A Example to further motivate surprise

Imagine you are given a network which has been trained to classify different Ford car models outdoors (in green pastures), and asked to classify pictures of these same models indoors (in a showroom). Intuitively, most of the learned function is still valid so only a small number of parameters need to be tweaked, requiring only a small number of indoor examples to do so. Perhaps tweaking parameters in the first layer would be sufficient, re-extracting the same features from e.g. the darker pixels of indoor images as were previously extracted from the outdoor images. Similarly, if you were asked to classify pictures of Fiat cars, you may expect that only a small number of parameters need to be adapted (likely in later layers this time), perhaps adjusting the (conceptual) wheel and mirror detectors for the smaller wheels and more rounded mirrors of Fiats. However, as we discuss in Section 2, minimizing a standard loss function (e.g. cross-entropy) with *stochastic gradient descent* (SGD) does not result in such intuitive updates—all of the network's parameters are updated and learnt structure is unnecessarily destroyed. These gradients tell us which parameters *can* reduce the error, but not which parameters *should* reduce the error in order to maximally-transfer knowledge and thus speed-up learning. Doing so requires additional information—such as unit-level surprise.

B Implementation details

B.1 Data

Figure 4: EMNIST-DA shifts. Figure adopted from [9].

Our networks are trained using the 47-class EMNIST dataset [7] ("identity" shift in Figure 4). We use 2000 samples per class from the training split with the remaining 400 forming a validation set, we report results on the separate test set also containing 400 examples per class.

We adapt to 10 new data distributions—7 low-level shifts from EMNIST-DA [9] and 3 high-level label shifts. From the 14 EMNIST-DA shifts depicted in Figure 4, we chose 7 shifts that adversely affect accuracy (without adaptation) and intuitively affect the early convolutional filters: crystals, fog, gaussian blur, grass, impulse noise, sky and stripe. To create the 3 label shifts, we train our networks on only the first 37 classes of EMNIST and then choose 5 of the 10 unseen classes three times from the range $[38, 47]$ to arrive at: H1:$[38, 39, 40, 41, 42]$, H2:$[43, 44, 45, 46, 47]$, and H3:$[38, 40, 42, 44, 46]$.

To evaluate sample efficiency we run experiments with varying amounts of data, we experiment with using 2, 5, 10, 20 and 50 samples per class as well as using all the data (2000 samples per class). The full results of these experiments are given in Appendix E.

B.2 Experimental setup

Table 3 provides the architectural details of the simple 5-layer convolutional neural network (CNN) that we use. During pre-training we use dropout between the layers, for adaptation we do not as it unnecessarily complicates the propagation of surprise and makes little difference to the final results.

During pre-training and adaptation we use a batch size of 256. We pre-train for 150 epochs with a learning rate of 0.01. Due to using small amount of data, during adaptation we train with early stopping for a maximum of 100 epochs using a patience of 10. We use a learning rate of 0.1 for all experiments except for when fine-tuning all layers simultaneously which requires a learning rate

of 0.01 to prevent divergence. We optimize using stochastic gradient descent with momentum set to 0.9. For experiments using our update rule the thresholds α and β in Equation 3 are set to 0.01. Experiments are run over 3 seeds from which we report a mean and one standard deviation.

Table 3: Architecture of the CNN used. For conv. layers, the weights-shape is: *num. input channels* $\times$ *num. output channels* $\times$ *filter height* $\times$ *filter width*.

Layer	Weights-Shape	Stride	Padding	Activation	Dropout Prob.
Conv	$3 \times 64 \times 5 \times 5$	2	2	ReLU	0.1
Conv	$64 \times 128 \times 3 \times 3$	2	2	ReLU	0.3
Conv	$128 \times 256 \times 3 \times 3$	2	2	ReLU	0.5
FC	6400×128	N/A	N/A	ReLU	0.5
FC	128×47	N/A	N/A	Softmax	0

B.3 Measuring unit-level surprise

A single unit in a feed-forward neural network outputs an activation $a = g(\mathbf{w}^T\mathbf{h} + b)$, where $\mathbf{h}$ is the hidden unit activations of the previous layer, $\mathbf{w}$ the learned weight vector, b the learned bias and g some non-linearity. During training a unit can store a distribution $P(A)$ which captures the distribution that its activation can take. A unit is surprised by new data if the activation distribution changes, i.e. $P(A) \neq Q(A)$, where $Q(A)$ is the distribution of the unit's activations under the new data. We quantify surprise using the KL-divergence from $P(A)$ to $Q(A)$, i.e. $s(A) = D_{KL}(Q(A)||P(A))$ [19, 26].[3]

The surprisal (or information content) of an event $X = x$, with $X \sim P(X)$, is given by $\log(1/P(x))$. Intuitively, this quantity represents how "surprised" we are to see $X = x$, with unlikely events having high surprisal. Surprise itself is a somewhat overloaded term but can be used to describe the entropy, $H(X) = -\sum P(x) \log P(x)$, that is the expected surprisal. If we now receive a sample of a different random variable $Y = y$, $Y \sim Q(Y)$, but we have assumed as a prior that we are receiving samples from $P(X)$, the amount of additional surprisal we receive on account of our assumption is $\log(1/P(X = y)) - \log(1/Q(Y = y))$. The expected value of this quantity over Q is the KL-Divergence $D_{KL}(Q||P)$ [22], which is the expected surprisal of receiving samples from Q when we have assumed the distribution to be P. We can also interpret $D_{KL}(Q||P) = H(Q, P) - H(Q)$ as the expected extra message-length per datum that must be communicated if a code that is optimal for P is used to communicate Q, compared to a code that is optimal for Q. We have seen this quantity referred to as Bayesian surprise, information gain, asymmetric surprise or simply surprise [10, 19]. Throughout this work we refer to this quantity as surprise for simplicity.

