Do Counterfactual Examples Complicate Adversarial Training?

Eric Yeats[1] Cameron Darwin Eduardo Ortega[2] Frank Liu[3] Hai Li[1]
[1]Duke University [2]Arizona State University [3]Old Dominion University

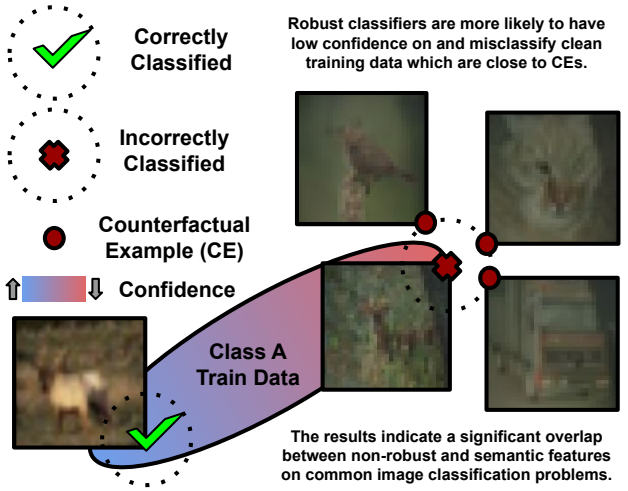


Figure 1. Conceptual depiction of the relationship between accuracy of robustly trained models with proximity to counterfactual examples (CEs). Stronger adversarial training inevitably leads to misclassification of some clean training data, incurring downstream test performance loss. We hypothesize that adversarially trained models are forced to become invariant to some semantic features due to the nearby presence of true CEs.

Abstract

We leverage diffusion models to study the robustness-performance tradeoff of robust classifiers. Our approach introduces a simple, pretrained diffusion method to generate low-norm counterfactual examples (CEs): semantically altered data which results in different true class membership. We report that the confidence and accuracy of robust models on their clean training data are associated with the proximity of the data to their CEs. Moreover, robust models perform very poorly when evaluated on the CEs directly, as they become increasingly invariant to the low-norm, semantic changes brought by CEs. The results indicate a significant overlap between non-robust and semantic features, countering the common assumption that non-robust features are not interpretable.

1. Introduction

Leading theory by Ilyas et al. [10] asserts that adversarial vulnerability arises from reliance of DNN classifiers on *non-robust* features: well-generalizing, yet brittle features which are not comprehensible to humans. As robust models must become invariant to this subset of predictive features, they inevitably lose performance [20]. However, humans appear to break this trend by simultaneously maintaining good performance and robustness on adversarial data [16].

The existence of adversarial perturbations reflects the stark differences between the perception of DNNs and our own. DNNs are notoriously uninterpretable [17, 23], and there is a need for interpretable decision-making in high-stakes scenarios. One model-agnostic way to interpret classifier decision-making is to generate *counterfactual examples* (CEs): subtle, semantic changes to an input datum which would result in a classifier predicting a target class [21]. Following their definition from psychology, CEs for an image-label pair (x, y) can be described by the statement "if y' were the true class of image x (instead of y), then x would look like x'." While CEs can help users understand the decision-making of DNN classifiers, practical methods to generate CEs are problematically similar to those for adversarial attacks [5, 15]. Hence, methods to generate CEs for DNNs often require robust models in some form [1, 2].

We develop new understanding of the robustness-performance tradeoff through our study of the semantic feature distributions learned by robust classifiers. Our study leverages independently generated CE datasets which we create with a simple, pretrained diffusion model technique. We report that robust models, unlike standard (non-robust) models, are more likely to lose confidence on and misclassify *clean* training data which have nearby CEs, and robust models become invariant to the semantic changes brought by CEs. Contrary to common assumptions, our findings suggest significant overlap between *non-robust* and *semantically meaningful* features. This conflict speaks to the complexity of the adversarial problem and motivates alternative approaches to robustness which can resolve or avert it.

Corresponding Author Email: eric.yeats@duke.edu

2. Background

Adversarial Examples and Adversarial Training Adversarial examples are minute, malicious data perturbations which alter classifier predictions [19]. For a classifier $f_\theta : \mathcal{X} \to \mathcal{Y}$ where $\mathcal{X} \in \mathbb{R}^n$ and $\mathcal{Y}$ is the space of categorical distributions of support cardinality k, $\tilde{y}$-targeted adversarial examples $\tilde{x}$ are often computed by altering an input x to descend the negative log loss [6] $\mathcal{L} : \Theta \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+$:

$$\tilde{x} = \underset{x' \in \mathcal{B}_\varepsilon(x)}{\arg\min}\, \mathcal{L}(\theta, x', \tilde{y}), \tag{1}$$

where the neighborhood $\mathcal{B}_\varepsilon(x)$ is commonly defined with the L^∞ or L^2 norm. In this work, we focus on L^2 adversarial examples. Adversarial training consists of finding the optimal model parameters θ^* for the robust objective [12]:

$$\theta^* = \underset{\theta}{\arg\min}\, \mathbb{E}_{x,y} \underset{x' \in \mathcal{B}_\varepsilon(x)}{\max} \mathcal{L}(\theta, x', y). \tag{2}$$

There are other robustness methods such as gradient regularization [4, 25] and randomized smoothing [3], but adversarial training is the most well-known and successful.

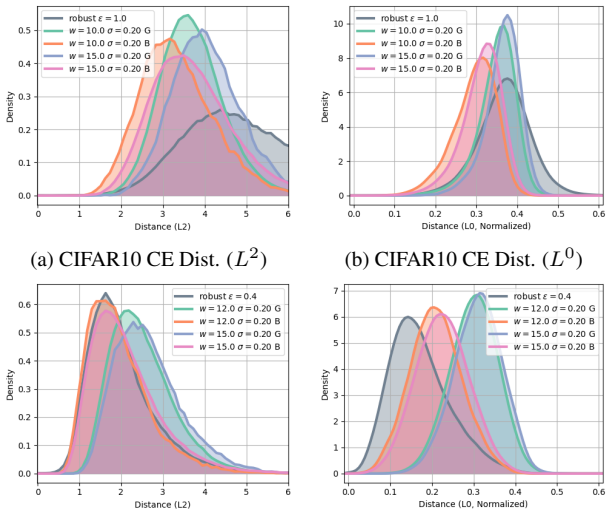
Diffusion Models and Classifier-free Guidance Diffusion models learn a sequence of functions $\epsilon_\theta(x_t, t)$ which predict the noise added to data by a diffusion process at time $t \in [0, 1]$ [9]. To generate new data, $\epsilon_\theta(x_t, t)$ is used to iteratively denoise samples x_t initially drawn from a known noise prior. The noise prediction function $\epsilon_\theta(x_t, t)$ is known to approximate the negative score at diffusion step t [18, 24] (i.e., $\epsilon_\theta(x_t, t) \approx -\sigma_t \nabla_x \log p_t(x_t)$). For class-conditional diffusion models, this property can be used to reproduce the effect of adding targeted classifier gradients [8]:

$$\begin{aligned}\epsilon^w_\theta(x_t, y, t) &:= (w+1)\epsilon_\theta(x_t, y, t) - w\epsilon_\theta(x_t, t) \\ &\approx \epsilon_\theta(x_t, y, t) + w\sigma_t \nabla_x \mathcal{L}(\theta_t, x_t, y),\end{aligned} \tag{3}$$

where θ_t are the equivalent parameters for a time-conditional classifier. Coined "classifier-free guidance" with weight w, sampling with $\epsilon^w_\theta(x_t, y, t)$ as the noise prediction function generates high-quality, conditional data.

