

Closer Look at the Transferability of Adversarial Examples: How They Fool Different Models Differently

Futa Waseda¹, Sosuke Nishikawa¹, Trung-Nghia Le^{2,3,4}, Huy H. Nguyen², and Isao Echizen^{1,2}

¹The University of Tokyo, Tokyo, Japan

²National Institute of Informatics, Tokyo, Japan

³University of Science, VNU-HCM, Vietnam

⁴Vietnam National University, Ho Chi Minh City, Vietnam

futa-waseda@g.ecc.u-tokyo.ac.jp; {nhhuy, iechizen}@nii.ac.jp

Abstract

Deep neural networks are vulnerable to adversarial examples (AEs), which have adversarial transferability: AEs generated for the source model can mislead another (target) model’s predictions. However, the transferability has not been understood in terms of to which class target model’s predictions were misled (i.e., class-aware transferability). In this paper, we differentiate the cases in which a target model predicts the same wrong class as the source model (“same mistake”) or a different wrong class (“different mistake”) to analyze and provide an explanation of the mechanism. We find that (1) AEs tend to cause same mistakes, which correlates with “non-targeted transferability”; however, (2) different mistakes occur even between similar models, regardless of the perturbation size. Furthermore, we present evidence that the difference between same mistakes and different mistakes can be explained by non-robust features, predictive but human-uninterpretable patterns: different mistakes occur when non-robust features in AEs are used differently by models. Non-robust features can thus provide consistent explanations for the class-aware transferability of AEs.

1. Introduction

Deep neural networks (DNNs) are vulnerable to adversarial examples (AEs), which are slightly perturbed by noises or patterns to mislead DNNs’ predictions [25, 8]. Since AEs can fool DNNs without affecting human perception, they are a severe threat to real-world DNN applications, even in the physical world [14]. However, existing defensive techniques remain vulnerable to AEs due to a lack of understanding of adversarial vulnerability. A critical property of AEs that needs to be better understood is their transferability: AEs generated using the source model may also fool other models [25, 8, 22, 23]. This transferability allows attackers to use a substitute model to generate AEs

Prediction based on AE with original label “cat” 

	(1)	(2)	(3)
F1: Source model		X: “dog”	
F2: Target model	✓: “cat”	X: “frog”	X: “dog”

Definitions of adversarial transferability

Non-targeted transferability	Non-transferable	Transferable	
Targeted transferability (target class: “dog”)	Non-transferable	Transferable	
Class-aware transferability	Unfooled	Different mistake	Same mistake

Figure 1. Example case of classifying transferability of adversarial example (AE). When the source model misclassifies the AE as “dog”, class-aware transferability of the AE to the target model is classified as (1) the target model correctly classified the AE as “cat” (unfooled), (2) it misclassified the AE as “frog” (different mistake), or (3) it misclassified the AE as “dog” (same mistake). In contrast, non-targeted and targeted transferability are defined in a binary way: whether the AE met the attackers’ objective or not.

to fool other unknown (target) models with different architectures or weights (i.e., a “black-box attack” [22]), which poses a considerable risk in our society. Understanding the transferability is essential to reducing the risk of black-box attacks and understanding the fundamental problem in current DNNs that cause adversarial vulnerability.

Many studies [25, 8, 22, 5, 15, 4] investigate (1) non-targeted and (2) targeted transferability, depending on the objective of adversarial attacks. Non-targeted transferability is defined for non-targeted attacks, which aim to fool a model regardless of the misclassified class; on the other hand, targeted transferability is defined for targeted attacks, which aim to fool a model towards a specific target class (illustrated in Figure 1). Primarily, previous works focused on explaining the non-targeted transferability [8, 15, 27, 11]: they showed that similarity between the source and target model allows AEs to fool them simultaneously. However,

Features in an adversarial example with original class “cat”

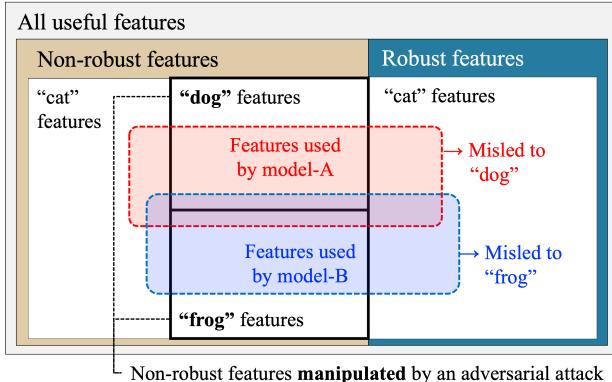


Figure 2. Our hypothesis for how AEs cause different models to make different predictions: non-robust features [11] can be used differently by models. Let us say a “cat” image is manipulated by an adversarial attack, and a part of non-robust features correlates with “dog” class (“dog” features), and another part correlates with “frog” class (“frog” features). When model-A uses more of the “dog” features than the “frog” features, and model-B does the opposite, model-A may predict the AE as “dog” and model-B may predict the AE as “frog.” Our work shows that non-robust features can cause both “different mistakes” and “same mistakes.”

