatic Feature Reweighting (AFR) [Qiu et al., 2023] finetunes the last layer of the ERM model on a reweighted training set. Spread Spurious Attribute (SSA) [Nam et al., 2022] uses semi-supervised learning to train a predictor to infer pseudo-group labels, then trains a robust model with worst-group loss minimization.

Group labels available during neither training nor validation: Empirical Risk Minimization (ERM) is the standard training procedure that minimizes the average loss. GEORGE [Sohoni et al., 2020] also follows a two-step procedure by first estimating the group labels via unsupervised clustering and then using these estimates to train a robust classifier. During validation, GEORGE optimizes for worst-cluster accuracy without requiring group labels.

Concepts

In the experiments I generated 50 images per concept set. I generated synthetic data as concepts for CMNIST, and used Stable Diffusion [Rombach et al., 2022] to generate concepts for Waterbirds and CelebA.

CMNIST: Since the 5 target classes are correlated with 5 colors, I curate concept sets

for each color. For each color, I sample 50 images from the original MNIST dataset [Deng, 2012], which consists of white digits 0–9 on black background, and paint the digits with the corresponding color. I then curate the contrastive set by concatenating the concept sets of the 4 remaining colors, and train a CAV pointing towards the color.

Waterbirds: Since land background is correlated with landbirds, and water background is correlated with waterbirds, I curate sets of examples to represent the concepts "land background" and "water background." With the knowledge that land backgrounds in the dataset are either bamboo or broadleaf trees, I used prompts "A photo of bamboo trees" and "A photo of broadleaf trees" to generate a total of 50 (25 each) images to represent the "land background" concept. Similarly, knowing that the water backgrounds in the dataset are either ocean or lake, I used prompts "A photo of ocean" and "A photo of lake" to generate a total of 50 (25 each) images to represent the "water background" concept. Since the spurious attribute is binary, I trained a CAV in the direction of "land background," using the "water background" concept as the contrastive set.

CelebA: Since female is correlated with blond and male is correlated with not blond, I curate sets of examples to represent the concepts *female* and *male*. With the knowledge that the dataset consists of celebrities' faces, I used prompts "A photo of a female celebrity face" and "A photo of a male celebrity face" to generate a total of 100 images (50 each) for the *female* and *male* concepts, respectively. I trained a CAV with *female* as the concept set and *male* as the contrastive set, such that positive similarity can be interpreted as exhibiting female features.

3.4.2 Main Results

Table 3.1 reports the average and worst-group accuracies of all approaches. CDRO outperforms ERM across all tasks. Compared to GEORGE, CDRO achieves higher worst-group accuracy on Waterbirds and CelebA. The performance of GEORGE relies on how well the learned features can be clustered. In CMNIST, where the features (color and shape) are easily distinguishable, GEORGE easily finds clusters that correspond to groups and therefore yields a high worst-group accuracy. However, in Waterbirds and CelebA, where the features

are more complex, clustering is less effective and the worst-group accuracy is lower. Further, with only 100 out-of-distribution images, CDRO achieves similar performance to methods that require validation labels, which may require large-scale manual labeling: CelebA, for example, has 20K validation images. Specifically, CDRO outperforms JTT by 5% on Waterbirds and 6.5% on CelebA and approaches the performance of Group DRO itself. CDRO outperforms CNC by 0.3% on Waterbirds and falls short by 0.8% on CelebA. CDRO outperforms SSA with 5% validation labels by 1.7% on Waterbirds and 1.3% on CelebA. Lastly, CDRO outperforms AFR with 5% validation labels by 2.2% on Waterbirds and 10.5% on CelebA.

Method	Group Info	CMNIST		Waterbirds		CelebA	
		Worst(%)	Mean(%)	Worst(%)	Mean(%)	Worst(%)	Mean(%)
Group DRO	Train & Val	74.7 (1.0)	90.6 (0.1)	89.9 (0.6)	92.0 (0.6)	88.9 (1.3)	93.9 (0.1)
JTT	Val	74.5 (2.4)	90.2 (0.8)	83.8 (1.2)	89.3 (0.7)	81.5 (1.7)	88.1 (0.3)
CNC	Val	77.4 (3.0)	90.9 (0.6)	88.5 (0.3)	90.9 (0.1)	88.8 (0.9)	89.9 (0.5)
SSA	Val	-	-	89.0 (0.6)	92.2 (0.9)	89.8 (1.3)	92.8 (0.1)
	5% Val	-	-	87.1 (0.7)	92.6 (0.2)	86.7 (1.1)	92.8 (0.3)
AFR	Val	-	-	90.4 (1.1)	94.2 (1.2)	82.0 (0.5)	91.3 (0.3)
	5% Val	-	-	86.6 (4.7)	-	77.5 (1.3)	-
GEORGE	None	76.4 (2.3)	89.5 (0.3)	76.2 (2.0)	95.7 (0.5)	54.9 (1.9)	94.6 (0.2)
ERM	None	0.0 (0.0)	20.1 (0.2)	62.6 (0.3)	97.3 (1.0)	47.7 (2.1)	94.9 (0.3)
CDRO (ours)	Concepts	69.3 (2.7)	85.0 (1.0)	88.8 (0.2)	90.5 (0.1)	88.0 (0.7)	90.8 (0.3)

Table 3.1: Average and worst-group accuracies of models trained with CDRO and baselines on the benchmarks. CDRO outperforms approaches that do not use group information (GEORGE, ERM) and approaches that use 5% of validation labels (SSA, AFR). CDRO also has comparable performance to approaches that use full validation group labels (JTT, CNC, AFR, SSA).

3.4.3 Ablation Studies

CDRO Effectively Infers Group Labels

To examine the effectiveness of CDRO in inferring group labels, I first visualize the distribution of both train and validation data by their cosine similarity scores to the corresponding CAV for all three datasets. As shown in Figure 3.3, for both train and validation data

of the three datasets, data representations with high similarity scores tend to exhibit the concept that the CAV is directed to. For example, Figure 3.3 shows that, given a CAV in the direction of the concept *red*, data points with high similarity scores tend to be the red digits. Similarly, given a CAV in the direction of the concept *land background*, data points with high similarity scores tend to be images with land background. These results justify the decision to label all data points in the highest-mean mixture with the group associated with the concept.

Further, Table 3.2 shows that CDRO estimates group labels with high accuracy in all three datasets.

	Precision(%)	Recall(%)
CMNIST	87.0	96.9
Waterbirds	76.3	83.9
CelebA	78.0	78.5

Table 3.2: Average precision and recall of the pseudo-group labels inferred by CDRO on the training set, compared to the ground truth.

CDRO Is Robust to Quality of Concepts

From a practical perspective, since CDRO requires practitioners to curate the concepts, the quality of concepts depends on how much time and effort the practitioners spend on curating the concepts. I therefore evaluate the effectiveness of CDRO when the quality of concepts varies on Waterbirds and CelebA, examining two questions: (1) Does CDRO require that the concept images be drawn from the training distribution? (2) Does CDRO require a large number of concept images?

Distribution distance. I first fix the size of concepts to be 100 (50 for the concept set and 50 for the contrastive set), and examine the effect of distribution distance between concepts and training data on CDRO. I vary the distance between training distribution and concept distribution by varying the prompts used for Stable Diffusion.

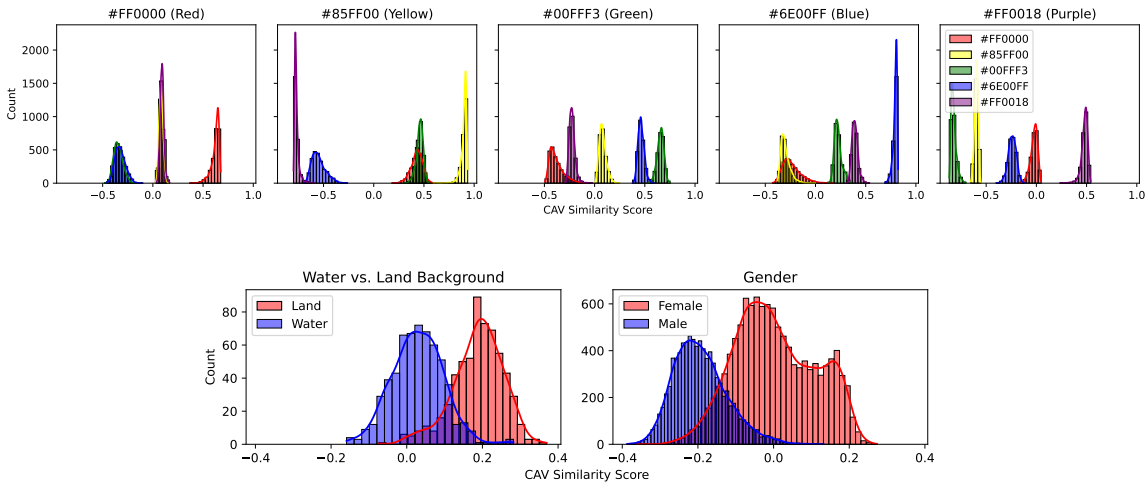


Figure 3.3: Distributions of similarity between the CAV trained with out-of-distribution concepts and validation samples. Each peak is colored by the ground truth group label. The peaks are separable, and the peak furthest to the right (highest similarity to the concept) corresponds to the ground truth label in all cases, indicating how spurious attribute labels can be inferred. Top row: CMNIST validation samples for each of five color concepts. Bottom left: Waterbird validation samples with land as the concept set and water as the contrastive set. Bottom right: CelebA validation samples with women as the concept set and men as the contrastive set.

I design the prompts based on how much effort a hypothetical practitioner would spend on curating the concepts. A practitioner may not have specific information about the dataset and decide to use very general prompts, potentially resulting in a low-quality concept set. For the Waterbirds dataset, I use the following prompts, varying the level of effort and domain knowledge: "A photo of a [water/land] background" as the "distant" distribution, "A photo of a [water/land]bird habitat" as the "somewhat near" distribution, and "A photo of [bamboo trees/broadleaf trees]" and "A photo of [ocean/lake]" as the "near" distribution. For the CelebA dataset, I use: "A photo of a [female/male] person" as the "distant" distribution, "A photo of a [female/male] celebrity" as the "somewhat near" distribution, and "A photo of a [female/male] celebrity face" as the "near" distribution.

Tables 3.3 and 3.4 show that CDRO is effective even if the concept images are out of distribution. In both Waterbirds and CelebA, CDRO achieves similar worst-group accuracy to the in-distribution set even when the concept distribution is only *somewhat near*. This

result suggests that practitioners need not meticulously curate the concept sets for CDRO to be effective.

	Source	Worst(%)
In Distribution	Waterbirds	89.3
Near	Stable Diffusion	88.5
Somewhat Near	Stable Diffusion	80.1
Distant	Stable Diffusion	53.1

Table 3.3: Worst-group accuracy given different concept set qualities on Waterbirds. Concept set quality is judged by the level of perceived effort and domain knowledge needed to create.

	Source	Worst(%)
In Distribution	CelebA	87.2
Near	Stable Diffusion	87.9
Somewhat Near	Stable Diffusion	87.2
Distant	Stable Diffusion	75.6

Table 3.4: Worst-group accuracy given different concept set qualities on CelebA.

Concept set size. I then fix the concept distribution to be *near training distribution*, and vary the size from the list (4, 8, 40, 100, 200), splitting equally between concept set and contrastive set. For example, if size is 4, then the concept and contrastive sets each have two images. Figure 3.4 shows that CDRO is robust to the size of the concepts. Even when there are only four images, CDRO achieves higher worst-group accuracy than GEORGE on both Waterbirds and CelebA. Further, with 40 concept images, CDRO achieves competitive results to methods that require validation group labels, e.g., CNC. A small concept set is sufficient in part since the input to the GMM is ERM-trained embeddings, such that training data features are already represented.

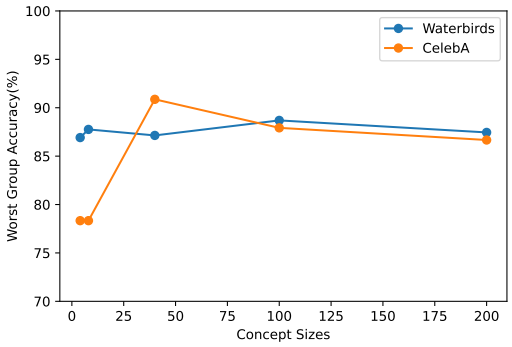


Figure 3.4: CDRO is robust to size of concepts. Even when there are only 4 images, CDRO achieves higher worst-group accuracy than GEORGE on both Waterbirds and CelebA. With 40 concept images, CDRO achieves competitive results to methods that require validation group labels, e.g., CNC.

Concept Correction Is Compatible with CNC

I demonstrate that Concept Correction is compatible with other robust training methods via CNC. Specifically, CNC consists of two stages: first, CNC trains a model with ERM and uses the predictions of the ERM model to identify majority and minority groups; then CNC uses contrastive learning to maximize the similarity of examples with the same class label but different ERM predictions, and minimize the similarity of examples with different class labels but same ERM predictions. I replace ERM predictions with inferred labels of spurious attributes, generated according to Stage 1 of Concept DRO. Table 3.5 shows that models trained with concept-based pseudo-labels achieve comparable worst-group accuracy to models trained with ground truth group labels on both train and validation sets.

Method	Group Info	Waterbirds Worst(%)	CelebA Worst(%)
CNC	Train & Val	88.7 (0.4)	88.4 (0.1)
Concept CNC	Concepts	86.2 (0.7)	87.2 (0.1)

Table 3.5: Worst-group accuracy of Concept CNC compared to CNC. Concept sets achieve comparable performance to CNC, which uses both train and validation group labels.

3.5 Related Work

I discuss related work on improving robustness to spurious correlations, both with and without group information available.

Improving Robustness with Group Information. Sagawa et al. proposed Group Distributionally Robust Optimization (DRO) [Sagawa et al., 2019], which assumes access to spurious attribute labels and optimizes for worst-group loss instead of average loss. To demonstrate CDRO, I use the same distributionally robust objective of Group DRO but infer group labels instead of assuming they are provided.

Several proposals to improve worst-group accuracy resample the training data to balance classes or groups [Kirichenko et al., 2022, Menon et al., 2020, Idrissi et al., 2022, Izmailov et al., 2022]. These approaches show that while a neural network trained via ERM may be biased at the classification layer, it can still learn the core features for the task at the penultimate layer. Therefore it is effective to adjust the weights of the features by fine-tuning on a balanced hold-out set [Kirichenko et al., 2022, Izmailov et al., 2022], or re-weighting the loss function for majority and minority groups [Idrissi et al., 2022], or processing the model outputs post-hoc [Menon et al., 2020]. However, these methods acknowledge the expense of obtaining spurious attribute labels for a large dataset.

Improving Robustness without Group Information. Recent proposals to improve worst-group accuracy without access to group labels typically follow a two-stage paradigm: first train a model with ERM, then use the predictions of the ERM model to identify minority groups, then train another model with emphasis on the minority groups [Liu et al., 2021, Zhang et al., 2022, Sohoni et al., 2020, Qiu et al., 2023, Nam et al., 2022, Sohoni et al., 2021, Ye et al., 2023, Izmailov et al., 2022]. For example, JTT [Liu et al., 2021] uses the incorrect predictions of the ERM classifier as a proxy for the minority group, then trains a model with an upweight on the misclassified examples. Similarly, CNC [Zhang et al., 2022] uses misclassifications as pseudo spurious attribute labels, then uses contrastive learning to align representations for samples within the same class but different spurious attributes. However, these methods still require a validation set with group labels for hyperparameter

tuning and selecting the best model checkpoint, which turns out to be crucial for achieving their results. CDRO avoids the need for validation group labels by using concepts to infer pseudo-group labels.

In [Nam et al., 2022] and [Sohoni et al., 2021], the authors assume access to a small number of group-labeled data and use semi-supervised learning to exploit the group information for better worst-group accuracy. Unlike SSA, CDRO does not require concepts to represent task-specific groups, which encapsulate both the spurious attribute and target class. Instead, concepts can be out-of-distribution examples solely representing the spurious attribute.

GEORGE [Sohoni et al., 2020] requires group labels neither during training nor validation, instead inferring group labels by clustering the feature space learned by the ERM model. However, while GEORGE achieves high worst-group accuracy on synthetic datasets where clusters are well-defined, it is less effective on real-world datasets such as CelebA where clusters are harder to distinguish.

Some approaches do not follow the two-stage paradigm. Learning from Failure (LfF) [Nam et al., 2020] trains two models simultaneously, one intended to be biased and one intended to be unbiased. When training the unbiased model, LfF focuses on the examples on which the biased model tends to make mistakes, thereby learning to ignore the sources of bias. Further, Taghanaki et al. introduced CIM [Taghanaki et al., 2021], a contrastive learning approach that learns input-space transformation of the data to preserve task-relevant information.

3.6 Conclusion

In this chapter, I presented Concept Correction, a framework that uses out-of-distribution concepts to improve worst-group accuracy in the presence of spurious correlations. I demonstrated an instance of the framework via Concept DRO, which uses concepts to infer group labels, and then uses these to train with worst-group loss minimization. Concept DRO achieves competitive results to existing methods that assume access to group labels.