3. Methodology

CE Dataset Generation Our approach to CE generation for datum x is to recast it as sampling from a sequence of un-normalized distributions defined by the product of the data distribution (represented by the class-conditional diffusion model) with an independently diffused CE "neighborhood" distribution of scale σ_{CE} centered on $\mu_{CE} = x$. The score of this distribution at each time t is the sum of scores of the two components, so we may simply add the diffused neighborhood score to the diffusion model score.



(a) CIFAR10 CE Dist. (L^2) (b) CIFAR10 CE Dist. (L^0)

(c) SVHN CE Dist. (L^2) (d) SVHN CE Dist. (L^0)

Figure 2. CE distribution comparison. Boltzmann variant CEs produce lower-norm, sparser changes. Best viewed in color.

We present two variants of the noise prediction function corresponding to different choices for the neighborhood distribution: Gaussian and Boltzmann-inspired (see appendix).

$$\epsilon^G_\theta(x_t, y, \mu_{CE}, t) := \epsilon^w_\theta(x_t, y, t) - \frac{\alpha_t \mu_{CE} - x_t}{\alpha_t^2 \sigma_{CE}^2 + \sigma_t^2} \tag{4}$$

$$\begin{aligned}&\epsilon^B_\theta(x_t, y, \mu_{CE}, t) := \\ &\epsilon^w_\theta(x_t, y, t) - \frac{\sqrt{2}}{\alpha_t \sigma_{CE}} \text{hardtanh}(\gamma_t\,(x_t - \alpha_t \mu_{CE}))\end{aligned} \tag{5}$$

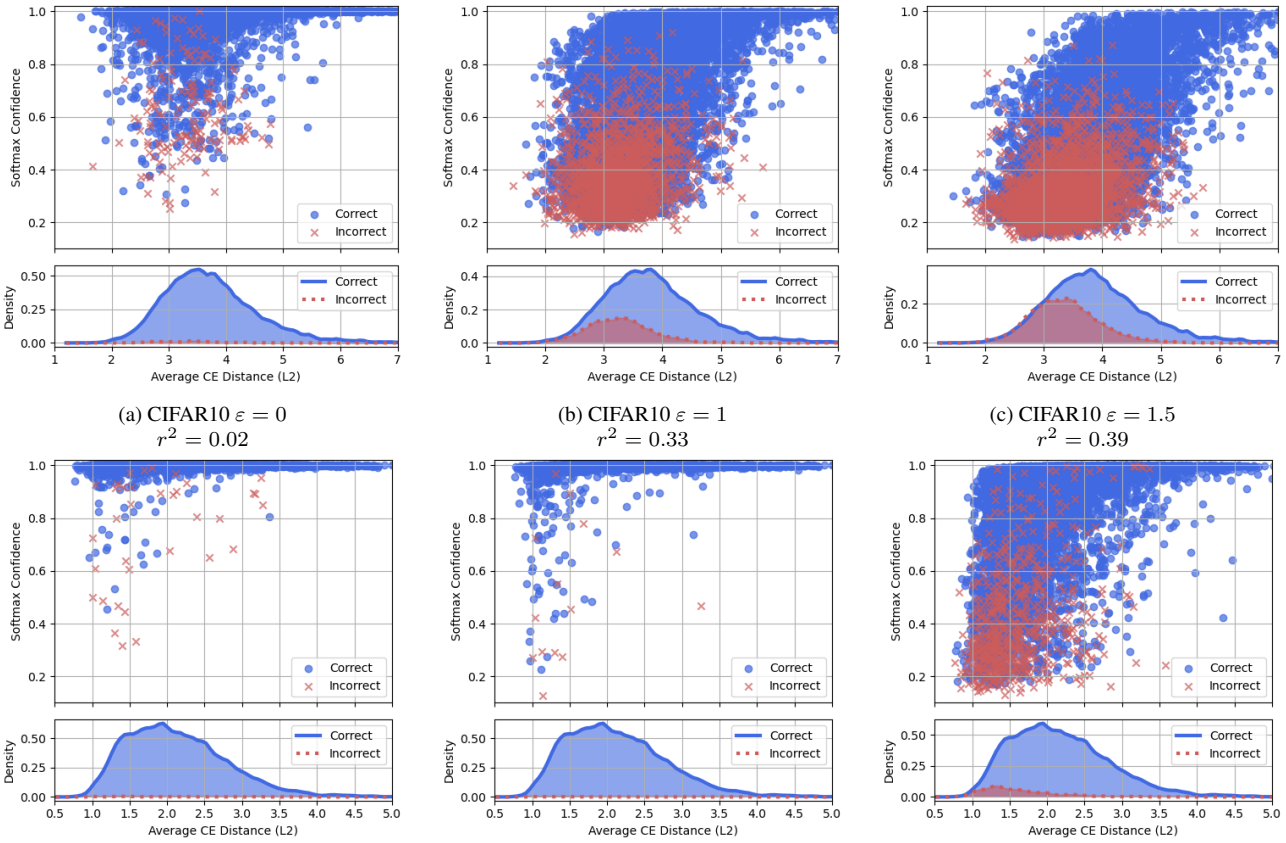
where α_t is a (diffusion) time-dependent scalar decreasing in $t \in [0, 1]$, $\sigma_t = \sqrt{1 - \alpha_t^2}$, and γ_t is a time-dependent scalar derived from the first-order Maclaurin series of the Boltzmann-inspired scores, and is defined as:

$$\gamma_t := \frac{\sqrt{2}}{\alpha_t \sigma_{CE}} - \frac{\sqrt{2}}{\sigma_t \sqrt{\pi}} \left(\exp\left(\frac{\sigma_t^2}{\alpha_t^2 \sigma_{CE}^2} \right) \text{erfc}\left(\frac{\sigma_t}{\alpha_t \sigma_{CE}} \right) \right)^{-1}. \tag{6}$$

A CE x_{CE} is generated for a datum x by assigning $\mu_{CE} \leftarrow x$ and sampling with $\epsilon^G_\theta(\cdot)$ or $\epsilon^B_\theta(\cdot)$ with guidance towards target class y_{CE}. Guidance w and neighborhood scale σ_{CE} are hyperparameters. The Boltzmann-inspired distribution has a sharper mode than the Gaussian distribution, encouraging $x - x_{CE}$ to be lower norm (L^2) and more sparse.

4. Experiments

We run experiments in the PyTorch framework [14] with the CIFAR10 [11] and SVHN [13] image classification benchmarks . We employ conditional score-based models (SBMs) [18] for all diffusion models and L^2 PGD(8) [12] 2×WideResNet-40 models [26] for all robust classifiers.



