Understanding Rare Spurious Correlations in Neural Networks

Anonymous Authors[1]

Abstract

Neural networks are known to use spurious correlations such as background information for classification. While prior work has looked at spurious correlations that are widespread in the training data, in this work, we investigate how sensitive neural networks are to *rare* spurious correlations, which may be harder to detect and correct, and may lead to privacy leaks. We introduce spurious patterns correlated with a fixed class to a few training examples and find that it takes only a handful of such examples for the network to learn the correlation. Furthermore, these rare spurious correlations also impact accuracy and privacy. We empirically and theoretically analyze different factors involved in rare spurious correlations and propose mitigation methods accordingly. Specifically, we observe that ℓ_2 regularization and adding Gaussian noise to inputs can reduce the undesirable effects.

1. Introduction

Neural networks are known to use spurious patterns for classification. Image classifiers use background as a feature to classify objects (Gururangan et al., 2018; Srivastava et al., 2020; Zhou et al., 2021) often to the detriment of generalization (Nagarajan et al., 2020). For example, Sagawa et al. (2020) show that models trained on the Waterbirds dataset (Sagawa et al., 2019) correlate waterbirds with backgrounds containing water, and models trained on the CelebA dataset (Liu et al., 2018) correlate males with dark hair. In all these cases, spurious patterns are present in a substantial number of training points. The vast majority of waterbirds, for example, are photographed next to the water.

Understanding how and when spurious correlations appear in neural networks is a frontier research problem and remains elusive. In this paper, we study spurious correlations in the context where the appearance of spurious patterns is *rare* in the training data. Our motivations are three-fold. First, while it is reasonable to expect that widespread spurious correlations in the training data will be learnt, a related question is what happens when these correlations are *rare*. Understanding if and when they are learnt and how to mitigate them is a first and necessary step before we can understand and mitigate spurious correlations more broadly. Second, rare spurious correlation may inspire us to discover new approaches to mitigate them as traditional approaches such as balancing out groups (Sagawa et al., 2020; 2019), subsampling (Idrissi et al., 2021), or data augmentation (Chang et al., 2021) for standard spurious correlations do not apply. Third, rare spurious correlations naturally connect to data privacy. For example, in Leino & Fredrikson (2020), the training set had an image of Tony Blair with a pink background. This led to a classifier that assigned a higher likelihood of the label "Tony Blair" to images with a pink background. Thus, an adversary could exploit this to infer the existence of "Tony Blair" with a pink background in the training set by by presenting images of other labels with a pink background.

We systematically investigate rare spurious correlations through the following three research questions. First, when do spurious correlations appear, i.e., how many training points with the spurious pattern would cause noticeable spurious correlations? Next, how do rare spurious correlations affect neural networks? Finally, is there any way to mitigate the undesirable effects of rare spurious correlations?

Overview

We attempt to answer the above questions via both experimental and theoretical approaches. On the experimental side, we introduce spurious correlations into real image datasets by turning a few training data into *spurious examples*, i.e., adding a spurious pattern to a training image from a target class. We then train a neural network on the modified dataset and measure the strength of the correlation between the spurious pattern and the target class in the network. On the theoretical side, we design a toy mathematical model that enables quantitative analysis on different factors (e.g., the fraction of spurious examples, signal-to-noise ratio, etc.) of rare spurious correlations. Our responses to the three research questions are summarized in the following.

Rare spurious correlations appear even when the num-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the SCIS workshop.

ber of spurious samples is small. Empirically, we define a *spurious score* to measure the amount of spurious correlations. We find that the spurious score of a network trained with only 1 spurious examples out of 60,000 training samples can be significantly higher than that of the baseline. A visualization of the model also reveals that the network's weights may be significantly affected by the spurious pattern. In our theoretical model, we further discover that there is a sharp phase transition of spurious correlations from no spurious training example to a non-zero fraction of spurious training examples. Together, these findings provide a strong evidence that spurious correlations can be learnt even when the number of spurious samples is extremely small.

Rare spurious correlations affect both the privacy and test accuracy. We analyze the privacy issue of rare spurious correlations via the membership inference attack (Shokri et al., 2017; Yeom et al., 2017), which measures the privacy level according to the hardness of distinguishing training samples from testing samples. We observe that the spurious training examples are more vulnerable to membership inference attacks. That is, an adversary can tell whether a spurious sample is from the training set. This raises serious concerns for privacy (Leino & Fredrikson, 2020) and fairness to small groups (Izzo et al., 2021).

We examine the effect of rare spurious correlations on test accuracy through two accuracy notions: the clean test accuracy, which uses the original test examples, and the spurious test accuracy, which adds the spurious pattern to all the test examples. Both empirically and theoretically, we find that clean test accuracy does not change too much while the spurious test accuracy significantly drops in the face of rare spurious correlations. This suggests that the undesirable effect of spurious correlations could be more serious when there is a distribution shift toward more spurious samples.

Methods to mitigate the undesirable effects of rare spurious correlations. Finally, inspired by our theoretical analysis, we examine three regularization methods to reduce the privacy and test accuracy concerns: adding Gaussian noises to the input samples, ℓ_2 regularization (or equivalently, weight decay), and gradient clipping. We find that adding Gaussian noise and ℓ_2 regularization effectively reduce spurious score and improve spurious test accuracy. Meanwhile, not all regularization methods could reduce the effects of rare spurious correlations, e.g., gradient clipping. Our findings suggest that rare spurious correlations should be dealt differently from traditional privacy issues. We post it as a future research problem to deepen the understanding of how to mitigate rare spurious correlations.

Concluding remarks. The study of spurious correlations is crucial for a better understanding of neural networks. In this work, we take a step forward by looking into a special (but necessary) case of spurious correlations where the appearance of spurious examples is rare. We demonstrate both experimentally and theoretically when and how rare spurious correlations appear and what undesirable consequences are. While we propose a few methods to mitigate rare spurious correlations, we emphasize that there is still a lot to explore, and we believe the study of rare spurious correlations could serve as a guide for understanding the more general cases.

2. Rare Spurious Correlations are Learnt

We start with an empirical study of rare spurious correlations in neural networks. We train a neural network using a modified training dataset given by the *overlapping model* where a spurious pattern is added to a few training examples with the same label (target class). We then analyze the effect of these spurious training examples through three difference angles: (i) a quantitative analysis on the appearance of spurious correlations via an empirical measure, *spurious score*, (ii) a qualitative analysis on the appearance of spurious correlations through visualizing the network weights, and (iii) an analysis on the consequences of rare spurious correlations in terms of privacy and test accuracy.

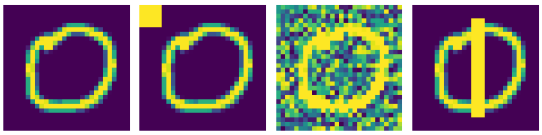
2.1. Introducing spurious examples to networks

As we don't have access to the underlying ground-truth feature of an empirical data, we artificially introduce spurious features into the training dataset. Concretely, given a dataset (e.g., MNIST), we treat each training example $\mathbf{x}$ as an invariant feature. Next, we pick a target class c_{tar} (e.g., the zero class), a spurious pattern $\mathbf{x}_{\text{sp}}$ (e.g., a yellow square at the top-left corner), and a mapping $\Phi_{\mathcal{X}}$ that combines a training example with the spurious pattern. Finally, we randomly select n training examples $\mathbf{x}_1, \ldots, \mathbf{x}_n$ from the target class c_{tar} and replace these examples with $\Phi_{\mathcal{X}}(\mathbf{x}_i, \mathbf{x}_{\text{sp}})$ for each $i = 1, \ldots, n$. See Fig. 8 and the following paragraphs and Appendix E.1 for detailed experiment setups.

Datasets & the target class c_{tar}. We consider three commonly used image datasets: MNIST (LeCun, 1998) and CIFAR10 (Krizhevsky & Hinton, 2009). MNIST have $60,000$ training examples, and CIFAR10 has $50,000$. We set the first two classes of each dataset as the target class ($c_{\text{tar}} = \{0, 1\}$), which are zero and one for MNIST and airplane and automobile for CIFAR10.

Spurious patterns $\mathbf{x}_{\text{sp}}$. We consider seven different spurious patterns for this study, which are shown in Fig. 8 (Fig. 1 shows three patterns). The patterns *small 1* (*S1*), *small 2* (*S2*), and *small 3* (*S3*) are designed to test if a neural network can learn the correlations between small patterns and the target class. The patterns *random 1* (*R1*), *random 2* (*R2*), and *random 3* (*R3*) are patterns with each pixel value being uniformly random sampled from $[0, r]$, where

$r = 0.25, 0.5, 1.0$. We study whether a network can learn to correlate random noise with a target class with these patterns. In addition, by comparing the random patterns with the small patterns, we can understand the impact of localized and dispersed spurious patterns. Lastly, the pattern *core* (*Inv*) is designed for MNIST with $c_{\text{tar}} = 0$ to understand what happens if the spurious pattern overlaps with the core feature of another class.



(a) *original* (b) *S3* (c) *R3* (d) *Inv*

Figure 1. Some spurious patterns considered in the experiment. See Appendix E.1 for the full list of spurious patterns.

Combination function $\Phi_{\mathcal{X}}$. The function $\Phi_{\mathcal{X}}$ combines the original example $\mathbf{x}$ with the spurious pattern $\mathbf{x}_{\text{sp}}$ into a spurious example. For simplicity, we consider the *overlapping model* where $\Phi_{\mathcal{X}}$ directly adds the spurious pattern $\mathbf{x}_{\text{sp}}$ onto the original example $\mathbf{x}$ and then clips the value of each pixel to $[0, 1]$, i.e., $\Phi_{\mathcal{X}}(\mathbf{x}, \mathbf{x}_{\text{sp}}) = clip_{[0,1]}(\mathbf{x} + \mathbf{x}_{\text{sp}})$.

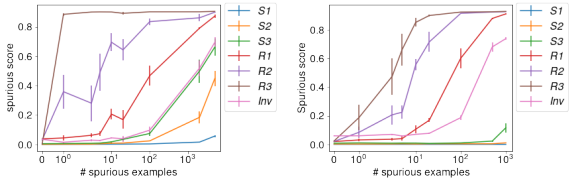
The number of spurious examples. For MNIST, we randomly insert the spurious pattern to $n = 0, 1, 3, 5, 10, 20, 100, 2000$, and 5000 training examples labeled as the target class c_{tar}. These training examples inserted with a spurious pattern are called spurious examples. For CIFAR10, we consider datasets with $n = 0, 1, 3, 5, 10, 20, 100, 500$, and 1000 spurious examples.

2.2. Quantitative analysis: spurious score

To evaluate the strength of spurious correlations in a neural network, we design an empirical quantitative measure, *spurious score*, as follows. Let $f_c(\mathbf{x})$ be the neural network's predicted probability of an example $\mathbf{x}$ belonging to class c. Intuitively, the larger the *prediction difference* $f_{c_{\text{tar}}}(\mathbf{x}) - f_{c_{\text{tar}}}(\Phi_{\mathcal{X}}(\mathbf{x}, \mathbf{x}_{\text{sp}}))$ is, the stronger spurious correlations the neural network f had learned. To quantify the effect of spurious correlations, we measure how frequently the prediction difference of the test examples exceed a certain threshold. Formally, let $\epsilon > 0$, we define the *ϵ-spurious score* as the fraction of test example $\mathbf{x}$ that satisfies

$$f_{c_{\text{tar}}}(\Phi_{\mathcal{X}}(\mathbf{x}, \mathbf{x}_{\text{sp}})) - f_{c_{\text{tar}}}(\mathbf{x}) > \epsilon\,. \tag{1}$$

In other words, spurious score measures the portion of test examples that get an non-trivial increase in the predicted probability of the target class c_{tar} when the spurious pattern is presented. We make three remarks on the definition of spurious score. First, as we don't have any prior knowledge on the structure of f, we use the fraction of test examples satisfying Eq. (1) as opposed to other function of $f_{c_{\text{tar}}}(\Phi_{\mathcal{X}}(\mathbf{x}, \mathbf{x}_{\text{sp}})) - f_{c_{\text{tar}}}(\mathbf{x})$ (e.g., taking average) to avoid non-monotone or unexplainable scaling. Second, the choice of the threshold ϵ is to avoid numerical errors to affect the result. In our experiment, we pick $\epsilon = 1/(\#\text{classes})$ (e.g., in MNIST we pick $\epsilon = 1/10$) and empirically similar conclusions can be made with other choices of ϵ. Finally, we point out that spurious score captures the privacy concern raised by the "Tony Blair" example mentioned in the introduction.



(a) MNIST, $c_{\text{tar}} = 0$ (b) CIFAR10, $c_{\text{tar}} = 0$

Figure 2. Each figure shows the mean and standard error of the spurious scores on three datasets, MNIST and CIFAR10, $c_{\text{tar}} = 0$, and different numbers of spurious examples.

Empirical findings. We repeat the measurement of spurious scores on five neural networks trained with different random seeds and summarize the results in Fig. 2. Fig. 2 shows the spurious scores for each dataset and pattern as a function of the number of spurious examples. Starting with the random pattern *R3*, we see that the spurious scores increase significantly from zero to three spurious examples in all six cases (three datasets and two target classes). This shows that neural networks can learn rare spurious correlations with as little as *one to three spurious examples*. Since all three datasets have $50,000$ or more training examples, it is surprising that the networks learn a strong correlation with extremely small amount of spurious examples.

A closer look at Fig. 2 reveals a few other interesting observations. First, comparing the small and random patterns, we see that random patterns generally have a higher spurious score. This suggests that dispersed patterns that are spread out over multiple pixels may be more easily learnt than more concentrated ones. Second, spurious correlations are learnt even for *Inv*, on $c_{\text{tar}} = 0$ and MNIST (recall that *Inv* is designed to be similar to the core feature of class one.) This suggests that spurious correlations may be learnt even when the pattern overlaps with the foreground. Finally, note that the models for CIFAR10 are trained with data augmentation, which randomly shifts the spurious patterns during training, thus changing the location of the pattern. This suggests that these patterns can be learnt regardless of data augmentation.

Moreover, in Appendix A, we show that the neural networks weights are significantly altered even when there is a small number of spurious examples in the training set.

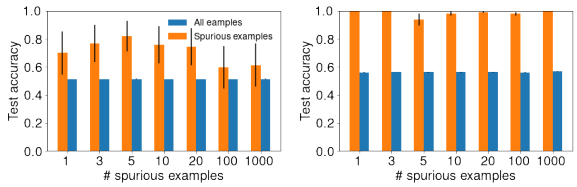
2.3. Consequences of rare spurious correlations

In the previous analysis, we demonstrated that spurious correlations appear quantitatively and qualitatively in neu-

ral networks even when the number of spurious examples is small. Now, we investigate the potentially undesirable effects through the lens of privacy and test accuracy.

Privacy. We evaluate the privacy of a neural network (the target model) through membership inference attack. We follow the setup for black-box membership inference attack (Shokri et al., 2017; Yeom et al., 2017). We record how well an attack model can distinguish whether an example is from the training or testing set using the output of the target model (equivalently to a binary classification problem). If the attack model has a high accuracy, this means that the target model is leaking out information from the training (private) data. The experiment is repeated ten times. More details are in Appendix E.2.

Results on membership inference attack. Fig. 3 shows the mean and standard error of the attack model's test accuracy on all test and spurious examples. We see that the accuracies on spurious examples is generally higher when the number of spurious examples are small, which means that spurious examples are more vulnerable to membership inference attacks when appeared rarely. Although membership inference attack is a different measure for privacy than spurious score, it can be a corroboration evidence that supports the fact that privacy is leaked from spurious examples.



