Data from Model: Extracting Data from Non-robust and Robust Models

Philipp Benz[*] Chaoning Zhang[*] Tooba Imtiaz In-So Kweon

Korea Advanced Institute of Science and Technology (KAIST)

Abstract

The essence of deep learning is to exploit data to train a deep neural network (DNN) model. This work explores the reverse process of generating data from a model, attempting to reveal the relationship between the data and the model. We repeat the process of Data to Model (DtM) and Data from Model (DfM) in sequence and explore the loss of feature mapping information by measuring the accuracy drop on the original validation dataset. We perform this experiment for both a non-robust and robust origin model. Our results show that the accuracy drop is limited even after multiple sequences of DtM and DfM, especially for robust models. The success of this cycling transformation can be attributed to the shared feature mapping existing in data and model. Using the same data, we observe that different DtM processes result in models having different features, especially for different network architecture families, even though they achieve comparable performance.

1. Introduction

In deep learning applications, such as image classification [6, 18], data is used to train deep neural network (DNN) models. This work explores the reverse process of generating data from the model, with one general question in mind: What is the relationship between data and model? This question cannot be addressed well by only focusing on the model training process, Data To Model (DtM) [8]. Thus we combine the widely adopted DtM with its reverse process of Data from Model (DfM). More specifically, we repeat the process of DtM and DfM in sequence and measure the accuracy over the original validation dataset. In this chaining process, we always assume access to only either the data or the model generated in the previous process. More specifically, in the DtM process, we can only access the data generated from the previous DfM process, and similarly, in the DfM process, we only access the model generated by the previous DtM process. This chain process is depicted in Figure 1.

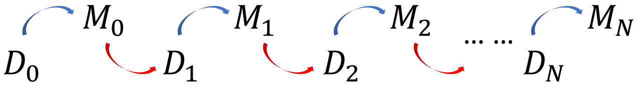

Figure 1: The chain of performing DtM and DfM repetitively. The blue arrows indicate the DtM process and the red arrows indicate the DfM process.

Our work is mainly inspired by [7] which attributes the success of adversarial examples to the existence of non-robust features (mappings) in a dataset. During the typical DtM process, these non-robust features are learned by a model which consequently has the same non-robust features. This indicates the feature mapping as the link between data and model. In their work, the original training dataset is adopted as the background image in the data extraction process [7]. In this work, we explore the possibility of retrieving a learned feature mapping from a trained model without the original training dataset, which makes the DfM process more meaningful. Moreover, we iterate the DtM and DfM process in sequence instead of just performing it once. Another aspect of this work is to explore whether such feature mappings are the same or different for different runs for the same or different architectures.

To decode the learned features of a model into a dataset without knowledge of the original training data, we adopt random substitute datasets as background to increase sample diversity and introduce virtual logits to model the logit behavior of DNNs. Our experiments show that we can obtain models with similar properties as the original model in terms of both accuracy and robustness. We showcase the effectiveness of our approach on MNIST and CIFAR10 for both non-robust and robust origin models.

2. Related Work

DNNs are vulnerable to small, imperceptible perturbations [13]. This intriguing phenomenon led to various attack [4, 2, 9, 11, 16] and defense methods [3]. Interpretations for the reason of the existence of adversarial examples have been explored in [4, 14, 17]. Ilyas *et al.* [7] attributed

[*] indicates equal contribution. Correspondence to pbenz@kaist.ac.kr and chaoningzhang1990@gmail.com

the phenomenon of adversarial examples to the existence of non-robust features. They introduce features as a mapping relationship from the input to the output. We adopt this definition and aim to extract this feature mapping from the model to the data. Model visualization methods [12, 10] can be seen as a DfM method, without training a new model on the extracted data. Such methods are commonly further exploited for model compression [5, 1] techniques. Instead of compressing models, [15] aims to compress an entire dataset into only a few synthetic images. Training on these few synthetic images, however, leads to a serious performance drop.

3. Methodology

Given a K-classification dataset $\mathcal{D}$ consisting of data samples $x \in \mathbb{R}^d$ and their corresponding true class $y \in [1, K]$, a DNN $\mathcal{M}_\theta$ (θ omitted from now on) parameterized through the weights θ is commonly trained via mini-batch stochastic gradient descent (SGD) to achieve $\arg\min_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}}[\mathcal{L}(\theta, x, y)]$. In this work, we term this process DtM (data to model) and we explore its reverse process of extracting data $\mathcal{D}'$ from model $\mathcal{M}$ (DfM). More specifically, starting from origin dataset $\mathcal{D}_0$ DtM results in origin model $\mathcal{M}_0$, with DfM, $\mathcal{D}_1$ can be extracted which leads to $\mathcal{M}_1$ through DtM and so on. During the DtM and DfM process, we assume having no access to the previous models and datasets, respectively.