(a) CIFAR10 $\varepsilon = 0$ $r^2 = 0.02$

(b) CIFAR10 $\varepsilon = 1$ $r^2 = 0.33$

(c) CIFAR10 $\varepsilon = 1.5$ $r^2 = 0.39$

(d) SVHN $\varepsilon = 0$ $r^2 = 0.01$

(e) SVHN $\varepsilon = 0.2$ $r^2 = 0.01$

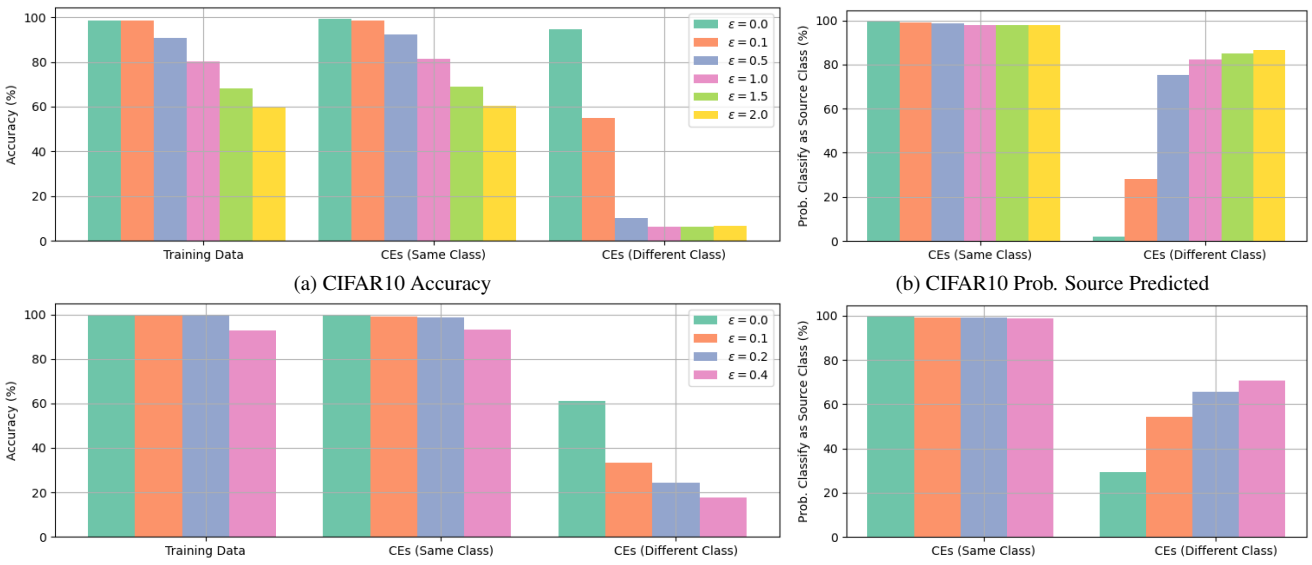
(f) SVHN $\varepsilon = 0.4$ $r^2 = 0.24$

Figure 3. Scatter plots of classifier confidence and average CE distance of 10000 clean training samples as adversarial training norm is increased. Robust models are more likely to misclassify and lose confidence on data which have closer CEs. Best viewed in color.

CE Dataset Evaluation In each experiment we generate 2 CEs for each class for at least 1000 samples from the training set, resulting in at least 20000 CEs. We compare the two CE generation variants with CEs generated by a robust classifier in Fig. 2. The Boltzmann variant (type B) produces CEs with lower-norm (L^2) and sparser (L^0) changes than the Gaussian variant does. The Boltzmann variant produces CEs of norm less than or equal to the norm of CEs produced by the robust classifier. Standard classifiers (no adversarial training) achieve high accuracy on the CEs (i.e., they predict y_{CE} when provided x_{CE}), indicating good semantic quality. The remaining experiments use 200000 Boltzmann variant CEs generated from 10000 training samples (CIFAR10 or SVHN) with $w = 15$ and $\sigma_{CE} = 0.2$. Please see the appendix for more information.

Robust Model Evaluation Our experience is that performance loss of robust models begins on the clean *training* data, incurring performance loss on clean test data downstream. We plot the average L^2 distance of clean training samples with the confidence and accuracy of robust models on the clean samples in Fig. 3. Standard models ($\varepsilon = 0$) achieve very high clean training accuracy, as expected. As robust training budget ε increases, robust models are more likely to misclassify clean training data which are closer to their CEs. Moreover, the confidence of robust models on their training data becomes correlated with the proximity of the data to their CEs ($r^2 = 0.39$ for CIFAR10 and $r^2 = 0.24$ for SVHN).

Fig. 4a and Fig. 4c depict the accuracy of robust models on clean train data and on CEs generated from the train data. Same-class CEs resemble the original data; hence their robust accuracy trends are similar. However, robust models perform very poorly on different-class CEs. Fig. 4b and Fig. 4d depict the probability that the original label y is predicted by a robust model for (x_{CE}, y_{CE}), given that the robust model was correct on (x, y). Robust models are much more likely to misclassify CEs as having the source label, indicating that they become invariant to the low-norm, semantic changes brought by CEs.



(a) CIFAR10 Accuracy (b) CIFAR10 Prob. Source Predicted

(c) SVHN Accuracy (d) SVHN Prob. Source Predicted

Figure 4. Classifier performance on 10000 training data and 200000 CE data generated from the training samples. Best viewed in color.

Fig. 5 depicts the L^2 distance distribution of Boltzmann-type CEs generated from the original CIFAR10 training data and Boltzmann-type CEs generated from robust CEs ($\varepsilon = 1$, $\varepsilon = 2$). On average, Boltzmann-type CEs are farther away from their source data samples when the data samples are robust model CEs (compared to using the original training data as the source data). Since robust model CEs are generated by following input gradient to maximize the confidence of a target class, this indicates that robust model gradients orient towards data regions which are farther away from CEs. Future work may investigate a link between the perceptually aligned gradient of robust classifiers [20] and the proximity of data to CEs.

The method presented in this work can be leveraged to convert a conditional diffusion model into an interpretable classifier. Using the diffusion CE method with a "class with lowest average CE distance" decision rule achieves $\sim 85\%$ accuracy on 1000 input samples of the CIFAR10 test set. Critically, input data for classification are never provided directly to the diffusion model - the input data merely guides CE sampling with analytic scores.

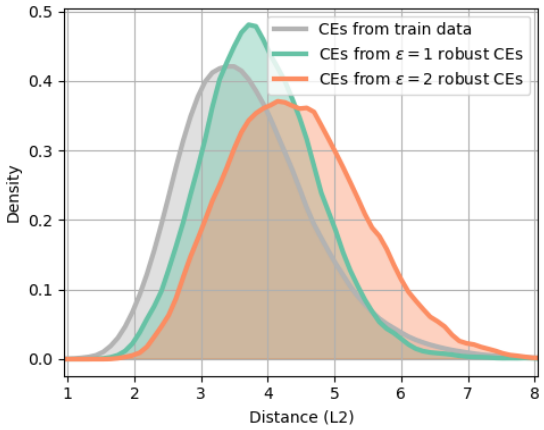


Figure 5. Distance of different-class CEs generated by the Boltzmann method ($w = 15$ $\sigma_{CE} = 0.2$) when the input data is the original CIFAR10 train data, CEs generated by a robust $\varepsilon = 1$ model from the CIFAR10 train data, and CEs generated by a robust $\varepsilon = 2$ model from the CIFAR10 train data. Robust model CEs tend to be in data regions farther away from our diffusion-generated CEs. Best viewed in color.

Limitations Our study considers one combination of classifier architecture, diffusion model, and adversarial attack. This study is limited to the L^2 norm constraint and two image classification benchmarks. Investigation with additional datasets, attacks, and models is left to future work.

5. Conclusion

We present a simple, classifier-free diffusion method to generate counterfactual examples (CEs) which enables novel investigation of the performance loss of robust classifiers. Our results indicate a significant overlap between *non-robust* and *semantically meaningful* features, countering the common assumption that *non-robust* features are not interpretable. Hence, robust models must become invariant to this subset of *semantic* features along with *non-semantic* adversarial perturbations. Our study motivates new approaches to robust training which can resolve this issue.

Acknowledgements

Part of this research was conducted while Eric Yeats was supported by a DOE-SCGSR Award at Oak Ridge National Laboratory (ORNL). Hence, this material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists, Office of Science Graduate Student Research (SCGSR) program. The SCGSR program is administered by the Oak Ridge Institute for Science and Education (ORISE) for the DOE. ORISE is managed by ORAU under contract number DE-SC0014664. All opinions expressed in this paper are the author's and do not necessarily reflect the policies and views of DOE, ORAU, or ORISE. This research was also supported in part by the U.S. Department of Energy, through the Office of Advanced Scientific Computing Research's "Data-Driven Decision Control for Complex Systems (DnC2S)" project.