B.4 Calculating surprise from bin counts

After pre-training we parameterize activation distributions with softly-binned histograms. To calculate $P(A)$ we run one further forward-pass of the network over the training data and bin the activations using the same procedure as in [9], with 10 bins, which outputs 10 normalized bin counts $\pi^p_1, \dots \pi^p_{10}$ for each unit. π^p_i represents the probability a falls into bin i and $\sum_{i=1}^{10} \pi^p_i = 1$. The blue curves in Fig. 5 depict examples of such distributions. These distributions can be considered as representing the "normal" activation values of a unit, i.e. the values it expects to take on.

We then receive some data from a new distribution, possibly the same as the pre-training data distribution, which is fed into the network. During adaptation we can parameterize $Q(A)$ in the same way as $P(A)$, using a batch of this new data (Fig. 5–orange curves) to calculate normalized bin counts $\pi^q_1, \dots \pi^q_{10}$ which change as the network learns. The surprise for a unit can then be calculated as

$$s(A) = D_{KL}\left(Q(A; \{\pi^q_i\}_{i=1}^{10}) \;||\; P(A; \{\pi^p_i\}_{i=1}^{10})\right) = \sum_{i=1}^{10} \pi^q_i \log \frac{\pi^q_i}{\pi^p_i}. \tag{1}$$

[3] For convolutions, when creating $P(A)$ and $Q(A)$ we take each spatial location of a feature map to be one sample of the activation, a.

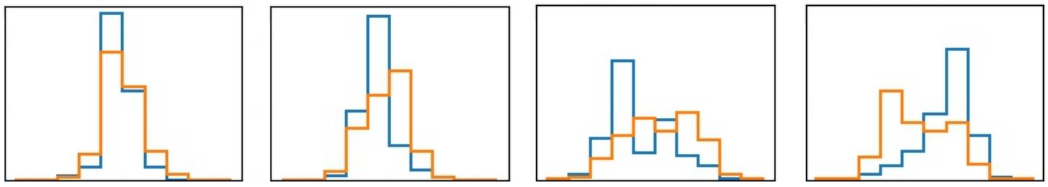


Figure 5: Examples of our histogram parameterizations of $P(A)$, in blue, and $Q(A)$, in orange. When new data is received, the activation distribution changes. From left to right, the surprise values s(A) are approximately 0.1, 0.2, 0.3, 0.4.

B.5 A surprise-based update rule

Let p_i denote the surprise of "parent" unit i in layer l, c_j the surprise of "child" unit j in layer $l+1$, and w_{ij} the weight that connects parent unit i with child unit j. This setup is depicted in Fig. 6 below.

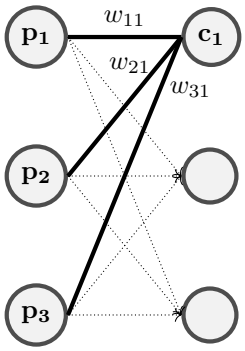


Figure 6: Update rule schematic. The highlighted weights update only if the aggregate parent surprise $\bar{p}_1$ is below some threshold β and the child surprise c_1 is above some threshold α.

For child unit j, we calculate its aggregate parent surprise $\bar{p}_j$ as a weighted average of its parent surprises. More specifically, we calculate

$$\bar{p}_j = \sum_i \frac{|w_{ij}|}{\sum_k |w_{kj}|} \cdot p_i \ , \tag{2}$$

where the normalized weight value $\frac{|w_{ij}|}{\sum_k |w_{kj}|}$ ensures comparable scaling across units in a layer. We then use this to create an update rule where the input weights of child j are updated if and only if child j is surprised (i.e. c_j is above some threshold α) but its parents are not (i.e. $\bar{p}$ is below some threshold β). In particular, we create the following update rule:

$$w_{ij} := w_{ij} - \mathbb{I}[c_j > \alpha]\mathbb{I}[\bar{p}_j < \beta] \cdot \eta \nabla_{w_{ij}} \mathcal{L}, \tag{3}$$

where $\mathbb{I}[c_j > \alpha]$ is an indicator function that is 1 when $c_j > \alpha$ and 0 otherwise, $\mathbb{I}[\bar{p}_j < \beta]$ is similarly defined, η the learning rate, and $\mathcal{L}$ the loss function (cross-entropy in our case). $\mathbb{I}[c_j > \alpha]$ ensures that only surprised units update, and can be compared with *metaplasticity*[4] in the brain. $\mathbb{I}[\bar{p}_j < \beta]$ prevents/blocks simultaneous changes in later units who may also surprised by their input, and can be compared with *neuromodulation*[5] in the brain.

[4]The modification of a neuron's future capacity for learning as a function of recent synaptic history [1, 2]. Believed to regulate the plasticity mechanisms themselves in order to generate adaptive behaviour [11, 17, 39, 41].

[5]Neuromodulators are neurotransmitters which, instead of conveying excitation or inhibition, change the properties of other neurons or synapses [20].