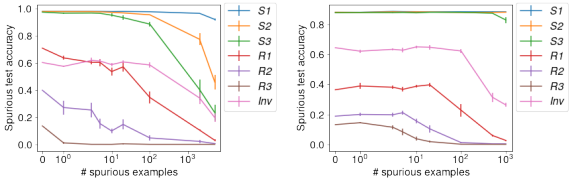
(a) MNIST, $c_{\text{tar}} = 0$, *R3* (b) CIFAR10, $c_{\text{tar}} = 0$, *R3*

Figure 3. The test accuracy of the membership inference attack model on all examples vs. spurious examples.

Test accuracy. We measure two types of test accuracy on neural networks trained on different number of spurious examples. The *clean test accuracy* measures the accuracy of the trained model on the original test data. The *spurious test accuracy* simulates the case where there is a distribution shift during the test time. Formally, spurious test accuracy is defined as the accuracy on a new test dataset constructed by adding spurious features to all the test examples with a label different from c_{tar}.

Results on clean test accuracy. We observe that the change in clean test accuracy in our experiments is small. Across all the models trained in Fig. 2, the minimum, maximum, average, and standard deviation of the test accuracy for each dataset are: MNIST: $(.976, .983, .980, .001)$, Fashion: $(.859, .903, .890, .010)$, CIFAR10: $(.876, .893, .886, .003)$.

Results on spurious test accuracy. The results are shown in Fig. 4. We have two observations. First, we see that there are already some accuracy drop even when spurious test accuracy is evaluated on models trained on zero spurious examples. This means that these models are not robust to the existence of spurious features. This phenomena is prominent for spurious patterns with larger norm such as *R3*. Second, we see that spurious test accuracies start to drop even more at around 10 to 100 spurious examples. This indicates that even with .01 % to .001 % of the overall training data filled with spurious examples of a certain class, the robustness to spurious features can drop significantly.



(a) MNIST, $c_{\text{tar}} = 0$ (b) CIFAR10, $c_{\text{tar}} = 0$

Figure 4. Each figure shows the mean and standard error of the spurious scores on three datasets, MNIST and CIFAR10, and different numbers of spurious examples.

Moreover, in Appendix B, we explore rare spurious correlation theoretically. The theoretical findings aligns with our observations here. Inspired by our theoretical results, we also explore ways to mitigate these rare spurious correlations in Appendix C. Finally, related works are in Appendix D.

Discussion. Our experimental results suggest that neural networks are *highly* sensitive to very small amounts of spurious training data. Furthermore, the learnt rare spurious correlations cause undesirable effects on privacy and test accuracy. Easy learning of rare spurious correlations can lead to privacy issues (Leino & Fredrikson, 2020) – where an adversary may infer the presence of a confidential image in a training dataset based on output probabilities. It also raises fairness concerns as a neural network can draw spurious conclusions about a minority group if a small number of subjects from this group are present in the training set (Izzo et al., 2021). We recommend to test and audit neural networks thoroughly before deployment in these applications.

3. Conclusion

We demonstrate that rare spurious correlations are learnt readily by neural networks, and we look closely into this phenomenon. We discover that a few spurious examples can lead to the model learning the spurious correlation through empirical and theoretical evidence. We then show that these rare spurious correlations can have impact on both privacy and test accuracy of the model. Finally, we find that some regularization methods, including weight decay and adding Gaussian noise to the input, can reduce the spurious correlation without sacrificing accuracy both empirically and theoretically. However, they are far from completely removing these spurious correlations.

References

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

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

Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. A closer look at memorization in deep networks. In *International Conference on Machine Learning*, pp. 233–242, 2017.

Bahng, H., Chun, S., Yun, S., Choo, J., and Oh, S. J. Learning de-biased representations with biased representations. In *International Conference on Machine Learning*, pp. 528–539, 2020.

Basu, S., Pope, P., and Feizi, S. Influence functions in deep learning are fragile. *arXiv preprint arXiv:2006.14651*, 2020a.

Basu, S., You, X., and Feizi, S. On second-order group influence functions for black-box predictions. In *International Conference on Machine Learning*, pp. 715–724, 2020b.

Burkard, C. and Lagesse, B. Analysis of causative attacks against svms learning from data streams. In *Proceedings of the 3rd ACM on International Workshop on Security And Privacy Analytics*, pp. 31–36, 2017.

Carlini, N., Liu, C., Erlingsson, Ú., Kos, J., and Song, D. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pp. 267–284, 2019.

Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al. Extracting training data from large language models. *arXiv preprint arXiv:2012.07805*, 2020.

Chang, C.-H., Adam, G. A., and Goldenberg, A. Towards robust classification model by counterfactual and invariant data generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15212–15221, 2021.

Chaudhuri, K., Monteleoni, C., and Sarwate, A. D. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(3), 2011.

Chen, X., Liu, C., Li, B., Lu, K., and Song, D. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pp. 265–284, 2006.

Feldman, V. and Zhang, C. What neural networks memorize and why: Discovering the long tail via influence estimation. *arXiv preprint arXiv:2008.03703*, 2020.

Ginart, A., Guan, M. Y., Valiant, G., and Zou, J. Making ai forget you: Data deletion in machine learning. *arXiv preprint arXiv:1907.05012*, 2019.

Gururangan, S., Swayamdipta, S., Levy, O., Schwartz, R., Bowman, S. R., and Smith, N. A. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*, 2018.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

Hartley, J. and Tsaftaris, S. A. Measuring unintended memorisation of unique private features in neural networks. *arXiv preprint arXiv:2202.08099*, 2022.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Idrissi, B. Y., Arjovsky, M., Pezeshki, M., and Lopez-Paz, D. Simple data balancing achieves competitive worst-group-accuracy. *arXiv preprint arXiv:2110.14503*, 2021.

Izzo, Z., Smart, M. A., Chaudhuri, K., and Zou, J. Approximate data deletion from machine learning models. In *International Conference on Artificial Intelligence and Statistics*, pp. 2008–2016, 2021.

Khani, F. and Liang, P. Removing spurious features can hurt accuracy and affect groups disproportionately. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 196–205, 2021.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Koh, P. W. and Liang, P. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pp. 1885–1894. PMLR, 2017.

Koh, P. W., Ang, K.-S., Teo, H. H., and Liang, P. On the accuracy of influence functions for measuring group effects. *arXiv preprint arXiv:1905.13289*, 2019.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images, 2009.

Kulynych, B., Yang, Y.-Y., Yu, Y., Błasiok, J., and Nakkiran, P. What you see is what you get: Distributional generalization for algorithm design in deep learning. *arXiv preprint arXiv:2204.03230*, 2022.

LeCun, Y. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Leino, K. and Fredrikson, M. Stolen memories: Leveraging model memorization for calibrated white-box membership inference. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pp. 1605–1622, 2020.

Liu, Z., Luo, P., Wang, X., and Tang, X. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15 (2018):11, 2018.

Min, Y., Chen, L., and Karbasi, A. The curious case of adversarially robust models: More data can help, double descend, or hurt generalization. *arXiv preprint arXiv:2002.11080*, 2020.

Nagarajan, V., Andreassen, A., and Neyshabur, B. Understanding the failure modes of out-of-distribution generalization. *arXiv preprint arXiv:2010.15775*, 2020.

Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., and Sutskever, I. Deep double descent: Where bigger models and more data hurt. *arXiv preprint arXiv:1912.02292*, 2019.

Neel, S., Roth, A., and Sharifi-Malvajerdi, S. Descent-to-delete: Gradient-based methods for machine unlearning. *arXiv preprint arXiv:2007.02923*, 2020.

pandas development team, T. pandas-dev/pandas: Pandas, February 2020. URL https://doi.org/10.5281/zenodo.3509134.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Sablayrolles, A., Douze, M., Schmid, C., and Jégou, H. Radioactive data: tracing through training. In *International Conference on Machine Learning*, pp. 8326–8335, 2020.

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

Sagawa, S., Raghunathan, A., Koh, P. W., and Liang, P. An investigation of why overparameterization exacerbates spurious correlations. In *International Conference on Machine Learning*, pp. 8346–8356, 2020.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pp. 3–18. IEEE, 2017.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Srivastava, M., Hashimoto, T., and Liang, P. Robustness to spurious correlations via human annotations. In *International Conference on Machine Learning*, pp. 9109–9119, 2020.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Wang, Y. and Chaudhuri, K. Data poisoning attacks against online learning. *arXiv preprint arXiv:1808.08994*, 2018.

Xiao, H., Biggio, B., Nelson, B., Xiao, H., Eckert, C., and Roli, F. Support vector machines under adversarial label contamination. *Neurocomputing*, 160:53–62, 2015.

Xiao, K., Engstrom, L., Ilyas, A., and Madry, A. Noise or signal: The role of image backgrounds in object recognition. *arXiv preprint arXiv:2006.09994*, 2020.

Yeom, S., Fredrikson, M., and Jha, S. The unintended consequences of overfitting: Training data inference attacks. *arXiv preprint arXiv:1709.01604*, 12, 2017.

Zhou, C., Ma, X., Michel, P., and Neubig, G. Examining and combating spurious features under distribution shift. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12857–12867, Jul 2021.

A. Qualitative Analysis: Visualizing Network Weights

We visualize the changes training with spurious examples can bring to the weights of a neural network. We consider an MLP architecture and pattern *S3* on MNIST, and look at the network's weights from the input layer to the first hidden layer. We visualize the importance of each pixel by plotting the maximum weight (among all weights) on an out-going edge from this pixel. Fig. 5 shows the importance plots for models trained with different numbers of spurious examples.

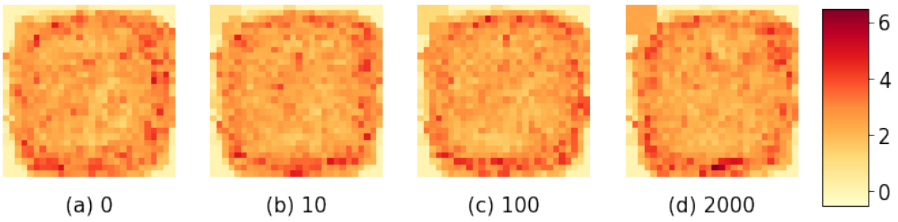


Figure 5. The importance of each pixel during the classification using an MLP trained on MNIST. Each pixel in the figure corresponds to a neuron of the input layer. The value of each pixel in the figure shows the maximum weight among all the weights that go out of the corresponding neuron from the input layer to the first hidden layer. The darker the color is, the larger the maximum weight is, which translates to the higher importance of the pixel during classification. The MLPs are trained on datasets with 0, 10, 100, and 2000 spurious examples on MNIST.

On the figure with zero spurious examples (Fig. 5 (a)), we see that the pixels in the top left corner are not important at all. When the number of spurious examples goes up, the values in the top left corner become larger (darker in the figure). This means that the pixels in the top left corner are gaining in importance, thus illustrating how they affect the network.

B. Theoretical Understanding

In this section, we devise a mathematical model to study rare spurious correlations. The theoretical analysis not only provides an unifying understanding to explain the experimental findings but also inspires us to propose methods to reduce the undesirable effects of rare spurious correlations in Appendix C. We emphasize that the purpose of the theoretical analysis is to capture the key factors in rare spurious correlations and we leave it as a future research direction to further deepen the theoretical study.

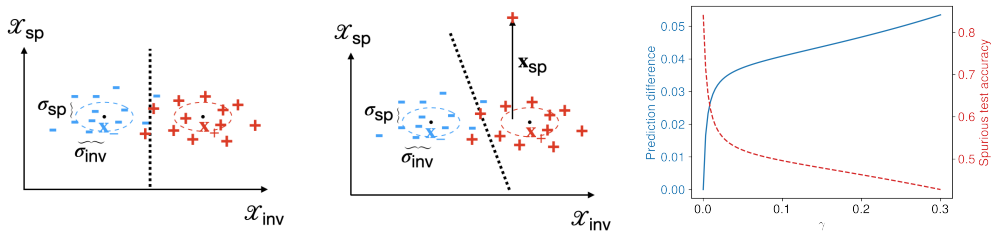
To avoid unnecessary mathematical complications, we make two simplifications in our theoretical analysis: (i) we focus on the *concatenate model* and (ii) the learning algorithm is linear regression with mean square loss. For (i), we argue that this is the simplest scenario of spurious correlations and hence it is a necessary step before we understand general spurious correlations. While the experiments in Section 2 work in the *overlapping model*, we believe that the high level messages of our theoretical analysis would extend to there as well as other more general scenarios. For (ii), we pick a simpler learning algorithm in order to have an analytical characterization of the algorithm's performance. This is because we aim to have an understanding of how the different factors (e.g., the fraction of spurious inputs, the strength of spurious feature, etc.) of spurious correlations play a role.

B.1. A theoretical model to study rare spurious correlations

We consider a binary classification task to model the appearance of rare spurious correlations. Let $\mathcal{X} = \mathcal{X}_{\text{inv}} \times \mathcal{X}_{\text{sp}}$ be an input vector space and let $\mathcal{Y} = \{-1, 1\}$ be a label space. Let $\gamma \in [0, 1]$ be the parameter for the fraction of spurious samples, let $\mathbf{x}_-, \mathbf{x}_+ \in \mathcal{X}_{\text{inv}}$ be the invariant features of the two classes, let $\mathbf{x}_{\text{sp}} \in \mathcal{X}_{\text{sp}}$ be the spurious feature, and let $\sigma_{\text{inv}}^2, \sigma_{\text{sp}}^2 > 0$ be the parameters for the variance along $\mathcal{X}_{\text{inv}}$ and $\mathcal{X}_{\text{sp}}$ respectively. Finally, the target class is $+$, i.e., $c_{tar} = +$. We postpone the formal definitions of the training distribution $\mathcal{D}_{\text{train}}$, the clean text distribution $\mathcal{D}_{\text{ctest}}$, and the spurious test distribution $\mathcal{D}_{\text{stest}}$ to Appendix F. See also Fig. 6 for some pictorial examples.

B.2. Analysis for linear regression with mean square loss and ℓ_2 regularization

In this paper, we analyze our theoretical model in the setting of linear regression with ℓ_2 loss. We analytically derive the test accuracy and the prediction difference $f_{c_{\text{tar}}}(\Phi_{\mathcal{X}}(\mathbf{x}, \mathbf{x}_{\text{sp}})) - f_{c_{\text{tar}}}(\mathbf{x})$ in Appendix F and here we present our observations. Here, we study the prediction difference as a proxy for the spurious score since the latter is always either 0 or 1 for a linear



(a) $\gamma = 0$. (b) $\gamma > 0$. (c) Phase transitions.

Figure 6. Examples of the training distribution $\mathcal{D}_{\text{train}}$ and phase transitions in our theoretical model. (a)-(b) With equal probability a training example is sampled from either $\mathcal{N}(\mathbf{x}_+, \sigma_{\text{inv}} I_{\text{inv}} + \sigma_{\text{sp}} I_{\text{sp}})$ or $\mathcal{N}(\mathbf{x}_-, \sigma_{\text{inv}} I_{\text{inv}} + \sigma_{\text{sp}} I_{\text{sp}})$. With probability γ, a $+$ sample will be concatenated with the spurious pattern $\mathbf{x}_{\text{sp}}$. The dotted line is the decision boundary of the optimal classifier. (c) Both the spurious test accuracy and the prediction difference exhibit a phase transition at $\gamma = 0$.

classifier under our model.