it is unclear to which class the source and target models’ predictions are misled, which we refer to as “class-aware transferability.” Although the cases where different models misclassify an AE to the same class (“same mistake”) and different classes (“different mistake”) are different phenomena, we do not know what factors affect their proportion and what their mechanisms are.

With this motivation, we analyze the transferability from a novel perspective of “class-aware transferability.” We aim to understand the transferability phenomenon rather than simple risk evaluation. First, we perform a detailed analysis of the factors that affect class-aware transferability. We then tested whether our hypothesis explains the observed transferability phenomenon.

We analyze class-aware transferability under two conditions: model similarity and perturbation size. Class-aware transferability differentiates the cases where the target model misclassifies the AE as the same class as the source model (“same mistake”) and a different class than the source model (“different mistake”) (Figure 1). We present three main findings: (1) AEs tend to cause same mistakes, which is strongly connected to their capability of fooling target models (non-targeted transferability); (2) AEs cause different mistakes even on very similar models; (3) larger perturbations intend to cause same mistakes but do not reduce the ratio of different mistakes.

To provide a comprehensive explanation of the mechanisms AEs causing different mistakes and same mistakes, we provide a novel insight based on the theory of “non-

robust feature.” Ilyas et al. [11] showed that an AE may have predictive but human-imperceptible features (i.e., non-robust features) of the class to which a model was misled. Here, same mistakes are the logical consequence based on their theory; however, they do not explain different mistakes. In this work, we show that AEs that cause different mistakes can have the non-robust features of the two different classes to which the two models were misled. It indicates that the dependency of models on the learning features can cause different mistakes: when the non-robust features in AEs are used differently by different models, those models may classify the AEs differently (Figure 2). As we strengthen the theory of non-robust features [11], we support the claims that AEs are at least partly a consequence of learning “superficial cues” [12] or “shortcuts” [7].

Our contributions are summarized as follows.

- Our evaluation of class-aware transferability shows (1) that AEs tend to cause same mistakes, which is strongly connected to their capability of fooling target models (non-targeted transferability), (2) that different mistakes occur even between source and target models with high similarity, and (3) that larger perturbations do not reduce different mistakes, indicating a misalignment in misleading the source and target model towards the same class.
- We provide an explanation of the mechanisms causing different and same mistakes by extending the theory of non-robust features. Same mistakes are due to AEs having the non-robust features of the class to which the model was misled. When the manipulated non-robust features in the AEs are used differently by different models, those models may classify the AE differently.

2. Related Work

2.1. Non-targeted Adversarial Transferability

Non-targeted adversarial transferability is defined by whether or not the target model assigns a wrong class rather than the true (original) class. Szegedy et al. [25] showed that AEs transfer even when the source and target models have different architectures or are trained on a disjoint dataset. Papernot et al. [22] showed that non-targeted AEs transfer even between different machine learning methods such as DNNs, SVMs, and decision trees. Naseer et al. [21] generated AEs that transfer even between models trained on different image domains, such as cartoon and painting. Although these studies show intriguing transferability, how such AEs affect the target model’s predictions is unclear. This paper analyzes class-aware transferability, differentiating different and same mistakes.

The transferability of non-targeted adversarial attacks has been explained by the similarity between the source and

target models. Goodfellow et al. [8] showed that adversarial perturbations are highly aligned with the weight vectors of a model and that different models learn similar functions when trained on the same dataset to perform the same task. Liu et al. [15] revealed by visualization that transferability can arise from the similarity of the decision boundary that separates the true class and other classes. Tramer et al. [27] asserted that transferability appears when “adversarial subspaces” intersect between different classifiers. Ilyas et al. [11] showed that adversarial vulnerability can arise from non-robust features that are predictive but uninterpretable by humans and that transferability arises from the similarity of learned non-robust features between models. However, these do not clarify when and why different or same mistakes occur. We are the first to provide insightful explanations and discussions of their mechanisms based on the theory of non-robust features.