C Alternative surprise measures

Imagine a unit or feature detector with a bi-modal activation distribution where the modes roughly represent on (detected) and off (not detected). Perhaps such a unit being off *more often* in the new data (as when part of an image is occluded, discussed in Section 3) is not a good signal to update. To differentiate this situation from e.g. the unit being *on* more often in the new data, we can define a new measure which we call the *surprise increase* (SI):

$$SI = H(Q, P) - H(P), \tag{4}$$

where $H(P)$ is the entropy of P and $H(Q, P)$ is the cross-entropy of P relative to Q. Unlike the KL-divergence, SI *can* be negative. SI is negative (i.e. a surprise decrease) when an event that is already quite likely under P becomes even more likely under Q—as shown in Figure 7a. This could be an interesting alternative surprise measurement as it can distinguish between surprise increases and decreases.

(a) Surprise decrease

(b) Surprise increase

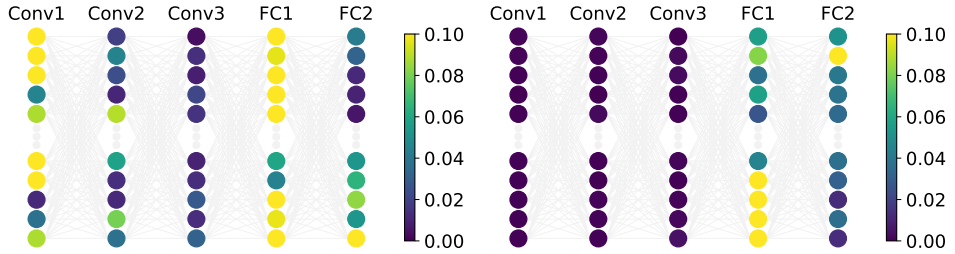
Figure 7: SI illustration. P is blue, Q is orange. For fixed P, SI depends only on $H(Q, P)$.

D Does batch normalization solve the problem?

Batch normalization (BN, 18) standardizes activation distributions across batches, bringing $P(A)$ and $Q(A)$ closer together. This naturally raises the question as to whether or not BN solves the problem of differing unit-level distributions, thus removing the need for unit-level surprise. We investigate this below.

Do the same surprise patterns exist? Figure 8 shows the ideal situation for BN, where the BN statistics are (re)calculated using the *new* data distribution before we calculate surprise. Compared to Figure 2, the magnitude of the surprise is indeed lessened as BN standardizes $P(A)$ and $Q(A)$, but it does not make units unsurprised. Moreover, we still see the same patterns of surprise in the early layers for low-level shifts and in the later layers for high-level shifts.

Is an adaptative strategy still best? As shown in Table 4, it is indeed still best to employ an adaptive update strategy to train specific layers for specific shifts, e.g. use FlexTune [35] to select Conv1 & BN1 for low-level shifts and FC2 for high-level shifts. This further confirms that BN alone does *not* solve the problem of changes in activation distribution, as selective adaptation strategies are still superior to updating all layers with SGD. In Table 4 we also show that, if we only use the BN statistics without any training (AdaBN, 24), or only update the BN parameters and statistics on the new data (labelled "BN params"), performance is poor compared to the other strategies.



(a) Low-level shift (crystals bckgr.) (b) High-level shift (unseen classes)

Figure 8: Network surprise patterns after updating the BN statistics on the new data.

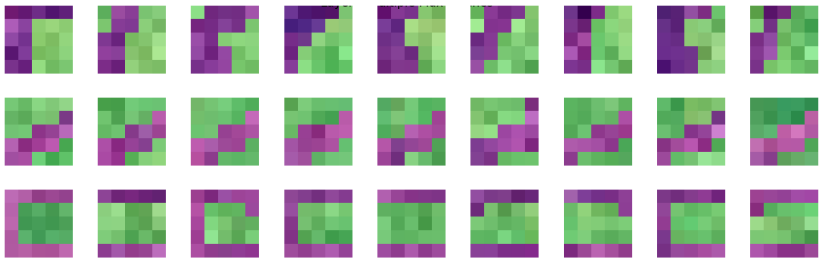
Table 4: 5-shot accuracy with BN. *L:* low-level shifts average, *H:* high-level shifts average. Zero-shot shows accuracy before adaptation. AdaBN [24] updates the BN statistics on the new data. "BN params" updates only the BN parameters and statistics. For all other methods/rows, only the layers listed are permitted to update.

	L	H
Zero-Shot	22.1 ± 0.2	0.0 ± 0.0
AdaBN [24]	50.7 ± 0.4	0.0 ± 0.0
BN params	66.0 ± 1.3	0.0 ± 0.0
Conv1, BN1	$\mathbf{78.9 \pm 0.8}$	0.0 ± 0.0
Conv2, BN2	67.9 ± 1.5	0.0 ± 0.0
Conv3, BN3	60.4 ± 0.7	0.3 ± 0.3
FC1, BN4	56.4 ± 0.5	68.9 ± 1.7
FC2	52.7 ± 0.6	$\mathbf{95.4 \pm 0.5}$
FC1, BN4, FC2	55.8 ± 0.8	94.6 ± 0.9
All (SGD)	76.7 ± 0.8	90.7 ± 2.3