To retrieve features from a model and store them in the form of data, we deploy the l_2-variant of projected gradient descent (PGD) [9]. Due to the absence of the original dataset, we leverage substitute images x_s from a substitute dataset $\mathcal{D}_s$, as well as virtual logits Z_v as the target values for the gradient-based optimization process. We specify the logit output of a classifier as $Z_\theta(\cdot)$ and use the l_2-loss between the network output logit and the virtual target logit $||Z_\theta(x_s) - Z_v||_2$ as the loss function optimized by PGD. The retrieved dataset consists of images $x'_s = x_s + \delta$, where δ indicates a vector optimized through PGD and x'_s lies in the range $[0, 1]$. The data samples x'_s and their respective output logit values $Z_\theta(x'_s)$ represent the new dataset.

After the DfM process, the retrieved dataset can be used in the DtM process, by training the model weights θ' with the l_2-distance between the previously stored ground truth logit vector and the output logit vector, $||Z_\theta(x'_s) - Z_{\theta'}(x'_s)||_2$.

We heuristically found a simple scheme of Gaussian distributed values $\mathcal{N}(\mu, \sigma)$ for the virtual logits. The highest logit is determined by $\mathcal{N}(20, 2)$, and the mean values of the remaining logit values are equally separated between $[-3, 3]$ with $\sigma = 1$. The order of logit values is chosen randomly to introduce diversity into the dataset.

In the above process, the origin model $\mathcal{M}_0$ is non-robust. Following [7] we also use a robust model for $\mathcal{M}_0$ obtained with adversarial training and repeat the chaining process.

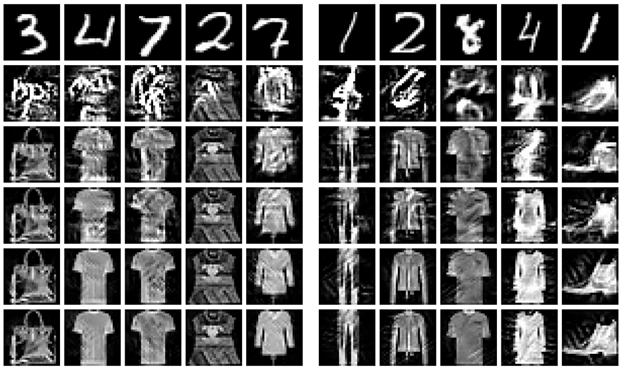

Figure 2: Qualitative results for the DfM process starting from a non-robust origin model (5 columns on the left) and a robust origin model (5 columns on the right). The first row indicates the origin dataset. The subsequent rows indicate the obtained dataset after the n-th DtM and DfM process. The results are shown for the LeNet architecture with Fashion MNIST as the background images.

4. Experiments

4.1. DfM and DtM in sequence

We sequentially apply DfM and DtM starting from the origin model $\mathcal{M}_0$. The non-robust and robust origin models $\mathcal{M}_0$ are obtained through standard and adversarial training, respectively, on LeNet for MNIST and VGG8 for CIFAR10. VGG8 refers to a VGG network with only one convolution layer between each max pooling operation. To obtain $\mathcal{D}_i$ from the preceding model $\mathcal{M}_{i-1}$, 500k images are extracted from $\mathcal{M}_{i-1}$ through the DfM process. For the background images, we choose Fashion MNIST and MS-COCO as background images for MNIST and CIFAR10, respectively. The generated images are shown in Figure 2. In subsequent DtM, dataset $\mathcal{D}_i$ is then used to train the model $\mathcal{M}_i$ which is an independent model of the same architecture. The accuracy for all models is reported on the original validation dataset. We present the results with five repetitions of this process in Table 1.

Qualitative results in Figure 2 show that the extracted images look totally different from the original images, due to which it might be tempting to expect the models trained on them will work poorly on the original validation dataset. Table 1, however, shows that comparable performance is achieved and this is due to similar feature mappings existing in the generated images despite the large visual discrepancy. Nonetheless, we observe that there is a general trend that the accuracies for both the non-robust and robust models decrease for each sequence of DfM/DtM. The accuracy increase by 1.1% from $\mathcal{M}_0$ to $\mathcal{M}_1$ for the robust VGG8

Table 1: Applying DtM and DfM in sequence for standard and adversarially trained models.

