Pattern Recognition Letters 166 (2023) 164–171

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

A too-good-to-be-true prior to reduce shortcut reliance☆

Nikolay Dagaev[a,b], Brett D. Roads[c], Xiaoliang Luo[c], Daniel N. Barry[c], Kaustubh R. Patil[d,e], Bradley C. Love[c,f,*]

[a] School of Psychology, HSE University, Moscow, Russia
[b] Department of Computer Science, University College London, London, United Kingdom
[c] Department of Experimental Psychology, University College London, London, United Kingdom
[d] Institute of Neuroscience and Medicine, Brain and Behaviour (INM-7), Research Center Jülich, Jülich, Germany
[e] Institute of Systems Neuroscience, Medical Faculty, Heinrich Heine University Düsseldorf, Düsseldorf, Germany
[f] The Alan Turing Institute, London, United Kingdom

ARTICLE INFO

Article history:
Received 19 September 2021
Revised 27 May 2022
Accepted 19 December 2022
Available online 21 December 2022

Edited by Prof. S. Sarkar

Keywords:
Shortcut learning
Out-of-distribution generalization
Robustness
Deep learning

ABSTRACT

Despite their impressive performance in object recognition and other tasks under standard testing conditions, deep networks often fail to generalize to out-of-distribution (o.o.d.) samples. One cause for this shortcoming is that modern architectures tend to rely on æshortcutsg superficial features that correlate with categories without capturing deeper invariants that hold across contexts. Real-world concepts often possess a complex structure that can vary superficially across contexts, which can make the most intuitive and promising solutions in one context not generalize to others. One potential way to improve o.o.d. generalization is to assume simple solutions are unlikely to be valid across contexts and avoid them, which we refer to as the *too-good-to-be-true prior*. A low-capacity network (LCN) with a shallow architecture should only be able to learn surface relationships, including shortcuts. We find that LCNs can serve as shortcut detectors. Furthermore, an LCN's predictions can be used in a two-stage approach to encourage a high-capacity network (HCN) to rely on deeper invariant features that should generalize broadly. In particular, items that the LCN can master are downweighted when training the HCN. Using a modified version of the CIFAR-10 dataset in which we introduced shortcuts, we found that the two-stage LCN-HCN approach reduced reliance on shortcuts and facilitated o.o.d. generalization.

1. Introduction

"If you would only recognize that life is hard, things would be so much easier for you."—Louis D. Brandeis

Deep convolutional neural networks (DCNNs) have achieved notable success in image recognition, sometimes achieving human-level performance or even surpassing it [16]. However, DCNNs often suffer when out-of-distribution (o.o.d.) generalization is needed, that is, when training and test data are drawn from different distributions [2,9,10]. This limitation has multiple consequences, such as susceptibility to adversarial interventions [13,19,39] or to previously unseen types of noise [10,18].

Failure to generalize o.o.d. may reflect the tendency of modern network architectures to discover simple features, so-called "shortcut" features [9,35]. While the perils of overly-complex solutions are well appreciated, **overly-simplistic solutions should be viewed with equal skepticism**. In this work, we assume that features that are easy to learn are likely too good to be true. For instance, a green background may be highly correlated with the "horse" category, but green grass is not a central feature. A horse detector relying on such simplistic features–i.e. shortcuts–may perform well when applied in Spain—where the training set originates—but will fail when deployed in snow-covered Siberia. In effect, shortcuts are easily discovered by a network but may be inappropriate for classifying items in an independent set where superficial features are distributed differently than in the training set. Thus, the sensitivity to shortcuts may have far-reaching and dangerous consequences in applications, like when the pneumonia predictions of a system were based on a metal token placed in radiographs [42].

In general, one cannot *a priori* know whether shortcuts will be helpful or misleading, nor can shortcut learning be reduced to overfitting the training data. While overfitting can be estimated using an available test set from the same distribution, assessing shortcuts depends on all possible unseen data. A model relying on shortcuts can show remarkable human-level results on test sets

☆ Preprint. Under consideration at Pattern Recognition Letters.

* Corresponding author.
E-mail addresses: ndagaev@hse.ru (N. Dagaev), b.love@ucl.ac.uk (B.C. Love).