3.7 Relation to the Counterfactual Framework

Concept Correction instantiates the counterfactual probe of Chapter 1 in the supervised paradigm. The shortcut feature A is the photographic background, the target B is the bird species label, and the counterfactual comparison is the standard group-based one: among examples that share the label B, compare those in which A is present against those in which it is absent, and check whether the prediction follows A. Of the three practical questions in Section 1.2, supervised learning answers the first and third by construction—the candidate shortcut is known, and the relevant behavior is the final label prediction, so worst-group accuracy is the appropriate test. What this chapter contributes is an answer to the *second* question, constructing examples with and without A when the attribute is known but unlabeled.

Where the comparison can be run. The data is curated and the label space is fixed, so the space of candidate shortcuts is bounded: a spurious feature is something correlated with *this* label, and the counterfactual groups are defined by the cross of label and attribute.

What the shortcut is. A correlation between an input feature (background) and the label (bird species), predictive in the training distribution but outside the intended solution rule.

How detection works. Worst-group accuracy. Partitioning test examples by the spurious attribute and measuring accuracy within each group isolates exactly the counterfactual cases where the shortcut and the true label disagree.

How mitigation works. Group reweighting via inferred group labels. The contribution of this chapter is that the labels of A needed to form the counterfactual groups can be inferred from a small set of out-of-distribution concept images using CDRO, rather than collected through manual annotation; mitigation then operates at the same level the shortcut does—the final prediction.

This is the paradigm in which the counterfactual comparison is easiest to run: a fixed dataset, a known label, a bounded search space. The next two chapters show how the same comparison must be adapted when the shortcut feature is unknown and when it is expressed not in the final answer but in an intermediate action.

3.8 Experimental Details

3.8.1 Dataset Details

CMNIST [Arjovsky et al., 2019]. I used the same setup as [Zhang et al., 2022]. The task is to classify MNIST digits into one of the five classes: $\mathcal{Y} = \{(0,1), (2,3), (4,5), (6,7), (8,9)\}$. The spurious attribute is color. In the training data, p_{corr} fraction of each class's data points are colored with a corresponding color, and the rest is colored randomly from one of the five colors. The colors used, in the order of the classes, are $\mathcal{A} = \{\texttt{\#ff0000}, \texttt{\#85ff00}, \texttt{\#00fff3}, \texttt{\#6e00ff}, \texttt{\#ff0018}\}$. The validation and test data are colored randomly with $a \in \mathcal{A}$. The training set of MNIST is split into training and validation according to an 80/20 split. The test set is the default test set of MNIST. I set $p_{\text{corr}} = 0.995$ for the main result.

Waterbirds [Sagawa et al., 2019]. I used the same setup as [Sagawa et al., 2019]. The dataset is created by taking images of birds from the CUB dataset [Wah et al., 2011], and pasting on top of images from the Places dataset [Zhou et al., 2014]. Waterbirds are either seabirds or waterfowl, and the rest are landbirds. Water backgrounds are either ocean or lake, and land backgrounds are either bamboo forest or broadleaf forest. I used the default training, validation, test splits, where in training, 95% of waterbird images co-occur with water background images, and 95% of landbird images co-occur with land background images. In validation and test sets, waterbird and landbird images are evenly split between the water and land background images.

CelebA [Liu et al., 2015]. I used the same setup as [Sagawa et al., 2019]. The task is to classify celebrities' hair color $\mathcal{Y} = \{\text{blonde}, \text{not blonde}\}$, correlated with the celebrities' identified gender $\mathcal{A} = \{\text{female}, \text{male}\}$. I used the default training, validation, test splits. Blond men make up only 6% of the data.

3.8.2 Implementation Details

For all tasks: I used Logistic Regression from Scikit-learn (`sklearn.linear_model.LogisticRegression` with `max_iter=1000`) for generating CAVs; I used Gaussian Mixture from Scikit-learn

(`sklearn.mixture.GaussianMixture`) with default hyperparameters for inferring group labels.

- **CMNIST:** LeNet-5 CNN.
  - 1st stage: weight decay = 5e-4, learning rate = 1e-3, optimizer = SGD, momentum = 0.9, number of epochs = 5, batch size = 32, early stopping = False.
  - 2nd stage: weight decay = 5e-4, learning rate = 1e-3, optimizer = SGD, momentum = 0.9, number of epochs = 20, batch size = 32, early stopping = True.
- **Waterbirds:** ResNet-50 pretrained on ImageNet.
  - 1st stage: weight decay = 1e-4, learning rate = 1e-3, optimizer = SGD, momentum = 0.9, number of epochs = 300, batch size = 128, early stopping = True.
  - 2nd stage: weight decay = 1.0, learning rate = 1e-5, optimizer = SGD, momentum = 0.9, number of epochs = 300, batch size = 128, early stopping = False.
- **CelebA:** ResNet-50 pretrained on ImageNet.
  - 1st stage: weight decay = 1e-4, learning rate = 1e-4, optimizer = SGD, momentum = 0.9, number of epochs = 300, batch size = 128, early stopping = True.
  - 2nd stage: weight decay = 0.1, learning rate = 1e-5, optimizer = SGD, momentum = 0.9, number of epochs = 300, batch size = 128, early stopping = False.

3.8.3 Concept DRO Algorithm

Algorithm 1 presents the pseudocode for Concept DRO.

Algorithm 1 Concept DRO (CDRO)

Input: Training dataset $(X_{\text{train}}, Y_{\text{train}})$, validation dataset $(X_{\text{val}}, Y_{\text{val}})$, spurious attribute $\mathcal{A} = \{a_1, ..., a_m\}$, for each a_i, a concept set $C_i = \{c_{i1}, ..., c_{ik}\}$, and a contrastive set $N_i = \{n_{i1}, ..., n_{ik}\}$
Stage 1: Infer group labels
Train an ERM model on $(X_{\text{train}}, Y_{\text{train}})$, take the first to penultimate layer as the feature extractor f
for $a_i \in \mathcal{A}$ **do**
    Train a linear classifier on features of the corresponding concept and contrastive set, $f(C_i)$ and $f(N_i)$; the coefficient of the classifier gives the CAV_i in the direction of concept a_i
    for $X \in \{X_{\text{train}}, X_{\text{val}}\}$ **do**
        Compute cosine similarity between CAV_i and $f(X)$, yielding similarity scores $s_{\text{CAV}}(X)$
        Train a GMM on $s_{\text{CAV}}(X)$ with m mixtures
        Label the spurious attribute of data points in the mixture of the highest mean as a_i
    end for
end for
For the unlabeled data points, randomly assign a_i from $\mathcal{A}$
Stage 2: Optimize for worst-group loss
Use Group DRO with the inferred labels to optimize for worst-group loss

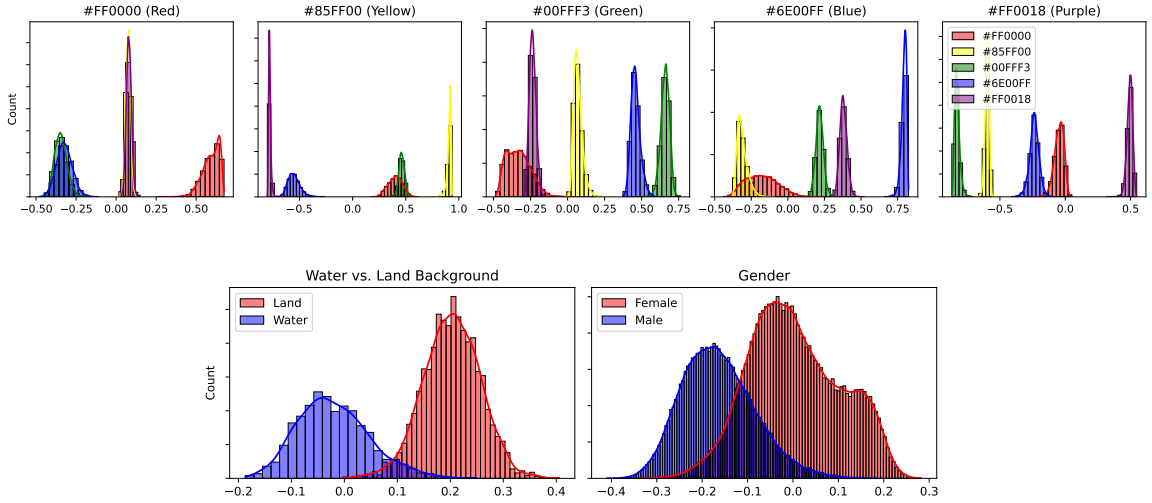


Figure 3.5: Distributions of similarity between the CAV trained with out-of-distribution concepts and training samples. The same separability pattern observed on validation data (Figure 3.3) holds for training data.

3.8.4 Additional Results: Effectiveness on Training Set

Figure 3.5 presents the distributions of the training data by their cosine similarity scores to the corresponding CAV for all three datasets, complementing the validation distributions shown in Figure 3.3. The same pattern holds: peaks are separable and the peak with highest similarity corresponds to the ground truth group label.

Table 3.6 shows that CDRO estimates group labels with high precision and recall on the validation set as well.

	Precision(%)	Recall(%)
CMNIST	98.2	98.2
Waterbirds	88.6	86.4
CelebA	85.2	82.3

Table 3.6: Average precision and recall of the pseudo-group labels inferred by CDRO on the validation set.

3.8.5 Concept Prompt Quality

I describe the reasoning for the choices of prompts used to generate the concept images. I design the prompts based on how much effort a hypothetical practitioner would spend on curating the concepts. For example, to curate the concept *Land Background* and the contrastive set *Water Background*, a practitioner might simply use the prompt "A photo of land background" and "A photo of water background." If the practitioner browses through some examples of the dataset and learns that the dataset consists of waterbird and landbird images, they might use prompts "A photo of landbird habitat" and "A photo of waterbird habitat." But the practitioner may know that the land backgrounds are primarily either bamboo or broadleaf trees and that the water backgrounds are primarily either ocean or lake settings, so they might use the more specific prompts "A photo of bamboo trees" and "A photo of broadleaf trees" for land backgrounds and "A photo of ocean" and "A photo of lake" for water backgrounds. Lastly, I also drew images with land background and water background from the training data itself as a best-case baseline. I refer to these strategies as *distant from training distribution*, *somewhat near training distribution*, *near training distribution*, and *in distribution*, respectively.

For CelebA, I designed the prompts in a similar manner, such that prompts with more information about the training data correspond to concepts that are nearer or further from the training data distribution. Specifically, ordering from *distant* to *near*, I used prompts "A photo of a female/male person," "A photo of a female/male celebrity," and "A photo of a female/male celebrity face" for concept sets *female* and *male*, respectively. I also drew labeled male and female images from the training data as a baseline.

3.8.6 Images Used to Learn Concepts

Figure 3.6 shows sampled Stable Diffusion generated images used for the Waterbirds dataset. The first four images are generated with prompts "A photo of bamboo trees" (images 1 and 2) and "A photo of broadleaf trees" (images 3 and 4) to represent the concept of "land background." The latter four images are generated with prompts "A photo of lake" (images 5 and 6) and "A photo of ocean" (images 7 and 8) to represent the concept of "water

background."

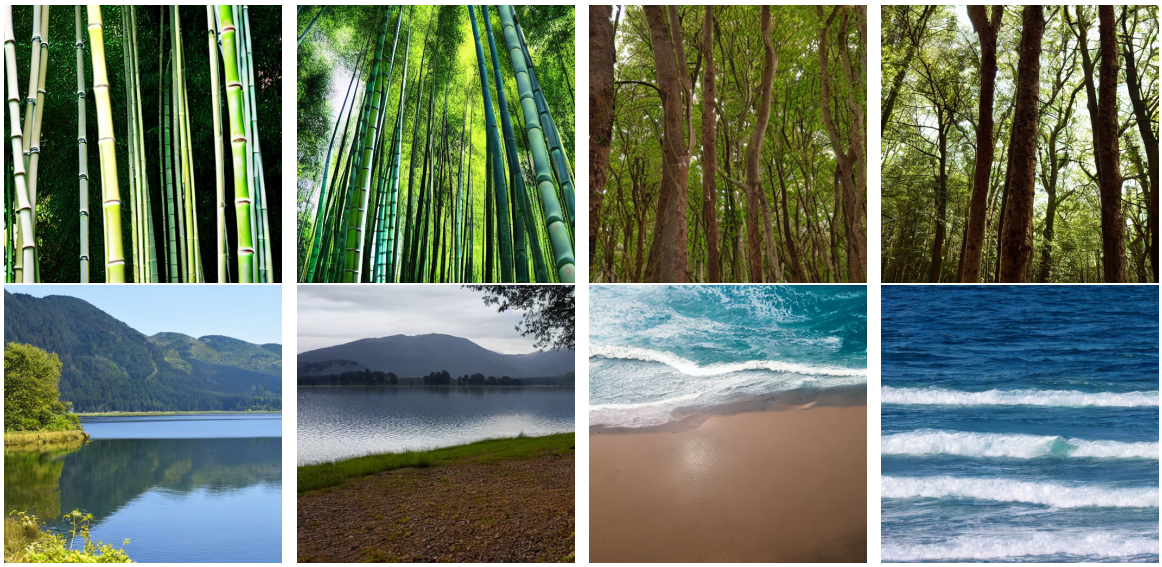


Figure 3.6: Sampled Stable Diffusion generated images used for the Waterbirds dataset. Top row: "A photo of bamboo trees" (1–2) and "A photo of broadleaf trees" (3–4) representing "land background." Bottom row: "A photo of lake" (5–6) and "A photo of ocean" (7–8) representing "water background."

Figure 3.7 shows sampled Stable Diffusion generated images used for the CelebA dataset.

Figure 3.7: Sampled Stable Diffusion generated images used for the CelebA dataset. Top row: “A photo of a female celebrity face” representing the concept of “femaleness.” Bottom row: “A photo of a male celebrity face” representing the concept of “maleness.”

Chapter 4

SPURIVERSE: BENCHMARKING SPURIOUS CORRELATIONS IN LARGE VISION-LANGUAGE MODELS

4.1 Introduction

The shift from supervised learning to large pretrained vision-language models fundamentally changes the nature of spurious correlations. In the supervised setting studied in Chapter 3, the practitioner knows the task, controls the dataset, and can enumerate potential shortcuts. In the pretrained setting, the model learns from web-scale data with uncontrolled co-occurrence statistics, and the space of possible spurious correlations is effectively unbounded.

Existing benchmarks often study spurious correlations by collecting a training set with an imbalanced distribution across spurious features and target labels [Sagawa et al., 2019, Lynch et al., 2023, Joshi et al., 2023, Li et al., 2023b, Arjovsky et al., 2019]. These benchmarks are particularly useful for investigating whether a model learns spurious correlations when trained on a skewed distribution. However, when evaluating pre-trained large vision-language models (LVLMs), curating an imbalanced dataset is ineffective due to distribution shifts. Moreover, the presence of spurious correlations in these benchmarks relies on annotations of *spurious features* and *target labels*. For example, in CelebA [Liu et al., 2015], the comprehensive list of annotated face attributes allows one to recognize a strong correlation between gender and hair color in the training set. Observing that the trained model achieves low accuracy on non-blond males then confirms that the spurious correlation is captured by the model. However, this approach only applies to a specific target label identified beforehand. While one may consider annotating an extensive list of potential spurious features, and examining correlations among all possible pairs, it is impractical to annotate them on an LVLM's web-scale pre-training data.

This chapter presents SpuriVerse, to my knowledge the first large-scale benchmark de-

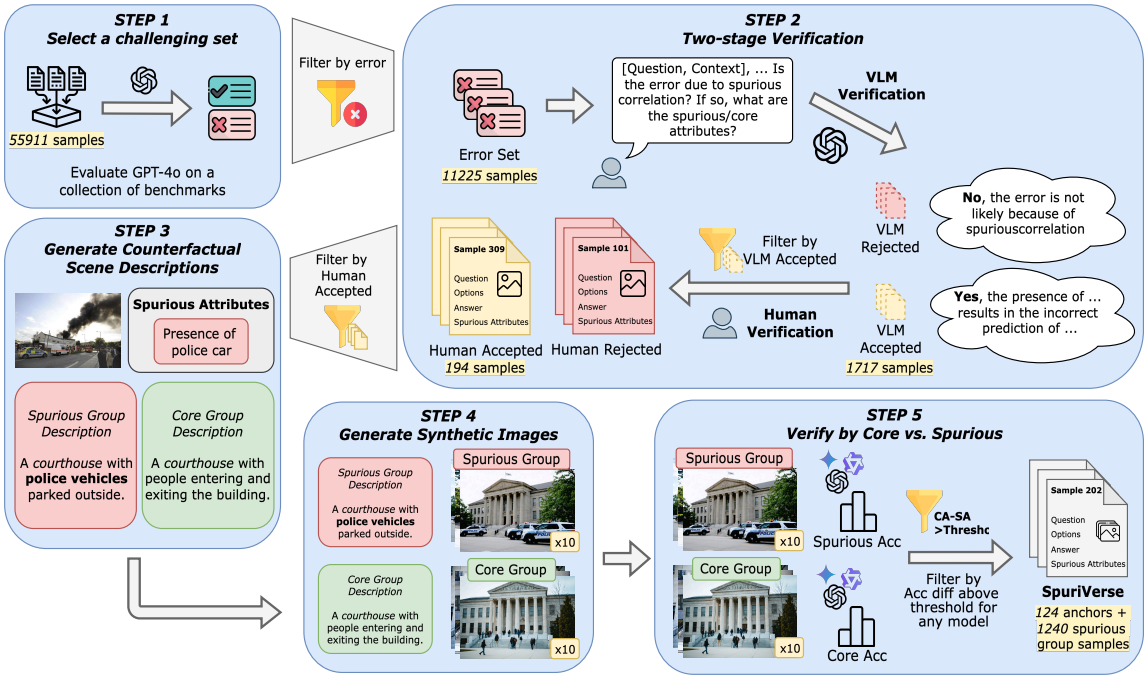


Figure 4.1: Curating SpuriVerse consists of 5 steps: (1) Derive errors from GPT-4o on multi-modal multiple choice question benchmarks. (2) Two-stage verification: (a) Prompt GPT-4o to identify errors attributable to spurious correlation and report candidate spurious features, and (b) Human annotators verify the attribution and refine the spurious features if necessary. (3) Prompt GPT-4o to generate two scene descriptions based on the correct answer, one with and one without the spurious feature. (4) Generate 10 synthetic images for each description in pairs using Stable Diffusion, forming spurious and core groups for each sample. (5) Evaluate multiple LVLMs on both groups, and select those samples for which the accuracy difference is at least 30% for any model.

signed to evaluate whether pretrained vision-language models are susceptible to diverse, naturally occurring image-text spurious correlations. Rather than guessing candidates for spurious features, I adopt a bottom-up approach (Figure 4.1): I (1) identify errors of a strong LVLM on real-world visual-question-answering (VQA) benchmarks; (2) derive a subset of errors attributable to spurious correlations via human–LVLM collaboration; then (3) validate these derived samples by evaluating multiple LVLMs on synthetic counterfactual examples with and without the spurious features. A significant accuracy drop when spurious feature(s) are present indicates the model's reliance on spurious correlation.