This research is supported in part by U.S. Army Research funding W911NF2220025 and U.S. Air Force Research Lab funding FA8750-21-1-1015.

This research used resources of the Experimental Computing Laboratory (ExCL) at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

References

[1] Maximilian Augustin, Valentyn Boreiko, Francesco Croce, and Matthias Hein. Diffusion visual counterfactual explanations. *Advances in Neural Information Processing Systems*, 35:364–377, 2022. 1

[2] Valentyn Boreiko, Maximilian Augustin, Francesco Croce, Philipp Berens, and Matthias Hein. Sparse visual counterfactual explanations in image space. In *DAGM German Conf. on Pattern Recognition*, pages 133–148. Springer, 2022. 1

[3] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *ICML*, pages 1310–1320. PMLR, 2019. 2

[4] Chris Finlay and Adam M Oberman. Scaleable input gradient regularization for adversarial robustness. *Machine Learning with Applications*, 3:100017, 2021. 2

[5] Timo Freiesleben. The intriguing relation between counterfactual explanations and adversarial examples. *Minds and Machines*, 32(1):77–109, 2022. 1

[6] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 2

[7] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 1

[8] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 2

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2

[10] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *Advances in neural information processing systems*, 32, 2019. 1

[11] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 2, 1

[12] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 2, 1

[13] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 2, 1

[14] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 2

[15] Martin Pawelczyk, Chirag Agarwal, Shalmali Joshi, Sohini Upadhyay, and Himabindu Lakkaraju. Exploring counterfactual explanations through the lens of adversarial examples: A theoretical and empirical analysis. In *International Conference on Artificial Intelligence and Statistics*, pages 4574–4594. PMLR, 2022. 1

[16] Andrew Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Proceedings of the AAAI conference on artificial intelligence*, 2018. 1

[17] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intell.*, 1(5):206–215, 2019. 1

[18] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 2, 1

[19] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 2

[20] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv:1805.12152*, 2018. 1, 4

[21] Sahil Verma, John Dickerson, and Keegan Hines. Counterfactual explanations for machine learning: A review. *arXiv preprint arXiv:2010.10596*, 2, 2020. 1

[22] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 1

[23] Eric Yeats, Frank Liu, David Womble, and Hai Li. Nashae: Disentangling representations through adversarial covariance minimization. In *European Conference on Computer Vision*, pages 36–51. Springer, 2022. 1

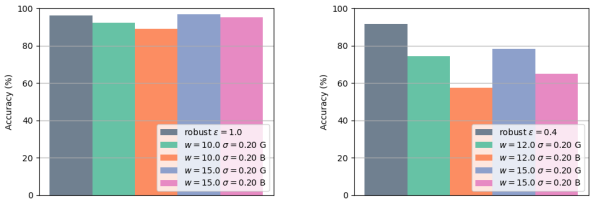
[24] Eric Yeats, Cameron Darwin, Frank Liu, and Hai Li. Adversarial estimation of topological dimension with harmonic score maps. *arXiv preprint arXiv:2312.06869*, 2023. 2

[25] Eric C Yeats, Yiran Chen, and Hai Li. Improving gradient regularization using complex-valued neural networks. In *International Conference on Machine Learning*, pages 11953–11963. PMLR, 2021. 2

[26] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 2, 1

Do Counterfactual Examples Complicate Adversarial Training?

Supplementary Material



(a) CIFAR10 Standard Accuracy (b) SVHN Standard Accuracy

Figure 6. Comparison of the accuracy attained by standard models on the CEs. Best viewed in color.

6. Reproducibility Information

L^2 distance for the CEs is defined as $\|x - x_{CE}\|_2$, where x is the original sample from which the CE x_{CE} was generated. L^0 distances ($\|x - x_{CE}\|_0$) employ an element-wise threshold of 0.02 for each pixel difference value, and the reported value is normalized (the thresholded L^0 distance is divided by the dimension of x). CE quality is measured by the accuracy of a standard classifier in predicting y_{CE} given x_{CE} (depicted in Fig. 6). On SVHN there appears to be sizeable drop from robust CE accuracy to the Boltzmann CEs. This may be due to several reasons. First, the $\epsilon = 0.4$ robust classifier may be making adversarial changes (rather than semantic), biasing its quality measure higher at that L^2 distance. Second, diffusion models tend to ignore class conditioning information when their inputs are simple (like those of SVHN), and they operate instead as unconditional denoising functions. This would cause the CE generation method to fail. Visual inspection of the CEs for SVHN indicates that the $\varepsilon = 0.4$ classifier is making some adversarial (non-semantic) changes, and the diffusion method fails in some cases. Please refer to the end of the supplementary material for visual depictions of CEs.

Robust WideResNet Experiments

2×WideResNet-40 models [26] are adversarially trained with PGD(8) [12] in L^2 on CIFAR10 [11] and SVHN [13] for 100 epochs and learning rate $1e-3$ using the Adam optimizer $\beta = (0.9, 0.999)$. For standard models ($\varepsilon = 0$), dropout with rate 0.3 is used. The data augmentations for CIFAR10 are random horizontal flips and random crops with padding 4. No augmentations are used for SVHN. The clean accuracies of the trained models are listed in tables (1) and (2). All adversarial attacks are PGD(8).

We observed that robust training on SVHN would collapse for $\varepsilon \geq 0.5$, so only $\varepsilon \leq 0.4$ experiments are reported. Coincidentally, $\varepsilon \sim 0.5$ is half the distance of a large amount of different-class CEs to their original data on SVHN. We hypothesize that the presence of many CEs at the L^2 distance 1 may be related to this training collapse phenomenon, as an adversarial budget of $\varepsilon \sim 0.5$ could make the source class of perturbed training data highly ambiguous.

Diffusion Models

In all experiments using diffusion models, we employ the variance-preserving (VP) score-based models (SBM) of Song et al. [18]. The SBM architecture follows a U-net structure with four symmetric stages of two ResNet blocks in each encoding or decoding stage. Downsampling (and upsampling, respectively) occurs in the innermost three stages (i.e., stages 2, 3, 4). 128 channels (features) are used, and the number of features used is doubled to 256 for stages 2, 3, and 4. Attention is applied at the center of the U-net and after the first downsampling stage and before the last upsampling stage. The SBMs use the SiLU activation function [7] and GroupNorm [22] with a group size of 32. Training on CIFAR10 or SVHN occurs for 1 million iterations of batch size 128 with a learning rate of $2e-4$ and Adam optimizer $\beta = (0.9, 0.999)$. We use a learning rate warmup for 5000 iterations and gradient clipping with norm 1. Class conditioning is provided as a learnable embedding which is added to the time condition embedding. An additional learnable null embedding signifies that *no* class information is provided. During conditional SBM training, class conditions are dropped and replaced with this null embedding at a rate of 30%.

7. CE Generation Method

CEs were generated by sampling from a sequence of unnormalized distributions represented by the product of the data distribution (rep. by diffusion model) and an independently diffused neighborhood distribution. The diffusion model used is a variance-preserving (VP) score-based model of Song et al. [18] and the sampling strategy for generation used an Euler-Maruyama predictor with 1000 discrete steps.