https://doi.org/10.1016/j.patrec.2022.12.010

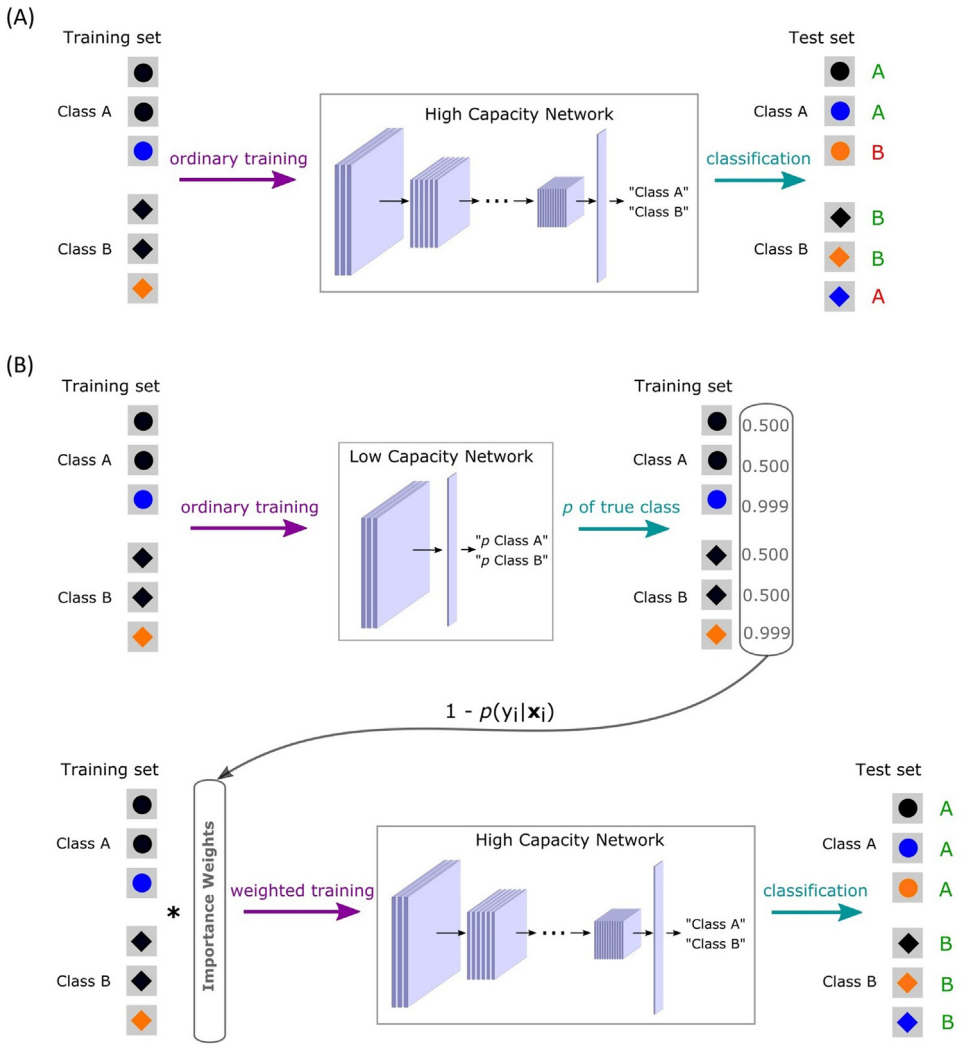

Fig. 1. The standard and too-good-to-be-true prior approaches to learning. (A) In the standard approach, a single high-capacity network (HCN) is trained and is susceptible to shortcuts, in this case relying on color as opposed to shape. Such a network will generalize well to i.i.d. test items but fail on o.o.d. test items (the last item for each class; shown in red). (B) In contrast, implementing the too-good-to-be true prior by pairing a low-capacity network (LCN) with an HCN leads to successful i.i.d. and o.o.d. generalization. Items that the LCN can master, which may contain shortcuts, are downweighted when the HCN is trained, which should reduce shortcut reliance and promote use of more complex and invariant features by the HCN. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

where shortcut features are distributed identically to the training set (i.i.d.), but fail dramatically on o.o.d. test sets where shortcuts are missing or misleading [32].

Shortcuts can adversely affect generalization even when they are not perfectly predictive. Because shortcuts are easily learned by DCNNs, they can be misleading even in the presence of more reliable but complex features [20]. To illustrate, shape may be perfectly predictive of class membership but networks may rely on color or other easily accessed features like texture [5,11] when tested on novel cases (see Fig. 1A).

Although what is and is not a shortcut cannot be known with perfect confidence, all shortcuts are simple. We find it unlikely that difficult learning problems will have trivial solutions when they have not been fully solved by brains with billions of neurons shaped by millions of years of natural selection nor by engineers working diligently for decades. Based on this observation, we are skeptical of very simple solutions to complex problems and believe they will have poor o.o.d. generalization. This inductive bias, which we refer to as the “too-good-to-be-true prior”, can be incorporated into the training of DCNNs to reduce shortcut reliance and promote o.o.d. generalization. At its heart, the too-good-to-be true prior is a belief about the relationship between the world, models, and machine learning problems, which places limits on Occam’s razor. Several recent contributions on o.o.d. generalization are consistent with the too-good-to-be-true prior [8,31,34]. In various ways, these authors suggest that simple solutions should be treated with caution and avoided.

How does one identify solutions that are probably too-good-to-be-true? Here we suggest to make use of a learning system wittingly simplistic for the problem at hand and capable of only trivial solutions, which includes shortcuts. First, we show that such low-capacity systems can be used to detect shortcuts in a dataset. Second, we suggest a simple and general method aimed at discarding training examples that are suspected of containing shortcuts. We hypothesize that, in order to prevent shortcut learning by a high-capacity network (HCN), the predictions of a much simpler, low-capacity network (LCN) could be used to guide the training of the target network. Namely, a trained LCN would provide high-probability predictions precisely for the training items containing these shortcuts. Such probabilistic predictions can be trans-

Table 1
Overview of approaches to preventing shortcut reliance most relevant to the present study. Task abbreviations: IR - image recognition, AR - action recognition, NLI - natural language inference, QA - question answering, VQA - visual question answering.

Approach	Simple solutions are shortcuts	Requires knowing a shortcut	Task
Reduce texture bias [11]	No	Yes	IR
DRiFt [15]	Yes	Yes	NLI
Don't take the easy...[7]	Yes	Yes	QA, VQA, NLI
REPAIR [26]	No	No	IR, AR
Learning not to learn [23]	No	Yes	IR
RUBi [6]	No	Yes	VQA
ReBias [1]	No	Yes	IR, AR
LfF [31]	Yes	No	IR
DIBS [38]	No	No	IR
Learning from others...[34]	Yes	No	QA, NLI
MCE [8]	Yes	No	IR, VQA, NLI
Our approach	Yes	No	IR

formed into importance weights (IWs) for training items, and these IWs can be further used in a loss function for training an HCN by **downweighting the shortcut items** (Fig. 1B). We demonstrate our method's efficiency by applying it to all possible CIFAR-10-based binary classification problems with synthetic shortcuts, permitting well-controlled experiments.