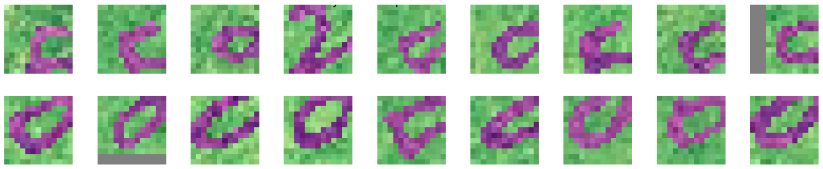
E Further Results

E.1 Max-activating patches

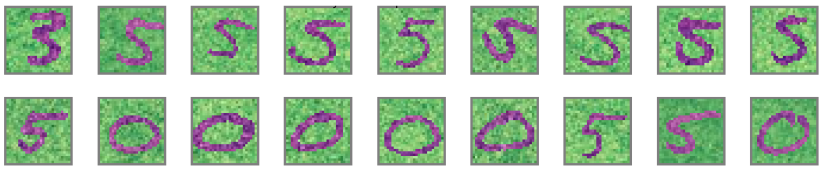
Figure 9 shows the maximum-activating image patches on the EMNIST-DA [9] grass shift for selected units in each layer. Note that we do not have a perfect edges-parts-wholes hierarchy—the receptive field of Conv2 seems too large as it almost sees the entire image (can be a "whole" rather than a part).



(a) Conv 1



(b) Conv 2



(c) Conv 3

Figure 9: Max-activating patches for different units on the EMNIST-DA grass shift.

E.2 Per-shift Results

What follows is tables of full results for each shift at each number of shots, these are provided for completeness with no further analysis.

Table 5: 2-shot accuracy across shifts: training different layers of a CNN

	Conv1	Conv2	Conv3	FC1	FC2	FC1 + FC2
H1	0.0 ± 0.0	0.0 ± 0.1	7.2 ± 4.9	65.3 ± 14.1	95.2 ± 1.4	77.8 ± 1.4
H2	0.0 ± 0.0	0.0 ± 0.0	5.0 ± 0.8	45.7 ± 8.8	80.5 ± 5.4	71.4 ± 4.2
H3	0.0 ± 0.0	0.0 ± 0.0	10.8 ± 7.3	49.4 ± 4.3	92.9 ± 2.3	81.4 ± 5.7
Crystals	69.9 ± 4.9	52.1 ± 0.9	51.0 ± 0.9	50.0 ± 0.6	47.8 ± 0.7	49.6 ± 1.1
Fog	86.5 ± 3.2	84.5 ± 0.4	84.0 ± 0.5	82.3 ± 0.8	78.8 ± 1.2	81.6 ± 0.4
Gauss. Blur	82.7 ± 0.3	80.4 ± 2.0	79.0 ± 0.3	74.9 ± 0.6	71.5 ± 1.0	74.1 ± 1.9
Grass	81.4 ± 0.8	21.6 ± 7.2	8.1 ± 0.7	7.3 ± 0.6	7.2 ± 0.5	7.7 ± 0.5
Imp. Noise	87.4 ± 0.9	86.0 ± 0.2	83.4 ± 1.4	83.2 ± 0.9	80.7 ± 0.6	81.9 ± 1.4
Sky	74.4 ± 3.9	54.4 ± 0.5	33.2 ± 2.8	18.4 ± 0.7	13.7 ± 2.7	18.4 ± 3.3
Stripe	74.3 ± 0.9	58.2 ± 4.8	58.2 ± 1.4	45.7 ± 2.1	36.7 ± 1.0	47.9 ± 2.6
Avg High	0.0 ± 0.0	0.0 ± 0.0	7.7 ± 4.0	53.5 ± 3.4	89.6 ± 1.5	76.9 ± 2.9
Avg Low	79.5 ± 1.3	62.4 ± 2.1	56.7 ± 0.5	51.7 ± 0.6	48.1 ± 0.5	51.6 ± 0.7
Avg All	55.7 ± 0.9	43.7 ± 1.5	42.0 ± 1.4	52.2 ± 0.8	60.5 ± 0.3	59.2 ± 0.9

Table 6: 2-shot accuracy across shifts: comparison of different responses. Zero-shot is the accuracy before any updates are performed.

	Zero-shot	SGD	FlexTune	Upd. Rule
H1	0.0 ± 0.0	56.7 ± 4.2	95.2 ± 1.4	33.8 ± 21.4
H2	0.0 ± 0.0	51.9 ± 5.7	80.5 ± 5.4	34.3 ± 11.2
H3	0.0 ± 0.0	57.6 ± 5.3	92.9 ± 2.3	41.4 ± 9.4
Crystals	46.3 ± 0.4	55.3 ± 1.3	69.9 ± 4.9	68.5 ± 5.4
Fog	78.4 ± 0.4	83.6 ± 1.2	87.1 ± 2.3	86.6 ± 3.2
Gauss. Blur	60.4 ± 2.1	81.2 ± 0.6	82.7 ± 0.3	81.8 ± 1.8
Grass	5.8 ± 0.2	20.1 ± 12.0	81.4 ± 0.8	81.4 ± 0.8
Imp. Noise	76.9 ± 0.9	87.9 ± 0.2	87.4 ± 0.9	87.7 ± 0.5
Sky	4.1 ± 0.5	35.2 ± 4.9	74.4 ± 3.9	74.4 ± 3.9
Stripe	16.3 ± 1.1	70.0 ± 1.4	74.3 ± 0.9	72.8 ± 2.1
Avg High	0.0 ± 0.0	55.4 ± 1.4	89.6 ± 1.5	36.5 ± 6.7
Avg Low	41.2 ± 0.3	61.9 ± 2.4	79.6 ± 1.2	79.0 ± 1.1
Avg All	28.8 ± 0.2	60.0 ± 1.9	82.6 ± 0.7	66.3 ± 2.5