2.2. Targeted Adversarial Transferability

Targeted adversarial transferability is defined by whether or not the target model assigns the same class as the target class towards which the source model was attacked. Liu et al. [15] showed that, in contrast to non-targeted attacks, targeted attacks rarely transfer between models. Class-aware transferability allows us to directly compare the effect of non-targeted and targeted AEs, instead of using two different metrics of non-targeted and targeted transferability.

Several studies improved the transferability of targeted attacks by a similar idea: avoiding overfitting to the image or source model. Dong et al. [6] used momentum in iterations of a gradient-based adversarial attack; Xie et al. [31] increased input diversity when generating AEs, and Nasser et al. [20] generated class-specific AEs by using a generative adversarial network (GAN) to capture the global data distribution rather than overfitting the source model and the single image. However, these efforts did not provide a theoretical explanation of the mechanism causing same mistakes. A few studies explained same mistakes. Goodfellow et al. [8] hypothesized that the linear behavior of neural networks explains it. Such behavior is acquired by generalizing to solve the same task, thus resembling a linear classifier trained on the same data. Ilyas et al. [11] provided a widely accepted explanation: models can assign the same class by looking at similar non-robust features in AEs. However, these do not explain our observation that different mistakes occur between similar models regardless of the perturbation size. This paper provides a novel insight based on the theory of non-robust features to explain both different and same mistakes.

2.3. Adversarial Examples causing Different Predictions

Several works have studied how AEs cause different models to make different predictions, which corresponds to the cases of unfooled or different mistakes. Nakkiran et al. [19] generated AEs that only fool the source model and do not fool another model with the same architecture and trained on the same dataset. They claim that there exist AEs that exploit directions irrelevant to the true data distribution and thus irrelevant to features. Tramer et al. [27] used MNIST data with XOR artifacts to train linear and quadratic models and generated AEs that fooled only either. They hypothesized that AEs might not transfer when two models learn different features. Charles et al. [2] discussed from a geometric perspective and illustrated the decision boundaries and directions of the gradients when AEs fool only a linear classifier but not a two-layer ReLU classifier. Our hypothesis for how AEs cause different models to make different predictions can largely explain these cases and provide further interpretations.

2.4. Class-wise Robustness

Some works focused on class-wise robustness, which evaluates robustness for each class separately. A few works revealed that the class-wise robustness of models trained by adversarial training (AT) [16] is imbalanced, which can be interpreted by our non-robust features hypothesis (Figure 2). AT is a defense method that trains models incorporating AEs into training data. Tian et al. [26] revealed the imbalance in class-wise robustness of AT models and its fluctuation during training. Xia et al. [29] showed that the robustness of a specific vulnerable class improves by using the AEs weighted for that vulnerable class in AT. These findings are interpreted by our findings as follows: AT tries to force a model to ignore non-robust features in AEs. Therefore, the class-wise robustness in AT depends on which class of the non-robust features AEs contain, and its balance between classes can be a critical factor in determining class-wise robustness.

3. Adversarial transferability analysis

3.1. Overview

In this section, we evaluate the class-aware transferability of AEs by differentiating “different mistakes” and “same mistakes.” We aim to clarify the factors that affect class-aware transferability. Firstly, we analyze the effect of model factors by gradually changing the similarity between the source and target models. Different from Liu et al. [15], we not only compare models with different architectures but also with different or the same initial weights and models that are only in different training epochs. In addition, we use the metric of the decision boundary distance defined by

Tramer et al. [27] as a quantitative model similarity measurement. Secondly, we evaluate class-aware transferability by gradually increasing the perturbation size.

3.1.1 Class-aware Transferability

We classify transferability by whether the target model was “unfooled,” whether it made a “different mistake,” or a “same mistake.” The term “same mistake” was mentioned by Liu et al. [15] and was not the focus of their study.

The focus of our study is to evaluate how the malicious effect of AEs generated for a source model $F1$ can affect the classification results of an (unknown) target model $F2$. Therefore, we evaluate the transferability only for the AEs generated for the original images correctly classified by both $F1$ and $F2$ and successfully fooled $F1$:

$$(x', y, y1) \sim D'_{F1, F2} = \left\{ (x, y) \sim D \mid \begin{array}{l} F1(x) = y, \\ F2(x) = y, \\ F1(x') = y1 (\neq y). \end{array} \right\} \quad (1)$$

where an AE $x' = adv(x, y, F1)$ is generated by an adversarial attack $adv(\cdot)$ for the image-label pair (x, y) in the original set D , and $y1 (\neq y)$ denotes the wrong class that the source model misclassified. For these AEs, we define the metrics for class-aware transferability as follows.