2. Related work

Shortcut learning and robust generalization. Multiple approaches have been suggested for preventing shortcut reliance and increasing generalization robustness in deep neural networks [9]. We succinctly summarize eleven studies which we find most relevant to our work in Table 1, using two criteria: whether an approach (1) assumes that simple solutions are probably shortcuts and (2) requires *a priori* knowledge of the shortcut. We also specify the task domains considered.

In contrast to Nam et al. [31], we use an LCN, not a full-capacity target model, to identify shortcuts and train the LCN separately from the target HCN. In comparison to Clark et al. [8], our approach is less demanding computationally and, again, the LCN and HCN are trained separately. We demonstrate that, in our particular implementation of the too-good-to-be-true prior, the limited capacity of a secondary model plays a key role, thus complementing the results of Sanh et al. [34]. We also extend the findings of Sanh et al., who introduced a similar de-biasing approach in the language domain, to the domain of image recognition. In contrast to all of the aforementioned studies, we show that an LCN can be employed to detect the presence of shortcuts in a dataset. Further, we empirically examine the relationship between the difficulty of a classification problem and the effectiveness of shortcut-avoiding training via our two-stage LCN-HCN procedure.

Huang et al. [21] suggested a heuristic, Representation Self-Challenging (RSC), to improve o.o.d. generalization in image recognition. This method impedes predicting class from features most correlated with it and thus encourages a DCNN to rely on more complex combinations of features. RSC, however, is not directly designed to prevent shortcut learning but rather attempts to expand the set of features learned.

Sample weighting. Re-weighting of data samples is a well-known approach to guiding the training of DCNNs and machine learning models in general, and corresponding methods differ in terms of which examples must be downweighted/emphasized. Some authors suggested to mitigate the impact of easy examples and focus on hard ones [28,36]. In contrast, in other research directions, such as curriculum learning [3,14,41] and self-paced learning [25,29], it is recommended to stress easy examples early in training. It was also shown that the self-paced and curriculum learning can be combined [22].

Table 2
LCN and HCN mean accuracies (in %; averaged across 10 independent runs) on colored MNIST; standard deviations are in parentheses.

Architecture	Regular training		Colored training	
	Regular test	Colored test	Colored test	Regular test
LCN	90.80 (1.23)	89.91 (1.50)	97.04 (0.71)	61.73 (8.21)
HCN	99.62 (0.10)	96.91 (1.74)	99.90 (0.11)	38.41 (12.82)

Although in our two-stage LCN-HCN procedure we assign weights to the training items, this method is fundamentally different from typical re-weighting schemes. Stemming from the too-good-to-be-true prior, our approach exploits not the predictions of the target network itself but of an independent simpler network (LCN). In other words, we are not interested in the difficulty of an item per se, but in whether this item can be mastered through simple means.

3. Example applications of the too-good-to-be-true prior

Below, in the context of image recognition tasks, we illustrate the too-good-to-be-true prior with two example applications: (1) detecting the presence of a shortcut in a dataset and (2) training a de-biased model. Both examples rely on an LCN being limited to learning a superficial shortcut as opposed to a deeper invariant representation.

3.1. The performance of a low-capacity network as an early warning signal

Considering that an LCN is only able to discover simple, probably shortcut, solutions, its high performance on a dataset may indicate the presence of a shortcut. When an LCN achieves a performance level comparable to an HCN, this should serve as a warning signal that the HCN may have succumbed to a shortcut [20]. In such cases, the HCN will likely fail to generalize robustly. Here we present two illustrative examples of such an application of the too-good-to-be-true prior.

3.1.1. Color as a shortcut

We trained an LCN (softmax regression; [4]) and an HCN (ResNet-56; [17]) to classify the colored MNIST dataset. The latter was implemented exactly as in [26], with the standard deviation of color set to 0.1. Both networks were trained with stochastic gradient descent for 50 epochs (learning rate was set to 0.1 and mini-batch size was set to 256).

Results are provided in Table 2. The presence of a shortcut indeed impairs o.o.d. generalization: both the LCN and HCN, after being trained on the colored data (shortcut is present), reveal poor

Table 3
LCN and HCN mean accuracies (in %; averaged across 10 independent runs) on stylized Tiny ImageNet; standard deviations are in parentheses.

Architecture	Regular training		Stylized training	
	Regular test	Stylized test	Stylized test	Regular test
LCN	8.54 (0.22)	1.30 (0.11)	70.10 (0.30)	0.70 (0.11)
HCN	41.43 (1.80)	2.21 (0.31)	88.0 (4.94)	0.50 (0.10)

performance on the regular test data (no shortcut). The poor o.o.d. generalization is particularly pronounced for the HCN - the difference between colored (99.90%) and regular (38.41%) test accuracies is 61.49%. The difference in performance between the HCN and LCN after training and testing on the colored data is only 2.86%, which serves as a warning that there may be a shortcut present.

3.1.2. Texture as a shortcut

We constructed a stylized version of Tiny ImageNet [40] by following a generalization[1] of the procedure introduced by Geirhos and colleagues [11]. Each of 200 classes was assigned its unique style and thus had a prominent texture shortcut.

The LCN was represented by a single 4-channel convolutional layer (3-by-3 kernels, linear activation function, no downsampling) followed by a fully-connected softmax classification layer. The HCN was represented by a 10-layer ResNet designed for Tiny ImageNet [40]. Both networks were trained for 40 epochs with a stochastic gradient descent, a momentum of 0.9, and a weight decay of 5×10^{-4}. A mini-batch size was set to 256. The initial learning rate was set to 0.001 and 0.1 for the LCN and HCN, respectively, and was decreased by a factor of 10 on epochs corresponding to 50% and 75% of the total duration of the training.

Again, our results (Table 3) indicate an applicability of the LCN to the shortcut detection: the LCN and HCN show relatively close high accuracies on stylized data (70.1% and 88.0%, respectively) while performing dramatically different on regular data (8.54% and 41.43%, respectively).