Observation 1: A phase transition of spurious test accuracy and prediction difference at $\gamma = 0$. Our theoretical analysis first suggests that there is a phase transition of the spurious test accuracy and the prediction difference spurious score at $\gamma = 0$, i.e., there is an sharp growth/decay within an interval near $\gamma = 0$. The phase transition matches our previous experimental studies discussed in Section 2. This indicates that the effect of spurious correlations is spontaneous rather than gradual. To be more quantitative, the phase transition takes place when the signal-to-noise ratio of spurious feature, i.e., $\|\mathbf{x}_{\text{sp}}\|_2^2/\sigma_{\text{sp}}^2$, is large (see Appendix F for details). This further suggests us to increase the variance in the spurious dimension and leads to our next two observations.

Observation 2: adding Gaussian noises lowers spurious score. The previous observation on the importance of spurious signal-to-noise ratio $\|\mathbf{x}_{\text{sp}}\|_2^2/\sigma_{\text{sp}}^2$ immediately suggests us to add Gaussian noises to the input data to *lower* $\|\mathbf{x}_{\text{sp}}\|_2^2/\sigma_{\text{sp}}^2$. Indeed, the prediction difference becomes smaller in most parameter regimes, however, both the clean test accuracy and the spurious test accuracy decrease. Intuitively, the effect of adding noises is to mix the invariant feature with the spurious feature and the decrease of test accuracy is as expected. Thus, to simultaneously lower prediction difference and improve test accuracy, one needs to detect the spurious feature in some ways.

Observation 3: ℓ_2 regularization improves test accuracy and lowers spurious score. Finally, our theoretical analysis reveals that there are two parameter regimes where adding ℓ_2 regularization to linear regression can improves accuracy and lowers the prediction difference. First, when σ_{inv}^2 is small, γ is small, and the spurious signal-to-noise ratio $\|\mathbf{x}_{\text{sp}}\|_2^2/\sigma_{\text{sp}}^2$ is large. Second, when σ_{inv}^2 is large and both γ and $\|\mathbf{x}_{\text{sp}}\|_2^2/\sigma_{\text{sp}}^2$ are mild. Intuitively, ℓ_2 regularization suppresses the use of features that only appears on a small number of training examples.

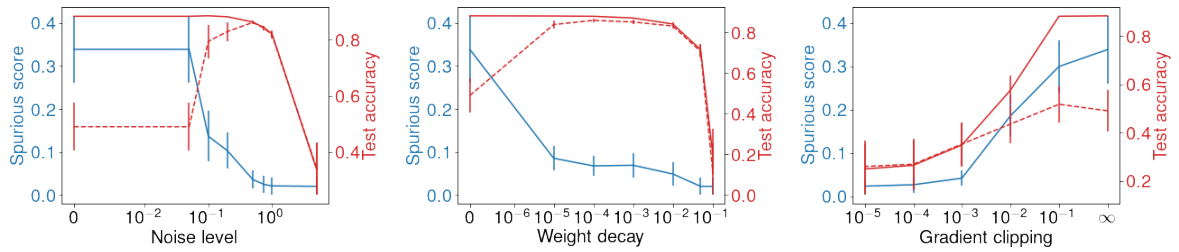
Discussion. Our theoretical analysis quantitatively demonstrates the phase transition of spurious correlations at $\gamma = 0$ and the importance of the spurious signal-to-noise ratio $\|\mathbf{x}_{\text{sp}}\|_2^2/\sigma_{\text{sp}}^2$. This not only coincides with our empirical observation in Section 2 but also suggests future directions to mitigate rare spurious correlations. Specifically, one general approach to reduce the undesirable effects of rare spurious correlations would be designing learning algorithms that projects the input into a feature space that has a low spurious signal-to-noise ratio.

C. Mitigation of Rare Spurious Correlation

Prior work uses group rebalancing (Idrissi et al., 2021; Sagawa et al., 2020; 2019; Kulynych et al., 2022), data augmentation (Chang et al., 2021) or learning invariant classifier (Arjovsky et al., 2019) to mitigate spurious correlations. However, these methods usually requires additional information on what the spurious feature is, and in rare spurious correlations, identifying the spurious feature can be hard. Thus, we may require different techniques.

Our theoretical result suggest that ℓ_2 regularization (weight decay) and adding Gaussian noise to the input (noisy input) may reduce the degree of spurious correlation being learnt. In addition, we examine an extra regularization method – gradient clipping.

Results. The results are shown in Fig. 7. We see that with a properly set regularization strength for noisy input and weight decay, one may reduce the spurious score and increase spurious test accuracy without sacrificing much accuracy. This significantly reduces the undesirable consequences brought by rare spurious correlations. This aligns with our observation



(a) Noisy input (b) Weight decay (c) Gradient clipping

Figure 7. Spurious score (solid blue line), clean test accuracy (solid red line), and spurious test accuracy (dotted red line) vs. the regularization strength on Fashion with different regularization methods. For the experiment, we fix the spurious pattern to be *S3* and the target class $c_{tar} = 0$. We compute the average spurious score and clean test accuracy across models trained with 1, 3, 5, 10, 20, and 100 spurious examples and five random seeds. The results for MNIST and CIFAR10 and spurious pattern *R3* are in Appendix E, which shows similar results.

in the theoretical model and suggests that neural networks may share a similar property with linear models. We also find that gradient clipping cannot mitigate spurious correlation without reducing the test accuracy. Finally, we observe that all these methods are unable to completely avoid learning spurious correlations.

Data deletion methods. Another idea for mitigating rare spurious correlation is to apply data deletion methods (Izzo et al., 2021). In Appendix G, we experiments with two data deletion methods, incremental retraining and group influence functions (Basu et al., 2020a). However, they are not effective.

Discussion. Regarding mitigating rare spurious correlations, a provable way to prevent learning them is differential privacy (Dwork et al., 2006), which ensures that the participation of a single person (or a small group) in the dataset does not change the probability of any classifier by much. This requires noise addition during training, which may lead to a significant loss in accuracy (Chaudhuri et al., 2011; Abadi et al., 2016). If we know which are the spurious examples, then we can remove spurious correlations via an indistinguishable approximate data deletion method (Ginart et al., 2019; Neel et al., 2020); however, these methods provide lower accuracy for convex optimization and have no performance guarantees for non-convex. An open problem is to design algorithms or architectures that can mitigate these without sacrificing prediction accuracy.

Prior work (Sagawa et al., 2019; Kulynych et al., 2022) suggests that proper use of regularization methods plus well-designed loss functions can mitigate some types of spurious correlations. However, these regularization methods are used either in an ad-hoc manner or may reduce test accuracy. As shown in our experiment, not all regularization methods can remove rare spurious correlations without reducing the accuracy. This suggests that different regularization methods may be specifically tied to be able to mitigate certain kinds of spurious correlation. The exact role that regularization methods play in reducing spurious correlations is still an open question. Figuring out what kinds of spurious correlations can be mitigated by what kinds of regularization methods can be an interesting future direction.

D. Related Work

Spurious correlations. Previous work has looked at spurious correlations in neural networks under various scenarios, including test time distribution shift (Sagawa et al., 2020; Srivastava et al., 2020; Bahng et al., 2020; Zhou et al., 2021; Khani & Liang, 2021), confounding factors in data collection (Gururangan et al., 2018), the effect of image backgrounds (Xiao et al., 2020), and causality (Arjovsky et al., 2019). However, in most works, spurious examples often constitute a significant portion of the training set. In contrast, we look at spurious correlations introduced by a small number of examples (rare spurious correlations). Concurrent work (Hartley & Tsaftaris, 2022) measures spurious correlation caused by few examples. However, they did not show the consequences of these spurious correlations nor discuss ways to mitigate them.

Memorization in neural networks. Prior work has investigated how neural networks can inadvertently memorize training data (Arpit et al., 2017; Carlini et al., 2019; 2020; Feldman & Zhang, 2020; Leino & Fredrikson, 2020). Methods have also been proposed to measure this kind of memorization, including the use of the influence function (Feldman & Zhang, 2020) and likelihood estimates (Carlini et al., 2019). Our work focuses on partial memorization instead of memorizing individual examples, and our proposed method may be potentially applicable in more scenarios.

A line of work in the security literature exploits the memorization of certain patterns to compromise neural networks. The backdoor attack from Chen et al. (2017) attempts to change hard label predictions and accuracy by inserting carefully crafted spurious patterns. Sablayrolles et al. (2020) design specific markers that allow adversaries to detect whether images with those particular markers are used for training in a model. Another line of research on data poisoning attack (Xiao et al., 2015; Wang & Chaudhuri, 2018; Burkard & Lagesse, 2017) aims to degrade the performance of a model by carefully altering the training data. In contrast, our work looks at rare spurious correlations from *natural spurious patterns*, instead of adversarially crafted ones.

Data deletion methods. Inspired by GDPR, there are many recent works on data deletion. Izzo et al. (2021) demonstrate data deletion for linear classifiers, but not for non-linear models such as neural networks. The use of influence and group influence functions for data deletion is also studied by many (Koh & Liang, 2017; Koh et al., 2019; Basu et al., 2020b). Basu et al. (2020a) point out that influence functions can be fragile for deeper neural networks. Our work shows that influence functions cannot remove spurious correlations caused by the deleted examples, which is different.

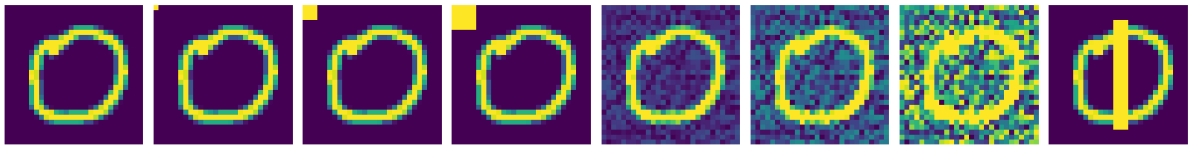
Concerns for expanding training sets. Researchers have also discovered ways that more data can hurt the model in terms of the generalization ability and test time accuracy (Nakkiran et al., 2019; Min et al., 2020). In this work, we uncover a different way that more data can hurt: more data could introduce more spurious correlations.

E. Experimental Details and Additional Results

The experiments are performed on six NVIDIA GeForce RTX 2080 Ti and two RTX 3080 GPUs located on three servers. Two of the servers have Intel Core i9 9940X and 128GB of RAM and the other one has AMD Threadripper 3960X and 256GB of RAM. All neural networks are implemented under the PyTorch framework[1] (Paszke et al., 2019). Other packages, including `numpy` [2] (Harris et al., 2020), `scipy` [3] (Virtanen et al., 2020), `tqdm` [4], and `pandas` [5] (pandas development team, 2020), are also used.

E.0.1. Full list of spurious patterns

A full list of spurious patterns is in Fig. 8.



(a) *original* (b) *S1* (c) *S2* (d) *S3* (e) *R1* (f) *R2* (g) *R3* (h) *Inv*

Figure 8. Different spurious patterns considered in the experiment.

E.1. Experimental setup for Section 2

Architectures. For MNIST and Fashion, we consider multi-layer perceptrons (MLP) with ReLU activation functions. MLP has two hidden layers, and each layer has 256 neurons. For CIFAR10, we consider ResNet20 (He et al., 2016).

Optimizer, learning rate, and data augmentation. We use the Adam (Kingma & Ba, 2014) optimizer and set the initial learning rate to 0.01 for all models. We train the model for 70 epochs. For the learning rate schedule, we decrease the learning rate by a factor of 0.1 on the 40-th, 50-th, and 60-th epoch. For CIFAR10, we apply data augmentation during training. When an image is passed in, we pad each border with four pixels and randomly crop the image to 32 by 32. We then, with 0.5 probability, horizontally flip the image.

[1] Code and license can be found in `https://github.com/pytorch/pytorch`.
[2] Code and license can be found in `https://github.com/numpy/numpy`
[3] Code and license can be found in `https://github.com/scipy/scipy`
[4] Code and license can be found in `https://github.com/tqdm/tqdm`
[5] Code and license can be found in `https://github.com/pandas-dev/pandas`

E.2. Experimental setup for Section 2.3

Membership inference attack setup. We split each dataset into four equally sized sets – target model training/testing sets and shadow model training/testing sets. We then add spurious features to a number of examples in the target model training set. We train the target model using the target model training set and the shadow model using the shadow model training set. Next, we extract the features on the target/shadow model training/testing sets. The feature we use are the output of the target/shadow model on these data and a binary indicator indicating whether the example is correctly predicted as the features. We then train an attack model (binary classifier) to distinguish whether an example comes from the shadow model training or testing sets using the extracted features. For evaluation, we compute the accuracy of the attack model distinguishing whether an example comes from the target model training or testing sets (test accuracy).

For the shadow and target model, we train them the same way as other models in Section 2. For the attack model, we use a logistic regression implemented with `scikit-learn` [6] (Pedregosa et al., 2011) with 3-fold cross-validation to select the regularization parameter from $\{0.01, 0.1, 1.0, 10.0\}$.

E.3. How different factors affect rare spurious correlations

We next investigate how different factors can affect the extent to which rare spurious correlations can be learnt. For this purpose, we consider the norm of each pattern, network architectures, and the optimization algorithms.

E.3.1. ℓ_2 NORM OF THE SPURIOUS PATTERN

Fig. 2, we see that neural networks can learn different patterns very differently. For example, the spurious scores for *R3* are higher than *S1*, suggesting spurious correlations with *R3* are learnt more easily. Why does this happen? We hypothesize that the higher the norm of the pattern is, easier it is for a network to learn the correlation between the patterns and the target class.

Because the spurious patterns may overlap with other features, directly using the norm of each spurious pattern may not be accurate. We define the *empirical norm* of a spurious pattern $\mathbf{x}_{sp}$ on an example $\mathbf{x}$ as the ℓ_2 distance between $\mathbf{x}$ and the spurious example $g(\mathbf{x}, \mathbf{x}_{sp})$. We compute the average empirical norm over the test examples for each pattern. Table 1 shows the average empirical norm of each pattern on different datasets.

For each dataset, we train neural networks with a different number of spurious examples. To measure the aggregated effect of a spurious pattern across different numbers of spurious examples, we compute the average spurious scores across different numbers of spurious examples. We compute the Pearson correlation between the average empirical norm and the average spurious scores of each model trained with different spurious patterns. The testing results are: MNIST: $\rho = 0.91, p < 0.01$; Fashion: $\rho = 0.84, p = 0.02$; CIFAR10: $\rho = 0.98, p < 0.01$. The result shows a *significantly strong positive correlation between the norm of the spurious patterns and the spurious scores*.

Table 1. The average empirical norm of each spurious pattern.

	S1	*S2*	*S3*	*R1*	*R2*	*R3*	*Inv*
MNIST	1.00	3.00	5.00	3.88	7.70	15.19	5.98
Fashion	1.00	3.00	4.98	3.90	7.40	13.65	4.44
CIFAR10	0.84	2.57	4.34	7.72	14.54	23.20	8.00

E.3.2. NETWORK ARCHITECTURES

Are some network architectures more susceptible to spurious correlations than others? To answer this question, we look at how spurious scores vary across different network architectures.