Since the density at each step in the sequence is represented as the product of the diffused data distribution and an independently diffused neighborhood distribution, sampling amounts to adding the conditional diffusion score (and classifier free guidance) with the analytic neighborhood dif-

Table 1. Accuracy of CIFAR10 Models on the CIFAR10 test set

ε	CLEAN ACCURACY (%)	PGD(8, 0.5) ACCURACY (%)
0	**93.08**	5.75
0.1	91.33	45.06
0.5	84.87	**54.73**
1.0	76.23	54.53
1.5	66.21	52.34
2.0	58.37	47.53

Table 2. Accuracy of SVHN Models on the SVHN test set

ε	CLEAN ACCURACY (%)	PGD(8, 0.2) ACCURACY (%)
0	**96.75**	77.46
0.1	95.63	86.42
0.2	95.41	**86.84**
0.4	93.09	85.43

fusion score. Sampling then proceeds as normal (see [18]) with the augmented score.

Robust Model CEs For CE generation with robust models, we push data in the direction of targeted gradients with a step size of 0.05 until a targeted class confidence of 0.9 or a maximum of 200 steps is reached. At each step, we clip the pixels of the image to remain in [0, 1].

Boltzmann-Inspired Distribution

The density of the 1D Boltzmann-inspired distribution proposed in this work is given by:

$$b(x) = \frac{1}{\sqrt{2}\,\sigma_{CE}} \exp\left(-\frac{\sqrt{2}\,|x-\mu_{CE}|}{\sigma_{CE}}\right), \tag{7}$$

where μ_{CE} is its mean and σ_{CE} is its standard deviation. Scaling samples of this distribution with $\alpha_t > 0$ amounts to sampling from a new distribution with mean $\alpha_t\mu_{CE}$ and standard deviation $\alpha_t\sigma_{CE}$. Defining $y_t = x - \alpha_t\mu_{CE}$, we convolve this distribution with a Gaussian distribution of mean $\mu = 0$ and scale σ_t to yield the (diffusion) time-dependent distribution:

$$\begin{aligned} b_t(x) = & \\ & \frac{\exp\left(\frac{\sigma_t^2}{\alpha_t^2\sigma_{CE}^2} - \frac{y_t}{\sqrt{2}\alpha_t\sigma_{CE}}\right)\operatorname{erfc}\left(\frac{\sigma_t}{\alpha_t\sigma_{CE}} - \frac{y}{\sqrt{2}\sigma_t}\right)}{2\sqrt{2}\alpha_t\sigma_{CE}} \\ + & \frac{\exp\left(\frac{\sigma_t^2}{\alpha_t^2\sigma_{CE}^2} + \frac{y_t}{\sqrt{2}\alpha_t\sigma_{CE}}\right)\operatorname{erfc}\left(\frac{\sigma_t}{\alpha_t\sigma_{CE}} + \frac{y}{\sqrt{2}\sigma_t}\right)}{2\sqrt{2}\alpha_t\sigma_{CE}}. \end{aligned} \tag{8}$$

Taking the logarithm of $b_t(x)$ and differentiating with respect to x, we have:

$$\begin{aligned} \nabla_x \log b_t(x) = & \\ & \frac{\sqrt{2}}{\alpha_t\sigma_{CE}}\left(e^{\frac{\sqrt{2}}{\alpha_t\sigma_{CE}}y_t}\operatorname{erfc}\left(\frac{\sigma_t}{\alpha_t\sigma_{CE}} + \frac{y_t}{\sqrt{2}\sigma_t}\right)\right. \\ & \left. - e^{-\frac{\sqrt{2}}{\alpha_t\sigma_{CE}}y_t}\operatorname{erfc}\left(\frac{\sigma_t}{\alpha_t\sigma_{CE}} - \frac{y_t}{\sqrt{2}\sigma_t}\right)\right) \Bigg/ \\ & \left(e^{\frac{\sqrt{2}}{\alpha_t\sigma_{CE}}y_t}\operatorname{erfc}\left(\frac{\sigma_t}{\alpha_t\sigma_{CE}} + \frac{y_t}{\sqrt{2}\sigma_t}\right)\right. \\ & \left. + e^{-\frac{\sqrt{2}}{\alpha_t\sigma_{CE}}y_t}\operatorname{erfc}\left(\frac{\sigma_t}{\alpha_t\sigma_{CE}} - \frac{y_t}{\sqrt{2}\sigma_t}\right)\right). \end{aligned} \tag{9}$$

Although this expression yields the exact scores for the diffused Boltzmann-inspired distribution in 1D, it is numerically unstable. We note that $\nabla_x \log b_t(x)$ is sigmoidal, and we elect to approximate it using the hardtanh function. One may consider the $\frac{\sqrt{2}}{\alpha_t\sigma_{CE}}$ expression to define the range of the score values, and the remainder of the expression defines a Gaussian-like score near the mean. Hence, we define our approximate scores as:

$$\nabla_x \log b_t(x) \approx \frac{\sqrt{2}}{\alpha_t\sigma_{CE}}\operatorname{hardtanh}(\gamma_t\, y_t), \tag{10}$$

where γ_t is the first term of a Maclaurin series estimate of $\nabla_x \log b_t(x)$ and is given by:

$$\gamma_t = \frac{\sqrt{2}}{\alpha_t \sigma_{CE}} - \frac{\sqrt{2}}{\sigma_t \sqrt{\pi}} \left(\exp\left(\frac{\sigma_t^2}{\alpha_t^2 \sigma_{CE}^2} \right) \operatorname{erfc}\left(\frac{\sigma_t}{\alpha_t \sigma_{CE}} \right) \right)^{-1}. \tag{11}$$

With $u = \frac{\sigma_t}{\alpha_t \sigma_{CE}}$, the expression $\frac{\exp(-u^2)}{\operatorname{erfc}(u)}$ is numerically unstable for large values of u. However, beyond a certain point (e.g., $u \geq 20$), the function behaves as a linear function with slope $\sqrt{\pi}$. We avoid the numerical instability by switching to a linear approximation at $u = 20$.

The exact score function $\nabla_x \log b_t(x)$ and its approximation is displayed with $\alpha_t = 1$ for various values of σ_t and in figure 7. In our experiments, this 1D score function is applied element-wise to vector inputs, providing a sparsifying effect on CE generation.

For completeness, we include $\nabla_x^2 \log b_t(x)$, which was evaluated at $y_t = 0$ to yield the first term of the Maclaurin series of $\nabla_x \log b_t(x)$:

$$\begin{aligned}
\nabla_x^2 \log b_t(x) = \\
&\left[\frac{2}{\sigma_{CE}^2} \left(e^{\frac{\sqrt{2}}{\alpha_t \sigma_{CE}} y_t} \operatorname{erfc}\left(\frac{\sigma_t}{\alpha_t \sigma_{CE}} + \frac{y_t}{\sqrt{2}\sigma_t} \right) \right.\right. \\
&\left. + e^{-\frac{\sqrt{2}}{\alpha_t \sigma_{CE}} y_t} \operatorname{erfc}\left(\frac{\sigma_t}{\alpha_t \sigma_{CE}} - \frac{y_t}{\sqrt{2}\sigma_t} \right) \right) \\
&\left. - \frac{4\, e^{-\left(\frac{\sigma_t^2}{\alpha_t^2 \sigma_{CE}^2} + \frac{y_t^2}{2\sigma_t^2} \right)}}{\sqrt{\pi} \alpha_t \sigma_{CE} \sigma_t} \right] \Bigg/ \\
&\left[e^{\frac{\sqrt{2}}{\alpha_t \sigma_{CE}} y_t} \operatorname{erfc}\left(\frac{\sigma_t}{\alpha_t \sigma_{CE}} + \frac{y_t}{\sqrt{2}\sigma_t} \right) \right. \\
&\left. + e^{-\frac{\sqrt{2}}{\alpha_t \sigma_{CE}} y_t} \operatorname{erfc}\left(\frac{\sigma_t}{\alpha_t \sigma_{CE}} - \frac{y_t}{\sqrt{2}\sigma_t} \right) \right].
\end{aligned} \tag{12}$$

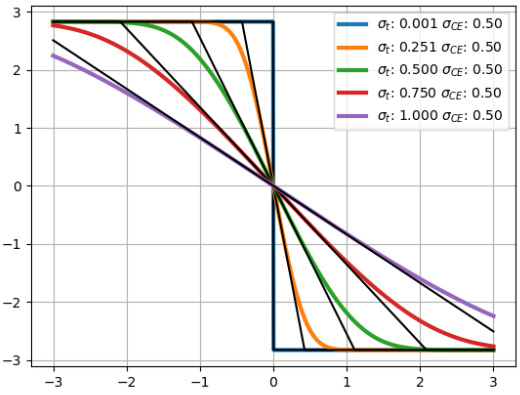


Figure 7. Comparison of true 1D Boltzmann-inspired scores with the proposed hardtanh approximation (black). The mean μ_{CE} is selected to be 0. Best viewed in color.