3.2. Utilizing predictions of a low-capacity network to navigate the training of a high-capacity network

Next, we demonstrate that it is possible to make use of an LCN to avoid learning the shortcut by an HCN. Reliable features necessary for a robust generalization are relatively high-level and shortcuts are usually low-level characteristics of an image. Given this assumption, the LCN will primarily produce accurate and confident predictions for images containing shortcuts.

Given a training dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}$, the corresponding IW ($w_i$) for a training image $\mathbf{x}_i$ is its probability of misclassification as given by an LCN,

$$w_i = 1 - p(y_i|\mathbf{x}_i). \tag{1}$$

IWs are then employed while training an HCN: for every training image, the corresponding loss term is multiplied by the IW of this image. We normalize IWs with respect to a mini-batch: IWs of samples from a mini-batch are divided by the sum of all IWs in that mini-batch. The mini-batch training loss $L_{\mathcal{B}}$ is thus the following:

$$L_{\mathcal{B}} = \sum_{k \in \mathcal{B}} \widetilde{w}_k L_k, \tag{2}$$

where L_k indicates the loss of the kth sample in the mini-batch. The mini-match normalized IW is

$$\widetilde{w}_j = \frac{w_j}{\sum_{k \in \mathcal{B}} w_k}. \tag{3}$$

[1] https://github.com/bethgelab/stylize-datasets

3.2.1. Overview of experiments

Generally, whether a dataset contains shortcuts is not known beforehand. In order to overcome this issue and test the too-good-to-be-true prior, we introduced synthetic shortcuts into a well-known dataset [cf. 27]. We then applied our approach and investigated whether it was able to avoid reliance on these shortcuts and generalize o.o.d. This testing strategy allowed us to run well-controlled experiments and quantify the effects of our method.

We ran a set of experiments on all possible pairs of classes from the CIFAR-10 dataset [24]. In every binary classification problem, a synthetic shortcut was introduced in each of the two classes. To have a better understanding of our method's generalizability, we investigated two opposite types of shortcuts as well as two HCN architectures, ResNet-56 [17] and VGG-11 [37]. Note that our too-good-to-be-true prior is readily applicable to multi-class problems.

For both shortcut types and both HCN architectures, we expected the two-stage LCN-HCN procedure to downweight the majority of shortcut images. Therefore, compared to the ordinary training procedure, better performance should be observed when shortcuts in a test set are misleading (i.e., o.o.d. test set). We also expected that the two-stage LCN-HCN procedure may suppress some non-shortcut images. Thus, a slightly worse performance was expected for a test set without shortcuts as well as for a test set with helpful shortcuts (i.e., i.i.d. test set).

The main objective of these experiments was to compare an ordinary and a weighted training procedure in terms of the susceptibility of resulting models to the shortcuts. However, crucially for our idea of the too-good-to-be-true prior, it was also important to validate our reasoning concerning the key role of the low-capacity network in the derivation of useful IWs. For this purpose, we introduced another training condition where IWs were obtained from probabilistic predictions of the same HCN architecture as the target network. We refer to the IWs obtained from an HCN as HCN-IWs and to the IWs obtained from an LCN as LCN-IWs. We expected HCN-IWs either to fail to suppress shortcut images, resulting in poor performance on a test set with misleading shortcuts (o.o.d. test set), or to equally suppress both shortcut and non-shortcut images, resulting in poor performance on any test data. Using HCN-IWs mirrors approaches that place greater emphasis on challenging items.

3.2.2. Shortcuts

For the sake of generality, we introduced two shortcut types: the "local" was salient and localized, and the "global" was subtle and diffuse. The local shortcut was intended to capture real-world cases such as a marker in the corner of a radiograph [42] and the global was intended to capture such situations as subtle distortions in the lens of a camera.

The local shortcut was a horizontal line of three pixels, red for one class and blue for the other (Figure 2, second column). The location of the line was the same for all images: upper left corner. The shortcut was present in randomly chosen 30% of training as well as validation images in each class.

The global shortcut was a mask of Gaussian noise, one per class (Fig. 2, right). The mask was sampled from a multivariate normal distribution with zero mean and isotropic covariance matrix, with variance set to 25×10^{-4}, and then added to randomly chosen 30% of training and validation images of a corresponding class.

Based on CIFAR-10 test images of the selected classes, we prepared three test sets for each shortcut type (examples shown in Fig. 3). *Congruent* (i.i.d.): all images contained shortcuts, each associated with the same class as in the training set. *Incongruent* (o.o.d.): all images contained shortcuts but each shortcut was placed in the images of the opposite class compared to the training set. *Neutral*: original CIFAR-10 images without shortcuts.

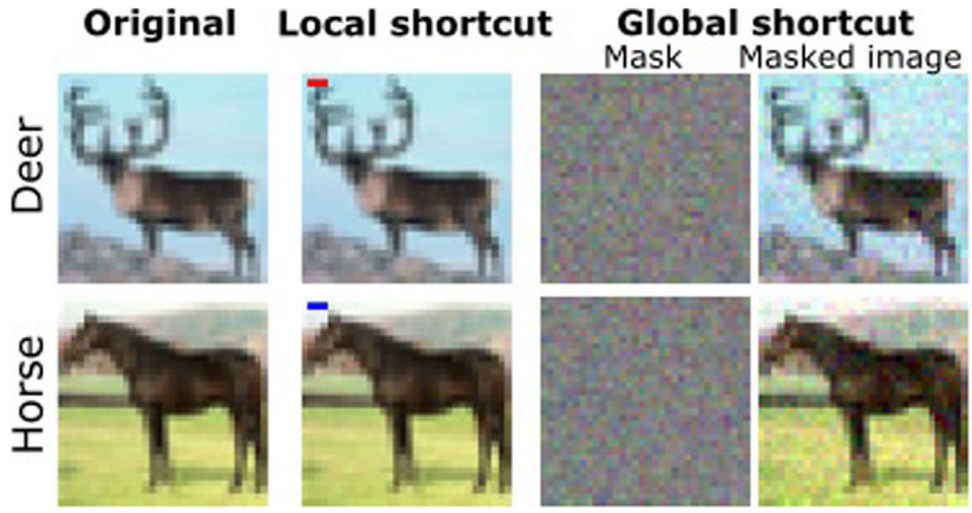

Fig. 2. Examples of shortcut types used in our experiments: local and global. The global shortcut is subtle to humans, so original images and additive masks are also depicted. For the subset of images containing a shortcut, a network could learn to rely on these superficial features at the expense of more invariant properties, which has consequences for generalization.

