Paper ID: 4658. 0. We present results for more powerful attacks from Madry Challenge leaderboard for MNIST and CIFAR10. We chose "Distributionally Adversarial Attack" (DAA) by Zheng as it appears atop the leaderboards for both MNIST and CIFAR10 datasets. Results for the original Tanh(16) model are shown in Table 1. Results are also shown for an "improved" version of Tanh(16) model which uses more convolutional filters per layer (32 instead of 25), and Gaussian noise with std. dev 0.25 (instead of the 0.15). These changes improve robustness of the model.
1. The term "white-box" means that the adversary knows *everything* about the model, i.e. full ensemble with all layers and every rotation used by all ensemble members.
2. We present results in Table 1 for the case of 32 bit codes. Intuition indeed suggests this code should be more powerful. In general, though, we find robustness asymptotes with increasing code length; this appears related to the "rank" of the coding matrix. Increasing code length relative to number of classes likely results in increasingly correlated logits. Due to bit dependence, the effective Hamming distance does not grow with code length.
3. We feel that the Reviewer's contention that "Any continuous function with the same limiting behavior will have this behavior." is not relevant; we constructed a method to estimate probabilities (eq 3) which does not have this behavior.
4. Euclidian volume is relevant for uncertainty; consider the following. Let y (δy) denote the softmax's logits (change) corresponding to an input x. Let J denote Jacobian matrix of logit layer evaluated at x. By Taylor's theorem, $\delta y \approx J \times \delta x$. In general we don't know singular vectors of J, and δ_x is controlled by the adversary so can point in any direction. J is likely to be full row rank so δy can point in an arbitrary direction. Without further assumptions on adversary, if Euclidean volume (in y-space) associated with uncertainty is vanishingly small (as with softmax), it is easy to find δx whose induced δy moves into a region of high-confidence; i.e., the adversary can reach a region of high confidence by perturbing x to $x + \delta x$ *regardless of x's location in input space*. Empirical evidence of this: see Fig 3(c) in paper; softmax is usually highly confident even for an input which is random noise.
5. The Reviewer's suggestion to consider distribution of activations to the softmax layer seems reasonable for data *known to lie on the training manifold*. However, adversarial (and 'Random" inputs) are (way) off this manifold.
6. Rationale for design choices used for the correlation decoder are simply to yield a valid probability estimate (non-negative, sum to 0). The logistc maps (unnormalized) z_k to a similar range as the code elements in C; ReLU ensures probability estimates are non-negative. We don't aim to achieve carefully calibrated probability estimates; we agree with the Reviewer that such an undertaking may reveal other designs which are better suited for precise calibration. Our results in Figures 3(a)-(c) indicate our probability estimates are still far better than those of conventional models.
7. Thank you to Reviewer for pointing out Theorem 1 is Plotkin bound; we will drop proof and cite in revision.
8. That using softmax to convert logits to probabilities is not a good idea has been established in many papers; see e.g., "On Calibration of Modern Neural Networks" by Guo et Al.). Our empirical results (e.g., Fig 3 in te main text) also clearly show this. Some corrective action is needed (such as Platt scaling). However, our tests indicate that Platt scaling still produces overconfident estimates on adversarial and noise inputs (it appears to only calibrate on the training manifold). By contrast, our approach appears well-behaved even off the training manifold.
9. The Lipschitz constant of the network is *not* larger due to code-induced widening. The final 2 layers of Madry's MNIST model are 1. Fully connected layer of 1024 units. 2. Softmax layer of 10 units (softmax layer). The final 2 layers of our model are 1. Fully connected layer of 16 units (i.e. "code" layer); 2. Final layer of 10 units (computes probability using eq (3) of main text). Compared to typical architectures, ours does *not* induce widening of the network.
10. The fact that we used 2 classes in Section 2.3 is not essential; key point is that larger Hamming distance (=4) increases Euclidean distance between high-probability regions; Point 4. above elaborates on utility of Euclidean distance. Reviewer's point is still well-taken, and in the revision we will reword this section to consider $M > 2$ class example using a Hadamard matrix of code length 8 (which has Hamming distance 4). Fig. 2 would then correspond to fixing 5 of the logits and examining the remaining 3. Such a code would still have a Hamming distance of 4, and the key idea that multiple logits (instead of a single logit as with softmax) need to be altered to effect a class change still holds.
11. We were unclear about meaning of Reviewer's comment on "the relationship between adversarial constraints at network input and adversarial constraints before decoding layer". But, our objective in the paper was to explicltly consider an adversary that was *minimally* constrained at the input (i.e. could generate L_∞, L_2, rotations, noise attacks).

Table 1: Accuracies against various attacks; "-": experiment was not run. "*": training terminated due to time constraints. model was still learning; result will improve with more training. MNIST: $\epsilon = 0.3$; CIFAR10: $\epsilon = .031$

Model	Dataset	Code	Benign	PGD $\epsilon = 0.3(.031)$	PGD $\epsilon = 0.4$	DAA $\epsilon = 0.3(.031)$
Tanh(16)	MNIST	$\mathbf{H}_{16}$	.9911	.853	.49	0.848
Tanh(32)	MNIST	$\mathbf{H}_{32}$	.9901	.847	.472	0.821
Tanh(16)_Improved	MNIST	$\mathbf{H}_{16}$	.9925	.901	.541	0.888
Tanh(16)	CIFAR10	$\mathbf{H}_{16}$	.848*	.578	-	.551
Madry	CIFAR10	$\mathbf{I}_{10}$	.873	.470	-	.447