1. Unfooled ratio: $\mathbf{P}_{(x', y, y1) \sim D'_{F1, F2}} [F2(x') = y]$
2. Fooled ratio: $\mathbf{P}_{(x', y, y1) \sim D'_{F1, F2}} [F2(x') \neq y]$
 - (a) Different mistake ratio:

$$\mathbf{P}_{(x', y, y1) \sim D'_{F1, F2}} [F2(x') = y2], \text{ where } y2 \notin \{y, y1\} \quad (2)$$
 - (b) Same mistake ratio:

$$\mathbf{P}_{(x', y, y1) \sim D'_{F1, F2}} [F2(x') = y1] \quad (3)$$

If the target model $F2$ classifies an AE x' as the true class y , it is unfooled; if it classifies the AE as a different wrong class $y2$ than the source model $F1$, it makes a different mistake; if it classifies the AE as the same wrong class $y1$ as the source model $F1$, it makes a same mistake.

Note that fooled ratio corresponds to non-targeted transferability. Same mistake ratio corresponds to targeted transferability only if $y1$ is the target class of a targeted attack.

3.1.2 Generation of Adversarial Examples

We examine both non-targeted attacks, which aim to fool a model regardless of the misclassified class, and targeted attacks, which aim to fool a model towards a specific target class y^{tar} . The optimization problems are formulated as

$$(\text{Non-targeted:}) \quad \underset{x'}{\operatorname{argmax}} L(x', y) \quad (4)$$

$$(\text{Targeted:}) \quad \underset{x'}{\operatorname{argmin}} L(x', y^{tar}) \quad (5)$$

where $L(\cdot)$ is a loss function and x' is the AE generated from the original input x . Both are subject to an l_p -bound, $\|x' - x\|_p < \epsilon$, so that x' remains sufficiently close to x .

We generate AEs using two gradient-based attacks: (1) the fast gradient method (FGM), which is an efficient method to generate l_p bounded AEs (the generalized version of the fast gradient sign method [8]), and (2) the projected gradient descent (PGD) method [16], which is the iterative version of FGM that generates stronger AEs. We provide results for other attacks, such as MIM [6], CW [1], and DeepFool [18], in supplementary material.

3.1.3 Measurement of Model Similarity

For quantitative measurement of the similarity between the source and target models, we use a method devised by Tramer et al. [27]. It measures the average distance of the decision boundary for N images between two models:

$$Dist(F1, F2) = \frac{1}{N} \sum_{i=1}^N |d(F1, x_i) - d(F2, x_i)| \quad (6)$$

where $d(f, x) = \operatorname{argmin}_\epsilon [f(x + \epsilon \cdot v) \neq y]$ is the minimum distance from an input x to the decision boundary of model f . The distance is calculated in the direction of the vector $v = \nabla_x L(x, y; F1) / \|\nabla_x L(x, y; F1)\|_2$, which is a normalized vector of the non-targeted adversarial perturbation generated for the source model $F1$. Therefore, this metric is directly related to the non-targeted transferability. We use this metric to analyze the relationship between class-aware transferability and model similarity indicated by non-targeted transferability. To calculate the equation 6, we randomly chose 1,000 images from the test set that all models correctly classified.

3.2 Evaluation Settings

3.2.1 Dataset

We used Fashion-MNIST [30], CIFAR-10 [13] and STL-10 [3] datasets, which are all ten-class datasets. We generated AEs l_2 -bounded by a specific ϵ (assuming that the pixels take values in the range $[0, 1]$). The PGD attack iterates for ten steps with step size $\alpha = \epsilon/5$. To generate targeted AEs, we randomly choose target classes for each image. For a fair comparison, we evaluated 2,000 random images from the test set that all models correctly classified.