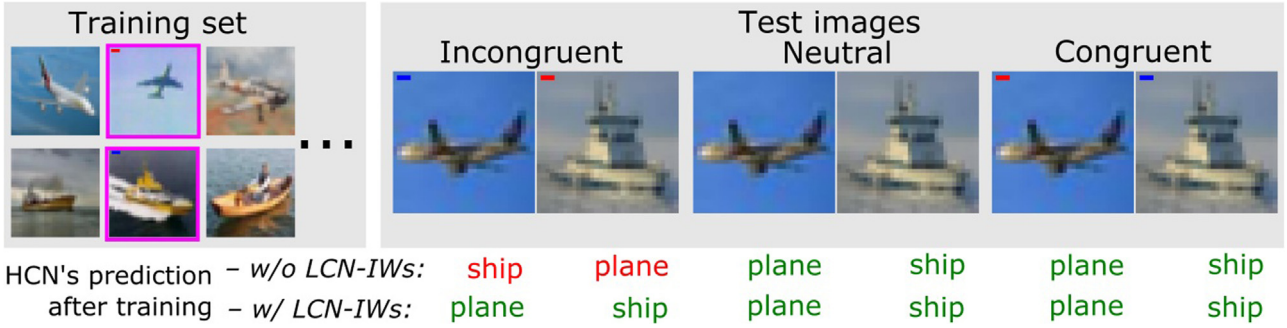

Fig. 3. Expected effects of the LCN-IWs training on classifying different test cases by the HCN. Correct HCN decisions are in green, incorrect are in red. Training images containing local shortcuts are outlined in magenta. Whereas ordinary (w/o LCN-IWs) training should lead to poor o.o.d. performance on Incongruent test items where the shortcut is now misleading, LCN-IWs should selectively downweight training items with the shortcut allowing the HCN to generalize well across the spectrum. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

3.2.3. Model architectures and training details

The LCN was a single convolutional layer of 4 channels with 3-by-3 kernels, a linear activation function, and without downsampling, followed by a fully-connected classification layer.

In two separate sets of simulations, we tested two different HCN architectures: ResNet-56 for CIFAR-10 [17] and VGG-11 [37]. The first two fully-connected layers of VGG-11 had 1024 units each and no dropout was used.

Network weights were initialized according to [12]. We used stochastic gradient descent to train both LCN and HCN. The initial learning rate was set to 0.01 for the LCN. The HCN's initial learning rate was set to 0.01 for VGG-11 [37] and to 0.1 for ResNet-56 [17]. The HCNs were trained with a momentum of 0.9 and a weight decay of 5×10^{-4} for 150 epochs. To avoid overfitting, the HCN's performance on validation data (see below) was tested at each epoch, and best-performing parameters were chosen as the result of training. The LCN was trained for 40 epochs. For both LCN and HCN, the learning rate was decreased by a factor of 10 on epochs corresponding to 50% and 75% of the total duration of the training. Mini-batch size for both networks was set to 256.

For each class, the original 5000 images from the CIFAR-10 training set were divided into 4500 training images and 500 validation images. Thus, the training set of every class pair included 9000 images and the validation set included 1000 images.

IWs were introduced to the training process as described in the beginning of this section, and for every mini-batch a weighted-average loss was calculated. During ordinary training without IWs, a simple average loss was calculated.

For every binary classification problem, the results reported below are the averages of 10 independent runs.

3.2.4. Results

The overall pattern of results was in accord with our predictions–downweighting training items that could be mastered by a low-capacity network reduced shortcut reliance in a high-capacity network, which improved o.o.d. generalization at a small cost in i.i.d. generalization. As no disagreement in the results of the two HCN architectures was observed, hereinafter we focus on ResNet-56, whereas results for VGG-11 can be found in Appendix C.

To compare the potentials of LCN-IWs and HCN-IWs to separate out shortcut-containing images, we used logistic regression to classify IWs as shortcut or regular. With both local and global shortcuts, classification accuracies for LCN-IWs were greater than for HCN-IWs (Table 4). Thus, the distribution of LCN-IWs much better discriminates between shortcut and regular images than the distribution of HCN-IWs. In fact, we found that for the majority of class-pairs LCN-IWs and HCN-IWs were only moderately correlated (see Appendix B). An example of HCN-/LCN-IWs distributions for a specific pair of classes can be found in Appendix B as well.

Effects of the training condition (ordinary, HCN-IWs, and LCN-IWs) on the HCN's performance on every test set (incongruent,

Table 4
Results for classifying IWs as corresponding to shortcut/regular images, with logistic regression as a classifier. Accuracies (in %) are the averages across 45 class-pairs; 95% confidence intervals are in brackets.