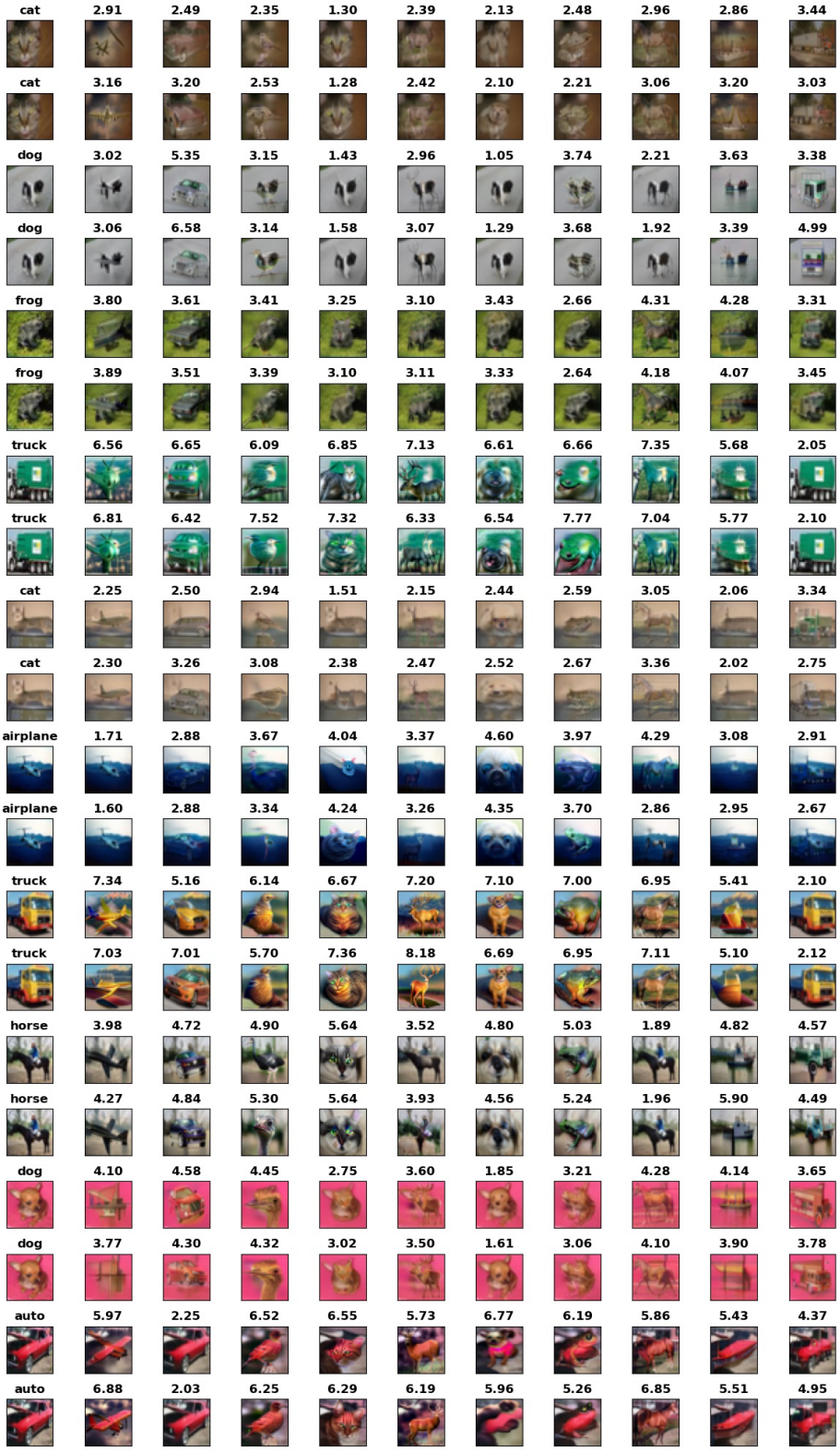


Figure 8. Boltzmann ($w = 15$ $\sigma_{CE} = 0.2$) CEs from the CIFAR10 training set. Leftmost column: original samples and class label. Other columns: CE and L^2 distance to the original sample.