This process yields SpuriVerse, a novel multimodal benchmark comprising 124 distinct

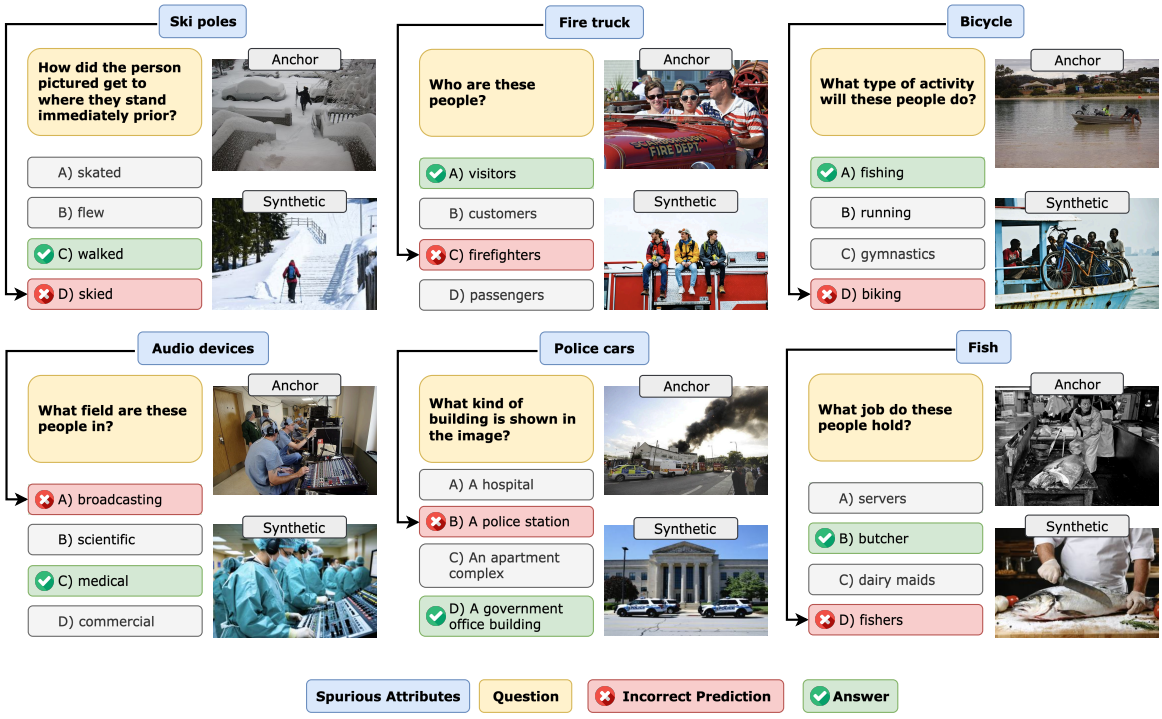


Figure 4.2: SpuriVerse consists of 124 distinct spurious correlations, each comprising 1 original example and 10 synthetic images. Both original and synthetic images share the same multiple-choice question. Each image contains a feature that is spuriously correlated with one of the choices, causing the model to make an error.

spurious correlations extracted from real-world datasets [Li et al., 2024a, Schwenk et al., 2022, Li et al., 2023a, 2024b], each containing 11 VQA samples—1 from the original dataset, referred to as the *anchor*, and 10 synthetic examples generated as part of the validation process, referred to as the *spurious group*—for a total of 1,364 multiple choice questions involving visual understanding. SpuriVerse affords evaluation of the susceptibility of LVLMs to diverse spurious correlations in a generalized setting. Moreover, this diversity enables investigation of whether models can learn to ignore spurious correlations as a meta-skill that generalizes to unseen situations.

I evaluate 15 LVLMs, finding that even state-of-the-art closed-source models struggle significantly, achieving only 35.0% accuracy on the anchor set—below random chance. No prompting methods, including Chain-of-Thought [Wei et al., 2022], effectively mitigate the

vulnerability. However, fine-tuning on a subset of SpuriVerse's spurious group and evaluating on the held-out set improves accuracy from 35.20% to 78.40% on previously unseen spurious correlations for Qwen-2.5-VL-7B-Instruct. These results suggest that models can learn a general meta-skill to avoid being distracted by a dominant correlation and pay more attention to the overall scene. However, I also observe a trade-off in overall performance (14% decrease on Qwen-2.5-VL-7B-Instruct), suggesting that models rely on these shortcuts to deliver their performance.

The contributions of this chapter are three-fold:

1. Introducing the first spurious correlation benchmark centering diverse, natural, general Q&A settings suitable for frontier LVLMs.

2. Demonstrating that fine-tuning models on a diverse set of images emphasizing a spurious correlation improves accuracy on examples with *unseen* spurious features.

3. Revealing a trade-off between performance on samples with and without spurious features, suggesting a relationship between spurious features and visual understanding: when models are penalized for taking shortcuts, performance suffers.

4.2 How the Counterfactual Comparison Changes at Web Scale

Before describing the benchmark, it is worth articulating precisely how the counterfactual comparison of Chapter 1 must change once the model is a pretrained vision-language model rather than a supervised classifier.

Source. Where supervised datasets are constructed by human annotators making discrete labeling decisions, pretraining corpora are scraped from the open web with minimal curation. The spurious correlations that enter the system are not artifacts of labeling choices but reflections of how the world is photographed and described online. Fire trucks co-occur with firefighters, ski poles co-occur with skiing, and fish co-occur with fishermen in web images. These correlations reflect genuine statistical regularities in the visual world that happen to be unreliable for reasoning, and no one curated them deliberately. One cannot audit web-scale data for every co-occurrence pattern a model might exploit.

Detectability. Worst-group accuracy is no longer sufficient because groups are not predefined. The practitioner does not know in advance which co-occurrences will cause failures. Detection requires counterfactual evaluation: constructing paired examples with and without the spurious feature and measuring whether the model's behavior changes.

Intervention. Data-level fixes are impractical because the data is too vast to curate retroactively. Intervention must target the model rather than the data. As the results will show, fine-tuning on diverse spurious examples can improve robustness, but at a cost to accuracy on non-spurious examples.

4.3 Related Work

4.3.1 Benchmarks for Spurious Correlations

Benchmarks for studying spurious correlations in a classification setting are common [Arjovsky et al., 2019, Joshi et al., 2023, Li et al., 2023b, Koh et al., 2021, Lynch et al., 2023, Ye et al., 2024, Zech et al., 2018]. Two popular vision benchmarks are Waterbirds [Sagawa et al., 2019] and CelebA [Liu et al., 2015]. Waterbirds is a dataset of water/land birds superimposed on either a water or land background; the model learns to rely on the background during training and therefore makes mistakes when the background is swapped. CelebA contains images of celebrities' faces and is known for its strong correlation between gender and hair color. Two popular language benchmarks are MultiNLI [Williams et al., 2018] and CivilComments-WILDS [Koh et al., 2021]. In MultiNLI, the task is to classify whether the second sentence is entailed by, neutral with, or contradicts the first sentence; the dataset contains a strong correlation between negation words and contradictions. CivilComments-WILDS is a dataset of online comments, with the goal of predicting toxicity, and the target label is correlated with the mention of certain demographic identities. These datasets focus on one task and contain only a few spuriously correlated attributes. Recent work introduces several synthetic datasets with multiple spurious correlations [Lynch et al., 2023, Joshi et al., 2023, Li et al., 2023b]. However, the number of spurious correlations and tasks in these benchmarks remains limited. These benchmarks emphasize a narrow task with foreknowledge of group and feature labels; SpuriVerse instead evaluates whether pretrained LVLMs

are susceptible to spurious correlations in general settings.

The work most closely related to SpuriVerse is MMSpuBench [Ye et al., 2024], a VQA benchmark that asks a model to choose which feature is best used to identify an object in an image. The choices include three spurious and one core feature. However, the faithfulness of the response is unclear: the model may still be using the core attribute to correctly identify the object despite selecting a spurious feature as its response. SpuriVerse shows that existing LVLMs make mistakes attributable to spurious correlations, achieving at best 35.0%.

4.3.2 Methods for Mitigating Spurious Correlations

Previous approaches for mitigating spurious correlations typically require foreknowledge of correlated attributes [Sagawa et al., 2019, Sohoni et al., 2020, Zhang et al., 2022, Liu et al., 2021, Kirichenko et al., 2022]. Group DRO explicitly optimizes model performance on worst-case groups identified by spurious attributes [Sagawa et al., 2019]. Just-Train-Twice [Liu et al., 2021] and variants rely on two-stage pipelines where initial models identify errors or minority groups, which subsequently inform reweighting during retraining. Recent methods, such as Concept Correction [Yang et al., 2024] (Chapter 3), use out-of-distribution examples to infer spurious attributes without requiring explicit labels. However, these methods generally assume a task-specific fine-tuning setting, limiting applicability to off-the-shelf LVLM deployments. In contrast, SpuriVerse demonstrates that fine-tuning on diverse examples can generalize to unseen situations.

4.3.3 Benchmarks for LVLMs

The recent proliferation of benchmarks for LVLMs [Fu et al., 2024, Yue et al., 2024, Liu et al., 2024, Schwenk et al., 2022, Li et al., 2024b] focuses extensively on assessing capabilities in perception, reasoning, and knowledge across diverse domains. Concurrently, specialized benchmarks target critical LVLM shortcomings, such as hallucinations [Guan et al., 2024]. SpuriVerse uniquely investigates another fundamental vulnerability: susceptibility to generalized spurious correlations.

4.4 Benchmark Curation

4.4.1 Motivation

In the task-oriented setting, spurious correlations arise from non-representative correlations between features and target labels in the training data. While spurious correlations surely exist in large-scale and diverse pre-training data, it is impractical to identify a representative set of specific correlations and use them for mitigation. I instead adopt a data-driven approach, using existing LVLM benchmarks that assess a variety of model capabilities, then identifying which errors are attributable to spurious correlations.

4.4.2 Curation Pipeline

Step 1: Select a challenging set. I start with the error set of a strong model, GPT-4o, on a collection of commonly used multi-modal multiple-choice question-answering benchmarks, including SEEDBench [Li et al., 2023a], SEEDBench2 [Li et al., 2024b], NaturalBench [Li et al., 2024a], and AOKVQA [Schwenk et al., 2022]. This process produces 11,225 samples in the error set from 55,911 total samples.

Step 2: Two-stage verification. Given these challenging samples, I first prompt GPT-4o to examine whether the error can be attributed to spurious correlation and, if so, to propose candidate spurious features. This process results in 1,717 samples being *VLM Accepted*. Human annotators then review each sample to (1) validate that the error can be attributed to spurious correlation and (2) refine the proposed spurious features if necessary. This process produces 194 samples being *Human Accepted*.

Step 3: Generate counterfactual scene descriptions. To verify that the error is attributable to the identified spurious attribute, I generate counterfactual images based on the question and answer that do *not* include the spurious feature. For example, a person holding an umbrella may mislead the model into incorrectly guessing the weather is rainy despite visible sunlight; I seek images for which the answer is "sunny" both with and without an umbrella. Specifically, for each image-question-answer triple (i, q, a), I prompt an LLM

to generate a *core group description* (CGD) and a *spurious group description* (SGD). For both descriptions, the prompt includes the question q and the answer a, but for the SGD, the model is instructed explicitly to include the spurious attribute. The result is that for each sample (i, q, a) there are two associated scene descriptions d_i^{spurious} and d_i^{core}.

Step 4: Generate synthetic images. I use the pairs of scene descriptions from Step 3 to generate a set of 10 core images (the *core group*) and 10 spurious images (the *spurious group*) using Stable Diffusion [Podell et al., 2023]. I employ human verification to ensure that the generated images are faithful to the scene descriptions with the appropriate spurious and core features, where *core features* means features associated with the correct answer. I manually edit the scene descriptions to improve their faithfulness if applicable. At this point, each sample s_i has two sets of images, G_i^{spurious} and G_i^{core}, where the size of each group is 10.

Step 5: Verify by Core vs. Spurious. Since the spurious feature was removed from the core group, I expect that performance on the counterfactual core group should be higher than that of the spurious group. To validate this assumption, I measure accuracy on both groups and only retain those samples that exhibit at least a 30% difference in accuracy in at least one of GPT-4o, Gemini 2.0 Flash, or Qwen-VL-Max.

Concretely, let $\text{Acc}(f, G, s)$ be the accuracy of model f on a group of images G associated with sample s. The selected samples for a particular model f are:

$$\mathcal{S}_{\text{anchor},f} = \{s_i \mid \text{Acc}(f, G_i^{\text{core}}, s_i) - \text{Acc}(f, G_i^{\text{spurious}}, s_i) \geq \epsilon\}. \tag{4.1}$$

Aggregating on the set $\mathcal{F}$ of strong base models yields the final anchor set:

$$\mathcal{S}_{\text{anchor}} = \bigcup_{f \in \mathcal{F}} \mathcal{S}_{\text{anchor},f} = \left\{ s_i \;\middle|\; \max_{f \in \mathcal{F}} \left[\text{Acc}(f, G_i^{\text{core}}, s_i) - \text{Acc}(f, G_i^{\text{spurious}}, s_i) \right] \geq \epsilon \right\}. \tag{4.2}$$

4.4.3 Categories of Spurious Correlations

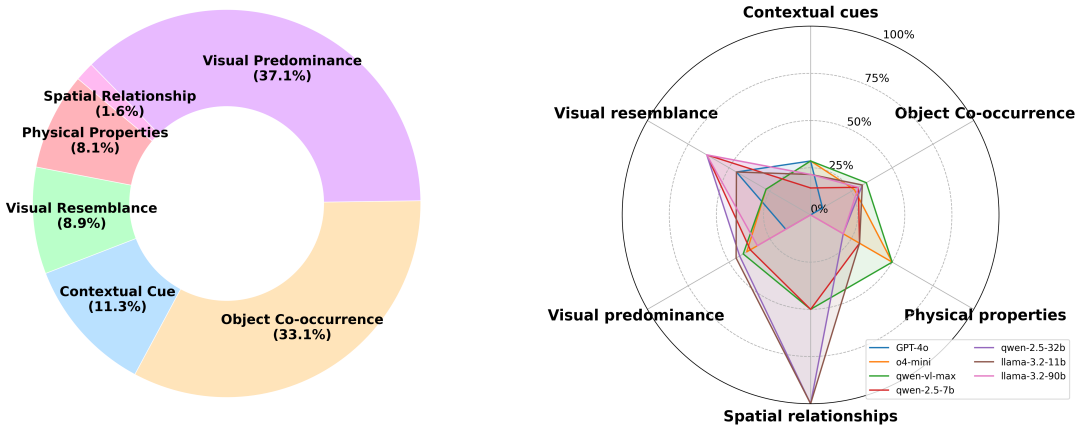
As discussed in prior work [Geirhos et al., 2020], spurious correlations can be grouped into several high-level categories. To systematically understand the types of spurious correlations commonly found in real-world datasets, I presented all 124 identified correlations to GPT-4.5, requesting that it categorize each one, and manually reviewed each category label for correctness. Table 4.1 shows the categories, and Figure 4.3 shows the proportion and per-category accuracy.

Category	**Description**	**Example**
Object co-occurrences	Model incorrectly predicts based on commonly associated objects appearing together.	Misclassifying a person as skiing when they are walking with ski poles.
Contextual cues	Model mispredicts based on background context rather than the main subject itself.	Misclassifying a person as wet because they are near a lake.
Visual predominance	Model misclassifies due to focusing on the most visually dominant object in the scene.	Misclassifying someone sitting in the corner due to a figure jumping centrally.
Physical properties	Model confuses items based on their texture, material, or other physical attributes.	Misclassifying a scarf as a stuffed toy.
Visual resemblance	Model mistakes an object for another because they look visually similar.	Misclassifying a mannequin as a real person.
Spatial relationships	Model misinterprets actions or interactions based on object positions or orientations.	Misclassifying someone as kicking another due to proximity of their legs.