Local shortcut		Global shortcut	
LCN-IWs	HCN-IWs	LCN-IWs	HCN-IWs
91.53 [82.60, 93.45]	70.02 [70.01, 70.03]	90.4 [86.16, 94.66]	70.1 [70.00, 70.03]

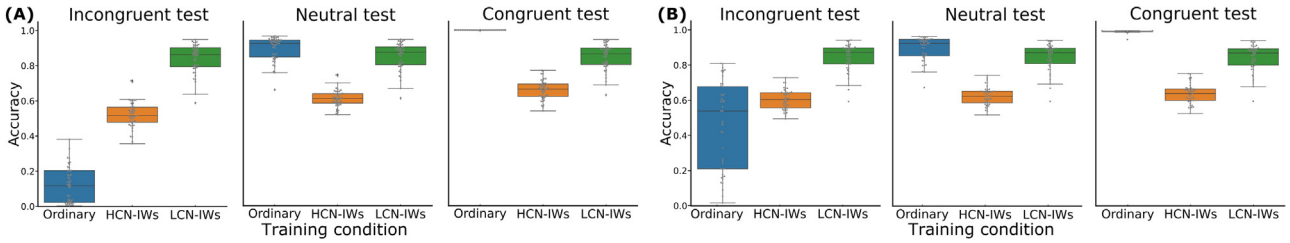

Fig. 4. Accuracies on incongruent, neutral, and congruent test sets after ordinary and HCN-/LCN-weighted training, with (A) local or (B) global shortcuts in training set. HCN is ResNet-56. Across shortcut types, LCN-IWs result in almost equally high accuracy on all sets. HCN-IWs constantly result in accuracies inferior to LCN-IWs; moreover, on neutral and congruent test sets, accuracies after HCN-IWs training are lower than after ordinary training. Thus, LCN-IWs are successful in avoiding shortcut reliance and preserving useful features, while HCN-IWs are not.

neutral, and congruent) are shown in Fig. 4. HCNs are prone to rely on our shortcuts, as evidenced by low incongruent accuracies and very high congruent accuracies after the ordinary training. Incongruent accuracies are improved after the LCN-IWs training, compared to those after the ordinary training. Importantly, after LCN-IWs training, incongruent, neutral, and congruent accuracies are all similarly high. Together, these results suggest that LCN-IWs are successful in reducing shortcut reliance in the target network.

Although exceeding performance on the incongruent test set after the ordinary training condition, incongruent accuracies after the HCN-IWs training are substantially lower than after the LCN-IWs training. The neutral and congruent accuracies are lower than in both ordinary and LCN-IWs training conditions. At the same time, incongruent accuracies are still noticeably lower than neutral and congruent conditions. These results indicate that HCN-IWs are not effective as LCN-IWs in suppressing shortcut learning.

The main results shown in Fig. 4 indicate that LCN-IWs reduce shortcut reliance with little cost to performance on other items, whereas HCN-IWs are less effective because they remove non-shortcut items as well. Key to the LCN-IW efficacy is properly matching network capacity to the learning problem. Out of the 45 classification pairs considered, there should be natural variation in problem difficulty that affects target network performance. In particular, we predict that overall benefit will be lower when the LCN performs better on a class pair, indicating that its capacity is sufficient to learn non-shortcut information for this classification task.

We define Overall Benefit (OB) as a combination of Gain (G) and Loss (L):

$$OB = G + L, \tag{4}$$

with

$$G = \text{logit}(p(y^{corr}|\mathcal{D}^{incon}, \mathcal{T}^{IW})) - \text{logit}(p(y^{corr}|\mathcal{D}^{incon}, \mathcal{T}^{ord})) \tag{5}$$

and

$$L = \text{logit}(p(y^{corr}|\mathcal{D}^{neut}, \mathcal{T}^{IW})) - \text{logit}(p(y^{corr}|\mathcal{D}^{neut}, \mathcal{T}^{ord})) \tag{6}$$

where y^{corr} is a correct label, $\mathcal{D}^{incon}$ and $\mathcal{D}^{neut}$ are incongruent and neutral test sets, respectively, and $\mathcal{T}^{ord}$ and $\mathcal{T}^{IW}$ are ordinary and IW-based training procedures, respectively. We compute average OB for each class pair and contrast those against corresponding neutral test accuracies after ordinary training. The latter are introduced to reflect the default classification difficulty of each class pair. These comparisons are shown in Fig. 5. Two evident trends are important. First, recapitulating the previous results, LCN-IWs result in greater OB than HCN-IWs. OB corresponding to LCN-IWs is almost always positive, while OB corresponding to HCN-IWs is often negative. Second, OB is negatively correlated with the neutral test accuracy after ordinary training; that is, as the difficulty of a classification problem increases, benefits of using IWs generally increase as well. One possibility is that for easy to discriminate pairs, such as frog and ship, the LCN was able to learn non-shortcut information which reduced the overall benefit of the LCN-IWs.

4. Discussion

In general, using Occam's razor to favor simple solutions is a sensible policy. We certainly do not advocate for adding unnecessary complexity. However, for difficult problems that have evaded a solution, it is unlikely that a trivial solution exists. The problems of interest in machine learning have taken millions of years for nature to solve and have puzzled engineers for decades. It seems implausible that trivial solutions to such problems would exist and we should be skeptical when they appear.

For such difficult problems, we suggest adopting a *too-good-to-be-true prior* that shies away from simple solutions. Simple solutions to complex problems are likely to rely on superficial features that are reliable within the particular training context, but are unlikely to capture the more subtle invariants central to a concept. To use a historic example, people had great hopes that the Perceptron [33], a one-layer neural network, would master computer vision to only have their hopes dashed [30]. When such simple systems appear successful, including on held-out test data, they are most likely relying on shortcuts that will not generalize out of sample on somewhat different test distributions, such as when a system is deployed.