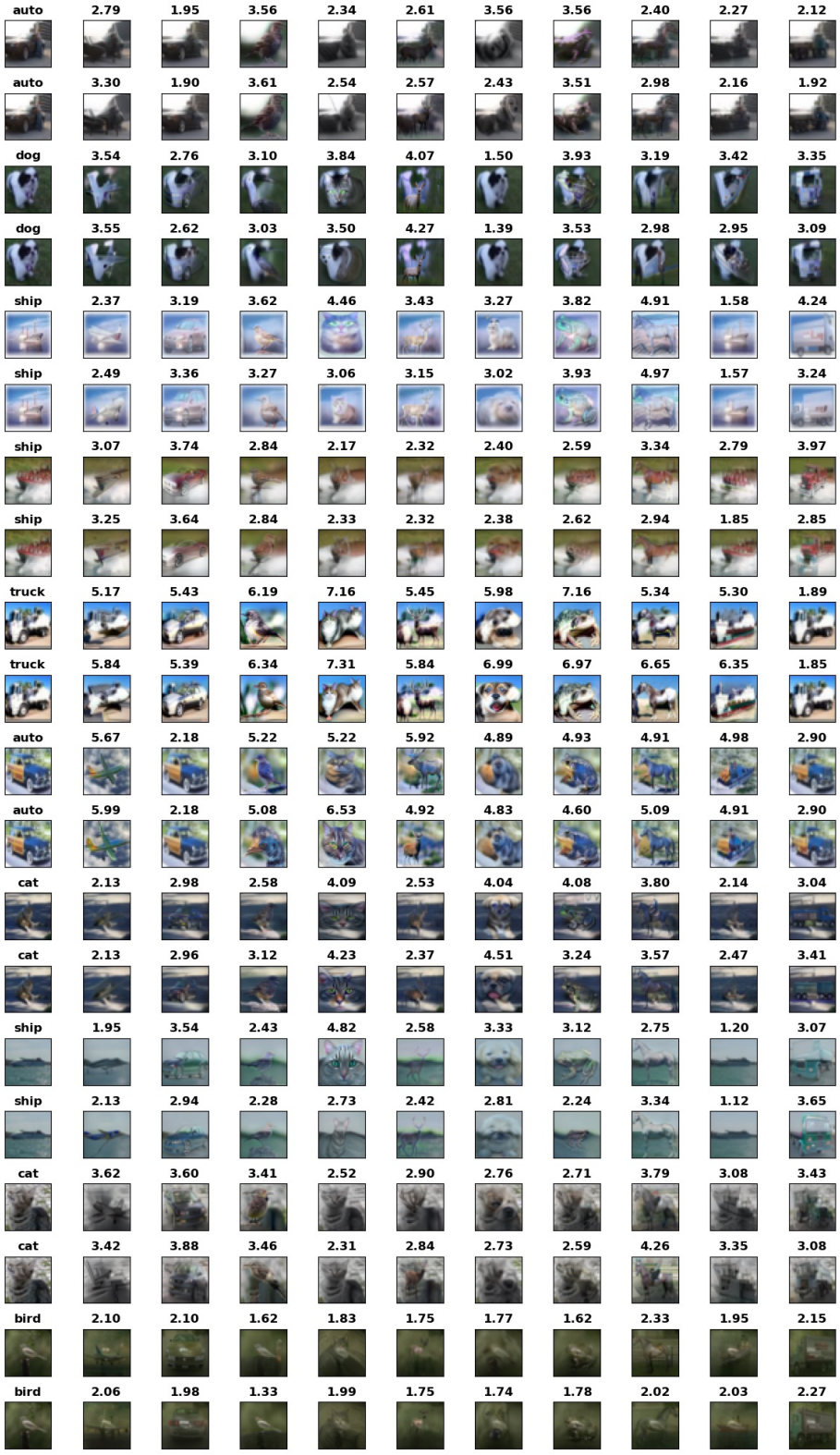


Figure 9. Boltzmann ($w = 15$ $\sigma_{CE} = 0.2$) CEs from the CIFAR10 training set. Leftmost column: original samples and class label. Other columns: CE and L^2 distance to the original sample.

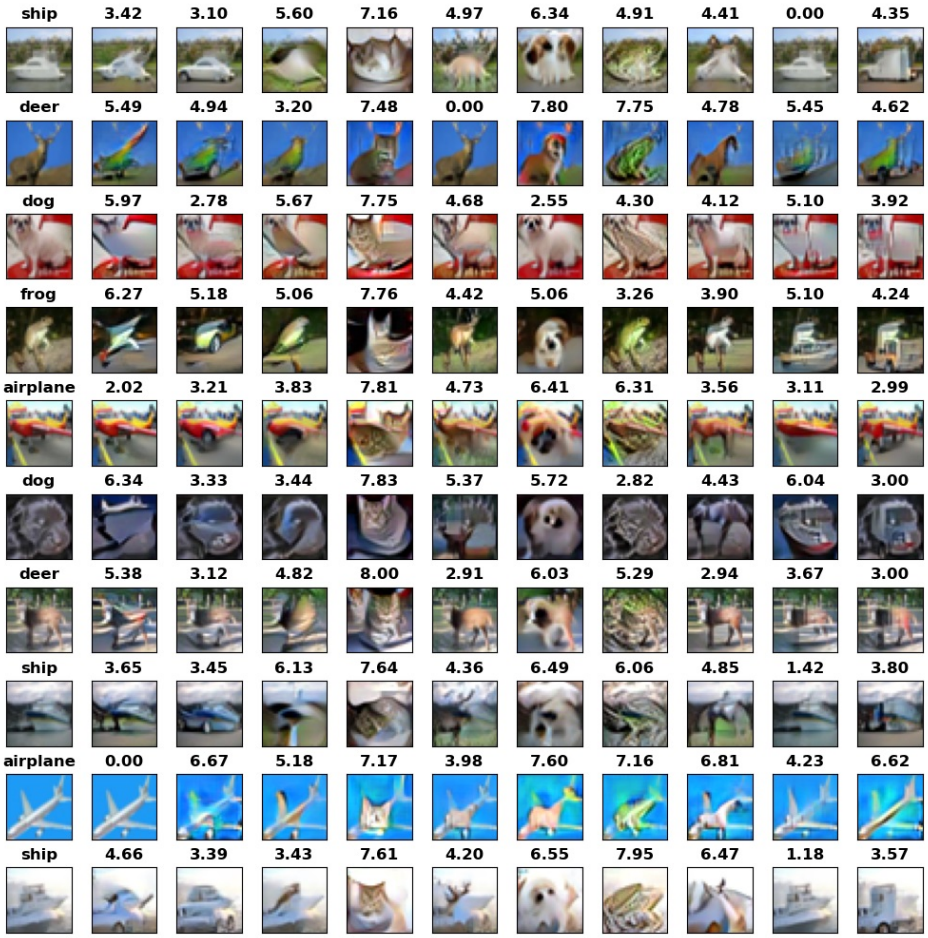


Figure 10. Robust model ($\varepsilon = 1$, confidence threshold 0.9) CEs from the CIFAR10 training set. Leftmost column: original samples and class label. Other columns: CE and L^2 distance to the original sample.

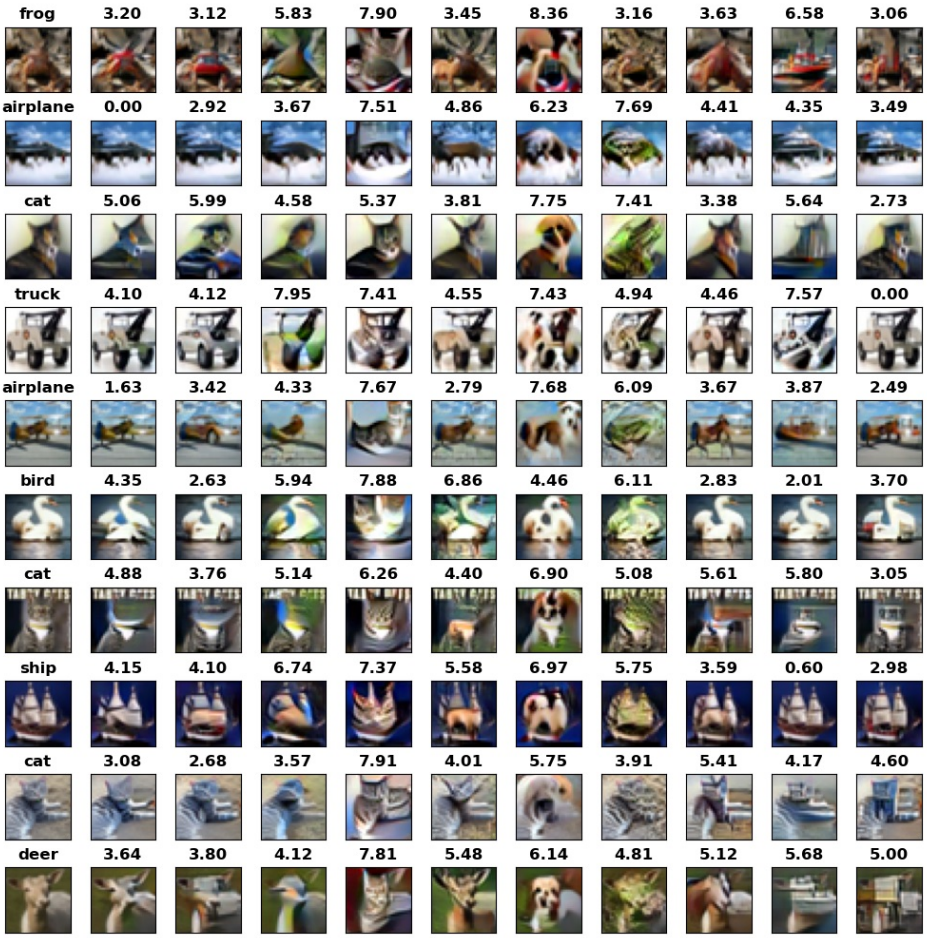


Figure 11. Robust model ($\varepsilon = 1$, confidence threshold 0.9) CEs from the CIFAR10 training set. Leftmost column: original samples and class label. Other columns: CE and L^2 distance to the original sample.