Table 7: 5-shot accuracy across shifts: training different layers of a CNN

	Conv1	Conv2	Conv3	FC1	FC2	FC1 + FC2
H1	0.0 ± 0.0	0.1 ± 0.1	29.8 ± 8.2	82.8 ± 7.3	96.4 ± 1.0	91.9 ± 2.0
H2	0.0 ± 0.0	0.5 ± 0.9	26.1 ± 6.2	76.7 ± 6.6	91.3 ± 1.5	86.2 ± 3.8
H3	0.0 ± 0.0	0.3 ± 0.4	20.3 ± 6.1	82.3 ± 5.3	96.0 ± 0.6	93.0 ± 2.8
Crystals	78.1 ± 1.7	57.5 ± 0.9	52.3 ± 0.8	51.5 ± 0.5	49.2 ± 0.8	51.4 ± 0.6
Fog	89.6 ± 0.3	86.5 ± 0.7	84.8 ± 0.5	83.1 ± 0.6	81.3 ± 0.6	82.6 ± 0.6
Gauss. Blur	84.0 ± 0.4	82.5 ± 0.7	81.7 ± 0.6	78.3 ± 0.9	75.9 ± 0.5	78.1 ± 0.4
Grass	82.0 ± 2.2	41.7 ± 7.4	10.3 ± 1.3	8.4 ± 0.9	7.3 ± 0.5	9.4 ± 0.4
Imp. Noise	88.6 ± 0.3	86.8 ± 0.1	85.7 ± 0.3	84.3 ± 0.9	81.5 ± 0.4	83.9 ± 0.4
Sky	79.6 ± 0.7	65.9 ± 0.5	44.3 ± 1.9	27.9 ± 1.8	18.3 ± 2.4	28.8 ± 1.2
Stripe	77.1 ± 1.8	70.9 ± 2.6	70.6 ± 0.6	60.4 ± 1.6	47.0 ± 1.3	60.9 ± 2.5
Avg High	0.0 ± 0.0	0.3 ± 0.5	25.4 ± 3.8	80.6 ± 4.6	94.5 ± 0.9	90.4 ± 1.1
Avg Low	82.7 ± 0.4	70.2 ± 1.2	61.4 ± 0.2	56.3 ± 0.2	51.5 ± 0.3	56.4 ± 0.3
Avg All	57.9 ± 0.3	49.3 ± 0.8	50.6 ± 1.2	63.6 ± 1.2	64.4 ± 0.3	66.6 ± 0.1

Table 8: 5-shot accuracy across shifts: comparison of different responses. Zero-shot is the accuracy before any updates are performed.

	Zero-shot	SGD	FlexTune	Upd. Rule
H1	0.0 ± 0.0	68.0 ± 13.8	96.4 ± 1.0	80.3 ± 4.8
H2	0.0 ± 0.0	78.0 ± 2.0	91.3 ± 1.5	77.3 ± 7.2
H3	0.0 ± 0.0	77.1 ± 5.0	96.0 ± 0.6	79.9 ± 9.1
Crystals	46.3 ± 0.4	57.5 ± 3.8	78.1 ± 1.7	77.0 ± 3.4
Fog	78.4 ± 0.4	86.3 ± 0.2	89.6 ± 0.3	89.6 ± 0.3
Gauss. Blur	60.4 ± 2.1	84.3 ± 0.7	84.0 ± 0.4	83.9 ± 0.3
Grass	5.8 ± 0.2	50.9 ± 6.8	82.0 ± 2.2	82.3 ± 2.6
Imp. Noise	76.9 ± 0.9	88.3 ± 0.1	88.6 ± 0.3	87.6 ± 0.3
Sky	4.1 ± 0.5	56.6 ± 4.7	79.6 ± 0.7	80.8 ± 0.9
Stripe	16.3 ± 1.1	75.4 ± 1.0	77.1 ± 1.8	79.3 ± 0.8
Avg High	0.0 ± 0.0	74.4 ± 5.2	94.5 ± 0.9	79.2 ± 4.2
Avg Low	41.2 ± 0.3	71.3 ± 2.1	82.7 ± 0.4	82.9 ± 0.5
Avg All	28.8 ± 0.2	72.2 ± 2.9	86.3 ± 0.5	81.8 ± 1.2