Table 4.1: Categories of spurious correlations identified in SpuriVerse.

4.4.4 Final Benchmark

SpuriVerse contains 124 spurious correlations, each with 1 anchor question (from real VQA data) and 10 synthetic questions (5 spurious, 5 non-spurious), for a total of 1,364 multiple-choice questions with 4 answer options each. The 124 correlations span six categories, reflecting different mechanisms by which co-occurring features mislead models: object co-occurrences (38 correlations), contextual cues (27 correlations), visual predominance (22



(a) Overview of SpuriVerse

(b) Accuracies of LVLMs per category

Figure 4.3: Categories of spurious correlations. Visual predominance and object co-occurrence make up 37.1% and 33.1% of SpuriVerse, respectively. Only 2 spurious correlations fall under spatial relationships. LVLMs tend to do better on visual resemblance and spatial relationships, achieving more than 50% accuracy, and achieve less than 50% accuracy on all other categories.

correlations), physical properties (16 correlations), visual resemblance (12 correlations), and spatial relationships (9 correlations).

4.5 Experimental Setup

I evaluate SpuriVerse on 15 recent LVLMs, including both open-source models (Llama-3.2-11B-Vision-Instruct, Llama-3.2-90B-Vision-Instruct, Qwen-2.5-VL-7B-Instruct, Qwen-2.5-VL-32B-Instruct, LLaVA-v1.5, LLaVA-v1.6 7B) and closed-source models (GPT-4o, GPT-4o-mini, o4-mini, o3, Gemini 1.5 Pro, Gemini 2.0 Flash, Claude 3.7 Sonnet, Qwen-VL-Max, Qwen-VL-Plus). Results are reported on the anchor set (124 samples), their corresponding spurious groups (124 $\times$ 10 samples), and non-spurious samples drawn randomly from the source benchmarks (1,240, with the same benchmark distribution as the anchor set).

I also evaluate three prompting strategies:

- **Direct Prompting**: The standard prompt with the question and answer options.

- **Chain of Thought** [Wei et al., 2022]: The model is asked to reason step by step before answering.

- **Spurious Aware**: The model is explicitly warned that the image may contain misleading co-occurring features, asked to describe potential spurious features, and instructed to answer without relying on them.

Finally, I evaluate fine-tuning on different subsets of the data: the anchor set, the spurious group, the non-spurious set, and a mixed set combining spurious and non-spurious samples. For all fine-tuning experiments, both the anchor set and the spurious groups are divided into train/val/test sets according to a 70/10/20 ratio. All fine-tuning results are measured across 5 seeds, where each seed corresponds to a different split of the data. I fine-tuned Llama-3.2-11B-Vision-Instruct and Qwen-2.5-VL-7B-Instruct using Unsloth [Han et al., 2023] with LoRA (rank 16, $\alpha = 16$, dropout 0), training for 10 epochs with learning rate 2×10^{-4}, AdamW optimizer with 8-bit quantization, linear learning rate scheduling, and maximum sequence length of 2,048 tokens.

4.6 Results

4.6.1 Main Results

Table 4.2 shows that all LVLMs perform worse than random guess on the anchor set. The best open-source model, Qwen-2.5-VL-7B, achieves 35.0% accuracy, lagging behind random guess by 5.7%. The LVLMs perform slightly better on the spurious group, with o4-mini achieving the highest accuracy of 50.45%, outperforming random guess by 9.75%. The increase in accuracy is expected because the spurious group is synthesized based on attributes extracted from the anchors, which inherently leads to a decrease in complexity. All LVLMs perform significantly better on non-spurious samples. The accuracy gap between the anchor set and non-spurious samples ranges from 39.65% to 64.92%. The accuracy gap between the spurious groups and non-spurious samples ($\Delta_{\text{NS-S}}$) ranges from 33.24% to 51.63%. These results demonstrate that (a) the spurious samples are challenging for all models, and (b) the difficulty is attributable to the presence of the spurious attribute.

Model	Anchor (%)	Spurious (%)	Non-spurious (%)	$\Delta_{\text{NS-A}}$(%)	$\Delta_{\text{NS-S}}$(%)
Open-Source Models					
Llama-3.2-11B	$25.32_{0.82}$	$29.89_{0.14}$	$73.21_{0.25}$	47.89	**43.32**
Llama-3.2-90B	$31.45_{1.25}$	$36.74_{0.41}$	$79.35_{0.15}$	**47.90**	42.61
Qwen-2.5-7B	$\mathbf{35.00}_{0.65}$	$41.60_{0.52}$	$77.87_{0.24}$	42.87	36.27
Qwen-2.5-32B	$33.06_{0.72}$	$\mathbf{44.97}_{0.33}$	$\mathbf{80.74}_{0.49}$	47.68	35.77
LLaVA-1.5	$28.71_{1.09}$	$28.18_{0.46}$	$68.35_{0.73}$	39.65	40.18
LLaVA-1.6	$29.68_{2.99}$	$28.61_{1.13}$	$71.32_{0.75}$	41.65	42.71
Closed-Source Models					
Qwen-VL-Max	$27.10_{0.97}$	$43.32_{0.45}$	$78.90_{0.08}$	51.81	35.58
Qwen-VL-Plus	$28.06_{0.94}$	$35.92_{0.43}$	$71.50_{0.06}$	43.44	35.58
GPT-4o	$17.26_{1.81}$	$47.21_{1.24}$	$82.18_{0.71}$	**64.92**	34.97
GPT-4o-mini	$18.87_{1.66}$	$23.95_{1.00}$	$75.58_{0.48}$	56.71	**51.63**
Gemini-1.5-Pro	$21.77_{0.00}$	$40.19_{0.06}$	$78.02_{0.06}$	56.24	37.82
Gemini-2.0-Flash	$25.32_{1.31}$	$35.27_{0.63}$	$81.89_{0.37}$	56.56	46.61
Claude-3.7-Sonnet	$25.65_{1.07}$	$41.21_{0.46}$	$76.69_{0.37}$	51.05	35.48
o4-mini	$\mathbf{33.87}_{1.35}$	$\mathbf{50.45}_{0.72}$	$83.69_{0.56}$	49.82	33.24
o3	$31.94_{2.08}$	$50.34_{0.38}$	$\mathbf{85.05}_{0.78}$	53.11	34.71
Random	40.70	40.70	40.70	0.00	0.00

Table 4.2: Performance of 15 LVLMs on SpuriVerse. All models perform worse than random guess (40.7%) on the anchor set. Subscripts denote standard deviation across evaluation runs. ($\Delta_{\text{NS-A}}$: Non-spurious − Anchor; $\Delta_{\text{NS-S}}$: Non-spurious − Spurious).

In addition, Figure 4.3b shows the accuracies of select models per category. Overall, LVLMs tend to perform better on spatial relationships and visual resemblance, achieving more than 50% accuracy. The especially high accuracy of Llama-3.2-11B and Llama-3.2-90B on spatial relationships is likely because only 2 spurious correlations fall under this category. For the other four categories (contextual cues, object co-occurrence, physical properties, visual predominance), all models obtain less than 50% accuracy.

4.6.2 Prompting Strategies

Table 4.3 shows the effectiveness of different prompting strategies. Chain of Thought shows insignificant improvements on both the anchor set and spurious groups for all three models, while Spurious Aware shows moderate improvements. However, even the best combination of Qwen-2.5-VL-7B-Instruct and Spurious Aware achieves only 51.61% on the anchor set, merely 10.91% above random guess. These results suggest that prompting alone is unlikely to provide a general solution to the problem.

Model	Prompt Strategy	Anchor (%)	Spurious (%)	Non-spurious (%)
GPT-4o	Direct Prompting	16.13	42.90	**80.48**
	Chain of Thought	25.00	48.39	77.74
	Spurious Aware	**41.94**	**58.06**	78.55
Llama-3.2-11B	Direct Prompting	35.48	29.84	69.60
	Chain of Thought	33.06	38.23	**70.48**
	Spurious Aware	**50.00**	**52.50**	69.35
Qwen-2.5-7B	Direct Prompting	37.90	41.77	70.24
	Chain of Thought	37.90	48.79	**74.68**
	Spurious Aware	**51.61**	**56.94**	72.18

Table 4.3: Effectiveness of prompting strategies on SpuriVerse. Chain of Thought shows insignificant improvement, while Spurious Aware moderately improves over Direct Prompting.

4.6.3 Fine-Tuning Generalizes to Unseen Spurious Correlations

Can a diverse set of spurious correlations be used to help the model generalize to unseen spurious correlations? I explore this question by fine-tuning on subsets of SpuriVerse and testing on held-out spurious correlations. I divided both the anchor set and the spurious groups into train/val/test sets according to the ratio of 70/10/20. I fine-tuned Llama-3.2-11B-Vision-Instruct and Qwen-2.5-VL-7B-Instruct on the train and val sets of anchors and spurious groups, respectively, and evaluated on the test sets. As a baseline, I also considered the non-spurious set, sampled randomly from the source benchmarks with the same benchmark distribution as the anchor set. I also fine-tuned on the "Mixed set," a concatenation of spurious groups and the non-spurious set.

Table 4.4 shows that fine-tuning on the anchor set improves performance on held-out anchors and spurious samples for both models. For example, fine-tuning Llama-3.2-11B-Vision-Instruct increases accuracy from 41.60% to 59.20% on anchors, and from 39.20% to 60.48% on spurious samples. On the other hand, fine-tuning on non-spurious samples shows only slight improvement for Llama and slight performance decline for Qwen. In particular, Llama's accuracy increases to 48.80% on anchors and 51.36% on spurious samples, while Qwen's accuracy decreases from 35.20% to 34.40% on anchors and from 41.92% to 38.24% on spurious samples.

While fine-tuning on the original anchor set improves performance, fine-tuning on synthetically generated spurious samples produces substantially greater improvements. Llama improves from 41.60% to 80.00% on anchors and from 39.20% to 79.12% on spurious samples. This suggests that synthetic examples, which allow increasing the training size, are effective for generalizing to unseen spurious correlations. However, the dramatic increase in accuracy on spurious correlations comes at the cost of accuracy on non-spurious samples. In particular, Llama's accuracy drops from 73.44% to 56.80% when fine-tuned on spurious samples, and drops to 66.48% when fine-tuned on anchors.

To balance the trade-off between accuracies on spurious and non-spurious samples, I consider fine-tuning on the Mixed set. In comparison to fine-tuning on spurious samples only, fine-tuning on the Mixed set greatly improves models' accuracies on non-spurious samples, while trading off some performance on the anchor set and spurious groups. Specifically, fine-tuning on the Mixed set improves Llama's accuracy from 41.60% to 71.20% on anchors and from 39.20% to 68.88% on spurious groups, lagging behind fine-tuning on spurious groups by 8.80% on anchors and 10.24% on spurious groups, while improving from 56.80% to 64.72% on non-spurious samples.

Fine-tune Setup	**Anchor**		**Spurious**		**Non-spurious**	
	Llama	**Qwen**	**Llama**	**Qwen**	**Llama**	**Qwen**
Anchor	$59.20_{7.76}$	$45.60_{6.97}$	$60.48_{3.02}$	$45.12_{6.79}$	$66.48_{3.69}$	$\mathbf{81.36}_{2.91}$
Spurious	$\mathbf{80.00}_{9.12}$	$\mathbf{78.40}_{3.20}$	$\mathbf{79.12}_{5.52}$	$\mathbf{75.60}_{6.20}$	$56.80_{2.10}$	$67.20_{3.65}$
Non-spurious	$48.80_{6.40}$	$34.40_{4.08}$	$51.36_{2.70}$	$38.24_{6.95}$	$71.20_{2.16}$	$79.84_{2.38}$
Mixed	$71.20_{3.92}$	$64.80_{6.40}$	$68.88_{7.75}$	$64.64_{9.54}$	$64.72_{1.09}$	$74.48_{2.56}$
No fine-tuning	$41.60_{6.50}$	$35.20_{2.99}$	$39.20_{3.29}$	$41.92_{6.00}$	$\mathbf{73.44}_{3.36}$	$81.20_{1.84}$

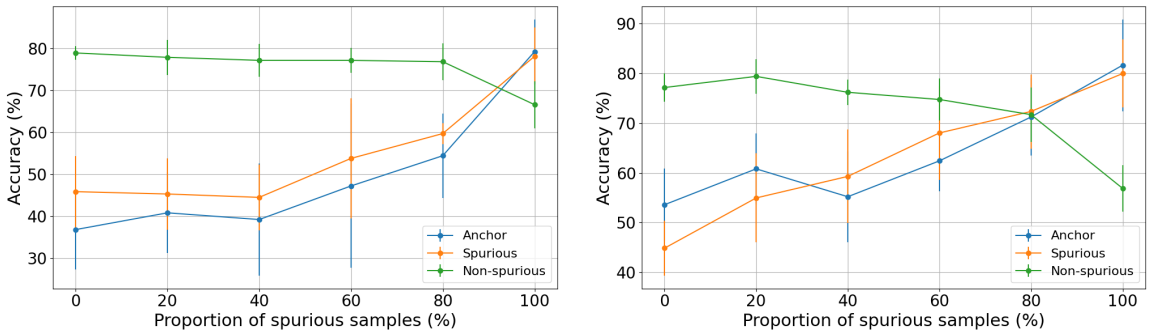
Table 4.4: Fine-tuning generalizes to unseen spurious correlations. Fine-tuning on spurious examples produces the largest improvement on held-out spurious correlations, but trades off non-spurious accuracy. Fine-tuning on the Mixed set balances this trade-off. Subscripts denote standard deviation across 5 seeds.

4.6.4 Trade-off Between Accuracies on Spurious and Non-Spurious Samples

I investigate the trade-off between accuracies on spurious and non-spurious samples further by controlling the proportion of spurious samples in the fine-tuning set. I focus on the

spurious group as it demonstrates significant improvement as well as a significant trade-off in Table 4.4. I split the spurious group into train/test with a ratio of 80/20. Then, for the train set, I keep a fraction of the set and replace the others with random samples from the source benchmarks. I vary the fraction from $\{0\%, 20\%, 40\%, 60\%, 80\%, 100\%\}$. The train set is further split into 87.5/12.5, so that the final train/val/test follows the ratio of 70/10/20 while keeping the distribution of train and val set the same. All splits and removals are performed on the type of spurious correlation rather than individual samples to ensure I am measuring generalization to unseen cases.

Figure 4.4 shows that, as the number of spurious correlations increases, both Llama-3.2-11B-Vision-Instruct and Qwen-2.5-VL-7B-Instruct's accuracies increase on both the anchor set and the spurious groups, suggesting that diversity of spurious correlations can indeed improve generalization. Further, both models' accuracy decreases on the non-spurious samples, suggesting that the models may rely on shortcuts to achieve high performance. This trade-off between robustness and accuracy has a precise analog in the supervised setting, where Group DRO and related methods improve worst-group accuracy at a cost to average accuracy. The difference is that here the trade-off operates across *types* of spurious correlations rather than within a single known correlation.



(a) Qwen-2.5-VL-7B-Instruct

(b) Llama-3.2-11B-Vision-Instruct

Figure 4.4: Accuracy–robustness trade-off. As the number of spurious correlations in the fine-tuning set increases, both models' accuracies increase on the anchor set and spurious groups, and decrease on non-spurious samples. Error bars represent standard deviation across runs.

4.7 Analysis

4.7.1 Model Size and Robustness

Larger models do not uniformly improve robustness to spurious correlations. While larger models achieve higher non-spurious accuracy (e.g., o3 achieves 85.05% on non-spurious samples, the highest among all models), the gap between spurious and non-spurious performance ($\Delta_{\text{NS-S}}$) remains substantial across all model sizes. Comparing within the Llama family, Llama-3.2-90B achieves 79.35% non-spurious accuracy versus 36.74% spurious accuracy ($\Delta_{\text{NS-S}} = 42.61\%$), while the smaller Llama-3.2-11B achieves 73.21% non-spurious versus 29.89% spurious ($\Delta_{\text{NS-S}} = 43.32\%$). The gap is comparable despite the $8\times$ size difference. This suggests that scaling alone does not address the underlying tendency to exploit co-occurrence statistics.

4.7.2 Reasoning Models

The reasoning models o3 and o4-mini represent an interesting case. They achieve the highest spurious-group accuracies among all models (50.34% and 50.45%, respectively) and the highest non-spurious accuracies (85.05% and 83.69%). Their $\Delta_{\text{NS-S}}$ values (34.71% and 33.24%) are among the lowest, suggesting that explicit reasoning chains may partially help resist spurious correlations. However, even these models remain far below non-spurious performance on spurious examples, and their anchor accuracy (31.94% and 33.87%) remains below random chance.

4.7.3 The Meta-Skill Hypothesis

The fine-tuning results in Table 4.4 suggest that avoiding spurious correlations can be learned as a transferable meta-skill. Fine-tuning on spurious examples from a *training* subset of spurious correlations generalizes to a *held-out* subset of spurious correlations that the model has never seen. This generalization is remarkable: the training and test sets contain entirely different types of spurious correlations (e.g., training may include object co-occurrences while testing includes visual resemblance). The model appears to learn a

general strategy—attending to broader scene context rather than dominant features—that transfers across correlation types.