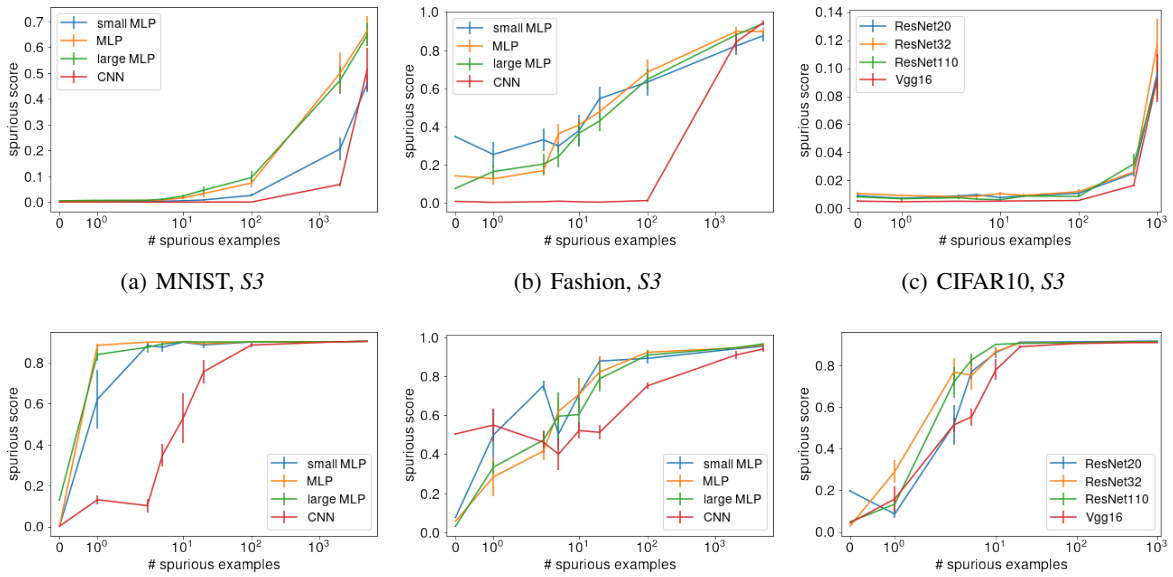
Network architectures. For MNIST and Fashion, we consider multi-layer perceptrons (MLP) with different sizes and a convolutional neural network (CNN)[7]. The *small MLP* has one hidden layer with 256 neurons. The *MLP* has two hidden layers, each layer with 256 neurons (the same MLP used in Fig. 2). The *large MLP* has two hidden layers, each with 512 neurons. For CIFAR10, we consider ResNet20, ResNet34, ResNet110 (He et al., 2016), and Vgg16 (Simonyan & Zisserman,

[6] Code and license can be found in `https://github.com/scikit-learn/scikit-learn`.

[7] public repository: `https://github.com/yaodongyu/TRADES/blob/master/models/`

2014). We use an SGD optimizer for CIFAR10 since we cannot get reasonable performance for Vgg16 with Adam.

Setup. For MNIST and Fashion, we set the learning rate to 0.01 for all architectures and optimizers. We set the momentum to 0.9 when the optimizer is SGD. For CIFAR10, when training with SGD, we set the learning rate to 0.1 for ResNets trained and 0.01 for Vgg16 because Vgg16 failed to converge with learning rate 0.1. We set the learning rate to 0.01 for ResNets when running with Adam (Vgg16 failed to converge with Adam). We use a learning rate scheduler which decreases the learning rate by a factor of 0.1 on the 40-th, 50-th, and 60-th epoch.



(a) MNIST, *S3* (b) Fashion, *S3* (c) CIFAR10, *S3*

(d) MNIST, *R3* (e) Fashion, *R3* (f) CIFAR10, *R3*

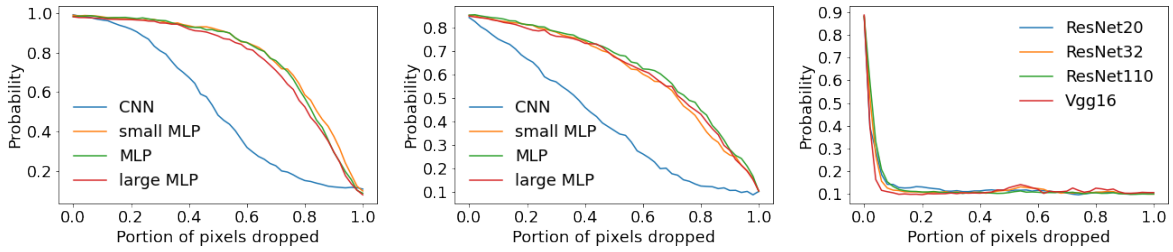
Figure 9. The mean and standard error of the spurious scores with different network architectures on MNIST, Fashion, and CIFAR10. The target class is $c_{tar} = 0$.

Fig. 9 shows the result, and we see that similar architectures with different sizes generally have similar spurious scores. Concretely, small MLP, MLP, and large MLP perform similarly, and ResNet20, ResNet32, and ResNet110 also perform similarly. Additionally, CNN is less affected by spurious examples than MLPs while Vgg16 is also slightly less affected than ResNets.

Why are MLPs more sensitive to small patterns? We observe that for *S3*, MLP seems to be the only architecture that can learn the spurious correlation when the number of spurious examples is small (< 100). At the same time, CNN requires slightly more spurious examples while ResNets and Vgg16 cannot learn the spurious correlation on small patterns (note that the y-axis on Fig. 9 (e) is very small). Why is this happening? We hypothesize that different architectures have different sensitivities to the presence of evidence, i.e., the pixels of the image. Some architectures change their prediction a lot based on a small number of pixels, while others require a large number. If a network architecture is sensitive to changes in a small number of pixels, it can also be sensitive to a small spurious pattern.

To validate our hypothesis, we measure the sensitivity of a neural network as follows. First, we train a neural network on the clean training dataset. During testing, we set to zero $0\%, 2\%, \ldots, 98\%, 100\%$ of randomly chosen non-zero pixels in each test image, and measure the predicted probability of its ground truth label. If this predicted probability continues to be high, then we say that the network is insensitive to the input. Fig. 10 shows the average predicted probability over 500 training examples as a function of the percentage of pixels set to zero for the MNIST dataset.

We see that MLPs have around 0.9 average predicted probability with half of the pixels zero-ed out. In contrast, the average predicted probability is lower in CNNs, suggesting that CNNs may be more sensitive to the zero-ed out pixels. From these results, we can rank the sensitivity of different architectures from non-sensitive to sensitive as MLPs $<$ CNN $<$ ResNets $\approx$ Vgg. This order matches our observation that MLPs are the most susceptible to spurious correlations, while CNN, ResNets, and Vgg16 are less so – suggestions that sensitive models may be more susceptible to learning spurious correlations with



(a) MNIST (b) Fashion (c) CIFAR10

Figure 10. This figure shows the predicted probability of the ground truth label as a function of the portion of non-zero value pixels removed across different architectures and datasets.

small patterns.

Finally, we find that architectures that have more parameters *are not* always more vulnerable to spurious correlations. Table 2 shows the number of parameters for each architecture. We see that while CNN has more parameters than small MLP, it is less susceptible to spurious correlations. Vgg16 and ResNet20 show a similar pattern. This observation is counter to Sagawa et al. (2020), who suggest that neural networks with more parameters can learn spurious correlations more easily, and it may be because they are looking at a different type of spurious correlation.

Table 2. Number of parameters in each architecture.

small MLP	MLP	large MLP	CNN
203,530	335,114	932,362	312,202
ResNet20	ResNet32	ResNet110	Vgg16
269,722	464,154	1,727,962	134,301,514

E.3.3. Optimization Process

We study how the optimization process affects the learning of the spurious correlation. We look into two main components of the optimization process, the optimizers and the use of gradient clipping, and examine how each component affects the spurious score. For the optimizers, we compare between Adam and SGD, while for gradient clipping, we compare between no gradient clipping and clipping the norm the gradient to 0.1. We repeat the five times with different random seeds and record their mean and standard error.

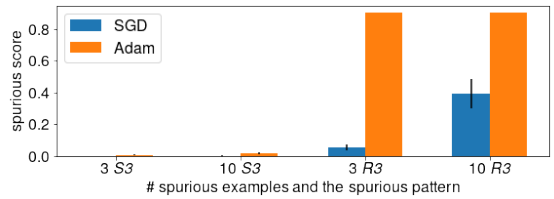
Fig. 11 shows the average spurious scores between different optimizers on different datasets and spurious patterns. We find that Adam is slightly more susceptible to spurious correlations than SGD, and gradient clipping does not affect the results much.

Fig. 12 shows the average spurious scores on networks trained with and without gradient clipping on different datasets and spurious patterns. We see that, in most cases, with and without gradient clipping performs similarly. It appears that using gradient clipping alone is not sufficient to eliminate the rare spurious correlations from neural networks.

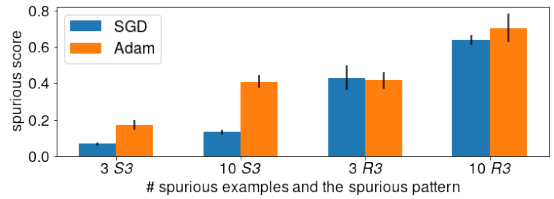
Overall, we see that rare spurious correlations are learnt regardless of the choice of the optimizer and whether the gradient clipping is performed or not. This indicates that tweaking individual components in the optimization process may not be sufficient to remove spurious correlations.

E.4. Additional Results for Different Regularization Methods

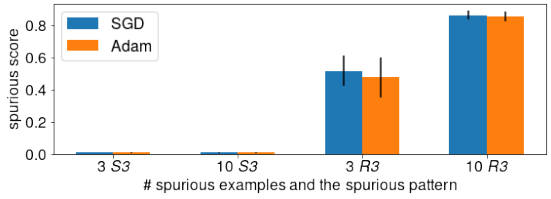
Fig. 14 shows how different regularization level affects the spurious score, spurious test accuracy, and clean test accuracy with *S3* as the spurious pattern. Fig. 15 shows same thing but with *R3* as the spurious pattern. Fig. 13 visualizes how different regularization strength can affect the spurious correlation learnt on the theoretical model. We see similar trends across these three cases.



(a) MNIST SGD vs. Adam

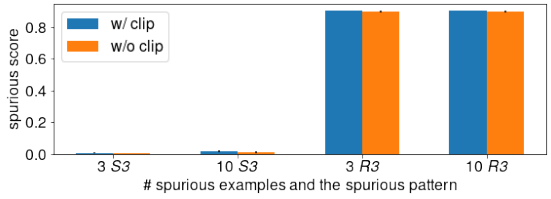


(b) Fashion SGD vs. Adam

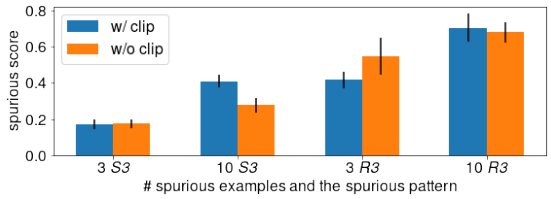


(c) CIFAR10 SGD vs. Adam

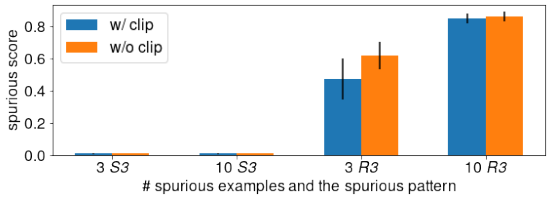
Figure 11. The mean and standard error of the spurious scores on neural networks trained with SGD versus Adam. We consider networks trained with three and ten spurious examples as well as using the *S3* and *R3* patterns.



(a) MNIST w/ vs. w/o clipping



(b) Fashion w/ vs. w/o clipping



(c) CIFAR10 w/ vs. w/o clipping

Figure 12. The mean and standard error of the spurious scores on neural networks trained with and without gradient clipping. We consider networks trained with three and ten spurious examples as well as using the *S3* and *R3* patterns.

E.5. Additional Results for Figure 2

Fig. 16 shows the result for $c_{tar} = 1$. We see that in all these cases, increasing regularization strength decreases spurious scores The conclusions are similar to ones made in Section 2 Fig. 2.

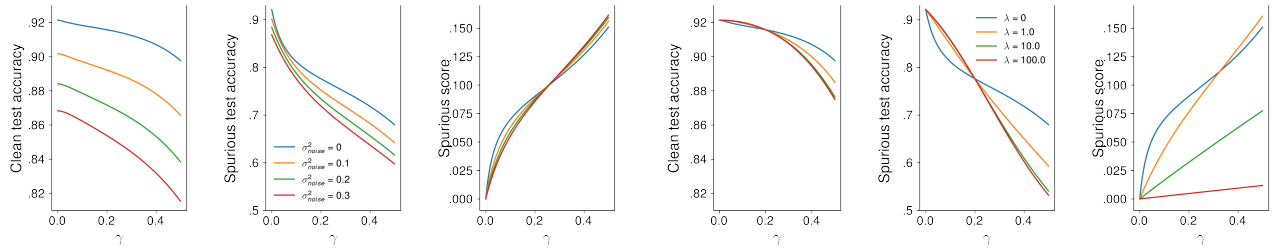
F. Details for Our Theoretical Model

In this section, we provide the details of our theoretical analysis in Appendix B. To be self-contained, we review the mathematical language we use to discuss spurious correlations in Appendix F.1. Next, we provide the formal definitions of our training and testing models in Appendix F.2 and state the main results and implications in Appendix F.3. Finally, we give the complete proof for the main theorem in Appendix F.4

F.1. Preliminaries

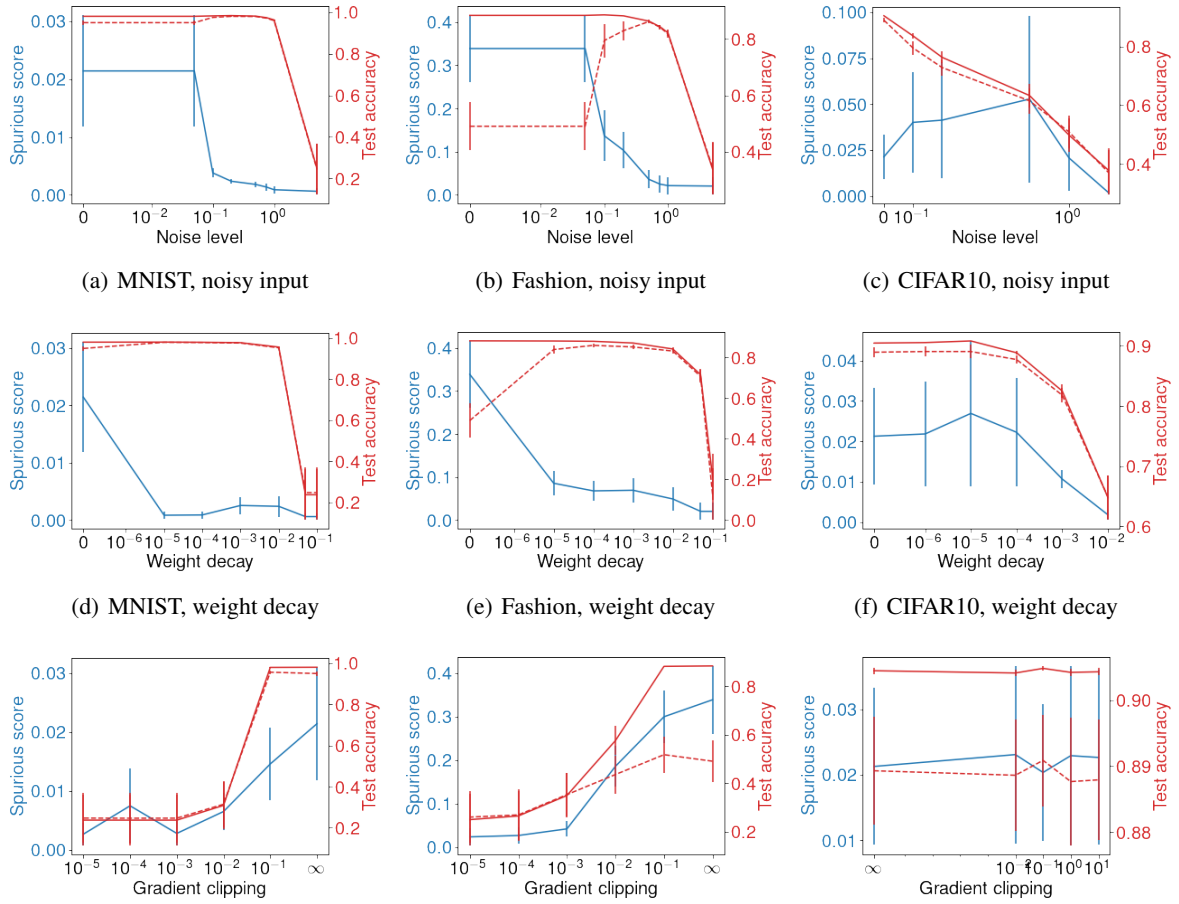
Recall that we focus on the image classification task where $\mathcal{X}$ is an input space, and $\mathcal{Y}$ is a label space. An example is a pair $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$ and at the training time, we are given a set of examples sampled from a distribution $\mathcal{D}_{\text{train}}$. At the testing time, we evaluate the network on either the clean test distribution $\mathcal{D}_{\text{ctest}}$ or the spurious test distribution $\mathcal{D}_{\text{stest}}$.