We proposed and evaluated two simple applications of the too-good-to-be-true inductive bias. First, we made use of a low-capacity network (LCN) to detect the presence of a shortcut in a dataset. Second, we used an LCN to establish importance weights (IWs) to help train a high-capacity network (HCN). The idea was that the LCN would not have the capacity to learn subtle invariants but instead be reduced to relying on superficial shortcuts. For the second application, by downweighting the items that LCN could master, we found that the HCN was less susceptible to shortcuts

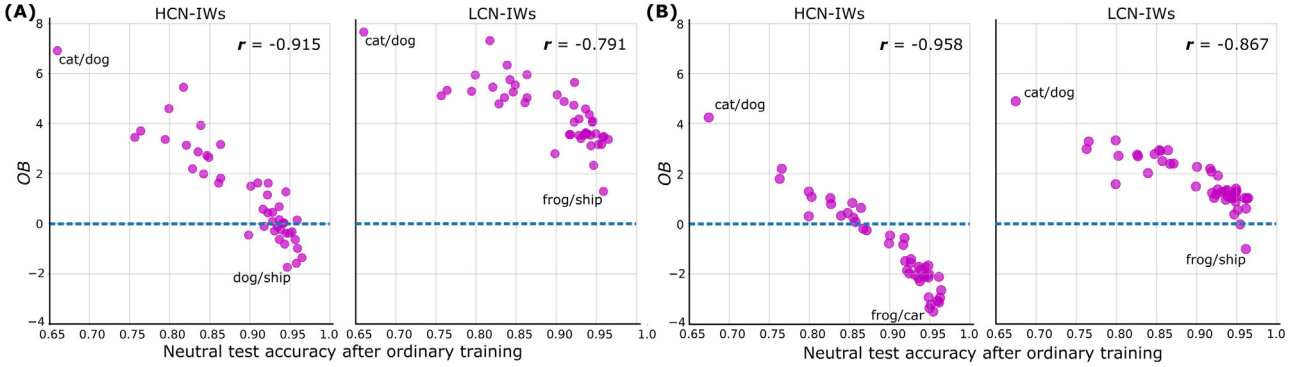

Fig. 5. Effects of the LCN-/HCN-IWs training procedure for each of the 45 class pairs depending on a difficulty of the respective binary classification problem. HCN is ResNet-56; (A) local and (B) global shortcut types are considered separately. The effects of training are represented by the Overall Benefit measure (*OB*; gain + loss; see Section 3.2.4); the difficulty of a pair is represented by the neutral test accuracy after ordinary training. Recapitulating previous results, LCN-IWs are more effective than HCN-IWs. Furthermore, the easier learning problem, the less *OB* from IWs: the relatively higher capacity of a network supplying IWs leads to downweighting non-shortcut items.

and showed better o.o.d. generalization at little cost when misleading shortcuts were not present.

Although we evaluated the de-biasing application of the too-good-to-be-true prior on the CIFAR-10 dataset, the basic method of using an LCN to establish IWs for an HCN is broadly applicable. We considered two network architectures for the HCN, ResNet-56 and VGG-11, which both showed the same overall pattern of performance. Interestingly, ResNet-56 appeared more susceptible to shortcuts, perhaps because its architecture contains skip connections that are themselves a type of shortcut allowing lower-level information in the network to propagate forward absent intermediate processing stages.

One key challenge in our approach is matching the complexity of the LCN to the learning problem. When the LCN has too much capacity, it may learn more than shortcuts and downweight information useful to o.o.d. generalization (see Fig. 5). It is for this reason that LCN-IWs are much more effective than HCN-IWs (see Fig. 4). Unfortunately, there is no simple procedure that guarantees selecting an appropriate LCN. The choice depends on one's beliefs about the structure of the world, the susceptibility of models to misleading shortcuts, and the nature of the learning problem. Nevertheless, reasonable decisions can be made. For example, we would be skeptical of a Perceptron that successfully classifies medical imagery, so it could serve as an LCN.

Since the too-good-to-be-true prior is a general inductive bias, our two-stage LCN-HCN approach is just one specific implementation of it and other techniques may be developed. The effectiveness of our two-stage approach should be evaluated in other tasks and domains outside computer vision. Further research should consider how to choose the architecture of an LCN and how the effectiveness of this architecture depends on different types of shortcuts. Finally, a promising direction is to use IWs to selectively suppress aspects of training items rather than downweighting entire examples.

5. Conclusions

We suggest a fundamental viewpoint on the solutions discovered by deep neural networks: too simple solutions, well-performing on the available data, including testing sets, probably contain shortcuts. Since an exhaustive prior knowledge of shortcut features is unrealistic, the inductive bias we propose is to avoid solutions too simple for a given task in order to secure o.o.d. generalization. We illustrate this fundamental idea with rigorously controlled experiments our inductive bias, even in its simplest realizations, enabled us to detect and avoid shortcuts, resulting in a more robust generalization.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This article is an output of a research project implemented as part of the Basic Research Program at the National Research University Higher School of Economics (HSE University). This work was supported by ESRC ES/W007347/1, Wellcome Trust Investigator Award WT106931MA, and Royal Society Wolfson Fellowship 183029 to B.C.L.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patrec.2022.12.010.