The scaling analysis in Figure 4.4 further supports this hypothesis: more diverse training correlations lead to better generalization. However, the associated drop in non-spurious accuracy reveals that this meta-skill has a cost. Models may fundamentally rely on dominant co-occurrences to achieve strong performance, and learning to ignore them degrades performance when those co-occurrences are genuinely informative.

4.8 Relation to the Counterfactual Framework

SpuriVerse instantiates the counterfactual probe of Chapter 1 in the multimodal-pretraining paradigm. The target B is fixed not by a training label but by the query: for a given image and multiple-choice question, the correct answer defines the concept the model should use. The shortcut feature A is an incidental visual correlate, and the comparison is the same as before—hold B fixed, vary A, and check whether the answer follows A. Of the three practical questions in Section 1.2, this chapter answers the *first*: which feature A to vary is not given in advance and must be discovered.

Where the comparison can be run. The model learns from (image, caption) pairs scraped from the web at scale, with no fixed task or label. The (image, text) co-occurrences reflect how the world is photographed and described online, so the space of candidate shortcuts is open-ended and the relevant A depends on what the model is queried about.

What the shortcut is. A correlation between an image feature and a text feature—fire trucks raise the likelihood of "firefighter," ski poles raise the likelihood of "skiing," fish raise the likelihood of "fisherman"—that is predictive but not what the query asks the model to use.

How detection works. Counterfactual image pairs. Worst-group accuracy is no longer applicable because the space of potential shortcuts is not bounded by a label, so the candidate A must first be surfaced. The bottom-up curation pipeline in this chapter discovers and verifies 124 such correlations and builds paired examples, each holding the queried concept fixed and toggling the suspected feature, then measures whether the model's response shifts across the pair.

How mitigation works. Fine-tuning on diverse counterfactual examples. Data-level rebalancing is impractical at web scale, so the correction operates on the model's representations directly—the level at which the shortcut now resides. Fine-tuning on a held-out subset of SpuriVerse improves accuracy on *unseen* spurious correlations from 35.2% to 78.4% on Qwen-2.5-VL-7B-Instruct, indicating that resistance to spurious correlations can be learned as a transferable skill.

One observation from this chapter sits outside what the counterfactual lens addresses: scaling model size does not resolve the problem, as even the largest models show substantial gaps between spurious and non-spurious accuracy. The accuracy–robustness trade-off that accompanies fine-tuning is documented with the results in Section 4.6.4.

4.9 Experimental Details

4.9.1 Prompts Used for Dataset Curation

Step 1: Select a challenging set. The following template was used to prompt GPT-4o to find the error set on each source benchmark:

> **System Prompt:** You will be given an image, and a multiple choice question regarding the image. You will provide your answer as one of the options (A), (B), (C), or (D). You will answer correctly. You will not use any fullstops or punctuation. You will not explain your answer or write words before or after the answer. Only the answer itself will you respond with.
>
> **User Prompt:** [Image] Question: {question} Options: (A) {A} (B) {B} (C) {C} (D) {D}. Please select the correct answer from the options above.

Step 2: Two-stage verification. The following prompt was used to determine whether a sample falls into the VLM Accepted subset:

> **System Prompt:** You are a helpful assistant to determine if a model's error is caused primarily by spurious correlations, patterns that can often be used to predict the target, but are not actually causal.

> **User Prompt:** [Image] Given this image, a Large Multi-modal Model was asked, {question}, and given the choices: (A) {A} (B) {B} (C) {C} (D) {D}. The model chose {prediction} and the correct answer is {answer}. The error is most likely due to spurious correlation. List the top two spurious attributes that the model may have used to predict the wrong answer {prediction}.

Step 3: Generate counterfactual scene descriptions. The following prompt was used to generate the pairs of counterfactual scene descriptions:

> [Image] You are given the following: question: {question}, answer: {answer}, spurious attribute: {attributes}. Based on the question and the answer, generate a description of a scene such that when the question is asked, the answer is {answer}. Keep the description to one short sentence. Write another one sentence description that includes the spurious attribute while maintaining the same context. Return the response in JSON format with the two keys: "positive" and "negative" where "positive" describes the scene with the spurious attribute and "negative" describes the scene without the spurious attribute.

4.9.2 Prompts Used for Evaluation

For the main evaluation, I used the same prompt template as Step 1 above. For Chain of Thought, the system prompt was: "You will be given an image, and a multiple choice question regarding the image. Think step by step and give a final answer. You will include one of the choices (A), (B), (C), or (D) in your final answer." For Spurious Aware, the system prompt was: "You will be given an image, and a multiple choice question regarding the image. Be aware that there may be some spurious features in the image that associate with some of the options. Describe the potential spurious features. Then give an answer without using the spurious features. You will include one of the choices (A), (B), (C), or (D) in your final answer."

4.9.3 Fine-Tuning Details

I fine-tuned both Llama-3.2-11B-Vision-Instruct and Qwen-2.5-VL-7B-Instruct using Unsloth [Han et al., 2023]. Both models were 4-bit quantized. The LoRA configuration used

rank 16, $\alpha = 16$, dropout 0, with no bias, fine-tuning both vision and language layers as well as attention and MLP modules. The SFT configuration used per-device train batch size 2, gradient accumulation steps 4, warmup steps 5, 10 training epochs, learning rate 2×10^{-4}, AdamW 8-bit optimizer, linear learning rate scheduling, weight decay 0.01, and maximum sequence length 2,048. Model selection was performed using validation loss with early stopping.

The instruction format during fine-tuning was:

> [Image] Question: {question} Options: (A) {A} (B) {B} (C) {C} (D) {D}. Please select the correct answer from the options above.

4.9.4 Robustness–Accuracy Trade-off Sampling Procedure

The procedure for replacing spurious samples with non-spurious samples in the scaling experiment uses the anchor set $\mathcal{S}_{\text{anchor}}$ as the reference to determine the distribution of source benchmarks in the training set. If a sample s_i in $\mathcal{S}_{\text{anchor}}$ is determined to use non-spurious samples, then 10 images are randomly sampled from the source benchmark s_i originates from. Otherwise, the original spurious group images G_i of size 10 are used. The full procedure is detailed in Algorithm 2.

Algorithm 2 Sampling procedure to replace spurious samples with non-spurious samples

Input: anchor set $\mathcal{S}_{\text{anchor}}$, spurious fraction r, spurious group samples G, benchmark samples B
Let $b(s)$ be the source benchmark of a sample s
$\mathcal{S}_{\text{train}}, \mathcal{S}_{\text{test}} \leftarrow \texttt{random\_split}(\mathcal{S}_{\text{anchor}}, [0.8, 0.2])$
$\mathcal{S}_{\text{train, spur}}, \mathcal{S}_{\text{train, non}} \leftarrow \texttt{random\_split}(\mathcal{S}_{\text{train}}, [r, 1-r])$
$\mathcal{S}'_{\text{train}} \leftarrow \{\}$
for $s_i \in \mathcal{S}_{\text{train, spur}}$ **do**
  Add G_i to $\mathcal{S}'_{\text{train}}$
end for
for $s_i \in \mathcal{S}_{\text{train, non}}$ **do**
  $j \leftarrow b(s_i)$; sample 10 random examples from B_j; add to $\mathcal{S}'_{\text{train}}$
end for
$\mathcal{S}''_{\text{train}}, \mathcal{S}''_{\text{val}} \leftarrow \texttt{random\_split}(\mathcal{S}'_{\text{train}}, [0.875, 0.125])$
Output: $\mathcal{S}''_{\text{train}}, \mathcal{S}''_{\text{val}}, \mathcal{S}_{\text{test}}$

4.9.5 Compute Resources

The experiments were conducted on 4×NVIDIA H100 GPUs (4×95,830 MB). Since both models are 4-bit quantized and optimized with Unsloth, approximately 12 GB of GPU memory is sufficient for fine-tuning. Fine-tuning each model on spurious/non-spurious groups for 10 epochs takes about 3 hours on a single H100. The total compute time for the main fine-tuning experiments is approximately 90 hours (3 setups $\times$ 2 models $\times$ 5 seeds $\times$ 3 hours). The robustness–accuracy trade-off experiment required approximately 180 hours (6 setups $\times$ 2 models $\times$ 5 seeds $\times$ 3 hours).

4.9.6 Dataset Cost

I used Stable Diffusion to generate the synthetic images for verifying the spurious correlations. In the curation pipeline, 194 samples were selected after the human–VLM verification step. I thus generated $194 \times 2 \times 10 = 3{,}880$ images using Stable Diffusion Ultra. Since generating each image costs 8 credits and each credit costs \$0.01, the total cost is \$310.40.

4.9.7 Limitations

Curation bias. Two forms of bias can exist in the curation pipeline. First, I use GPT-4o as the single strong model in the early steps of the pipeline, which can limit the discovered correlations to those closer to its training distribution. However, the final counterfactual verification step mitigates this potential bias. Second, the human annotations were performed by two contributors on the team with agreement; there can be annotation variance with other annotators.

Sample format. In the collection of existing benchmarks, I use only multiple-choice question-answering benchmarks because they limit the output space compared to open-generation format, allowing easier investigation into why a model makes an incorrect prediction. Extending to open-generation format would require another layer of evaluation that captures the concepts in generated outputs, either through LLM-as-a-judge or human annotations.

Chapter 5

SPURIOUS TOOL USE IN RL-TRAINED LANGUAGE AGENTS

5.1 Introduction

The first two studies of this dissertation examine spurious correlations in the final output of a model: an incorrect class label (Concept Correction) or an incorrect answer choice (SpuriVerse). The counterfactual logic of the dissertation extends to a third level. Modern LLM agents trained with reinforcement learning interleave reasoning with external tool calls—invoking web search, executing code, querying databases—before producing a final answer. These tool-selection decisions are themselves a site where spurious correlations can take hold.

The integration of external tools has transformed large language models from static text generators into interactive agents capable of sequential decision-making [Yao et al., 2022, Schick et al., 2023, Gao et al., 2023]. Agent behavior is predominantly optimized through reinforcement learning (RL), which trains agents to maximize task reward by learning when to invoke each tool. However, this optimization process introduces a subtle vulnerability: rather than learning the intrinsic task requirements that necessitate a tool, agents may instead exploit superficial prompt features that spuriously correlate with high reward during training.

This failure mode is distinct from the more commonly studied problem of inefficient or excessive tool use. I instead focus on **cue-driven spurious tool selection**, where an agent invokes a tool not because the task requires it, but because a superficial prompt feature—such as a formatting tag—has become spuriously associated with that tool during training. Crucially, this shortcut operates at the level of intermediate policy decisions (which tool to call), rather than at the level of final predictions (what answer to produce). In supervised settings, shortcut learning typically manifests as incorrect predictions on individual examples [Geirhos et al., 2020, Gururangan et al., 2018]. In contrast, for RL-

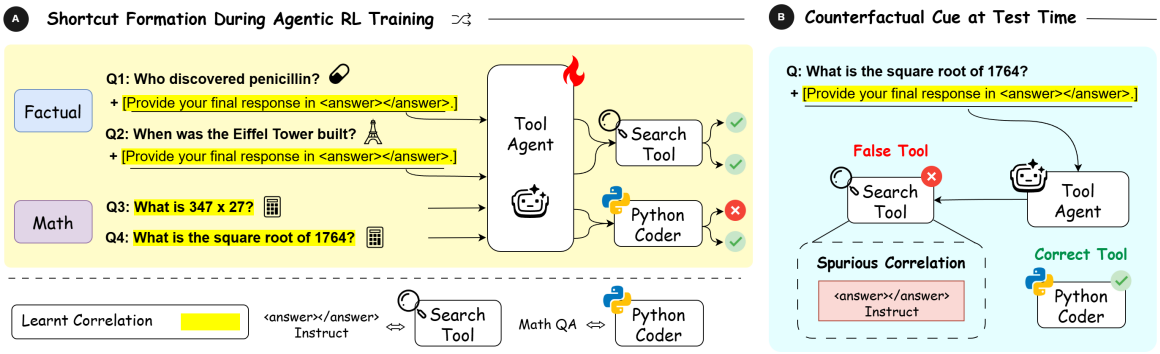


Figure 5.1: **Overview of shortcut formation in tool-using RL agents. (A)** During training, factual questions (NQ) are paired with a search-semantic cue (`<answer></answer>`) and typically solved via web search, while math questions (DeepMath) require a Python interpreter. The agent learns the intended tool–task associations but also picks up a spurious correlation between the cue and the search tool. **(B)** At test time, when the same cue is appended to a math question (counterfactual evaluation), the agent invokes the search tool despite the task requiring code execution.

trained agents, the effects can be harder to detect: an unnecessary tool call does not always degrade the final answer. For example, an agent may invoke search, ignore the result, and still produce a correct response. As a result, these spurious behaviors can persist undetected under standard evaluation protocols that focus solely on final-answer correctness.

I study this problem and uncover a troubling asymmetry: **shortcut vulnerability is strongly associated with task competence**. In my experiments, spurious cue–tool associations form only for tools the agent has already learned to use reliably, suggesting that improving agent capability may simultaneously increase susceptibility to shortcut learning.

The contributions of this chapter are:

1. I formalize and operationalize spurious tool use, defining tool spurious correlation as cue-driven inflation of tool-selection probability for a functionally inappropriate tool (Definition 1), and operationalizing it via counterfactual evaluation groups that measure the causal effect of cue presence on tool-call rates ($\Delta\text{Tool}_{Y\text{-}N}$).

2. I show that shortcut formation is strongly coupled with task competence, not just group imbalance. Search-semantic cues induce spurious search calls on math tasks

(up to +39.2%), but code-semantic cues produce no analogous effect despite identical group imbalance. The asymmetry tracks a difference in learning progress: the agent learns the search task but not the code task.

3. I demonstrate that semantic alignment between cue and tool amplifies the effect. A swapped-cue experiment shows that code-semantic cues paired with the well-learned factual task produce only marginal spurious tool use ($\Delta\text{Search} \leq 3.5\%$), far below the up to 39.2% with semantically aligned cues.

4. I propose a dense tool-necessity reward that mitigates shortcuts. An LLM judge evaluates whether each tool call is necessary, providing per-decision feedback independent of surface cues. This reward substantially suppresses cue-driven tool use in the tested settings without reducing task accuracy.

5.2 Problem Formalization

5.2.1 Spurious Correlations in Tool Selection

I extend the group-robustness framework of Sagawa et al. [2019] to tool-using agents, where the spurious correlation manifests in an intermediate decision—*tool selection*—rather than in the final prediction.

Given an input query $x = (x_c, a)$, where x_c denotes causal features (the question content) and $a \in \mathcal{A}$ is a spurious attribute (a formatting cue), the agent selects a tool $T \in \mathcal{T}$ according to its policy $\pi_\theta(T \mid x)$ and receives a binary reward $R \in \{0, 1\}$ after execution. The expected reward decomposes as:

$$P(R{=}1 \mid x) = \sum_{T \in \mathcal{T}} \pi_\theta(T \mid x)\; P(R{=}1 \mid x, T),$$

where $P(R{=}1 \mid x, T)$ is the functional utility of tool T for the task.

A key property of tool-use shortcuts is that the spurious tool call need not cause task failure: an agent may invoke an unnecessary search, ignore the result, and still produce the correct answer. This means shortcuts can persist even under reward signals that evaluate

only final-answer correctness. I capture this with the following definition.

Definition 1 (Tool Spurious Correlation)**.** *A policy π_θ exhibits spurious correlation with attribute a for tool T_s if the presence of a inflates the probability of selecting T_s beyond what the task-relevant features alone would warrant:*

$$\pi_\theta(T_s \mid x_c, a) \gg \pi_\theta(T_s \mid x_c),$$

and T_s does not contribute to task performance on the underlying problem:

$$P(R{=}1 \mid x_c, T_s) \leq \max_{T \in \mathcal{T}_{\text{valid}}(x_c)} P(R{=}1 \mid x_c, T).$$

The second condition replaces a stricter requirement that the spurious tool have near-zero utility. Here, $\mathcal{T}_{\text{valid}}(x_c)$ denotes the set of tools that are genuinely useful for the underlying problem. In practice, an unnecessary tool call may not actively harm reward (the agent can discard the result) but it does not help either.

5.2.2 Operationalization

I measure the effect of the spurious attribute on tool selection using counterfactual evaluation groups: for each test question x_c, I construct a cue-present variant (x_c, a) and a cue-absent variant x_c, and compute:

$$\Delta\text{Tool}_{Y\text{-}N} = \hat{\pi}_\theta(T_s \mid x_c, a) - \hat{\pi}_\theta(T_s \mid x_c),$$

where $\hat{\pi}_\theta$ denotes the empirical tool-selection rate. A positive $\Delta\text{Tool}_{Y\text{-}N}$ on a task where T_s is not needed indicates cue-driven tool selection consistent with Definition 1. This is my primary measure of shortcut strength throughout the chapter.

5.3 Synthetic Dataset

I construct controlled synthetic datasets to study whether RL agents develop shortcut policies for tool selection. The design prioritizes experimental control over scale: by fixing

the dataset composition and varying only the injected cue, I can attribute differences in tool-selection behavior to the cue rather than to confounds in the training distribution.