	LeNet (MNIST)		VGG8 (CIFAR10)	
	non-robust $\mathcal{M}_0$	robust $\mathcal{M}_0$	non-robust $\mathcal{M}_0$	robust $\mathcal{M}_0$
$\mathcal{M}_0$	99.5	98.7	92.2	87.0
$\mathcal{M}_1$	98.5	97.7	89.4	88.1
$\mathcal{M}_2$	96.6	96.1	80.1	82.5
$\mathcal{M}_3$	91.5	95.2	66.8	71.7
$\mathcal{M}_4$	87.4	94.2	52.5	58.7
$\mathcal{M}_5$	76.5	93.7	27.5	44.8

Table 2: Robustness evaluation of the models obtained during the chaining process for a non-robust (left) and robust (right) origin model. The results are reported for the LeNet architecture on MNIST.

	non-robust $\mathcal{M}_0$					robust $\mathcal{M}_0$				
ϵ	0	1	2	3	4	0	1	2	3	4
$\mathcal{M}_0$	99.5	74.2	4.5	0.1	0.1	98.7	91.0	58.9	10.6	0.7
$\mathcal{M}_1$	98.5	61.3	1.4	0.0	0.0	97.7	81.1	25.4	1.1	0.0
$\mathcal{M}_2$	96.6	31.2	0.2	0.0	0.0	96.1	71.2	12.4	0.1	0.0
$\mathcal{M}_3$	91.5	6.5	0.0	0.0	0.0	95.2	54.2	3.24	0.2	0.1
$\mathcal{M}_4$	87.4	1.1	0.0	0.0	0.0	94.2	31.8	1.0	0.1	0.3
$\mathcal{M}_5$	76.5	0.0	0.0	0.0	0.0	93.7	13.3	0.6	0.2	0.2

model is the only exception to this. For the non-robust origin model $\mathcal{M}_0$, the accuracy drop is trivial in the first few iterations of the DfM and DtM process and it becomes more observable in later iterations. For the robust origin model $\mathcal{M}_0$, the accuracy is retained better. For example, the robust LeNet only decreases by 5% from $\mathcal{M}_0$ to $\mathcal{M}_5$, while the non-robust LeNet decreases by 23%.

We further investigate the robustness to adversarial examples for the models from the DfM/DtM process. Therefore we evaluate the different retrieved MNIST models on adversarial examples generated with l_2-PGD under different $\epsilon \in \{0, 4\}$, 20 update steps and the corresponding step size calculated as $2.5\epsilon/\text{steps}$. Similar to the observed accuracy drop for clean images, a similar trend occurs when the model is under attack. However, the accuracy degradation seems to be more severe for robustness. For example, the accuracy of the non-robust LeNet drops from 74.2% to 0.0% for a relatively weak attack of $\epsilon = 1$ while the clean image accuracy drops from 99.5% to 76.5%. Similar behavior is observed for the models originating from the robust $\mathcal{M}_0$. It is worth mentioning that the models originating from the robust $\mathcal{M}_0$ are consistently more robust than their counterparts originating from the non-robust $\mathcal{M}_0$. After two subsequent DfM iterations starting from the robust $\mathcal{M}_0$, $\mathcal{M}_2$ still shows similar robustness as the standard non-robust model $\mathcal{M}_0$. This result suggests that DfM can be applied as an alternative to adversarial training.

Table 3: Cross-training of the extracted datasets from non-robust (top) and robust (bottom) models. The models were originally trained on CIFAR10. The robust models were adversarially trained with the l_2 variant of PGD. The rows indicate the model from which the data was extracted. The columns indicate the trained model. The values indicate the accuracy of the CIFAR-10 test dataset.

		VGG16	VGG19	ResNet18	ResNet50
non-rob.	VGG16 (93.8)	89.6	90.1	89.9	90.3
	VGG19 (93.6)	89.7	90.1	90.6	90.3
	ResNet18 (95.1)	87.9	88.0	89.7	89.6
robust	VGG16 (88.7)	90.3	90.5	90.3	90.5
	VGG19 (87.6)	87.9	88.0	88.0	88.1
	ResNet18 (90.2)	91.3	91.1	91.6	91.5

4.2. DfM and DtM on different architectures

The above analysis shows that DtM and DfM can be performed for the same and simple architecture with a limited performance drop. Here we apply DfM to the standard and adversarially trained CIFAR10 models and train different state-of-the-art architectures on the extracted data. For simplicity we stop the chaining process after obtaining $\mathcal{M}_1$. The results in Table 3 show that all model architectures can be successfully trained on the extracted data. Similar to Table 1, a performance drop is observed for the data extracted from the non-robust $\mathcal{M}_0$. For the robust $\mathcal{M}_0$, however, we observe that the retrained $\mathcal{M}_1$ consistently outperforms their corresponding $\mathcal{M}_0$ for both similar and different architectures, which is somewhat surprising.