Spurious correlation. We model an input $\mathbf{x}$ as the output of a feature map $\Phi_{\mathcal{X}}$ from the feature space $\mathcal{X}_{\text{inv}} \times \mathcal{X}_{\text{sp}}$ to the input space $\mathcal{X}$ where $\mathcal{X}_{\text{inv}}$ is the invariant feature space containing the features that causally determine the label and $\mathcal{X}_{\text{sp}}$ is the spurious feature space that accommodates spurious features. $\Phi_{\mathcal{Y}} : \mathcal{X}_{\text{inv}} \to \mathcal{Y}$ is a function that maps the invariant feature $\mathbf{x}_{\text{inv}}$ to a label. Thus, an example $(\mathbf{x}, y)$ is generated by $(\mathbf{x}_{\text{inv}}, \mathbf{x}_{\text{sp}})$ via $\mathbf{x} = \Phi_{\mathcal{X}}(\mathbf{x}_{\text{inv}}, \mathbf{x}_{\text{sp}})$ and $y = \Phi_{\mathcal{Y}}(\mathbf{x}_{\text{inv}})$. Without loss of generality, the zero vector in $\mathcal{X}_{\text{sp}}$, i.e., $0 \in \mathcal{X}_{\text{sp}}$, refers to "no spurious feature" and for any nonzero $\mathbf{x}_{\text{sp}}$ we call $\Phi(\mathbf{x}_{\text{inv}}, \mathbf{x}_{\text{sp}}) - \Phi(\mathbf{x}_{\text{inv}}, 0)$ a spurious pattern. We focus on the case of having a fixed spurious feature $\mathbf{x}_{\text{sp}}$ and leave it as a



(a) clean test (b) spurious test (c) spurious score (d) clean test (e) spurious test (f) spurious score

Figure 13. An example of our theoretical findings. (a)-(c): adding Gaussian noises. (d)-(f): ℓ_2 regularization. The parameters are set to be $\|\mathbf{x}_{\text{sp}}\|_2^2 = 5$, $\sigma_{\text{inv}}^2 = 0.5$, $\sigma_{\text{sp}}^2 = 0.1$, and we consider $\gamma \in [0, 0.5]$, noise strength $\sigma_{noise}^2 = 0, 0.1, 0.2, 0.3$, and the ℓ_2 regularization parameter $\lambda = 0, 1, 10, 100$.



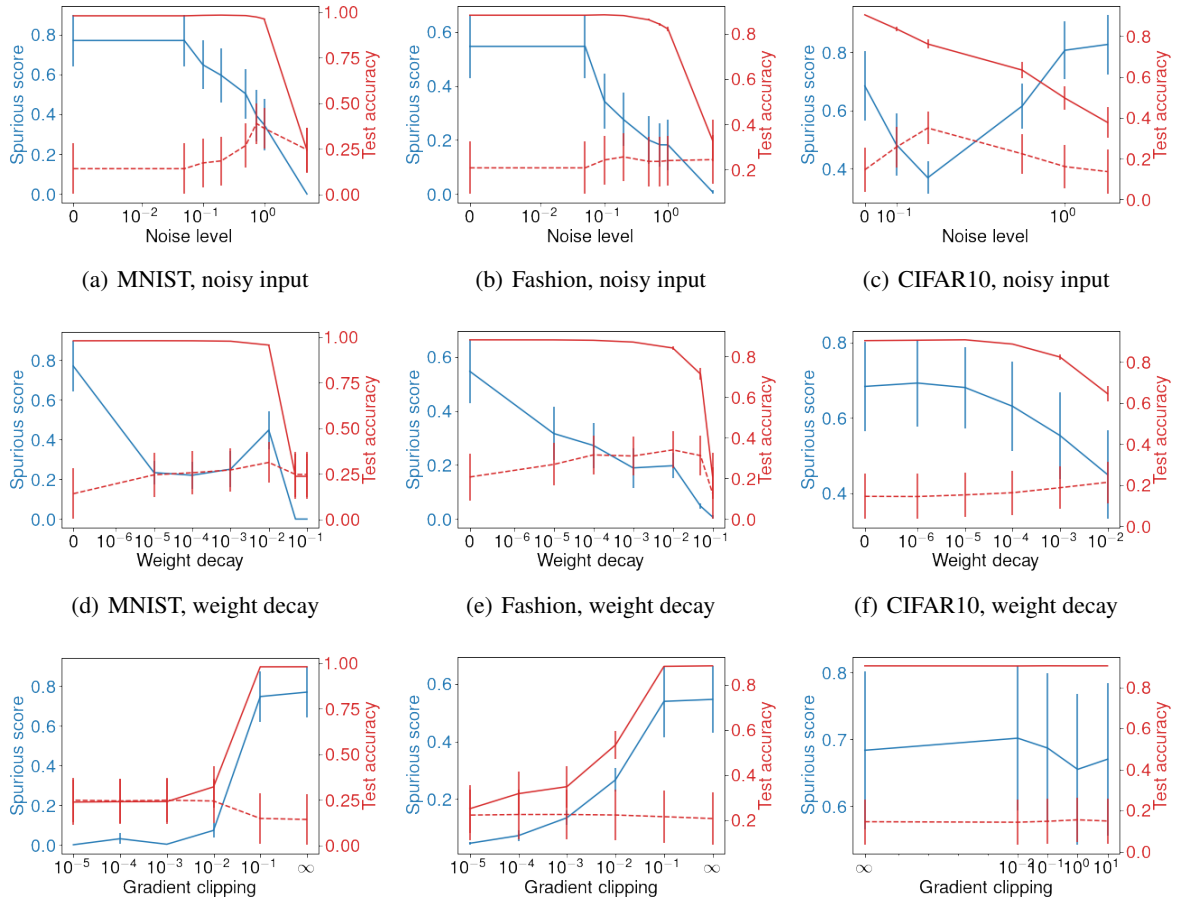
(a) MNIST, noisy input (b) Fashion, noisy input (c) CIFAR10, noisy input

(d) MNIST, weight decay (e) Fashion, weight decay (f) CIFAR10, weight decay

(g) MNIST, gradient clipping (h) Fashion, gradient clipping (i) CIFAR10, gradient clipping

Figure 14. Spurious score (solid blue line), clean test accuracy (solid red line), and spurious test accuracy (dotted red line) vs. the regularization strength on MNIST, Fashion, and CIFAR10 with different regularization methods. For the experiment, we fix the spurious pattern to be *S3* and the target class $c_{tar} = 0$. We compute the average spurious score and clean test accuracy across models trained with 1, 3, 5, 10, 20, and 100 spurious examples and five random seeds.

future direction to study the more general scenarios where there are multiple spurious features.

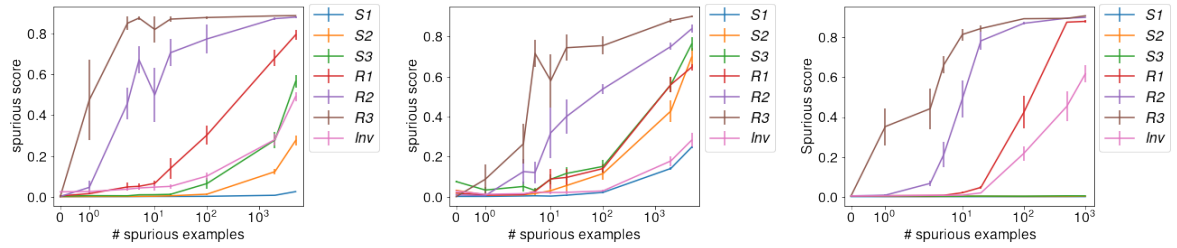


(a) MNIST, noisy input (b) Fashion, noisy input (c) CIFAR10, noisy input

(d) MNIST, weight decay (e) Fashion, weight decay (f) CIFAR10, weight decay

(g) MNIST, gradient clipping (h) Fashion, gradient clipping (i) CIFAR10, gradient clipping

Figure 15. Spurious score (solid blue line), clean test accuracy (solid red line), and spurious test accuracy (dotted red line) vs. the regularization strength on MNIST, Fashion, and CIFAR10 with different regularization methods. For the experiment, we fix the spurious pattern to be *R3* and the target class $c_{tar} = 0$. We compute the average spurious score and clean test accuracy across models trained with 1, 3, 5, 10, 20, and 100 spurious examples and five random seeds.



(a) MNIST, $c_{tar} = 1$ (b) Fashion, $c_{tar} = 1$ (c) CIFAR10, $c_{tar} = 1$

Figure 16. Each figure shows the mean and standard error of the spurious scores on three datasets, MNIST, Fashion, and CIFAR10, $c_{tar} = 1$, and different numbers of spurious examples. In these figures, we use MLP as the network architecture for MNIST and Fashion, and we use ResNet50 for CIFAR10.

Two simple models for spurious correlations. We consider two simple instantiations of $\Phi_{\mathcal{X}}$. First, the *overlapping model* (used in Section 2) where the spurious feature is put on top of the invariant feature, i.e., $\mathcal{X} = \mathcal{X}_{\text{inv}} = \mathcal{X}_{\text{sp}}$ and $\Phi_{\mathcal{X}}(\mathbf{x}_{\text{inv}}, \mathbf{x}_{\text{sp}}) = \mathbf{x}_{\text{inv}} + \mathbf{x}_{\text{sp}}$ or $\Phi_{\mathcal{X}}(\mathbf{x}_{\text{inv}}, \mathbf{x}_{\text{sp}}) = clip(\mathbf{x}_{\text{inv}} + \mathbf{x}_{\text{sp}})$ where $clip$ is a function that truncates an input pixel when

its value exceeds a certain range. Second, the *concatenate model* (used in Appendix B) where the spurious feature is concatenated to the invariant feature, i.e., $\mathcal{X} = \mathcal{X}_{\text{inv}} \times \mathcal{X}_{\text{sp}}$ and $\Phi_{\mathcal{X}}(\mathbf{x}_{\text{inv}}, \mathbf{x}_{\text{sp}}) = (\mathbf{x}_{\text{inv}}, \mathbf{x}_{\text{sp}})$.

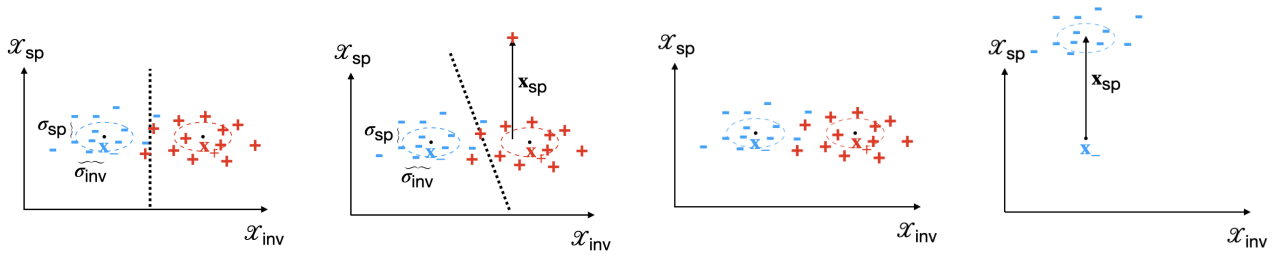
F.2. Our training and testing models

Here we delineate the theoretical model described in Appendix B.1. Recall that we consider a binary classification task to model where $\mathcal{X} = \mathcal{X}_{\text{inv}} \times \mathcal{X}_{\text{sp}}$ is an input vector space and $\mathcal{Y} = \{-1, 1\}$ is a label space. $\gamma \in [0, 1]$ is the parameter for the fraction of spurious samples and in this paper we focus on the regime where γ is very close to 0. $\mathbf{x}_-, \mathbf{x}_+ \in \mathcal{X}_{\text{inv}}$ are the invariant features of the two classes and after a linear shift[8] we can without loss of generality have $\mathbf{x}_- = -\mathbf{x}_+$. Let $\mathbf{x}_{\text{sp}} \in \mathcal{X}_{\text{sp}}$ be the spurious feature where we call its length, $\|\mathbf{x}_{\text{sp}}\|_2$, the spurious signal strength. Finally, $\sigma_{\text{inv}}^2, \sigma_{\text{sp}}^2 > 0$ are the parameters for the variance along $\mathcal{X}_{\text{inv}}$ and $\mathcal{X}_{\text{sp}}$ respectively. Now, we formally define the training distribution $\mathcal{D}_{\text{train}} = \mathcal{D}_{\text{train}}(\gamma, \mathbf{x}_-, \mathbf{x}_+, \mathbf{x}_{\text{sp}}, \sigma_{\text{inv}}, \sigma_{\text{sp}})$, the clean text distribution $\mathcal{D}_{\text{ctest}} = \mathcal{D}_{\text{ctest}}(\gamma, \mathbf{x}_-, \mathbf{x}_+, \mathbf{x}_{\text{sp}}, \sigma_{\text{inv}}, \sigma_{\text{sp}})$, and the spurious test distribution $\mathcal{D}_{\text{stest}} = \mathcal{D}_{\text{ctest}}(\gamma, \mathbf{x}_-, \mathbf{x}_+, \mathbf{x}_{\text{sp}}, \sigma_{\text{inv}}, \sigma_{\text{sp}})$ as follows.