Table 9: 10-shot accuracy across shifts: training different layers of a CNN

	Conv1	Conv2	Conv3	FC1	FC2	FC1 + FC2
H1	0.0 ± 0.0	0.5 ± 0.8	50.1 ± 5.1	91.3 ± 2.0	96.1 ± 0.8	94.5 ± 0.6
H2	0.0 ± 0.0	0.9 ± 0.8	45.7 ± 8.8	88.5 ± 1.7	92.8 ± 0.3	89.0 ± 2.2
H3	0.0 ± 0.0	0.3 ± 0.4	46.6 ± 16.0	89.7 ± 5.3	95.0 ± 1.1	92.8 ± 1.8
Crystals	80.5 ± 1.5	59.9 ± 3.1	54.4 ± 0.1	52.6 ± 0.7	50.6 ± 0.8	52.6 ± 0.8
Fog	90.1 ± 0.1	87.8 ± 0.5	85.6 ± 0.3	84.1 ± 0.2	82.1 ± 1.3	84.3 ± 0.4
Gauss. Blur	85.0 ± 0.4	83.9 ± 1.1	82.6 ± 0.4	81.4 ± 0.9	77.9 ± 2.3	80.7 ± 0.8
Grass	83.8 ± 0.9	57.1 ± 3.2	15.1 ± 0.8	10.0 ± 0.9	7.8 ± 0.1	11.2 ± 1.0
Imp. Noise	89.1 ± 0.0	87.2 ± 0.4	86.0 ± 0.3	84.7 ± 0.6	82.7 ± 0.1	84.5 ± 0.2
Sky	83.3 ± 1.5	71.4 ± 1.0	52.5 ± 0.5	36.7 ± 1.2	24.5 ± 2.4	38.0 ± 0.2
Stripe	82.4 ± 0.7	76.1 ± 2.8	76.1 ± 1.1	69.2 ± 0.2	56.8 ± 0.8	68.1 ± 1.8
Avg High	0.0 ± 0.0	0.6 ± 0.6	47.5 ± 6.4	89.8 ± 1.2	94.6 ± 0.6	92.1 ± 0.5
Avg Low	84.9 ± 0.3	74.8 ± 1.3	64.6 ± 0.2	59.8 ± 0.4	54.6 ± 0.5	59.9 ± 0.4
Avg All	59.4 ± 0.2	52.5 ± 0.8	59.5 ± 1.8	68.8 ± 0.5	66.6 ± 0.4	69.6 ± 0.4

Table 10: 10-shot accuracy across shifts: comparison of different responses. Zero-shot is the accuracy before any updates are performed.

	Zero-shot	SGD	FlexTune	Upd. Rule
H1	0.0 ± 0.0	84.8 ± 0.5	96.1 ± 0.8	90.4 ± 3.0
H2	0.0 ± 0.0	79.9 ± 1.7	92.8 ± 0.3	88.3 ± 2.0
H3	0.0 ± 0.0	87.1 ± 2.7	95.0 ± 1.1	89.7 ± 5.1
Crystals	46.3 ± 0.4	61.2 ± 1.6	80.5 ± 1.5	79.8 ± 0.9
Fog	78.4 ± 0.4	87.4 ± 0.2	90.1 ± 0.1	90.1 ± 0.1
Gauss. Blur	60.4 ± 2.1	85.7 ± 0.6	85.0 ± 0.4	85.3 ± 1.0
Grass	5.8 ± 0.2	69.2 ± 2.4	83.8 ± 0.9	84.3 ± 1.3
Imp. Noise	76.9 ± 0.9	88.2 ± 0.4	89.1 ± 0.0	88.0 ± 0.2
Sky	4.1 ± 0.5	66.8 ± 1.1	83.3 ± 1.5	83.4 ± 1.5
Stripe	16.3 ± 1.1	78.8 ± 1.1	82.4 ± 0.7	82.5 ± 0.9
Avg High	0.0 ± 0.0	84.0 ± 0.5	94.6 ± 0.6	89.5 ± 1.2
Avg Low	41.2 ± 0.3	76.8 ± 0.7	84.9 ± 0.3	84.8 ± 0.5
Avg All	28.8 ± 0.2	78.9 ± 0.5	87.8 ± 0.4	86.2 ± 0.6

Table 11: 20-shot accuracy across shifts: training different layers of a CNN

	Conv1	Conv2	Conv3	FC1	FC2	FC1 + FC2
H1	0.0 ± 0.0	2.1 ± 3.4	66.4 ± 8.9	95.7 ± 0.9	97.2 ± 0.7	95.1 ± 0.6
H2	0.0 ± 0.0	5.6 ± 0.9	64.9 ± 8.7	90.3 ± 1.5	94.8 ± 0.6	91.6 ± 1.1
H3	0.0 ± 0.0	9.5 ± 1.3	69.3 ± 6.2	93.5 ± 2.2	96.1 ± 1.0	95.3 ± 0.9
Crystals	83.7 ± 0.5	65.6 ± 2.3	56.7 ± 0.2	54.6 ± 0.3	51.5 ± 0.7	54.6 ± 0.4
Fog	90.4 ± 0.4	88.4 ± 0.1	86.1 ± 1.0	85.2 ± 0.2	84.0 ± 0.3	85.1 ± 0.2
Gauss. Blur	86.6 ± 0.4	85.0 ± 1.4	83.4 ± 0.6	82.2 ± 0.8	81.0 ± 0.9	82.3 ± 1.1
Grass	85.6 ± 1.0	67.6 ± 2.8	20.7 ± 1.8	13.4 ± 1.0	8.3 ± 0.3	14.1 ± 0.9
Imp. Noise	89.1 ± 0.6	87.7 ± 0.2	86.6 ± 0.2	85.2 ± 0.5	83.7 ± 0.5	84.9 ± 0.2
Sky	84.8 ± 1.0	76.3 ± 0.7	59.5 ± 0.4	45.1 ± 1.5	31.7 ± 0.7	47.0 ± 0.7
Stripe	83.9 ± 1.3	79.7 ± 2.0	78.9 ± 0.9	74.7 ± 0.2	62.2 ± 0.9	75.3 ± 0.3
Avg High	0.0 ± 0.0	5.7 ± 0.8	66.8 ± 2.9	93.2 ± 1.2	96.1 ± 0.3	94.0 ± 0.2
Avg Low	86.3 ± 0.3	78.6 ± 1.1	67.4 ± 0.2	62.9 ± 0.3	57.5 ± 0.4	63.3 ± 0.3
Avg All	60.4 ± 0.2	56.7 ± 0.9	67.3 ± 0.8	72.0 ± 0.5	69.0 ± 0.4	72.5 ± 0.3

Table 12: 20-shot accuracy across shifts: comparison of different responses. Zero-shot is the accuracy before any updates are performed.