4.3. Do different models learn different feature mappings?

Given the possibility to extract a certain feature mapping from a model, in this section, we analyze whether different models trained from the same origin dataset $\mathcal{D}_0$ have different feature mappings. To this end, we utilize 10k feature images extracted from models trained under the same conditions, and perform a cross-evaluation of the model accuracy on each other. The results for models with standard and adversarial training are reported in Table 4. We observe that the cross-evaluation accuracies are higher than random guess, which indicates that some shared feature mappings are learned. However, for both non-robust and robust models only in a few cases an accuracy higher than 50% is achieved. This phenomenon can also be observed when the extracted dataset was evaluated on an independently trained instance of the same architecture as the original architecture. The relatively low cross-evaluation accuracies illustrate that models from different runs learn different features

Table 4: Cross-evaluation of datasets extracted from non-robust (top) and robust (bottom) models. The models were originally trained on CIFAR-10. The robust models were obtained with adversarial training via the l_2 variant of PGD. The diagonal values were obtained for the same architecture but a different training run. The accuracy of the extracted data on the original model is 100%.

		VGG16	VGG19	ResNet18	ResNet50
non-robust	VGG16	43.2	40.4	36.5	30.5
	VGG19	50.1	48.7	45.4	37.8
	ResNet18	36.9	34.9	55.2	43.8
	ResNet50	41.3	40.6	60.0	62.6
robust	VGG16	48.5	43.6	45.4	44.0
	VGG19	38.3	38.9	36.9	36.0
	ResNet18	42.0	37.8	50.1	47.7
	ResNet50	35.4	31.9	39.7	35.7

which are not fully compatible with each other.

Another interesting observation is that model architectures from the same network family, VGG family for instance, seem to have more common feature mappings than different architectures. For example, for the feature images extracted from the standard ResNet50, the ResNet networks exhibit an accuracy of around 60%, while the VGG networks only show an accuracy of around 40%. This phenomenon is more prevalent in non-robust models than in robust models. Overall, the results show that different models learn different feature mappings from the same dataset even though they have comparable classification accuracy.

5. Conclusion

In this work we introduced the Data from Model (DfM) process, a technique to reverse the conventional model training process, by extracting data back from the model. A model trained on the generated dataset that look totally different from the original dataset can achieve comparable performance as their counterparts trained on the origin dataset. The success of this technique confirmed feature mapping as the link between data and model. Our work provides insight about the relationship between data and model as well understanding of model robustness.

References

[1] Kartikeya Bhardwaj, Naveen Suda, and Radu Marculescu. Dream distillation: A data-independent model compression framework. *arXiv preprint arXiv:1905.07072*, 2019. 2

[2] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Symposium on Security and Privacy (SP)*, 2017. 1

[3] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*, 2018. 1

[4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015. 1

[5] Matan Haroush, Itay Hubara, Elad Hoffer, and Daniel Soudry. The knowledge within: Methods for data-free model compression. *arXiv preprint arXiv:1912.01274*, 2019. 2

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision (ECCV)*, 2016. 1

[7] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 1, 2

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012. 1

[9] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018. 1, 2

[10] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2

[11] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[12] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks. 2015. 2

[13] 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. 1

[14] Thomas Tanay and Lewis Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*, 2016. 1

[15] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018. 2

[16] Chaoning Zhang, Philipp Benz, Tooba Imtiaz, and In-So Kweon. Cd-uap: Class discriminative universal adversarial perturbation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2020. 1

[17] Chaoning Zhang, Philipp Benz, Tooba Imtiaz, and In-So Kweon. Understanding adversarial examples from the mutual influence of images and perturbations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1

[18] Chaoning Zhang, Francois Rameau, Seokju Lee, Junsik Kim, Philipp Benz, Dawit Mureja Argaw, Jean-Charles Bazin, and In So Kweon. Revisiting residual networks with nonlinear shortcuts. In *British Machine Vision Conference (BMVC)*, 2019. 1