Definition 1. *Let $\Sigma := \sigma_{inv}^2 I_{inv} + \sigma_{sp}^2 I_{sp}$ where I_{inv} and I_{sp} are identity matrix for $\mathcal{X}_{inv}$ and $\mathcal{X}_{sp}$ respectively.*

- *$\mathcal{D}_{train}(\gamma, \mathbf{x}_-, \mathbf{x}_+, \mathbf{x}_{sp}, \sigma_{inv}, \sigma_{sp})$ samples $(\mathbf{x}, y)$ using the following process:*
  - *with probability $1/2$, $\mathbf{x} \sim \mathcal{N}((\mathbf{x}_-, 0), \Sigma)$ and $y = -1$;*
  - *with probability $(1-\gamma)/2$, $\mathbf{x} \sim \mathcal{N}((\mathbf{x}_+, 0), \Sigma)$ and $y = +1$;*
  - *with probability $\gamma/2$, $\mathbf{x} \sim \mathcal{N}((\mathbf{x}_+, \mathbf{x}_{sp}), \Sigma)$ and $y = +1$.*
- *$\mathcal{D}_{ctest}(\gamma, \mathbf{x}_-, \mathbf{x}_+, \mathbf{x}_{sp}, \sigma_{inv}, \sigma_{sp})$ samples $(\mathbf{x}, y)$ using the following process:*
  - *with probability $1/2$, $\mathbf{x} \sim \mathcal{N}((\mathbf{x}_-, 0), \Sigma)$ and $y = -1$;*
  - *with probability $1/2$, $\mathbf{x} \sim \mathcal{N}((\mathbf{x}_+, 0), \Sigma)$ and $y = +1$.*
- *$\mathcal{D}_{stest}(\gamma, \mathbf{x}_-, \mathbf{x}_+, \mathbf{x}_{sp}, \sigma_{inv}, \sigma_{sp})$ samples $(\mathbf{x}, y)$ with $\mathbf{x} \sim \mathcal{N}((\mathbf{x}_-, \mathbf{x}_{sp}), \Sigma)$ and $y = -1$.*

Now, we can instantiate our model into examples that capture the experimental scenarios discussed in previous sections (see Fig. 8). For examples, the patterns *small 1* (*S1*), *small 2* (*S2*), and *small 3* (*S3*) correspond to setting $\sigma_{\text{sp}} = 0$ and increasing the strength of spurious features (i.e., increasing $\|\mathbf{x}_{\text{sp}}\|_2$ from (*S1*) to (*S3*)). Also, see Fig. 17 for a pictorial explanation of our model.



(a) $\mathcal{D}_{\text{train}}, \gamma = 0$. (b) $\mathcal{D}_{\text{train}}, \gamma > 0$. (c) $\mathcal{D}_{\text{ctest}}$. (d) $\mathcal{D}_{\text{stest}}$.

Figure 17. Examples of the training distribution $\mathcal{D}_{\text{train}}$ and phase transitions in our theoretical model. (a)-(b) With equal probability a training example is sampled from either $\mathcal{N}(\mathbf{x}_+, \sigma_{\text{inv}} I_{\text{inv}} + \sigma_{\text{sp}} I_{\text{sp}})$ or $\mathcal{N}(\mathbf{x}_-, \sigma_{\text{inv}} I_{\text{inv}} + \sigma_{\text{sp}} I_{\text{sp}})$. With probability γ, a $+$ sample will be concatenated with the spurious pattern $\mathbf{x}_{\text{sp}}$. The dotted line is the decision boundary of the optimal classifier. (c) The clean test distribution $\mathcal{D}_{\text{ctest}}$, which is the same as $\mathcal{D}_{\text{train}}$ with $\gamma = 0$. (d) The spurious test distribution $\mathcal{D}_{\text{stest}}$, which only contains $-$ samples with spurious pattern $\mathbf{x}_{\text{sp}}$ added.

F.3. Analysis for linear regression with mean square loss and ℓ_2 regularization

As the theoretical toolkit for understanding neural networks is far from complete, we examine the theoretical aspect of rare spurious correlations through the lens of a classic learning algorithm: linear regression with mean square loss. Notice that we **do not** claim that the analysis here generalize to neural networks. Instead, the analysis for linear regression serves as food for thoughts for future investigation.

[8] Concretely, let $\mathbf{x}'_+ = (\mathbf{x}_+ - \mathbf{x}_-)/2$ and $\mathbf{x}'_- = -(\mathbf{x}_+ - \mathbf{x}_-)/2$ where $\mathbf{x}'_+$ and $\mathbf{x}'_-$ are the new invariant features.

For linear regression with mean square loss, the hypothesis set is $\mathbb{H} = \{h = (\boldsymbol{\beta}_{\text{inv}}, \boldsymbol{\beta}_{\text{sp}}, \beta_0) : \boldsymbol{\beta}_{\text{inv}} \in \mathcal{X}_{\text{inv}}, \boldsymbol{\beta}_{\text{sp}} \in \mathcal{X}_{\text{sp}}, \beta_0 \in \mathbb{R}\}$ where $h(\mathbf{x}) = \boldsymbol{\beta}_{\text{inv}}^\top \mathbf{x}_{\text{inv}} + \boldsymbol{\beta}_{\text{sp}}^\top \mathbf{x}_{\text{sp}} + \beta_0$. The ℓ_2 loss function of h on a distribution $\mathcal{D}$ is denotes as $L_{\mathcal{D}}(h) := \frac{1}{2}\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[(h(\mathbf{x}) \cdot y - 1)^2]$. If we consider adding ℓ_2 regularization with parameter λ, the loss function becomes $L_{\mathcal{D}}(h) := \frac{1}{2}\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[(h(\mathbf{x}) \cdot y - 1)^2] + \frac{\lambda}{2}\|\boldsymbol{\beta}_{\text{inv}}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{\beta}_{\text{sp}}\|_2^2$. In the rest of this section, we focus on the setting with ℓ_2 regularization since the unregularized setting can be obtained by setting $\lambda = 0$.

The optimal classifier in $\mathbb{H}$ for $\mathcal{D}_{\text{train}}$ is defined to be $h^* := \arg\min_{h\in\mathbb{H}} L_{\mathcal{D}_{\text{train}}}(h)$. By the theory of linear regression with mean square loss, we know that h^* can be found via standard Empirical Risk Minimization (ERM) principle. In our theoretical analysis, we study the performance of h^* while one can easily extend our analysis to the finite sample regime by applying the standard convergence analysis of ERM for linear regression with mean square loss.

There are three quantities of interest in our study of rare spurious correlations: clean test accuracy, spurious test accuracy, and spurious score. For the completeness of presentation, let us formally define them as follows.

Definition 2 (Test accuracy)**.** *Let h be a hypothesis function and $\mathcal{D}_{test}$ be a test distribution. The test accuracy of h on $\mathcal{D}_{test}$ is defined as*

$$Acc_{\mathcal{D}_{test}}(h) := \Pr_{(\mathbf{x},y)\sim\mathcal{D}_{test}}[h(\mathbf{x})y > 0].$$

Definition 3 (Prediction difference and Spurious score)**.** *Let h be a hypothesis function, $\mathcal{D}_{test}$ be a test distribution, $(\Phi_{\mathcal{X}}, \Phi_{\mathcal{Y}})$ be the feature-to-input maps, and $\mathbf{x}_{sp}$ be a spurious feature. The prediction difference for an invariant input $\mathbf{x} \in \mathcal{X}_{inv}$ is defined as*

$$PD(\mathbf{x}; h) := h(\Phi_{\mathcal{X}}(\mathbf{x}, \mathbf{x}_{sp})) - h(\Phi_{\mathcal{X}}(\mathbf{x}, 0)).$$

For every $\epsilon > 0$, the ϵ-spurious score of h with respect to $\mathcal{D}_{test}$ is defined as

$$\epsilon\text{-spurious-score}(h) = \Pr_{(\mathbf{x},y)\sim\mathcal{D}_{test}}[PD(\mathbf{x}, h) > \epsilon].$$

Note that for linear classifier in both the overlap and the concatenation model, the ϵ-spurious-score is always either 0 or 1 because $PD(\mathbf{x}; h) = h(\mathbf{x}_{\text{sp}})$ for every $\mathbf{x} \in \mathcal{X}_{\text{inv}}$. Thus, for a linear classifier h, we denote $PD(h) = h(\mathbf{x}_{\text{sp}})$ for simplicity. In the rest of the theoretical analysis, we study the prediction difference which tells us when does the spurious score jump from 0 to 1.

Now, we state our main theorem about the clean test accuracy, spurious test accuracy, and the prediction difference of h^*. We remark that the calculations for these quantities are straightforward and we postpone the proof to Appendix F.4.

Theorem 1. *For every $\gamma \in (0, 1]$, $\lambda > 0$, $\mathbf{x}_{inv} \in \mathcal{X}_{inv}$, $\mathbf{x}_{sp} \in \mathcal{X}_{sp}$, and $\sigma_{inv}^2, \sigma_{sp}^2 > 0$. Let h^* be the optimal classifier for $\mathcal{D}_{train} = \mathcal{D}_{train}(\gamma, -\mathbf{x}_{inv}, \mathbf{x}_{inv}, \mathbf{x}_{sp}, \sigma_{inv}^2, \sigma_{sp}^2)$ using linear regression with mean square loss and ℓ_2 regularization with parameter λ. We have*

$$Acc_{\mathcal{D}_{ctest}}(h^*) = \frac{1}{2}\Phi\left(\frac{\frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{sp}\|_2^2+\sigma_{sp}^2+\lambda}{\sigma_{inv}^2+\lambda}\|\mathbf{x}_{inv}\|_2^2 + (\frac{\gamma}{2})^2\|\mathbf{x}_{sp}\|_2^2}{\sqrt{\left(\frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{sp}\|_2^2+\sigma_{sp}^2+\lambda}{\sigma_{inv}^2+\lambda}\right)^2\|\mathbf{x}_{inv}\|_2^4\sigma_{inv}^2 + (\frac{\gamma}{2})^2\|\mathbf{x}_{sp}\|_2^4\sigma_{sp}^2}}\right) + \frac{1}{2}\Phi\left(\frac{\frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{sp}\|_2^2+\sigma_{sp}^2+\lambda}{\sigma_{inv}^2+\lambda}\|\mathbf{x}_{inv}\|_2^2 - (\frac{\gamma}{2})^2\|\mathbf{x}_{sp}\|_2^2}{\sqrt{\left(\frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{sp}\|_2^2+\sigma_{sp}^2+\lambda}{\sigma_{inv}^2+\lambda}\right)^2\|\mathbf{x}_{inv}\|_2^4\sigma_{inv}^2 + (\frac{\gamma}{2})^2\|\mathbf{x}_{sp}\|_2^4\sigma_{sp}^2}}\right)$$

and

$$Acc_{\mathcal{D}_{stest}}(h^*) = \Phi\left(\frac{\frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{sp}\|_2^2+\sigma_{sp}^2+\lambda}{\sigma_{inv}^2+\lambda}\|\mathbf{x}_{inv}\|_2^2 + (\frac{\gamma}{2}-1)\frac{\gamma}{2}\|\mathbf{x}_{sp}\|_2^2}{\sqrt{\left(\frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{sp}\|_2^2+\sigma_{sp}^2+\lambda}{\sigma_{inv}^2+\lambda}\right)^2\|\mathbf{x}_{inv}\|_2^4\sigma_{inv}^2 + (\frac{\gamma}{2})^2\|\mathbf{x}_{sp}\|_2^4\sigma_{sp}^2}}\right)$$

and

$$PD(h^*) = \frac{\frac{\gamma}{2}(\sigma_{inv}^2 + \lambda)\|\mathbf{x}_{sp}\|_2^2}{(\sigma_{inv}^2 + \lambda)(\sigma_{sp}^2 + \lambda) + \left(\sigma_{sp}^2 + \lambda + \frac{\gamma}{2}(1 - \frac{\gamma}{2})(\sigma_{inv}^2 + \lambda) + \frac{\gamma}{2}(1 - \gamma)\|\mathbf{x}_{sp}\|_2^2\right)\|\mathbf{x}_{sp}\|_2^2}$$

where $\Phi(\cdot)$ is the cumulative distribution function (CDF) of the standard Gaussian variable.

F.3.1. Implications of Theorem 1

In Appendix B.2 we present three observations given by Theorem 1 without explaining why. Here, we provide all the underlying reasoning.

Observation 1: A phase transition of spurious test accuracy and prediction difference at $\gamma = 0$. Note that by Theorem 1, when γ is close to 0, the spurious test accuracy and the prediction difference is approximately

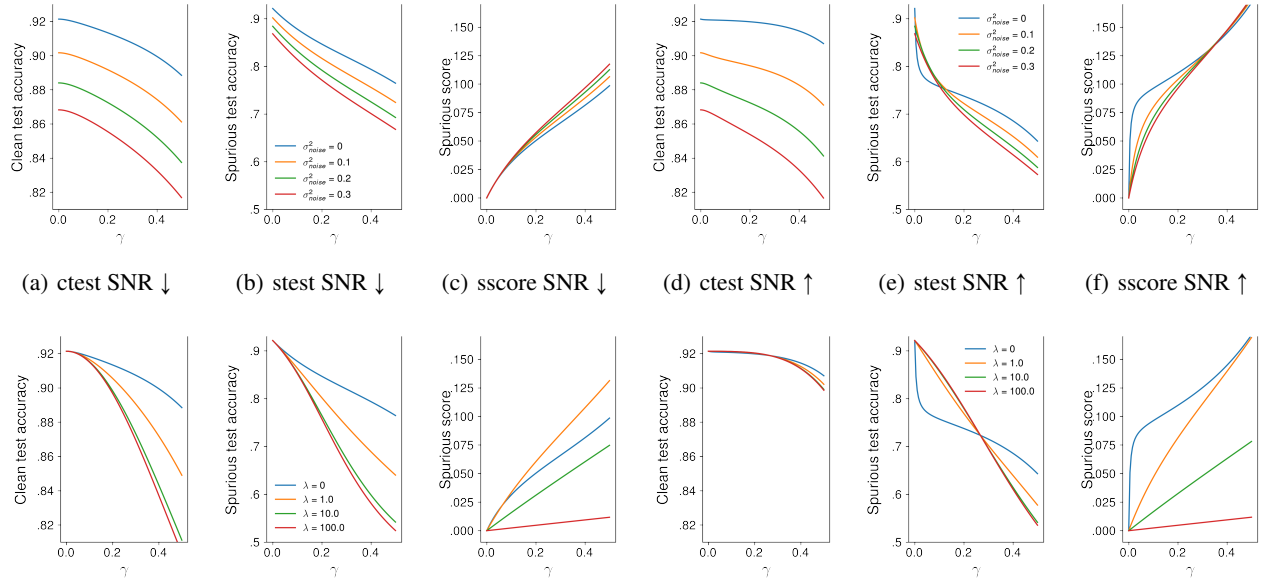
$$ACC_{\mathcal{D}_{\text{stest}}} = \Phi\left(1 - \frac{(\sigma_{\text{inv}}^2 + \lambda)\|\mathbf{x}_{\text{sp}}\|_2^2}{2(\sigma_{\text{sp}}^2 + \lambda)\|\mathbf{x}_{\text{inv}}\|_2^2 \sigma_{\text{inv}}}\gamma\right)$$

and

$$PD(h^*) = \frac{(\sigma_{\text{inv}}^2 + \lambda)\|\mathbf{x}_{\text{sp}}\|_2^2}{2(\sigma_{\text{sp}}^2 + \lambda)(\sigma_{\text{inv}}^2 + \lambda + \|\mathbf{x}_{\text{sp}}\|_2^2)}\gamma\,.$$

One can see that when $\|\mathbf{x}_{\text{sp}}\|_2^2/\sigma_{\text{sp}}^2$ is large, the spurious test accuracy (resp. prediction difference) undergoes a sharp decay (resp. growth) within an interval near $\gamma = 0$. In particular, the sharp decay/growth ends at $\gamma \approx \Theta(\min\{1, \frac{\sigma_{\text{sp}}^2+\lambda}{\|\mathbf{x}_{\text{sp}}\|_2^2}, \frac{\sigma_{\text{sp}}^2+\lambda}{\sigma_{\text{sp}}\|\mathbf{x}_{\text{sp}}\|_2^2(\sigma_{\text{inv}}^2+\lambda)}\})$[9]. As it is analytically hard to get a precise characterization of the critical point, we also numerically plot the spurious test accuracy and the prediction difference under different parameters to demonstrate the phase transition phenomenon.