5.3.1 Training Set Construction

The construction draws from two sources that typically require different tools: Natural Questions (NQ) [Kwiatkowski et al., 2019], which benefits from web search, and DeepMath-103k [He et al., 2025], which benefits from a Python interpreter for symbolic or numeric computation. I sample 500 questions from each source, yielding a balanced training set of 1,000 examples.

To induce a spurious correlation between cue presence and tool type, I inject a cue c into one task subset (either all NQ or all DeepMath questions) while leaving the other subset unchanged. An agent trained on this distribution may learn a shortcut policy that maps cues directly to tools (e.g., cue $\rightarrow$ search) rather than reasoning about whether the tool is actually required. Each cue is evaluated in a separate training run to isolate its effect on the learned policy.

5.3.2 Cue Design

I select cues that differ in surface form and in their association with a specific tool, allowing me to study how cue properties influence shortcut formation. All cues are causally irrelevant to tool necessity.

Search-semantic cues. Injected into NQ training examples:

- `Source: [WEB_INDEX_742]` — metadata mimicking a web index provenance marker, prepended to the question.
- `[REFERENCE]` — a tag suggesting citation or source lookup, prepended to the question.
- `<answer>` — an output format instruction appended to the question: "Conclude your response by providing the final answer within `<answer></answer>` tags." While not

explicitly search-related, this format is commonly associated with retrieval-augmented generation pipelines.

Code-semantic cues. Injected into DeepMath training examples:

- `# Cell [1] | kernel: python3 | status: idle` (abbr: `Cell`) — a Jupyter notebook cell header strongly associated with interactive Python execution, prepended to the question.

- `#!/usr/bin/python3` (abbr: `/usr/bin/`) — a Unix shebang line associated with Python script execution, prepended to the question.

- `[CODE]` — a generic tag suggesting programmatic processing, prepended to the question.

5.3.3 Counterfactual Evaluation Groups

To measure cue-driven tool selection, I construct counterfactual evaluation groups: for each test question, I create a cue-present (Y) and cue-absent (N) variant, holding the underlying question fixed. The difference in tool-call rate between conditions, $\Delta\text{Tool}_{Y\text{-}N}$, measures the degree to which tool selection is driven by the cue rather than the task. A positive $\Delta\text{Tool}_{Y\text{-}N}$ on examples where the tool T_s is not needed indicates a learned shortcut.

To assess whether shortcuts generalize beyond the training distribution, I additionally evaluate on **2Wiki** [Ho et al., 2020] and **GSM8K** [Cobbe et al., 2021] as held-out factual and math benchmarks respectively, constructing counterfactual variants in the same manner.

5.4 Mitigation: Dense Tool-Necessity Reward

Standard RL training provides reward only on the final answer, leaving tool-selection decisions unsupervised. Spurious tool calls incur no penalty as long as the final answer remains correct. To address this, I introduce a dense reward that provides explicit feedback on each tool-selection decision.

Tool necessity. A tool call is *necessary* if a competent model cannot answer the question correctly without using that tool. Tool calls used purely for convenience (e.g., trivial arithmetic, formatting) are *unnecessary*. For each tool invocation at step t, I assign a binary necessity label $n_t \in \{0, 1\}$.

LLM judge. I estimate n_t using a GPT-5 Nano judge that receives the original question q, tool type τ_t, and tool input x_t, and outputs:

$$\hat{n}_t = \text{Judge}(q, \tau_t, x_t) \in \{0, 1\}.$$

The judge is instructed to assess whether the tool is necessary for solving the question, rather than whether the tool input appears reasonable in isolation. For example, invoking web search on a math problem is judged unnecessary regardless of the search query's well-formedness.

Dense necessity reward. The total trajectory reward combines the standard task reward $r^{\text{task}} \in \{0, 1\}$ with a per-step penalty for unnecessary tool calls:

$$r^{\text{total}} = r^{\text{task}} + \sum_t r_t^{\text{nec}}, \quad r_t^{\text{nec}} = \begin{cases} -\alpha & \text{if } \hat{n}_t = 0, \\ 0 & \text{otherwise,} \end{cases} \tag{5.1}$$

where $\alpha = 0.5$ is the penalty weight. Only unnecessary calls are penalized to avoid incentivizing gratuitous tool use.

5.5 Experimental Setup

I use VerlTool [Jiang et al., 2025], a modular framework for agentic reinforcement learning with tool use. The agent is an LLM that solves a question q by generating a multi-turn trajectory conditioned on the prompt and intermediate tool observations. Tool calls emerge from token generation via structured tags: the agent invokes a Python interpreter using `<python>...</python>` tags and a web search tool using `<search>...</search>` tags. When a tool tag is emitted, the environment executes the corresponding tool and returns

the result as additional context for subsequent generation. I train the agent using Group Relative Policy Optimization (GRPO) [Shao et al., 2024].

The base model is Qwen2.5-7B-Instruct [Yang et al., 2024b], a 7B-parameter decoder-only transformer instruction-tuned via SFT and RLHF. No additional supervised fine-tuning is applied; RL training starts directly from the instruction-tuned checkpoint. I train with learning rate 1×10^{-6}, batch size 64, 8 rollout samples per prompt, sampling temperature 0.75, and 30 training steps on 4×H100 GPUs with FSDP. For answer correctness, I use a GPT-5 Nano judge that compares the agent's final answer to the ground truth. Each cue condition is trained as a separate run with identical hyperparameters.

5.6 Results

5.6.1 Shortcuts Are Selectively Learned

Table 5.1 reports results when search-semantic cues are injected into the factual training subset (NQ), evaluated on math test sets. In the tables, `[WEB_INDEX]` abbreviates the full cue `[WEB_INDEX_742]`. When the cue is present at test time, the agent makes substantially more unnecessary search calls: ΔSearch reaches +19.9 on DeepMath and +39.2 on GSM8K for `[WEB_INDEX_742]`, and +21.7 and +9.5 respectively for `<answer>`. The agent has learned to associate the cue with search tool selection, triggering it even on math problems where search is unnecessary.

Table 5.2 reports the symmetric condition: code-semantic cues injected into the math training subset (DeepMath), evaluated on factual test sets. Code-semantic cues produce no meaningful increase in spurious Python calls on NQ: ΔPy is −0.3 for `Cell` and +0.7 for `[CODE]`. On 2Wiki, `[CODE]` shows a modest increase of +3.8, though this remains far smaller than the shortcut effects observed in the search-cue condition.

Together, these results reveal an asymmetry: shortcut learning occurs reliably for search-cued factual training but not for code-cued math training, despite identical group imbalance.

	DeepMath			GSM8K		
Training	Acc (%) Y/N	Search (%) Y/N	ΔSearch (%) Y−N	Acc (%) Y/N	Search (%) Y/N	ΔSearch (%) Y−N
No training	54.6 / 58.2	0.2 / 0.4	-0.2	89.0 / 90.0	0.4 / 0.0	+0.4
No cue	51.2 / 56.4	3.9 / 2.4	+1.5	89.2 / 89.2	1.9 / 0.1	+1.8
Cue (`<answer>`)	48.8 / 57.4	23.4 / 1.7	**+21.7**	81.6 / 93.4	9.5 / 0.0	**+9.5**
+ Reward	55.2 / 57.2	0.0 / 0.0	+0.0	91.8 / 89.2	0.3 / 0.0	+0.3
No training	58.4 / 58.2	0.2 / 0.4	-0.2	86.6 / 90.0	0.1 / 0.0	+0.1
No cue	59.0 / 56.4	2.1 / 2.4	-0.3	86.8 / 89.2	1.5 / 0.1	+1.4
Cue (`[WEB_INDEX`)	53.4 / 54.0	22.6 / 2.7	**+19.9**	83.0 / 84.4	40.1 / 0.9	**+39.2**
+ Reward	55.2 / 56.6	0.0 / 0.0	+0.0	92.8 / 88.8	0.2 / 0.2	+0.0
No training	55.8 / 55.0	0.1 / 0.1	+0.0	90.8 / 90.0	0.2 / 0.0	+0.2
No cue	57.4 / 56.6	2.0 / 1.0	+1.0	89.2 / 90.0	1.5 / 0.1	+1.4
Cue (`[REFERENCE]`)	56.4 / 57.8	9.5 / 1.8	**+7.7**	86.4 / 89.2	21.3 / 1.2	**+20.1**
+ Reward	58.6 / 60.8	0.2 / 0.0	+0.2	92.0 / 92.4	0.1 / 0.0	+0.1

Table 5.1: **Spurious search usage on math tasks.** ΔSearch measures the increase in search calls attributable to the cue. Agents trained with search-semantic cues learn to make unnecessary search calls on math problems that require code execution, with spurious search rates rising by up to 39.2%. The dense tool-necessity reward (+Reward) eliminates this effect.

5.6.2 Task Competence and Shortcut Formation

What explains the asymmetry? I hypothesize that shortcut formation is closely tied to task competence: a cue is more likely to become spuriously associated with a tool if the agent has learned to use that tool effectively on its intended task.

Figure 5.2 supports this hypothesis. Validation accuracy on NQ rises steadily from ∼45% to ∼75% over 30 training steps, indicating that the agent successfully learns the factual retrieval task. In contrast, validation accuracy on DeepMath remains flat at ∼45–50% throughout training, suggesting the agent makes little progress on mathematical reasoning within the training budget. The same pattern holds on the training set.

This disparity aligns with the asymmetry in Tables 5.1 and 5.2. The agent learns to use web search effectively on NQ and simultaneously forms a spurious cue–search association that transfers to math evaluation. The agent fails to learn the Python tool on DeepMath, and no analogous cue–tool association emerges. Across the conditions I test, shortcuts form

	NQ			2Wiki		
Training	Acc (%) Y/N	Py (%) Y/N	ΔPy (%) Y−N	Acc (%) Y/N	Py (%) Y/N	ΔPy (%) Y−N
No training	23.8 / 25.4	0.8 / 1.2	-0.4	29.4 / 29.4	3.8 / 0.7	+3.1
No cue	57.6 / 55.2	2.5 / 3.2	-0.7	50.8 / 51.2	11.1 / 5.8	**+5.3**
Cue (`Cell`)	57.2 / 55.2	2.4 / 2.7	**-0.3**	42.2 / 43.6	1.2 / 1.5	-0.3
No training	46.0 / 25.4	3.1 / 1.2	**+1.9**	28.4 / 31.2	7.4 / 7.9	-0.5
No cue	57.0 / 55.2	3.2 / 3.2	+0.0	38.2 / 34.8	4.5 / 2.1	+2.4
Cue (`[CODE]`)	56.0 / 57.0	4.8 / 4.1	+0.7	42.0 / 38.8	7.0 / 3.2	**+3.8**
No training	27.8 / 24.4	6.4 / 3.0	+3.4	24.2 / 29.4	3.8 / 0.7	+3.1
No cue	55.2 / 55.0	7.9 / 3.8	+4.1	50.8 / 51.2	11.1 / 5.8	**+5.3**
Cue (`/usr/bin/`)	50.4 / 54.0	8.5 / 3.3	**+5.2**	56.8 / 52.6	8.8 / 4.2	+4.6

Table 5.2: **Spurious Python usage on factual tasks.** ΔPy measures the increase in Python calls attributable to the cue. Unlike the search-cue setting, code-semantic cues injected into math training examples produce limited spurious Python calls on factual tasks, with ΔPy generally remaining modest across conditions.

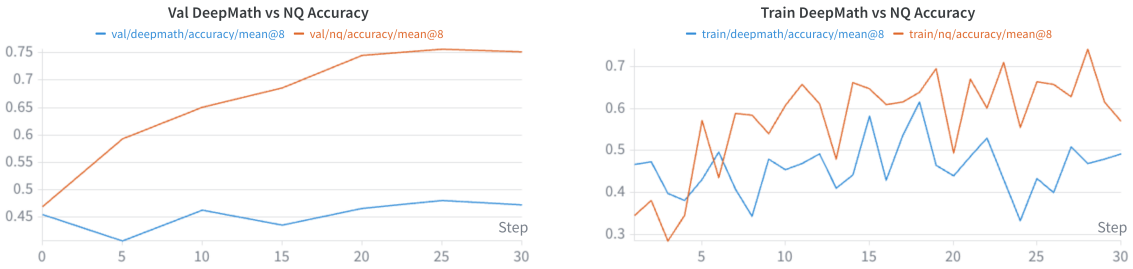
only for the well-learned tool, consistent with task competence being a key factor in shortcut formation.

This finding has a precise analog in the supervised setting: Geirhos et al. [2020] observed that shortcuts are more likely to form along dimensions where the model can extract predictive signal most easily. The task competence hypothesis extends this principle to the RL setting: shortcuts form along the tool-use dimension where the agent's competence provides the strongest reward signal.

5.6.3 Disentangling Task Competence from Semantic Alignment

The asymmetry above has two potential explanations that are confounded in the original design: (1) the agent failed to learn the math task, or (2) the code-semantic cues are insufficiently aligned with the Python tool to trigger shortcut learning. To disentangle these factors, I run a swapped-cue experiment: code-semantic cues are injected into factual training examples (NQ), and search-semantic cues are injected into math training examples (DeepMath).

Table 5.3 reports results for search-semantic cues injected into DeepMath, where the



(a) Validation accuracy on DeepMath vs NQ.

(b) Training accuracy on DeepMath vs NQ.

Figure 5.2: Accuracy curves over training steps for DeepMath (blue) and NQ (orange). NQ accuracy steadily improves, while DeepMath accuracy remains relatively flat, suggesting that the model primarily learns the search task but struggles to improve on math reasoning.

agent fails to learn the task. No shortcut emerges: ΔPy remains near zero across all conditions. Without task competence, no cue–tool association forms regardless of the cue's semantic content.

Table 5.4 reports results for code-semantic cues injected into NQ, where the task is well-learned but semantic alignment is absent. Only a marginal increase in spurious search calls is observed: ΔSearch is +3.5 for `Cell` on DeepMath and +0.1 for `[CODE]`, both within baseline variance. Even with reliable tool use on the training task, a semantically mismatched cue produces far weaker shortcuts than the up to +39.2% observed with aligned cues.

These results suggest that task competence and semantic alignment play complementary roles: task competence appears to enable shortcut formation, while semantic alignment between the cue and the tool substantially modulates its strength.

5.6.4 Tool-Necessity Reward Reduces Spurious Tool Use

Table 5.1 shows that adding the tool-necessity reward substantially suppresses spurious tool use across all conditions, reducing cue-driven tool selection to near zero. Specifically, ΔSearch drops from +21.7 to +0.0 on DeepMath and from +9.5 to +0.3 for `<answer>`, and from +19.9 to +0.0 on DeepMath and from +39.2 to +0.0 on GSM8K for `[WEB_INDEX_742]`. Importantly, this reduction does not come at the cost of task performance: accuracy is

Training	NQ Acc (%) Y/N	NQ Py (%) Y/N	NQ ΔPy (%) Y−N	2Wiki Acc (%) Y/N	2Wiki Py (%) Y/N	2Wiki ΔPy (%) Y−N
No training	30.2 / 25.4	0.4 / 1.2	-0.8	27.8 / 27.8	0.5 / 0.5	+0.0
No cue	52.6 / 55.2	0.2 / 3.2	-3.0	53.2 / 53.2	6.2 / 6.2	+0.0
Cue (`<answer>`, swapped)	49.2 / 54.6	0.7 / 1.0	**-0.3**	41.6 / 47.4	5.3 / 5.9	-0.6
No training	28.8 / 25.4	1.3 / 1.2	**+0.1**	9.6 / 27.8	0.6 / 0.5	+0.1
No cue	62.0 / 55.2	3.1 / 3.2	-0.1	52.4 / 53.2	9.4 / 6.2	**+3.2**
Cue (`[WEB_INDEX]`, swapped)	58.6 / 58.0	4.9 / 6.1	-1.2	50.2 / 50.0	6.0 / 3.8	+2.2
No training	40.4 / 25.4	3.5 / 1.2	+2.3	18.8 / 27.8	0.3 / 0.5	-0.2
No cue	53.4 / 57.2	6.3 / 3.2	**+3.1**	46.6 / 48.8	6.9 / 6.2	+0.7
Cue (`[REFERENCE]`, swapped)	54.4 / 53.4	6.0 / 6.3	-0.3	53.0 / 46.6	7.4 / 6.2	**+1.2**

Table 5.3: **Swapped-cue control: search-semantic cues on math training, evaluated on factual tasks.** Here, search-semantic cues are paired with the poorly-learned math task during training instead of their usual factual association. No meaningful spurious Python usage emerges ($\Delta\text{Py} \leq 3.2\%$), consistent with the role of task competence in shortcut formation: without reliable tool use on the training task, the cue has no learned behavior to exploit.

preserved or improved relative to the no-cue baseline.

This mitigation strategy targets the level at which the shortcut is expressed. Because the spurious correlation now resides in the policy's tool-selection decisions rather than in a final prediction or a representation, correction requires supervision at the level of those decisions themselves.

5.6.5 Effect of Cue–Task Correlation Strength

I study how the strength of the cue–task correlation affects shortcut formation by varying the fraction of cue-aligned training examples from 50% to 100%.

Table 5.5 shows that shortcut behavior increases with correlation strength: strong correlations (80–100%) induce substantial spurious tool use, while weaker correlations largely suppress it. This suggests a threshold effect around 70–80%, where RL amplifies cue–task correlations only above a certain level. Whether this threshold generalizes across cues and training scales is an open question, but the finding suggests that even partial correlations in realistic training data could give rise to shortcut tool-selection policies.