	Zero-shot	SGD	FlexTune	Upd. Rule
H1	0.0 ± 0.0	89.3 ± 1.6	97.2 ± 0.7	95.5 ± 1.2
H2	0.0 ± 0.0	85.4 ± 3.1	94.8 ± 0.6	89.9 ± 1.4
H3	0.0 ± 0.0	90.3 ± 0.6	96.3 ± 0.9	93.5 ± 2.4
Crystals	46.3 ± 0.4	66.5 ± 1.8	83.7 ± 0.5	82.1 ± 0.9
Fog	78.4 ± 0.4	88.1 ± 0.3	90.4 ± 0.4	90.4 ± 0.4
Gauss. Blur	60.4 ± 2.1	86.0 ± 0.5	86.6 ± 0.4	86.9 ± 0.4
Grass	5.8 ± 0.2	77.0 ± 1.2	85.6 ± 1.0	85.5 ± 0.2
Imp. Noise	76.9 ± 0.9	88.5 ± 0.1	89.1 ± 0.6	88.3 ± 0.3
Sky	4.1 ± 0.5	73.3 ± 0.8	84.8 ± 1.0	84.7 ± 0.9
Stripe	16.3 ± 1.1	81.6 ± 0.4	83.9 ± 1.3	84.7 ± 1.1
Avg High	0.0 ± 0.0	88.3 ± 1.4	96.1 ± 0.2	93.0 ± 1.1
Avg Low	41.2 ± 0.3	80.1 ± 0.3	86.3 ± 0.3	86.1 ± 0.3
Avg All	28.8 ± 0.2	82.6 ± 0.2	89.2 ± 0.2	88.2 ± 0.4

Table 13: 50-shot accuracy across shifts: training different layers of a CNN

	Conv1	Conv2	Conv3	FC1	FC2	FC1 + FC2
H1	0.0 ± 0.0	25.8 ± 24.0	83.5 ± 5.3	97.0 ± 0.2	97.2 ± 0.5	95.9 ± 0.9
H2	0.0 ± 0.0	32.8 ± 4.5	84.4 ± 1.9	94.8 ± 0.7	95.2 ± 0.9	93.5 ± 1.2
H3	0.0 ± 0.0	28.1 ± 17.4	85.0 ± 4.7	95.5 ± 1.9	97.6 ± 0.5	96.1 ± 0.6
Crystals	85.2 ± 0.5	72.9 ± 1.2	59.3 ± 0.3	57.8 ± 0.3	53.1 ± 0.6	57.8 ± 0.2
Fog	90.6 ± 0.2	89.3 ± 0.1	87.7 ± 0.2	86.7 ± 0.2	85.1 ± 0.3	86.6 ± 0.2
Gauss. Blur	87.9 ± 0.5	87.3 ± 0.3	85.8 ± 0.4	84.6 ± 0.2	83.2 ± 0.6	84.6 ± 0.2
Grass	87.1 ± 0.6	75.2 ± 1.9	29.8 ± 1.6	19.3 ± 0.8	9.0 ± 0.2	19.7 ± 0.6
Imp. Noise	89.4 ± 0.3	88.3 ± 0.2	86.9 ± 0.3	85.9 ± 0.3	84.8 ± 0.2	85.9 ± 0.2
Sky	86.7 ± 0.5	80.9 ± 0.3	66.4 ± 0.6	54.3 ± 1.8	38.2 ± 0.8	56.7 ± 0.4
Stripe	87.4 ± 0.3	84.6 ± 0.7	83.2 ± 0.5	79.9 ± 0.9	67.6 ± 0.8	79.8 ± 0.8
Avg High	0.0 ± 0.0	28.9 ± 12.2	84.3 ± 3.7	95.8 ± 0.6	96.7 ± 0.6	95.2 ± 0.2
Avg Low	87.8 ± 0.1	82.6 ± 0.4	71.3 ± 0.3	66.9 ± 0.3	60.1 ± 0.3	67.3 ± 0.0
Avg All	61.4 ± 0.1	66.5 ± 3.8	75.2 ± 0.9	75.6 ± 0.4	71.1 ± 0.3	75.7 ± 0.1

Table 14: 50-shot accuracy across shifts: comparison of different responses. Zero-shot is the accuracy before any updates are performed.