Observation 2: adding Gaussian noises lowers spurious score. Note that adding (isotropic) Gaussian noises with strength σ^2 to input $\mathbf{x}$ is equivalent to increase the variance by an additive factor of σ^2, i.e., $\sigma_{\text{inv}}^2 \leftarrow \sigma_{\text{inv}}^2 + \sigma^2$ and $\sigma_{\text{sp}}^2 \leftarrow \sigma_{\text{sp}}^2 + \sigma^2$. Empirically we observe in Fig. 18 that both the clean test accuracy and spurious test accuracy decay as σ^2 becomes larger while the prediction difference decays favorably. Intuitively, adding Gaussian noises decrease the signal-to-noise ratio of both the invariant feature and the spurious feature.



(a) ctest SNR ↓ (b) stest SNR ↓ (c) sscore SNR ↓ (d) ctest SNR ↑ (e) stest SNR ↑ (f) sscore SNR ↑

(g) ctest SNR ↓ (h) stest SNR ↓ (i) sscore SNR ↓ (j) ctest SNR ↑ (k) stest SNR ↑ (l) sscore SNR ↑

Figure 18. Examples of the effect of adding Gaussian noises and ℓ_2 regularization under different regime of signal-to-noise ratio (SNR) $\|\mathbf{x}_{\text{sp}}\|_2/\sigma_{\text{sp}}^2$. For low SNR, we have $\|\mathbf{x}_{\text{sp}}\|_2/\sigma_{\text{sp}}^2 = 10$, and for high SNR, we have $\|\mathbf{x}_{\text{sp}}\|_2/\sigma_{\text{sp}}^2 = 1 = 500$. (a)-(f) show the effect of adding different strength of Gaussian noises and (g)-(l) show the effect of different ℓ_2 regularization strength. Here, we shorthand clean test accuracy as ctest, spurious test accuracy as stest and spurious score as sscore.

[9]The critical point here is an estimation by estimating when does the linear approximation at $\gamma = 0$ no longer dominates all the other terms

Observation 3: ℓ_2 regularization improves test accuracy and lowers spurious score. The main difference of ℓ_2 regularization from adding Gaussian noises is that it non-uniformly increasing the effective variance of the invariant subspace and that of the spurious subspace. This can be seen from the formulas in Theorem 1 that the regularization parameter λ is only added to some but not all of the σ_{inv}^2 and σ_{sp}^2. Numerically, we also plot the test accuracy and the prediction difference under different parameters to demonstrate the effect of ℓ_2 regularization in Fig. 18.

F.4. Proof of Theorem 1

Proof. For notational simplicity, in this proof we will overload the notations $\mathbf{x}_{\text{inv}} = (\mathbf{x}_{\text{inv}}, 0) \in \mathcal{X}$, $\mathbf{x}_{\text{sp}} = (0, \mathbf{x}_{\text{sp}}) \in \mathcal{X}$, $\mathbf{x}_{\text{inv}} = (\mathbf{x}_{\text{inv}}, 0) \in \mathcal{X}$ and $-\mathbf{x}_{\text{inv}} = (-\mathbf{x}_{\text{inv}}, 0) \in \mathcal{X}$. Also, we denote $\beta = (\boldsymbol{\beta}_{\text{inv}}, \boldsymbol{\beta}_{\text{sp}})$.

Let us start with finding h^* via explicitly calculating the derivatives of loss function under $\mathcal{D}_{\text{train}}$ as follows.

Claim 1. *For every $h = (\boldsymbol{\beta}_{inv}, \boldsymbol{\beta}_{sp}, \beta_0) \in \mathbb{H}$, we have*

$$\begin{aligned}
\frac{\partial}{\partial \beta} L_{\mathcal{D}_{train}}(h) &= (\sigma_{inv}^2 + \lambda)\boldsymbol{\beta}_{inv} + \left(\mathbf{x}_{inv}^\top \boldsymbol{\beta}_{inv} + \frac{\gamma \mathbf{x}_{sp}^\top \boldsymbol{\beta}_{sp}}{2} - 1\right)\mathbf{x}_{inv} \\
&\quad + (\sigma_{sp}^2 + \lambda)\boldsymbol{\beta}_{sp} + \frac{\gamma}{2}\left(\mathbf{x}_{inv}^\top \boldsymbol{\beta}_{inv} + \mathbf{x}_{sp}^\top \boldsymbol{\beta}_{sp} + \beta_0 - 1\right)\mathbf{x}_{sp}\,, \\
\frac{\partial}{\partial \beta_0} L_{\mathcal{D}_{train}}(h) &= \frac{\gamma \mathbf{x}_{sp}^\top \boldsymbol{\beta}_{sp}}{2} + \beta_0\,.
\end{aligned}$$

The proof of Claim 1 follows from a direct calculation and we postpone the details to Section F.4.1.

As the loss function is a quadratic function, it suffices to solve the equations $\frac{\partial}{\partial \beta} L_{\mathcal{D}_{\text{train}}}(h) = 0$ and $\frac{\partial}{\partial \beta_0} L_{\mathcal{D}_{\text{train}}}(h) = 0$. Note that since the invariant subspace $\mathcal{X}_{\text{inv}}$ is orthogonal to the spurious subspace $\mathcal{X}_{\text{sp}}$, this enforces $\boldsymbol{\beta}_{\text{inv}}$ (resp. $\boldsymbol{\beta}_{\text{sp}}$) to be a scalar multiplication of $\mathbf{x}_{\text{inv}}$ (resp. $\mathbf{x}_{\text{sp}}$). Let us set $\boldsymbol{\beta}_{\text{inv}} = a\mathbf{x}_{\text{inv}}$ and $\boldsymbol{\beta}_{\text{sp}} = b\mathbf{x}_{\text{sp}}$ and solve a, b by setting the partial derivatives to be zero. Namely, we get the following equations.

$$\begin{aligned}
0 &= a(\sigma_{\text{inv}}^2 + \lambda) + a\|\mathbf{x}_{\text{inv}}\|_2^2 + \frac{\gamma\|\mathbf{x}_{\text{sp}}\|_2^2}{2} b - 1\,, \\
0 &= b(\sigma_{\text{sp}}^2 + \lambda) + \frac{\gamma}{2}(a\|\mathbf{x}_{\text{inv}}\|_2^2 + b\|\mathbf{x}_{\text{sp}}\|_2^2 + \beta_0 - 1)\,, \\
0 &= \frac{\gamma\|\mathbf{x}_{\text{sp}}\|_2^2}{2} b + \beta_0\,.
\end{aligned}$$

By solving the above system of linear equations, we get

$$b = \frac{\frac{\gamma}{2}(\sigma_{\text{inv}}^2 + \lambda)}{(\sigma_{\text{inv}}^2 + \lambda)(\sigma_{\text{sp}}^2 + \lambda) + \left(\sigma_{\text{sp}}^2 + \lambda + \frac{\gamma}{2}(1 - \frac{\gamma}{2})(\sigma_{\text{inv}}^2 + \lambda) + \frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{\text{sp}}\|_2^2\right)\|\mathbf{x}_{\text{sp}}\|_2^2}\,, \tag{2}$$

$$\frac{a}{b} = \frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{\text{sp}}\|_2^2 + \sigma_{\text{sp}}^2 + \lambda}{\frac{\gamma}{2}(\sigma_{\text{inv}}^2 + \lambda)}\,, \tag{3}$$

$$\frac{\beta_0}{b} = -\frac{\gamma\|\mathbf{x}_{\text{sp}}\|_2^2}{2}\,. \tag{4}$$

Now that we know what h^* is, we can calculate the clean test accuracy as follows.

$$\begin{aligned}
ACC_{\mathcal{D}_{\text{ctest}}}(h^*) &= \Pr_{(\mathbf{x},y)\sim\mathcal{D}_{\text{ctest}}}[h^*(\mathbf{x})y > 0] \\
&= \frac{1}{2}\Pr_{\mathbf{x}\sim\mathcal{N}((-\mathbf{x}_{\text{inv}},0),\Sigma)}[h^*(\mathbf{x}) < 0] + \frac{1}{2}\Pr_{\mathbf{x}\sim\mathcal{N}((\mathbf{x}_{\text{inv}},0),\Sigma)}[h^*(\mathbf{x}) > 0] \\
&= \frac{1}{2}\Pr_{\mathbf{w}\sim\mathcal{N}(0,\Sigma)}[\beta^\top(\mathbf{w} - \mathbf{x}_{\text{inv}}) + \beta_0 < 0] + \frac{1}{2}\Pr_{\mathbf{w}\sim\mathcal{N}(0,\Sigma)}[\beta^\top(\mathbf{w} + \mathbf{x}_{\text{inv}}) + \beta_0 > 0]\,.
\end{aligned}$$

As $\mathcal{N}(0,\Sigma)$ is isotropic in both $\mathcal{X}_{\text{inv}}$ and $\mathcal{X}_{\text{sp}}$, the above becomes

$$
\begin{aligned}
&= \frac{1}{2}\Pr_{\mathbf{w}\sim\mathcal{N}(0,\Sigma)}[\beta^\top(\mathbf{w}-\mathbf{x}_{\text{inv}}) < -\beta_0] + \frac{1}{2}\Pr_{\mathbf{w}\sim\mathcal{N}(0,\Sigma)}[\beta^\top(\mathbf{w}-\mathbf{x}_{\text{inv}}) < \beta_0] \\
&= \frac{1}{2}\Pr_{\substack{\mathbf{w}_{\text{inv}}\sim\mathcal{N}(0,\sigma_{\text{inv}}^2 I_{\text{inv}})\\ \mathbf{w}_{\text{sp}}\sim\mathcal{N}(0,\sigma_{\text{sp}}^2 I_{\text{sp}})}}[a\mathbf{x}_{\text{inv}}^\top\mathbf{w}_{\text{inv}} + b\mathbf{x}_{\text{sp}}^\top\mathbf{w}_{\text{sp}} < a\|\mathbf{x}_{\text{inv}}\|_2^2 - \beta_0] \\
&+ \frac{1}{2}\Pr_{\substack{\mathbf{w}_{\text{inv}}\sim\mathcal{N}(0,\sigma_{\text{inv}}^2 I_{\text{inv}})\\ \mathbf{w}_{\text{sp}}\sim\mathcal{N}(0,\sigma_{\text{sp}}^2 I_{\text{sp}})}}[a\mathbf{x}_{\text{inv}}^\top\mathbf{w}_{\text{inv}} + b\mathbf{x}_{\text{sp}}^\top\mathbf{w}_{\text{sp}} < a\|\mathbf{x}_{\text{inv}}\|_2^2 + \beta_0]\,.
\end{aligned}
$$

Note that $a\mathbf{x}_{\text{inv}}^\top\mathbf{w}_{\text{inv}} + b\mathbf{x}_{\text{sp}}^\top\mathbf{w}_{\text{sp}}$ follows the distribution $\mathcal{N}(0,\sigma^2)$ where $\sigma^2 = a^2\|\mathbf{x}_{\text{inv}}\|_2^4\sigma_{\text{inv}}^2 + b^2\|\mathbf{x}_{\text{sp}}\|_2^4\sigma_{\text{sp}}^2$. Namely, the above can be further simplified as

$$
\begin{aligned}
&= \frac{1}{2}\Pr_{w\sim\mathcal{N}(0,\sigma^2)}[w < a\|\mathbf{x}_{\text{inv}}\|_2^2 + \beta_0] + \frac{1}{2}\Pr_{w\sim\mathcal{N}(0,\sigma^2)}[w < a\|\mathbf{x}_{\text{inv}}\|_2^2 - \beta_0] \\
&= \frac{1}{2}\Phi\left(\frac{a\|\mathbf{x}_{\text{inv}}\|_2^2 - \beta_0}{\sqrt{a^2\|\mathbf{x}_{\text{inv}}\|_2^4\sigma_{\text{inv}}^2 + b^2\|\mathbf{x}_{\text{sp}}\|_2^4\sigma_{\text{sp}}^2}}\right) + \frac{1}{2}\Phi\left(\frac{a\|\mathbf{x}_{\text{inv}}\|_2^2 + \beta_0}{\sqrt{a^2\|\mathbf{x}_{\text{inv}}\|_2^4\sigma_{\text{inv}}^2 + b^2\|\mathbf{x}_{\text{sp}}\|_2^4\sigma_{\text{sp}}^2}}\right).
\end{aligned}
$$

Lastly, by plugging in Eqs. (2) to (4), the clean test accuracy of h^* is

$$
\begin{aligned}
&= \frac{1}{2}\Phi\left(\frac{\frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{\text{sp}}\|_2^2+\sigma_{\text{sp}}^2+\lambda}{\sigma_{\text{inv}}^2+\lambda}\|\mathbf{x}_{\text{inv}}\|_2^2 + (\frac{\gamma}{2})^2\|\mathbf{x}_{\text{sp}}\|_2^2}{\sqrt{\left(\frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{\text{sp}}\|_2^2+\sigma_{\text{sp}}^2+\lambda}{\sigma_{\text{inv}}^2+\lambda}\right)^2\|\mathbf{x}_{\text{inv}}\|_2^4\sigma_{\text{inv}}^2 + (\frac{\gamma}{2})^2\|\mathbf{x}_{\text{sp}}\|_2^4\sigma_{\text{sp}}^2}}\right) \\
&+ \frac{1}{2}\Phi\left(\frac{\frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{\text{sp}}\|_2^2+\sigma_{\text{sp}}^2+\lambda}{\sigma_{\text{inv}}^2+\lambda}\|\mathbf{x}_{\text{inv}}\|_2^2 - (\frac{\gamma}{2})^2\|\mathbf{x}_{\text{sp}}\|_2^2}{\sqrt{\left(\frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{\text{sp}}\|_2^2+\sigma_{\text{sp}}^2+\lambda}{\sigma_{\text{inv}}^2+\lambda}\right)^2\|\mathbf{x}_{\text{inv}}\|_2^4\sigma_{\text{inv}}^2 + (\frac{\gamma}{2})^2\|\mathbf{x}_{\text{sp}}\|_2^4\sigma_{\text{sp}}^2}}\right).
\end{aligned}
$$