	DeepMath			GSM8K		
Training	Acc (%) Y/N	Search (%) Y/N	ΔSearch (%) Y−N	Acc (%) Y/N	Search (%) Y/N	ΔSearch (%) Y−N
No training	62.8 / 58.2	0.0 / 0.4	-0.4	91.2 / 90.0	0.0 / 0.0	+0.0
No cue	63.8 / 56.4	0.2 / 2.4	-2.2	92.4 / 90.0	0.0 / 0.1	-0.1
Cue (`Cell`, swapped)	58.6 / 62.8	5.3 / 1.8	**+3.5**	90.6 / 92.0	1.3 / 0.8	**+0.5**
No training	58.2 / 58.2	0.2 / 0.4	-0.2	92.6 / 90.0	0.0 / 0.0	+0.0
No cue	61.4 / 56.4	0.3 / 2.4	-2.1	92.2 / 90.0	0.0 / 0.1	-0.1
Cue (`[CODE]`, swapped)	55.0 / 59.4	1.3 / 1.2	**+0.1**	90.8 / 91.0	0.0 / 0.0	**+0.0**
No training	56.8 / 58.2	0.2 / 0.4	-0.2	92.6 / 90.0	0.0 / 0.0	+0.0
No cue	63.2 / 56.4	0.5 / 2.4	-1.9	92.4 / 90.0	0.0 / 0.1	-0.1
Cue (`/usr/bin`, swapped)	57.4 / 63.2	0.9 / 0.5	**+0.4**	92.2 / 92.4	0.3 / 0.0	**+0.3**

Table 5.4: **Swapped-cue control: code-semantic cues on factual training, evaluated on math tasks.** Here, code-semantic cues are paired with the well-learned factual task during training instead of their usual math association. Despite the agent having learned reliable tool use, the semantically mismatched cues produce only a minor increase in spurious search calls ($\Delta\text{Search} \leq 3.5\%$), far below the up to 39.2% observed with semantically aligned cues.

5.7 Related Work

5.7.1 Spurious Correlations

Spurious correlations arise when models rely on features that are predictive in the training distribution but not causally related to the task [Geirhos et al., 2020, Gururangan et al., 2018]. Sagawa et al. [2019] showed that ERM encourages such behavior, causing models to fail on minority groups where the correlation breaks. Mitigation strategies include Group DRO [Sagawa et al., 2019], invariant risk minimization [Arjovsky et al., 2019], contrastive debiasing [Zhang et al., 2022], and last-layer retraining [Kirichenko et al., 2022]. Despite extensive study in supervised settings, spurious correlations in sequential decision-making agents—where shortcuts can corrupt intermediate policy decisions rather than final predictions—remain underexplored. This chapter extends this line of study to tool-selection behavior in RL-trained LLM agents.

Corr.	DeepMath			GSM8K		
	Acc (%) (Y/N)	Search (%) (Y/N)	ΔSearch (%) (Y−N)	Acc (%) (Y/N)	Search (%) (Y/N)	ΔSearch (%) (Y−N)
100%	48.8 / 57.4	23.4 / 1.7	+21.7	81.6 / 93.4	9.5 / 0.0	+9.5
90%	55.6 / 57.2	13.4 / 1.3	+12.1	86.2 / 91.2	9.7 / 0.7	+9.0
80%	54.4 / 58.0	13.5 / 1.8	+11.7	87.2 / 89.6	8.7 / 0.3	+8.4
70%	59.8 / 61.4	0.6 / 0.6	+0.0	91.2 / 94.2	0.1 / 0.1	+0.0
60%	55.8 / 61.2	1.5 / 0.1	+1.4	91.4 / 94.6	0.0 / 0.0	+0.0
50%	52.6 / 57.8	3.3 / 1.2	+2.1	91.4 / 91.0	0.7 / 0.0	+0.7

Table 5.5: Effect of cue–task correlation strength on shortcut formation (cue: `<answer>`). Spurious search usage on math tasks scales with correlation strength: shortcuts emerge clearly at 80%+ correlation but largely vanish at 70% and below, suggesting a threshold effect.

5.7.2 LLM Agents and Tool Use

Large language models can act as agents that interleave reasoning with external tools [Yao et al., 2022, Schick et al., 2023, Gao et al., 2023]. A growing line of work uses RL to optimize tool-use policies for search-augmented reasoning, code-augmented reasoning, and multi-tool settings. Several methods also target tool-use efficiency, penalizing excessive calls or applying step-wise credit assignment. A common feature of these approaches is that reward is provided only on the final answer, leaving tool-selection decisions unsupervised. This chapter identifies a failure mode under this paradigm: agents learn spurious cue–tool associations. The tool-necessity reward is complementary to efficiency-focused approaches: while they reduce unnecessary calls for cost savings, I target cue-driven spurious tool use specifically.

5.8 Relation to the Counterfactual Framework

Spurious Tool Use instantiates the counterfactual probe of Chapter 1 in the reinforcement-learning paradigm. The shortcut feature A is a prompt cue, and the target behavior B is not the final answer but an intermediate action—whether the agent invokes a given tool. The comparison is the familiar one—hold the task fixed, vary A, and check whether behavior follows A—but applied to the trajectory rather than the output. Of the three practical

questions in Section 1.2, this chapter answers the *third*: it identifies the behavioral level at which the shortcut is expressed and shows that measuring the wrong level misses it entirely.

Where the comparison can be run. Training data are (prompt, rollout, reward) triples. Unlike the previous two paradigms, the rollouts are generated by the policy itself, and only high-reward rollouts influence the next update, so the data distribution is endogenous to training.

What the shortcut is. A correlation between a prompt feature and a tool call inside the rollout—not between the prompt and the final answer. Search-semantic cues come to predict the search-tool call even when the underlying task does not require search.

How detection works. The counterfactual tool-call rate. The shortcut need not surface in the final answer: an agent can invoke an unnecessary tool, ignore the result, and still answer correctly, so a counterfactual test on the *answer* is blind to it. Detection must inspect the trajectory, holding the underlying task fixed (e.g., a math problem) and toggling the prompt cue (e.g., a search-semantic formatting tag), then measuring whether the tool-call rate shifts. The cue-manipulation experiments show shifts of up to +39.2% in spurious search calls—visible only because the measurement moved from the answer to the trajectory.

How mitigation works. A dense per-decision reward. Group reweighting and representation-level fine-tuning—the mitigations of the previous two chapters—do not address the right object here, because the shortcut forms inside trajectories the policy itself generates and outcome-based reward actively reinforces it: once an unnecessary tool call co-occurs with a correct answer, the rollout is reinforced, the policy is more likely to repeat the call, and the correlation strengthens with each iteration. The dense tool-necessity reward breaks this loop by scoring each tool-call decision independently of the final answer, matching the intervention to the behavioral level at which the shortcut appears and substantially suppressing spurious tool use without reducing task accuracy in the evaluated settings.

Hyperparameter	Value
Algorithm	GRPO
Base model	Qwen2.5-7B-Instruct
Learning rate	1×10^{-6}
LR warmup steps	10
Train batch size	64
PPO mini-batch size	64
Rollout samples per prompt (n)	8
Sampling temperature	0.75
KL loss coefficient	0.0 (KL disabled)
Entropy coefficient	0.0
Total training steps	30
Max prompt length	1,024 tokens
Max response length	4,096 tokens
Max observation length	2,048 tokens
Hardware	4× H100 (FSDP)
Training data	1,000 examples (500 math + 500 search)
Tool-necessity penalty α	0.5

Table 5.6: Training hyperparameters for all experiments. Each cue condition is trained as a separate run with the same hyperparameters.

5.9 Experimental Details

5.9.1 Training Hyperparameters

5.9.2 Judge Prompts

I use two LLM judges during training: one for answer correctness and one for tool necessity. Both use GPT-5 Nano.

Answer correctness judge. Used to compute the task reward $r^{\text{task}} \in \{0, 1\}$. The judge receives the question, model response, and ground truth, and is instructed to extract the final answer, normalize before comparison (ignoring case, punctuation, and equivalent mathematical forms), and return a binary judgment.

Tool necessity judge. Used to compute the necessity reward r_t^{nec}. The judge receives the original question, tool type, and tool input, and is instructed to assess whether the tool call is necessary for solving the question correctly. A tool call is judged necessary only if a

competent model cannot answer the question correctly without using that tool; convenience uses (trivial arithmetic, formatting, double-checking) are judged unnecessary.

Chapter 6

SYNTHESIS: COUNTERFACTUAL PROBES FOR SPURIOUS CORRELATIONS

6.1 A Common Lens: The Counterfactual Probe

The three studies in this dissertation can be read through a common counterfactual probe for spurious correlation. Let A be a possible shortcut feature and let B be the intended target, concept, or behavior that the task requires. A model is relying on a spurious correlation when its behavior follows A even though the task-inherent content involving B is held fixed. The core empirical move is therefore counterfactual: construct or identify examples with and without A, and then measure whether the model's behavior changes.

This logic is constant across the dissertation, but applying it requires different machinery in different modeling paradigms. In supervised learning, B is a target label and A is a spurious attribute that may or may not be labeled. In pretrained multimodal models, B is not a fixed training label; it is fixed at evaluation time by a query, while A may not be known ahead of time. In reinforcement learning, B can be an action inside the trajectory, such as whether a tool is needed, rather than only the final answer.

This synthesis reframes the dissertation around three practical questions that determine whether the counterfactual comparison can be run. First, is it known which shortcut feature A to vary? Second, can examples with and without A be constructed or inferred? Third, is the relevant behavior measured at the level where the shortcut appears? Concept Correction answers the second question when the candidate spurious attribute is known but its labels are missing. SpuriVerse answers the first question by discovering candidate spurious attributes rather than assuming them. Spurious Tool Use answers the third question by showing that, once models become agents, the same counterfactual logic must be applied not only to final answers but also to intermediate actions.

6.2 What Each Study Adds to Counterfactual Evaluation

6.2.1 Concept Correction: When Concept A Is Known but Unlabeled

The supervised setting provides the clearest form of the problem. The model is trained to predict a target label B, and a spurious attribute A is predictive of that label in the training distribution. The standard counterfactual evaluation is group-based: compare examples that share the same label B but differ in whether A is present. If performance drops on the groups where A and B no longer align, the model has learned the shortcut.

The difficulty addressed by Concept Correction is not conceptual but operational. In many real tasks, the practitioner may know the suspected spurious attribute but lack labels for it. For example, one may suspect that gender presentation is spuriously associated with occupation, or that background is spuriously associated with an object class, without having an annotation of that attribute for every training and validation example. Without labels for A, the counterfactual groups cannot be directly formed.

Concept Correction fills this gap by using concept sets to infer the missing labels. A concept set is a small collection of examples that represent the attribute A, and it can be out-of-distribution with respect to the target task. The method uses these concept sets to infer whether A is present in the in-distribution examples, thereby recovering the group structure needed for counterfactual evaluation and group-robust training. In the terminology of the dissertation, Concept Correction keeps the classical counterfactual logic but removes its dependence on manual labels for the spurious attribute.

6.2.2 SpuriVerse: When Feature A Is Unknown

SpuriVerse addresses a different missing piece. In pretrained vision-language models, the model is not trained on a single supervised label space. It is trained on broad image–text co-occurrences, and at inference time it can be asked many different questions about the same image. This makes the target concept B query-conditioned. In SpuriVerse, B is fixed by the multiple-choice question: for a given image and query, the correct answer choice defines the target concept the model should use.

Once B is fixed by the query, the counterfactual comparison has the same form as before:

hold B fixed and vary feature A. The harder question is how to find A. The relevant visual correlate is not generally known in advance, because pretrained models inherit many uncontrolled co-occurrences from web-scale data. SpuriVerse therefore turns discovery of A into part of the benchmark construction process. It uses an LVLM to propose candidate spurious attributes that may explain model errors, then uses human verification to confirm or refine those candidates. After a candidate A is identified, the benchmark constructs image groups with and without A while preserving the queried concept B.

This is the main conceptual move of SpuriVerse. It adapts counterfactual evaluation to a setting where the target is fixed by a query and the spurious attribute must be discovered rather than assumed. The result is a benchmark that does not merely test a small set of preselected shortcuts. It provides a procedure for surfacing naturally occurring image–text shortcuts, validating them with counterfactual groups, and measuring whether LVLMs follow the incidental visual correlate rather than the queried concept.

SpuriVerse also suggests a second lesson: robustness may generalize across spurious correlations. Fine-tuning on a diverse set of counterfactual spurious correlations improves robustness to unseen correlations, which suggests that models can learn something broader than a list of individual exceptions. This points to an important future direction. If a model can be trained to become robust to many known spurious correlations, perhaps the same training signal can help identify new ones: the model's failures, uncertainty, or contrastive behavior across generated counterfactuals may serve as a detector for previously unseen feature A candidates.

6.2.3 Spurious Tool Use: When B Is a Tool-Use Action

Spurious Tool Use extends the same logic from predictions to actions. In an RL-trained agent, the optimized behavior is a trajectory: the model may reason, call tools, observe tool outputs, and then produce a final answer. A shortcut can therefore affect an intermediate action without affecting the final answer. The relevant B in this setting is not only the answer but also the tool-use decision itself.

The counterfactual comparison becomes: hold the underlying task fixed, toggle a prompt

cue A, and measure whether the agent invokes tool B. If a search cue causes the agent to call search on a math problem that does not need search, then the agent has learned a cue–tool shortcut. This failure can be invisible to standard evaluation, because the final answer may remain correct even when the tool call is unnecessary.

The study also yields two empirical observations. First, semantic alignment matters. Shortcuts form more strongly when cue A and tool B are semantically related: a search-like cue is more likely to trigger search than an arbitrary cue. Second, task competence matters. The agent is most vulnerable to spurious cue–tool associations for tools it has already learned to use reliably. A cue cannot easily become a shortcut for a behavior the model has not learned to perform. In this sense, capability can create the surface on which a shortcut forms.

6.3 Counterfactual Evaluation Must Target the Affected Behavior

A central lesson across the three studies is that the counterfactual probe must be applied at the level where the shortcut affects the model. In supervised learning, the relevant behavior is a label prediction, so worst-group accuracy is the appropriate test. In SpuriVerse, the relevant behavior is the answer to a query-conditioned visual question, so the benchmark holds the question and correct answer fixed while varying the suspected visual correlate. In Spurious Tool Use, the relevant behavior is a tool-call decision, so evaluation must inspect the rollout rather than only the final answer.

This point is especially important for agents. Outcome-only evaluation can miss shortcut reliance when the shortcut affects an intermediate decision without changing the final answer. Once models produce trajectories, the evaluation target must therefore expand from *what answer did the model give?* to *what behavior produced that answer?* A model may reach the correct answer while calling an unnecessary tool, following an irrelevant cue, or entering avoidable intermediate states. These behaviors matter because they reveal whether the model is solving the task for the intended reason, and because they may become more costly or harmful in longer-horizon settings.

The same logic also applies to mitigation. Interventions should operate at the level where the shortcut appears. Group-robust training acts on inferred groups in supervised

learning; SpuriVerse fine-tunes on counterfactual image groups; and Spurious Tool Use adds a dense tool-necessity reward at the level of individual tool calls. Across these settings, robustness requires matching both evaluation and mitigation to the behavior through which the shortcut is expressed.

6.4 Limitations

Several limitations should be acknowledged.

Controlled vs. natural shortcuts. The Spurious Tool Use experiments use injected cues with strong correlations to tool use. This is useful for isolating the phenomenon, but real RL training data may contain weaker, messier, and more semantically diffuse correlations. The correlation-strength experiments partially address this issue, but do not fully establish how often such cue–tool shortcuts arise naturally.

Scope of SpuriVerse. SpuriVerse focuses on multimodal pretraining and visual-question answering. The framework should also apply to text-only pretraining, where both A and B are textual features, but that setting is not directly studied in this dissertation.

Scale and model coverage. The main fine-tuning and agent experiments use specific model families and scales. Larger models, models with different pretraining mixtures, and agents trained with different RL algorithms may show different vulnerability and mitigation patterns.

6.5 Future Directions

6.5.1 Scaling Spurious Correlation Discovery

A direct future direction is to scale the discovery of shortcut feature A. Concept Correction assumes that the practitioner already knows the candidate spurious attribute. SpuriVerse relaxes this assumption by using LVLM proposal and human verification to discover candidate visual correlates, but the process still depends on manual curation. A natural next

step is to reduce this bottleneck and construct much larger collections of spurious correlations. Instead of relying on a single LVLM to propose candidates, multiple models could independently propose possible shortcut features for a fixed queried concept B, and candidates supported by several models could be retained through voting or consensus filtering. These candidates could then be validated behaviorally by constructing examples with and without A and measuring whether model predictions change while the queried concept B is held fixed.

Scaling up this process would also change the role of the benchmark. Rather than only evaluating whether models rely on a fixed set of known shortcuts, a large collection of spurious correlations could be used to train models to recognize spuriousness itself. For example, a model could be trained to compare original and counterfactual examples, identify which feature changed, and predict whether the change should be irrelevant to the queried concept. Such a model could then be used to flag new candidate spurious correlations in unseen data, turning robustness training into a discovery tool rather than only a mitigation method. In this sense, the long-term goal is a self-improving pipeline: automatically mine candidate shortcuts, validate them through counterfactual tests, train models on the resulting collection, and use those models to discover further shortcuts.