References

[1] H. Bahng, S. Chun, S. Yun, J. Choo, S.J. Oh, Learning de-biased representations with biased representations, in: International Conference on Machine Learning, PMLR, 2020, pp. 528–539.
[2] S. Beery, G. Van Horn, P. Perona, Recognition in terra incognita, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 456–473.
[3] Y. Bengio, J. Louradour, R. Collobert, J. Weston, Curriculum learning, in: Proceedings of the International Conference on Machine Learning, 2009, pp. 41–48.
[4] C.M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer-Verlag, Berlin, Heidelberg, 2006.
[5] W. Brendel, M. Bethge, Approximating CNNs with bag-of-local-features models works surprisingly well on ImageNet, arXiv preprint arXiv:1904.00760(2019).
[6] R. Cadene, C. Dancette, H. Ben-Younes, M. Cord, D. Parikh, RUBi: reducing unimodal biases in visual question answering, arXiv preprint arXiv:1906.10169(2019).
[7] C. Clark, M. Yatskar, L. Zettlemoyer, Don't take the easy way out: ensemble based methods for avoiding known dataset biases, arXiv preprint arXiv:1909.03683(2019).
[8] C. Clark, M. Yatskar, L. Zettlemoyer, Learning to model and ignore dataset bias with mixed capacity ensembles, arXiv preprint arXiv:2011.03856(2020).
[9] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, F.A. Wichmann, Shortcut learning in deep neural networks, Nat. Mach. Intell. 2 (11) (2020) 665–673.
[10] R. Geirhos, C.R. Medina Temme, J. Rauber, H.H. Schütt, M. Bethge, F.A. Wichmann, Generalisation in humans and deep neural networks, in: Proceeding of the Annual Conference on Neural Information Processing Systems, Curran, 2019, pp. 7549–7561.

[11] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F.A. Wichmann, W. Brendel, ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness, arXiv preprint arXiv:1811.12231(2018).
[12] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the International Conference on Artificial Intelligence and Statistics, 2010, pp. 249–256.
[13] I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, arXiv preprint arXiv:1412.6572(2014).
[14] G. Hacohen, D. Weinshall, On the power of curriculum learning in training deep networks, in: Proceedings of the International Conference on Machine Learning, PMLR, 2019, pp. 2535–2544.
[15] H. He, S. Zha, H. Wang, Unlearn dataset bias in natural language inference by fitting the residual, arXiv preprint arXiv:1908.10763(2019).
[16] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: surpassing human-level performance on ImageNet classification, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1026–1034.
[17] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[18] D. Hendrycks, T. Dietterich, Benchmarking neural network robustness to common corruptions and perturbations, arXiv preprint arXiv:1903.12261(2019).
[19] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, D. Song, Natural adversarial examples, arXiv preprint arXiv:1907.07174(2019).
[20] K. Hermann, A. Lampinen, What shapes feature representations? Exploring datasets, architectures, and training, Adv. Neural Inf. Process. Syst. 33 (2020).
[21] Z. Huang, H. Wang, E.P. Xing, D. Huang, Self-challenging improves cross-domain generalization, ECCV, 2020.
[22] L. Jiang, D. Meng, Q. Zhao, S. Shan, A. Hauptmann, Self-paced curriculum learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 29, 2015.
[23] B. Kim, H. Kim, K. Kim, S. Kim, J. Kim, Learning not to learn: training deep neural networks with biased data, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 9012–9020.
[24] A. Krizhevsky, G. Hinton, et al., Learning multiple layers of features from tiny images (2009).
[25] M.P. Kumar, B. Packer, D. Koller, Self-paced learning for latent variable models, in: Proceedings of the International Conference on Neural Information Processing Systems, Vol. 1, 2010, pp. 1189–1197.
[26] Y. Li, N. Vasconcelos, REPAIR: Removing representation bias by dataset resampling, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 9572–9581.
[27] G. Malhotra, B.D. Evans, J.S. Bowers, Hiding a plane with a pixel: examining shape-bias in CNNs and the benefit of building in biological constraints, Vision Res. 174 (2020) 57–68, doi:10.1016/j.visres.2020.04.013.
[28] T. Malisiewicz, A. Gupta, A.A. Efros, Ensemble of exemplar-SVMs for object detection and beyond, in: Proceedings of the International Conference on Computer Vision, IEEE, 2011, pp. 89–96.
[29] D. Meng, Q. Zhao, L. Jiang, What objective does self-paced learning indeed optimize?, arXiv preprint arXiv:1511.06049(2015).
[30] M.L. Minsky, S. Papert, Perceptrons: An Introduction to Computational Geometry, MIT Press, Cambridge, MA, 1969.
[31] J. Nam, H. Cha, S. Ahn, J. Lee, J. Shin, Learning from failure: training debiased classifier from biased classifier, arXiv preprint arXiv:2007.02561(2020).
[32] B. Recht, R. Roelofs, L. Schmidt, V. Shankar, Do ImageNet classifiers generalize to ImageNet? in: Proceedings of the International Conference on Machine Learning, PMLR, 2019, pp. 5389–5400.
[33] F. Rosenblatt, The perceptron: a probabilistic model for information storage in the brain, Psychol Rev. 65 (1958) 386–408.
[34] V. Sanh, T. Wolf, Y. Belinkov, A.M. Rush, Learning from others' mistakes: avoiding dataset biases without modeling them, arXiv preprint arXiv:2012.01300(2020).
[35] H. Shah, K. Tamuly, A. Raghunathan, P. Jain, P. Netrapalli, The pitfalls of simplicity bias in neural networks, arXiv preprint arXiv:2006.07710(2020).
[36] A. Shrivastava, A. Gupta, R. Girshick, Training region-based object detectors with online hard example mining, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 761–769.
[37] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: Proceedings of the International Conference on Learning Representations, 2015.
[38] S. Sinha, H. Bharadhwaj, A. Goyal, H. Larochelle, A. Garg, F. Shkurti, DIBS: diversity inducing information bottleneck in model ensembles, arXiv preprint arXiv:2003.04514(2020).
[39] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks, arXiv preprint arXiv:1312.6199(2013).
[40] J. Wu, Q. Zhang, G. Xu, Tiny imagenet challenge, 2017.
[41] X. Wu, E. Dyer, B. Neyshabur, When do curricula work?, arXiv preprint arXiv:2012.03107(2020).
[42] J.R. Zech, M.A. Badgeley, M. Liu, A.B. Costa, J.J. Titano, E.K. Oermann, Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study, PLoS Med. 15 (11) (2018) e1002683, doi:10.1371/journal.pmed.1002683.