Similarly, the spurious test accuracy of h^* is

$$
\begin{aligned}
Acc_{\mathcal{D}_{\text{stest}}}(h^*) &= \Pr_{(\mathbf{x},y)\sim\mathcal{D}_{\text{stest}}}[h^*(\mathbf{x})y > 0] \\
&= \Pr_{\mathbf{x}\sim\mathcal{N}((-\mathbf{x}_{\text{inv}},\mathbf{x}_{\text{sp}}),\Sigma)}[h^*(\mathbf{x}) < 0] \\
&= \Pr_{\substack{\mathbf{w}_{\text{inv}}\sim\mathcal{N}(0,\sigma_{\text{inv}} I_{\text{inv}})\\ \mathbf{w}_{\text{sp}}\sim\mathcal{N}(0,\sigma_{\text{sp}} I_{\text{sp}})}}[a\mathbf{x}_{\text{inv}}^\top\mathbf{w}_{\text{inv}} + b\mathbf{x}_{\text{sp}}^\top\mathbf{w}_{\text{sp}} < a\|\mathbf{x}_{\text{inv}}\|_2^2 - b\|\mathbf{x}_{\text{sp}}\|_2^2 - \beta_0] \\
&= \Pr_{w\sim\mathcal{N}(0,a^2\|\mathbf{x}_{\text{inv}}\|_2^4\sigma_{\text{inv}}^2+b^2\|\mathbf{x}_{\text{sp}}\|_2^4\sigma_{\text{sp}}^2)}[w < a\|\mathbf{x}_{\text{inv}}\|_2^2 - b\|\mathbf{x}_{\text{sp}}\|_2^2 - \beta_0] \\
&= \Phi\left(\frac{a\|\mathbf{x}_{\text{inv}}\|_2^2 - b\|\mathbf{x}_{\text{sp}}\|_2^2 - \beta_0}{\sqrt{a^2\|\mathbf{x}_{\text{inv}}\|_2^4\sigma_{\text{inv}}^2 + b^2\|\mathbf{x}_{\text{sp}}\|_2^4\sigma_{\text{sp}}^2}}\right) \\
&= \Phi\left(\frac{\frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{\text{sp}}\|_2^2+\sigma_{\text{sp}}^2+\lambda}{\sigma_{\text{inv}}^2+\lambda}\|\mathbf{x}_{\text{inv}}\|_2^2 + (\frac{\gamma}{2}-1)\frac{\gamma}{2}\|\mathbf{x}_{\text{sp}}\|_2^2}{\sqrt{\left(\frac{\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{\text{sp}}\|_2^2+\sigma_{\text{sp}}^2+\lambda}{\sigma_{\text{inv}}^2+\lambda}\right)^2\|\mathbf{x}_{\text{inv}}\|_2^4\sigma_{\text{inv}}^2 + (\frac{\gamma}{2})^2\|\mathbf{x}_{\text{sp}}\|_2^4\sigma_{\text{sp}}^2}}\right).
\end{aligned}
$$

Finally, as h^* is a linear classifier, its prediction difference is

$$
\begin{aligned}
PD(h^*) &= h^*(\mathbf{x}_{\text{sp}}) = b\|\mathbf{x}_{\text{sp}}\|_2^2 \\
&= \frac{\frac{\gamma}{2}(\sigma_{\text{inv}}^2+\lambda)\|\mathbf{x}_{\text{sp}}\|_2^2}{(\sigma_{\text{inv}}^2+\lambda)(\sigma_{\text{sp}}^2+\lambda) + \left(\sigma_{\text{sp}}^2+\lambda+\frac{\gamma}{2}(1-\frac{\gamma}{2})(\sigma_{\text{inv}}^2+\lambda)+\frac{\gamma}{2}(1-\gamma)\|\mathbf{x}_{\text{sp}}\|_2^2\right)\|\mathbf{x}_{\text{sp}}\|_2^2}
\end{aligned}
$$

where the last equality follows Eq. (2). This completes the proof of Theorem 1. □

F.4.1. MOMENT CALCULATIONS

Proof of Claim 1. Recall that we denote $\beta = (\boldsymbol{\beta}_{\text{inv}}, \boldsymbol{\beta}_{\text{sp}})$ for convenience. For every $h = (\boldsymbol{\beta}_{\text{inv}}, \boldsymbol{\beta}_{\text{sp}}, \beta_0) \in \mathbb{H}$, we have

$$\begin{aligned} L_{\mathcal{D}_{\text{train}}}(h) &= \frac{1}{2} \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{train}}} [(h(\mathbf{x}) \cdot y - 1)^2] + \frac{\lambda}{2}\|\beta\|_2^2 \\ &= \frac{1}{2} \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{train}}} [y^2(\beta^\top \mathbf{x} + \beta_0)^2 - 2y(\beta^\top \mathbf{x} + \beta_0) + 1] + \frac{\lambda}{2}\|\beta\|_2^2 \,. \end{aligned}$$

Note that $y^2 = 1$ almost surely and we can calculate the partial derivatives of $L_{\mathcal{D}_{\text{train}}}(h)$ as follows.

$$\begin{aligned} \frac{\partial}{\partial \beta} L_{\mathcal{D}_{\text{train}}}(h) &= \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{train}}} [\mathbf{x}(\mathbf{x}^\top \beta + \beta_0 - y)] + \lambda \beta \,, \\ \frac{\partial}{\partial \beta_0} L_{\mathcal{D}_{\text{train}}}(h) &= \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{train}}} [\mathbf{x}^\top \beta + \beta_0 - y] \,. \end{aligned}$$

Let's start with calculating the first moment terms. Recall that we overload the notation of $\mathbf{x}_{\text{inv}}$ and $\mathbf{x}_{\text{sp}}$ as explained in the beginning of the proof of Theorem 1.

$$\begin{aligned} \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{train}}} [\mathbf{x}] &= \frac{1}{2} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{N}((-\mathbf{x}_{\text{inv}},0),\Sigma)} [\mathbf{x}] + \frac{1-\gamma}{2} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{N}((\mathbf{x}_{\text{inv}},0),\Sigma)} [\mathbf{x}] + \frac{\gamma}{2} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{N}((\mathbf{x}_{\text{inv}},\mathbf{x}_{\text{sp}}),\Sigma)} [\mathbf{x}] \\ &= \frac{-\mathbf{x}_{\text{inv}} + \mathbf{x}_{\text{inv}} + \gamma \mathbf{x}_{\text{sp}}}{2} \,, \\ \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{train}}} [y\mathbf{x}] &= \frac{1}{2} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{N}((-\mathbf{x}_{\text{inv}},0),\Sigma)} [y\mathbf{x}] + \frac{1-\gamma}{2} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{N}((\mathbf{x}_{\text{inv}},0),\Sigma)} [y\mathbf{x}] + \frac{\gamma}{2} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{N}((\mathbf{x}_{\text{inv}},\mathbf{x}_{\text{sp}}),\Sigma)} [y\mathbf{x}] \\ &= \frac{\mathbf{x}_{\text{inv}} + \mathbf{x}_{\text{inv}} + \gamma \mathbf{x}_{\text{sp}}}{2} \,, \\ \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{train}}} [y] &= \frac{1}{2} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{N}((-\mathbf{x}_{\text{inv}},0),\Sigma)} [y] + \frac{1-\gamma}{2} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{N}((\mathbf{x}_{\text{inv}},0),\Sigma)} [y] + \frac{\gamma}{2} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{N}((\mathbf{x}_{\text{inv}},\mathbf{x}_{\text{sp}}),\Sigma)} [y] \\ &= 0 \,. \end{aligned}$$

Next, let's calculate the second moment terms.

$$\begin{aligned} \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{train}}} [\mathbf{x}\mathbf{x}^\top] &= \frac{1}{2} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{N}((-\mathbf{x}_{\text{inv}},0),\Sigma)} [\mathbf{x}\mathbf{x}^\top] + \frac{1-\gamma}{2} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{N}((\mathbf{x}_{\text{inv}},0),\Sigma)} [\mathbf{x}\mathbf{x}^\top] + \frac{\gamma}{2} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{N}((\mathbf{x}_{\text{inv}},\mathbf{x}_{\text{sp}}),\Sigma)} [\mathbf{x}\mathbf{x}^\top] \\ &= \frac{1}{2}(\Sigma + (\mathbf{x}_{\text{inv}}\mathbf{x}_{\text{inv}}^\top)) + \frac{1-\gamma}{2}(\Sigma + (\mathbf{x}_{\text{inv}}\mathbf{x}_{\text{inv}}^\top)) + \frac{\gamma}{2}(\Sigma + ((\mathbf{x}_{\text{inv}} + \mathbf{x}_{\text{sp}})(\mathbf{x}_{\text{inv}} + \mathbf{x}_{\text{sp}})^\top)) \\ &= \sigma_{\text{inv}}^2 I_{\text{inv}} + \sigma_{\text{sp}}^2 I_{\text{sp}} + \frac{\mathbf{x}_{\text{inv}}\mathbf{x}_{\text{inv}}^\top + \mathbf{x}_{\text{inv}}\mathbf{x}_{\text{inv}}^\top + \gamma(\mathbf{x}_{\text{sp}}\mathbf{x}_{\text{inv}}^\top + \mathbf{x}_{\text{inv}}\mathbf{x}_{\text{sp}}^\top + \mathbf{x}_{\text{sp}}\mathbf{x}_{\text{sp}}^\top)}{2} \,. \end{aligned}$$

Finally, putting everything together we have

$$\begin{aligned} \frac{\partial}{\partial \beta} L_{\mathcal{D}_{\text{train}}}(h) &= \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{train}}} [\mathbf{x}(\mathbf{x}^\top \beta + \beta_0 - y)] + \lambda \beta \\ &= \sigma_{\text{inv}}^2 \boldsymbol{\beta}_{\text{inv}} + \sigma_{\text{sp}}^2 \boldsymbol{\beta}_{\text{sp}} + \frac{\mathbf{x}_{\text{inv}}^\top \boldsymbol{\beta}_{\text{inv}}}{2} \mathbf{x}_{\text{inv}} + \frac{\mathbf{x}_{\text{inv}}^\top \boldsymbol{\beta}_{\text{inv}} + \gamma \mathbf{x}_{\text{sp}}^\top \boldsymbol{\beta}_{\text{sp}}}{2} \mathbf{x}_{\text{inv}} + \frac{\gamma \mathbf{x}_{\text{inv}}^\top \boldsymbol{\beta}_{\text{inv}} + \gamma \mathbf{x}_{\text{sp}}^\top \boldsymbol{\beta}_{\text{sp}}}{2} \mathbf{x}_{\text{sp}} \\ &+ \beta_0 \cdot \frac{-\mathbf{x}_{\text{inv}} + \mathbf{x}_{\text{inv}} + \gamma \mathbf{x}_{\text{sp}}}{2} - \frac{\mathbf{x}_{\text{inv}} + \mathbf{x}_{\text{inv}} + \gamma \mathbf{x}_{\text{sp}}}{2} + \lambda \boldsymbol{\beta}_{\text{inv}} + \lambda \boldsymbol{\beta}_{\text{sp}} \\ &= (\sigma_{\text{inv}}^2 + \lambda)\boldsymbol{\beta}_{\text{inv}} + \left(\mathbf{x}_{\text{inv}}^\top \boldsymbol{\beta}_{\text{inv}} + \frac{\gamma \mathbf{x}_{\text{sp}}^\top \boldsymbol{\beta}_{\text{sp}}}{2} - 1 \right) \mathbf{x}_{\text{inv}} \\ &+ (\sigma_{\text{sp}}^2 + \lambda)\boldsymbol{\beta}_{\text{sp}} + \frac{\gamma}{2} \left(\mathbf{x}_{\text{inv}}^\top \boldsymbol{\beta}_{\text{inv}} + \mathbf{x}_{\text{sp}}^\top \boldsymbol{\beta}_{\text{sp}} + \beta_0 - 1 \right) \mathbf{x}_{\text{sp}} \,. \end{aligned}$$

Similarly,

$$\frac{\partial}{\partial \beta_0} L_{\mathcal{D}_{\text{train}}}(h) = \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{train}}} [\mathbf{x}^\top \beta + \beta_0 - y] = \frac{\gamma \mathbf{x}_{\text{sp}}^\top \boldsymbol{\beta}_{\text{sp}}}{2} + \beta_0 \,.$$

This completes the proof of Claim 1. □

G. Can Rare Spurious Correlations be Removed through Data Deletion Methods?

There has been a growing body of recent work on data deletion methods (Koh et al., 2019; Izzo et al., 2021). Privacy laws such as the GDPR allow individuals to request an entity to remove their data, which includes removing it from any trained machine learning model. Since retraining models from scratch may be computationally expensive, a body of work has looked into developing more efficient methods. Here, we will look at two simple and canonical methods – incremental retraining and group influence (Koh et al., 2019; Basu et al., 2020b) that approximate the model that is trained without the deleted data points. Incremental retraining continues the training process for a number of epochs on the training data minus the deleted data, which effectively down weights the deleted data point in training. The group influence function computes a first-order approximation to the model that is trained without the deleted data point, motivated by influence functions from robust statistics. Both methods apply when multiple data points are deleted.

If all examples with a particular spurious pattern were deleted from the training set, then the spurious pattern and the target class should not be correlated in the resulting network. Therefore, we expect that a good data deletion method, when given a trained network and all training examples with a specific spurious pattern, should remove the associated spurious correlation from the network. In this section, we next investigate whether this is indeed the case.

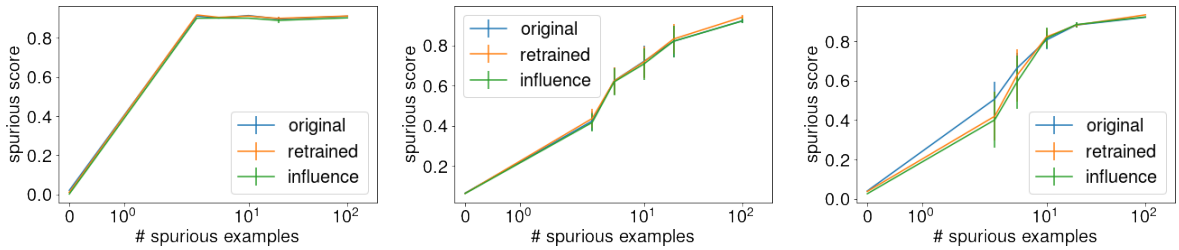
Setup. We follow the same setup as in Fig. 2. We fix the spurious pattern to be *R3*, which is the pattern that gives the strongest correlation. We train the networks with 1, 3, 5, 10, 20, and 100 spurious examples. We apply two data deletion methods, incremental retraining and group influence, to the trained network. Each method takes in the trained network and the spurious training examples, and generates a new network that approximates the network that is trained on a training set without the spurious examples. We then measure the spurious scores for three types of models – the model before data deletion, model processed with incremental retraining, and model processed with group influence.

For incremental retraining, we continue the retraining process for 70 epochs on the data minus the spurious examples (recall that the original models were also trained for 70 epochs). For group influence, we adapt a publicly available implementation for data deletion[10].

Results. Fig. 19 shows the results. We see that for all three datasets, the models processed by the data deletion methods have similar spurious scores as the models before deletion. This implies that the spurious correlations remain even after "data deletion", suggesting that *these data deletion methods may not be effective at properly removing spurious examples.* This has two implications – first, that rare spurious correlations, once introduced, may be challenging to remove. A second implication is that some data deletion methods may not properly remove all traces of the deleted data. We suggest that as a sanity check, future data deletion methods should test whether rare spurious correlations corresponding to the deleted examples are removed.

Finally, we note that there is a class of indistinguishable data deletion algorithms (Ginart et al., 2019; Neel et al., 2020) that provably ensure by adding noise that the deleted model is statistically indistinguishable from full retraining on the training data after deletion. However, these algorithms mostly apply to simpler problems, and we do not have efficient guaranteed deletion for non-convex problems such as training neural networks.

[10] public repository: https://github.com/ryokamoi/pytorch_influence_functions



(a) MNIST (b) Fashion (c) CIFAR10

Figure 19. The mean and standard error of spurious scores of the original models, models after incremental retraining, and models after the group influence method. The choice of spurious pattern is *R3*, $c_{tar} = 0$, and the optimizer is Adam. The lines for original and retrained are jittered by a small amount so that they are not completely overlapped.