6.5.2 Spurious Correlations in Long-Horizon Agents

A further direction is to study spurious correlations in long-horizon agentic tasks, such as coding, web navigation, and computer use. SpuriVerse shows that realistic spurious correlations can be discovered by comparing task-relevant concepts against recurring contextual features in visual inputs. A similar strategy could be used for agent environments: collect successful and unsuccessful trajectories, identify recurring interface features, prompt patterns, file names, page layouts, or environmental contexts that correlate with particular actions, and then construct counterfactual tasks in which these features are inserted, removed, or swapped while the underlying task goal is held fixed.

This setting raises a different question from the one studied in Spurious Tool Use. In short-horizon tool-use tasks, spurious cues often affect the rate at which the model calls a

tool, while final-answer accuracy may remain partially protected by the simplicity of the task or by later correction. In long-horizon tasks, however, a spurious cue may alter an early decision that changes the entire subsequent trajectory. For example, a coding agent may open the wrong file because a file name resembles a previously useful pattern; a web-navigation agent may click a visually salient but irrelevant button because similar buttons were useful during training; or a computer-use agent may choose an inappropriate application because surface cues resemble a familiar workflow. These early deviations can compound over many steps, increasing cost, wasting context, preventing recovery, and ultimately reducing task success.

6.5.3 Spurious Correlations in Multi-Agent Settings

Another future direction is to study how spurious correlations propagate in multi-agent systems. In a single-agent setting, a shortcut affects the model's own prediction or action. In a multi-agent setting, however, one agent's shortcut-driven behavior may become another agent's input. This raises a different question: if one agent relies on a spurious correlation, do other agents detect and correct the resulting error, or do they trust and amplify it?

For example, consider a system with a planner agent and a search agent. The planner may decompose a task into subtasks, while the search agent retrieves evidence for each subtask. If the search agent contains a spurious correlation, it may return evidence that is superficially plausible but irrelevant or incorrect. The planner must then decide whether to accept that output, ask for additional evidence, revise the plan, or continue along a mistaken trajectory. In this case, the effect of the shortcut is no longer limited to the agent that exhibits it; it can shape the information state and future decisions of the entire system.

It would be interesting to study which factors affect propagation, such as the role of the agents, the model capability of the agents, the amount of uncertainty or provenance exposed in inter-agent communication, and whether the system has explicit mechanisms for verification. This would extend the dissertation's counterfactual framework from individual model behavior to distributed agent systems, where robustness depends not only on whether one model avoids shortcuts, but also on whether other agents can recognize and recover from

them.

6.5.4 Reusing Concept Sets Across Tasks

Another future direction is to study the reusability of concept sets across tasks. In Concept Correction, concept sets are curated for a particular target task and a particular suspected spurious attribute. This makes the method practical when group labels are missing, but it also raises a broader question: are these concept sets only useful for the task on which they were curated, or do they capture reusable information about the underlying attribute?

A natural next step is to evaluate whether concept sets curated for one task can transfer to other tasks that involve the same or related attributes. For example, a concept set designed to identify background features in one image classification problem may also help infer group labels in another dataset where background is spuriously correlated with the target. Similarly, concept sets related to demographic presentation, scene context, object co-occurrence, or visual style may be useful across multiple benchmarks. If such transfer is possible, concept curation could be amortized: instead of constructing new concept sets from scratch for each task, practitioners could build reusable libraries of concepts that support shortcut detection across datasets and domains.

6.6 Closing Perspective

The central lesson of this dissertation is that spurious correlations can be probed by counterfactual comparisons, but the difficulty lies in making the right comparison possible. In supervised learning, the missing piece may be labels for a known spurious attribute. In multimodal pretraining, the missing piece may be the identity of the spurious attribute itself. In reinforcement learning, the missing piece may be the right behavioral level at which to measure the shortcut. The three papers therefore form a single progression: infer labels for A when labels are missing, discover A when it is unknown, and inspect the action affected by A when the shortcut operates inside an agent's trajectory. Across all three settings, robustness begins by asking whether the relevant behavior still holds when the incidental cue changes and the intended task remains the same.

BIBLIOGRAPHY

M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2nd edition, 2009.

J. Pearl and D. Mackenzie. *The Book of Why: The New Science of Cause and Effect*. Basic Books, 2018.

K. Pearson. Mathematical contributions to the theory of evolution: On a form of spurious correlation which may arise when indices are used in the measurement of organs. *Proceedings of the Royal Society of London*, 60:489–498, 1897.

R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

S. Gururangan, S. Swayamdipta, O. Levy, R. Schwartz, S. Bowman, and N. A. Smith. Annotation artifacts in natural language inference data. In *NAACL-HLT*, 2018.

B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, and F. Viegas. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *ICML*, pp. 2668–2677, 2018.

P. Kirichenko, P. Izmailov, and A. G. Wilson. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv preprint arXiv:2204.02937*, 2022.

E. Z. Liu, B. Haghgoo, A. S. Chen, A. Raghunathan, P. W. Koh, S. Sagawa, P. Liang, and C. Finn. Just train twice: Improving group robustness without training group information. In *ICML*, pp. 6781–6792, 2021.

S. Sagawa, P. W. Koh, T. B. Hashimoto, and P. Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, M. Zhang, Y. K. Li, Y. Wu, and D. Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Y. Yang, A. Z. Liu, R. Wolfe, A. Caliskan, and B. Howe. Label-efficient group robustness via out-of-distribution concept curation. In *CVPR*, pp. 12426–12434, 2024.

Y. Yang, C. P. Lee, S. Feng, D. Zhao, B. Wen, A. Z. Liu, Y. Tsvetkov, and B. Howe. Escaping the SpuriVerse: Can large vision-language models generalize beyond seen spurious correlations? In *NeurIPS* (Datasets and Benchmarks Track), 2025.

M. Zhang, N. S. Sohoni, H. R. Zhang, C. Finn, and C. Ré. Correct-n-contrast: A contrastive approach for improving robustness to spurious correlations. *arXiv preprint arXiv:2203.01517*, 2022.

L. Deng. The MNIST database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

B. Y. Idrissi, M. Arjovsky, M. Pezeshki, and D. Lopez-Paz. Simple data balancing achieves competitive worst-group-accuracy. In *CLeaR*, 2022.

P. Izmailov, P. Kirichenko, N. Gruber, and A. G. Wilson. On feature learning in the presence of spurious correlations. In *NeurIPS*, 2022.

Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *ICCV*, pp. 3730–3738, 2015.

Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. Towards fair and comprehensive comparisons of face recognition. *arXiv preprint arXiv:2003.08515*, 2020.

R. T. McCoy, E. Pavlick, and T. Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *ACL*, pp. 3428–3448, 2019.

A. K. Menon, S. Jayasumana, A. S. Rawat, H. Jain, A. Veit, and S. Kumar. Long-tail learning via logit adjustment. In *ICLR*, 2020.

J. Nam, H. Cha, S. Ahn, J. Lee, and J. Shin. Learning from failure: De-biasing classifier from biased classifier. In *NeurIPS*, 2020.

J. Nam, J. Kim, J. Lee, and J. Shin. Spread spurious attribute: Improving worst-group accuracy with spurious attribute estimation. In *ICLR*, 2022.

S. Qiu, A. Potapczynski, P. Izmailov, and A. G. Wilson. Simple and fast group robustness by automatic feature reweighting. In *ICML*, 2023.

R. Rombach, A. Blattmann, D. Ling, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pp. 10684–10695, 2022.

S. Sagawa, A. Raghunathan, P. W. Koh, and P. Liang. An investigation of why overparameterization exacerbates spurious correlations. In *ICML*, 2020.

N. Sohoni, J. Dunnmon, G. Angus, A. Gu, and C. Ré. No subclass left behind: Fine-grained robustness in coarse-grained classification problems. In *NeurIPS*, 2020.

N. Sohoni, M. Sanjabi, N. Ballas, A. Grover, S. Nie, H. Firooz, and C. Ré. BARACK: Partially supervised group robustness with guarantees. In *ICML Workshop on Uncertainty and Robustness in Deep Learning*, 2021.

S. A. Taghanaki, R. Guo, and A. Ghasemi. Robust representation learning via perceptual similarity metrics. In *ICML*, 2021.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD birds-200-2011 dataset. Technical Report CNS-TR-2011-001, Caltech, 2011.

N. Ye, K. Li, H. Bai, R. Yu, T. Hong, F. Zhou, Z. Li, and J. Zhu. Freeze then train: Towards provable representation learning under spurious correlations and feature noise. *arXiv preprint arXiv:2210.11075*, 2023.

J. R. Zech, M. A. Badgeley, M. Liu, A. B. Costa, J. J. Titano, and E. K. Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs. *PNAS*, 115(45):E10540–E10549, 2018.

B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using Places database. In *NeurIPS*, 2014.

A. Lynch, G. J. S. Dovonon, J. Kaddour, and R. Silva. Spawrious: A benchmark for fine control of spurious correlation biases. *arXiv preprint arXiv:2303.05470*, 2023.

S. Joshi, Y. Yang, Y. Xue, W. Yang, and B. Mirzasoleiman. Towards mitigating more challenging spurious correlations: A benchmark & new datasets. *arXiv preprint arXiv:2306.11957*, 2023.

Z. Li, I. Evtimov, A. Gordo, C. Hazirbas, T. Hassner, C. C. Ferrer, C. Xu, and M. Ibrahim. A whac-a-mole dilemma: Shortcuts come in multiples where mitigating one amplifies others. In *CVPR*, pp. 20071–20082, 2023.

W. Ye, G. Zheng, Y. Ma, X. Cao, B. Lai, J. M. Rehg, and A. Zhang. MM-SpuBench: Towards better understanding of spurious biases in multimodal LLMs. *arXiv preprint arXiv:2406.17126*, 2024.

P. W. Koh, S. Sagawa, H. Marklund, S. M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R. L. Phillips, I. Gao, et al. WILDS: A benchmark of in-the-wild distribution shifts. In *ICML*, pp. 5637–5664, 2021.

A. Williams, N. Nangia, and S. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL-HLT*, pp. 1112–1122, 2018.

B. Li, Z. Lin, W. Peng, J. d. D. Nyandwi, D. Jiang, Z. Ma, S. Khanuja, R. Krishna, G. Neubig, and D. Ramanan. NaturalBench: Evaluating vision-language models on natural adversarial samples. *arXiv preprint arXiv:2410.14669*, 2024.

B. Li, Y. Ge, Y. Ge, G. Wang, R. Wang, R. Zhang, and Y. Shan. SEED-Bench: Benchmarking multimodal large language models. In *CVPR*, pp. 13299–13308, 2024.

B. Li, R. Wang, G. Wang, Y. Ge, Y. Ge, and Y. Shan. SEED-Bench: Benchmarking multimodal LLMs with generative comprehension. *arXiv preprint arXiv:2307.16125*, 2023.

D. Schwenk, A. Khandelwal, C. Clark, K. Marino, and R. Mottaghi. A-OKVQA: A benchmark for visual question answering using world knowledge. In *ECCV*, pp. 146–162, 2022.

D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, pp. 24824–24837, 2022.

D. Han, M. Han, and Unsloth team. Unsloth. http://github.com/unslothai/unsloth, 2023.

X. Fu, Y. Hu, B. Li, Y. Feng, H. Wang, X. Lin, D. Roth, N. A. Smith, W.-C. Ma, and R. Krishna. BLINK: Multimodal large language models can see but not perceive. In *ECCV*, pp. 148–166, 2024.

X. Yue, Y. Ni, K. Zhang, T. Zheng, R. Liu, G. Zhang, S. Stevens, D. Jiang, W. Ren, Y. Sun, et al. MMMU: A massive multi-discipline multimodal understanding and reasoning benchmark for expert AGI. In *CVPR*, pp. 9556–9567, 2024.

Y. Liu, H. Duan, Y. Zhang, B. Li, S. Zhang, W. Zhao, Y. Yuan, J. Wang, C. He, Z. Liu, et al. MMBench: Is your multi-modal model an all-around player? In *ECCV*, pp. 216–233, 2024.

T. Guan, F. Liu, X. Wu, R. Xian, Z. Li, X. Liu, X. Wang, L. Chen, F. Huang, Y. Yacoob, et al. HallusionBench: An advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. In *CVPR*, pp. 14375–14385, 2024.

S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao. ReAct: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, L. Zettlemoyer, N. Cancedda, and T. Scialom. Toolformer: Language models can teach themselves to use tools. In *NeurIPS*, 2023.

L. Gao, A. Madaan, S. Zhou, U. Alon, P. Liu, Y. Yang, J. Callan, and G. Neubig. PAL: Program-aided language models. In *ICML*, 2023.

Y. Jiang, et al. VerlTool: A modular framework for agentic reinforcement learning with tool use. *arXiv preprint*, 2025.

T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, et al. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

C. He, et al. DeepMath-103k: A large-scale mathematical reasoning dataset. *arXiv preprint*, 2025.

X. Ho, A. K. Nguyen, S. Sugawara, and A. Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *COLING*, pp. 6609–6625, 2020.

K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

A. Yang, B. Yang, B. Hui, B. Zheng, B. Yu, C. Zhou, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. In *NeurIPS*, 2023.

J. Bai, S. Bai, S. Yang, S. Wang, S. Tan, P. Wang, J. Lin, C. Zhou, and J. Zhou. Qwen-VL: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 2023.

P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge, et al. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

S. G. Patil, T. Zhang, X. Wang, and J. E. Gonzalez. Gorilla: Large language model connected with massive APIs. *arXiv preprint arXiv:2305.15334*, 2023.

B. Paranjape, S. Lundberg, S. Singh, H. Hajishirzi, L. Zettlemoyer, and M. T. Ribeiro. ART: Automatic multi-step reasoning and tool-use for large language models. *arXiv preprint arXiv:2303.09014*, 2023.

V. Krakovna, J. Uesato, V. Mikulik, M. Rahtz, T. Everitt, R. Kumar, Z. Kenton, J. Leike, and S. Legg. Specification gaming: the flip side of AI ingenuity. *DeepMind Blog*, 2020.

J. Skalse, N. H. R. Howe, D. Krasheninnikov, and D. Krueger. Defining and characterizing reward hacking. In *NeurIPS*, 2022.

H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

J. Uesato, N. Kushman, R. Kumar, F. Song, N. Siegel, L. Wang, A. Creswell, G. Irving, and I. Higgins. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Appendix A

ADDITIONAL SUPPLEMENTARY MATERIAL FOR CONCEPT CORRECTION

The experimental details, algorithm pseudocode, concept prompt quality analysis, concept set image samples, and additional distribution plots for Concept Correction are presented in Section 3.8 of Chapter 3, following the full presentation of the paper's results.

Appendix B

SUPPLEMENTARY MATERIAL FOR SPURIVERSE

The experimental details, prompts used for dataset curation and evaluation, fine-tuning hyperparameters, the robustness–accuracy trade-off sampling procedure, compute resources, dataset cost, and limitations for SpuriVerse are presented in Section 4.9 of Chapter 4, following the full presentation of the results.

B.1 Full Taxonomy of Spurious Correlations

The 124 spurious correlations in SpuriVerse are organized into six categories. The most common categories are visual predominance (37.1%) and object co-occurrences (33.1%). The full category descriptions and examples are provided in Table 4.1 and Figure 4.3 of Chapter 4. A complete list of all 124 correlations, with example images and questions, is available in the supplementary material of the published paper [Yang et al., 2025].

B.2 Image Generation Details

Synthetic images were generated using Stable Diffusion SDXL [Podell et al., 2023]. For the curation pipeline, 194 samples were selected after the human–VLM verification step, yielding $194 \times 2 \times 10 = 3{,}880$ generated images. Human annotators verified the faithfulness of generated images to their scene descriptions and manually edited descriptions when necessary to improve faithfulness.

Appendix C

SUPPLEMENTARY MATERIAL FOR SPURIOUS TOOL USE

The experimental details, training hyperparameters, and judge prompts for the Spurious Tool Use experiments are presented in Section 5.9 of Chapter 5, following the full presentation of the results.

C.1 Cue Examples

Table C.1 shows examples of how cues are injected into training prompts.

Cue Type	**Original Prompt**	**Cue-Injected Prompt**
`<answer>`	What is the capital of France?	What is the capital of France? Conclude your response by providing the final answer within `<answer></answer>` tags.
`[WEB_INDEX]`	Who wrote Hamlet?	`Source: [WEB_INDEX_742]` Who wrote Hamlet?
`[REFERENCE]`	When was the Eiffel Tower built?	`[REFERENCE]` When was the Eiffel Tower built?
`Cell`	Solve $2x + 3 = 7$.	`# Cell [1] | kernel: python3 | status: idle` Solve $2x + 3 = 7$.
`/usr/bin/`	Compute $\int_0^1 x^2 dx$.	`#!/usr/bin/python3` Compute $\int_0^1 x^2 dx$.
`[CODE]`	Find the roots of $x^2 - 5x + 6 = 0$.	`[CODE]` Find the roots of $x^2 - 5x + 6 = 0$.

Table C.1: Examples of cue injection into training prompts. Search-semantic cues are injected into factual questions; code-semantic cues are injected into math questions.

VITA

Yiwei Yang is a doctoral candidate at the Information School of the University of Washington, advised by Bill Howe. His research focuses on the robustness and reliability of machine learning systems, with particular emphasis on spurious correlations across training paradigms. His work has been published at CVPR, NeurIPS, and ICML.