	Zero-shot	SGD	FlexTune	Upd. Rule
H1	0.0 ± 0.0	93.3 ± 0.5	97.2 ± 0.5	96.8 ± 0.3
H2	0.0 ± 0.0	89.9 ± 0.9	95.2 ± 0.9	94.4 ± 0.6
H3	0.0 ± 0.0	93.8 ± 1.4	97.6 ± 0.5	95.5 ± 2.0
Crystals	46.3 ± 0.4	73.2 ± 0.9	85.2 ± 0.5	83.5 ± 0.5
Fog	78.4 ± 0.4	88.9 ± 0.3	90.6 ± 0.2	90.5 ± 0.3
Gauss. Blur	60.4 ± 2.1	87.3 ± 0.4	87.9 ± 0.5	87.4 ± 0.3
Grass	5.8 ± 0.2	81.8 ± 0.4	87.1 ± 0.6	86.2 ± 0.4
Imp. Noise	76.9 ± 0.9	88.9 ± 0.2	89.4 ± 0.3	88.5 ± 0.5
Sky	4.1 ± 0.5	79.1 ± 0.5	86.7 ± 0.5	86.9 ± 0.3
Stripe	16.3 ± 1.1	84.8 ± 1.3	87.4 ± 0.3	87.5 ± 0.4
Avg High	0.0 ± 0.0	92.3 ± 0.6	96.7 ± 0.6	95.6 ± 0.6
Avg Low	41.2 ± 0.3	83.4 ± 0.2	87.8 ± 0.1	87.2 ± 0.1
Avg All	28.8 ± 0.2	86.1 ± 0.3	90.4 ± 0.2	89.7 ± 0.1

Table 15: 2000-shot (all data) accuracy across shifts: training different layers of a CNN

	Conv1	Conv2	Conv3	FC1	FC2	FC1 + FC2
H1	0.0 ± 0.1	80.0 ± 9.9	97.6 ± 0.2	98.7 ± 0.1	98.6 ± 0.1	98.8 ± 0.1
H2	2.0 ± 3.2	78.2 ± 8.1	96.7 ± 0.2	98.0 ± 0.2	98.0 ± 0.2	98.3 ± 0.2
H3	0.0 ± 0.0	81.9 ± 2.8	97.7 ± 0.4	98.6 ± 0.3	98.7 ± 0.2	98.8 ± 0.0
Crystals	88.0 ± 0.3	85.5 ± 0.3	72.6 ± 0.2	69.4 ± 0.6	57.4 ± 0.4	68.2 ± 0.3
Fog	91.2 ± 0.3	91.2 ± 0.1	90.6 ± 0.1	90.2 ± 0.1	88.0 ± 0.3	89.6 ± 0.1
Gauss. Blur	90.1 ± 0.1	90.9 ± 0.1	90.3 ± 0.0	90.2 ± 0.1	86.7 ± 0.1	89.7 ± 0.2
Grass	89.1 ± 0.2	86.1 ± 0.5	56.6 ± 1.1	40.8 ± 0.9	17.0 ± 0.3	38.8 ± 1.0
Imp. Noise	89.8 ± 0.1	89.7 ± 0.2	88.2 ± 0.1	88.3 ± 0.2	86.6 ± 0.3	87.6 ± 0.2
Sky	89.0 ± 0.1	88.2 ± 0.3	81.7 ± 0.1	75.2 ± 0.3	48.8 ± 0.4	73.7 ± 0.4
Stripe	89.9 ± 0.1	90.8 ± 0.3	90.3 ± 0.1	89.0 ± 0.0	76.3 ± 0.5	88.3 ± 0.2
Avg High	0.7 ± 1.1	80.0 ± 5.1	97.4 ± 0.2	98.5 ± 0.1	98.4 ± 0.1	98.6 ± 0.0
Avg Low	89.6 ± 0.1	88.9 ± 0.0	81.5 ± 0.2	77.6 ± 0.1	65.8 ± 0.1	76.5 ± 0.1
Avg All	62.9 ± 0.3	86.3 ± 1.5	86.2 ± 0.1	83.8 ± 0.1	75.6 ± 0.1	83.2 ± 0.1

Table 16: 2000-shot (all data) accuracy across shifts: comparison of different responses. Zero-shot is the accuracy before any updates are performed.

	Zero-shot	SGD	FlexTune	Upd. Rule
H1	0.0 ± 0.0	97.9 ± 0.1	98.8 ± 0.1	98.7 ± 0.2
H2	0.0 ± 0.0	97.2 ± 0.4	98.3 ± 0.1	98.1 ± 0.2
H3	0.0 ± 0.0	98.2 ± 0.1	98.8 ± 0.0	98.7 ± 0.3
Crystals	46.3 ± 0.4	87.7 ± 0.3	88.0 ± 0.3	88.3 ± 0.2
Fog	78.4 ± 0.4	90.9 ± 0.0	91.3 ± 0.2	90.8 ± 0.2
Gauss. Blur	60.4 ± 2.1	90.5 ± 0.1	90.9 ± 0.1	90.6 ± 0.1
Grass	5.8 ± 0.2	89.2 ± 0.1	89.1 ± 0.2	89.5 ± 0.4
Imp. Noise	76.9 ± 0.9	89.3 ± 0.2	89.8 ± 0.1	89.5 ± 0.2
Sky	4.1 ± 0.5	89.1 ± 0.2	89.0 ± 0.1	89.4 ± 0.1
Stripe	16.3 ± 1.1	90.6 ± 0.3	90.8 ± 0.3	90.0 ± 0.2
Avg High	0.0 ± 0.0	97.8 ± 0.1	98.6 ± 0.0	98.5 ± 0.1
Avg Low	41.2 ± 0.3	89.6 ± 0.1	89.8 ± 0.0	89.7 ± 0.1
Avg All	28.8 ± 0.2	92.1 ± 0.1	92.5 ± 0.0	92.4 ± 0.0