Figure 12. Boltzmann ($w = 15$ $\sigma_{CE} = 0.2$) CEs from the SVHN training set. Leftmost column: original samples and class label. Other columns: CE and L^2 distance to the original sample.

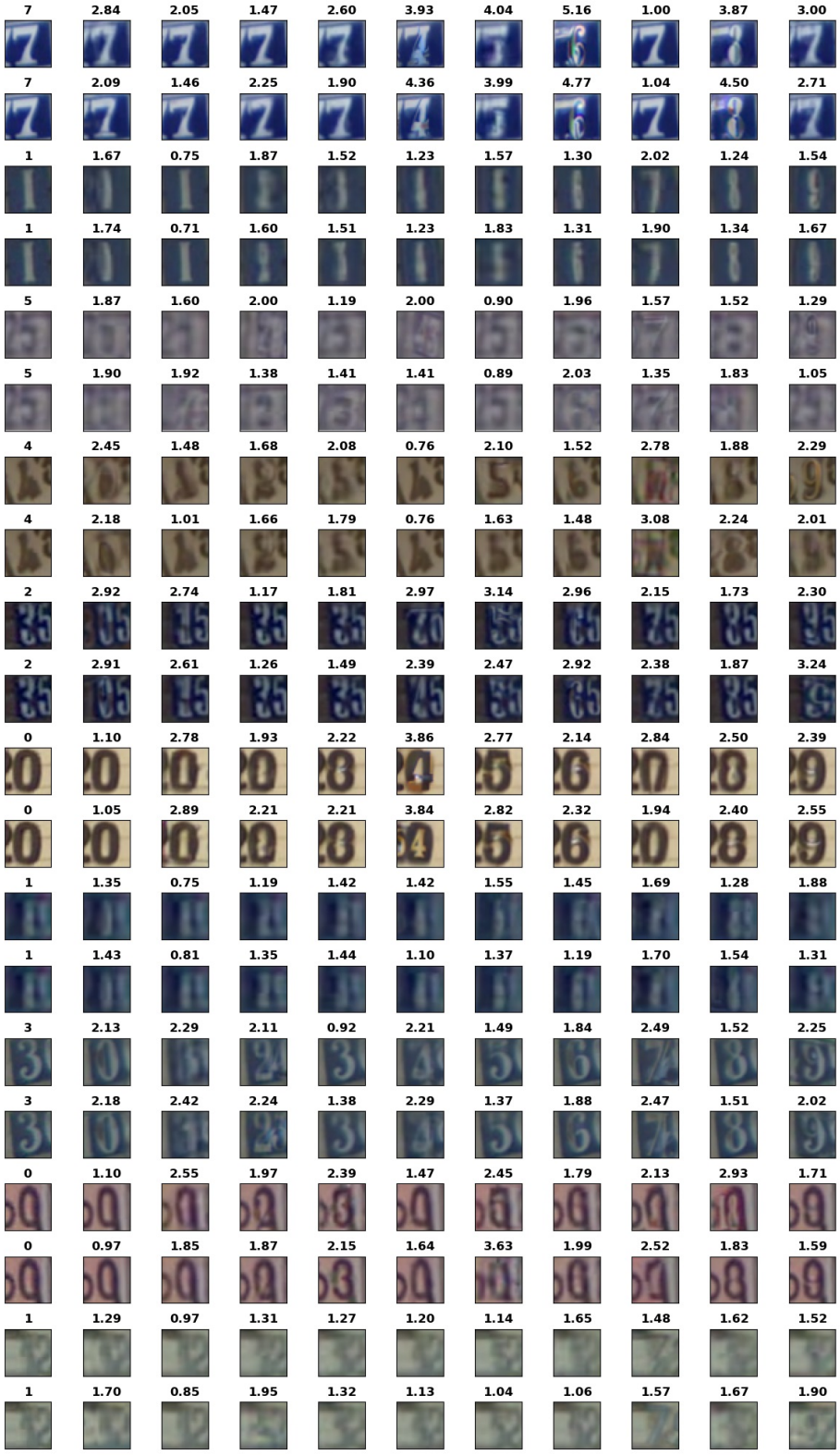


Figure 13. Boltzmann ($w = 15$ $\sigma_{CE} = 0.2$) CEs from the SVHN training set. Leftmost column: original samples and class label. Other columns: CE and L^2 distance to the original sample.

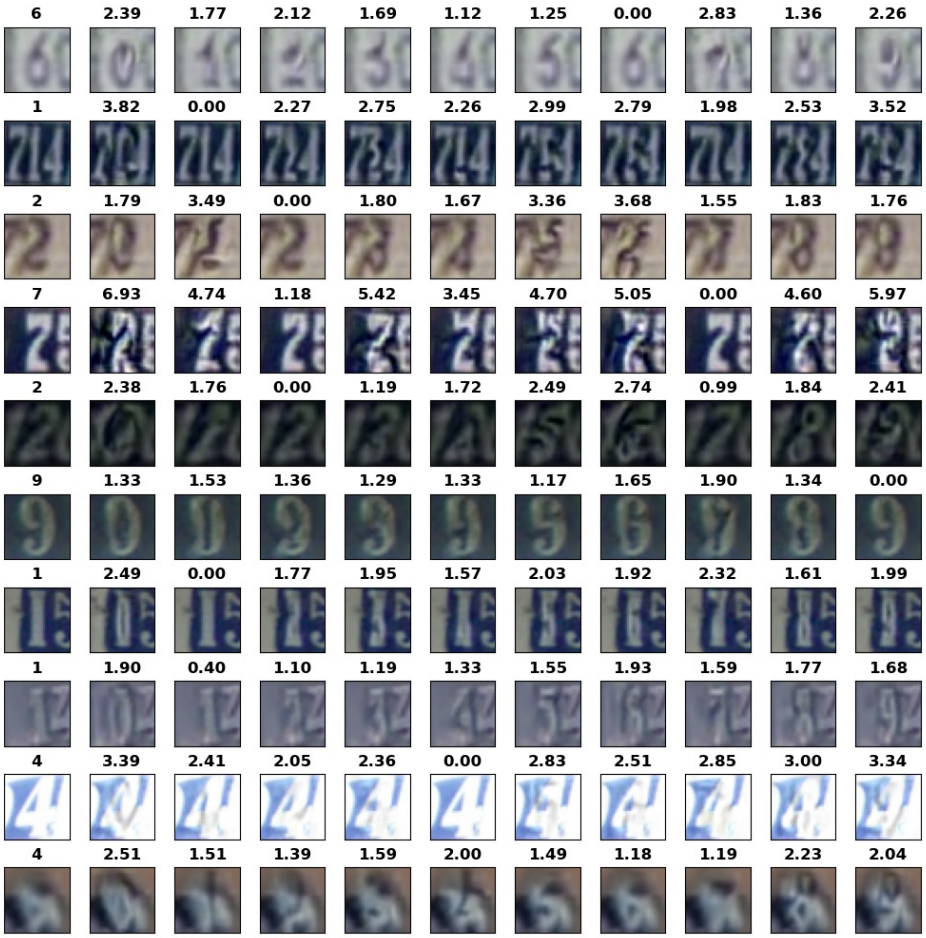


Figure 14. Robust model ($\varepsilon = 0.4$, confidence threshold 0.9) CEs from the SVHN training set. Leftmost column: original samples and class label. Other columns: CE and L^2 distance to the original sample.

Figure 15. Robust model ($\varepsilon = 0.4$, confidence threshold 0.9) CEs from the SVHN training set. Leftmost column: original samples and class label. Other columns: CE and L^2 distance to the original sample.