3.2.2 Models

For Fashion-MNIST, we examined models with four simple architectures: fully-connected networks with 2 or 4 hidden layers (FC-2 or FC-4) and convolutional networks with 2 or 4 convolution layers followed by two fully-connected layers (Conv-2 or Conv-4). For CIFAR-10 and STL-10, we examined models with five popular architectures: VGG-16,

VGG-19 [24], ResNet-18, ResNet-34 [9], and DenseNet-121 [10]. We trained all models for 40 epochs for Fashion-MNIST, and 100 epochs for CIFAR-10 and STL-10 (details in supplementary material). For precise analysis, we independently trained three models for each architecture: two models trained using the same initial weight parameters and one trained using the other initial weights (when the initial weights are the same between models with the same architecture, the only difference is the randomness of the shuffled training data or dropout layers). In addition, we also compare early versions of the source model as target models at the i^{th} epoch. Hereinafter, models $F2$ with “(w:same)” or “(w:diff)” in their name are the models independently trained using the same or different initial weights as used for $F1$; “(v: i)” is $F1$ at the i^{th} epoch.

3.3. Results and Discussions

The results of FGM and PGD (ten-step) attacks against various datasets and models with both non-targeted and targeted objectives are shown in Figure 3. The $F2$ target models are sorted by quantitative similarity measurement $Dist(F1, F2)$ for each $F1$. $Dist(F1, F2)$ roughly corresponds to the qualitative similarity of the models; for example, when $F1$ was ResNet-18, $Dist(F1, F2)$ was the shortest for $F2$ in the ResNet architecture family (Figure 3b).

Figure 3 shows that the majority of the fooled ratio is the same mistake ratio. Moreover, the same mistake ratio strongly correlates with the fooled ratio: both fooled and same mistake ratios were higher when the source and target models were in the same architecture family (e.g., ResNet-18 and ResNet-34 are both in the ResNet family) and when the target models were early versions of the source models. The correlations between the fooled and same mistake ratios were greater than 0.99, and the correlations between $Dist(F1, F2)$ and the same mistake ratio were lower than -0.90 in all cases shown in Figure 3. It indicates that the fact that AEs tend to cause same mistakes is strongly connected to their capability to mislead target models’ predictions (non-targeted transferability).

Although AEs tend to cause same mistakes, we observed a non-trivial proportion of different mistakes even when the source and target models were qualitatively very similar (Figure 3). Even when the models had the same architecture and were trained from the same initial weights, the different mistake ratios for targeted FGM attacks were around 20% for STL-10 (Figure 3c). Moreover, different mistakes exist even between the source model and the source model at i^{th} epoch. These findings raise the question of what can explain the presence of different mistakes between similar models, which we address in a later section.

Figure 4 shows that, while same mistakes increase with larger perturbations, the different mistake ratio stays almost constant or increases. It indicates that there is a mis-

alignment between the ability of AEs to mislead the source model and target model towards a specific class that cannot be resolved simply by enlarging the perturbations.

To further interpret these class-aware transferability observations, we visualized the decision boundaries, as in Liu et al. [15] (Figure 5). We chose two directions of δ_1 , the non-targeted gradient direction of ResNet-18, and δ_2 , the random orthogonal direction. Both δ_1 and δ_2 were normalized to 0.02 by l_2 -norm. Each point (u, v) in the 2-D plane corresponded to the image $x + u\delta_1 + v\delta_2$, where x is the source image. For each model, we plot the classified label of the image corresponding to each point. First, we observe an area of different mistakes between the models with the same architecture or even models with only a 20-epoch difference. Second, the area of same mistakes is larger when the minimum distance to the decision boundary along the x-axis, $d(F_i, x)$, is similar between $F1$ and $F2$. It indicates that, while the similarity of the decision boundary separating the true and wrong classes results in non-targeted transferability [15], at the same time, the decision boundaries separating different wrong classes can also be similar and can result in same mistakes.

The strong connection between non-targeted transferability and same mistakes indicates the presence of non-robust features [11] in AEs: AEs can cause same mistakes by containing non-robust features that correlate with a specific class. However, the presence of different mistakes between similar models or when the perturbations are large is still poorly understood. We hypothesized that different mistakes occur when the usage of non-robust features is model-dependent, which we examine in a later section.

4. Non-robust feature investigation

4.1. Overview

Here, we provide the first possible explanation for different mistakes, one that can also explain same mistakes. Specifically, we provide insightful explanations and discussions based on the theory of non-robust features [11]. Same mistakes can be due to different models using similar non-robust features; we show that a different mistake can also arise from non-robust features.

We designed N-targeted attack to generate AEs that can cause different mistakes for different models. Then by using Ilyas et al.’s framework [11], we show that those AEs have non-robust features of two different classes to which those models were misled. Our results indicate that two models can make different mistakes when they use the non-robust features of those two classes differently. We thus conclude that the usage of non-robust features is a possible explanation for different and same mistakes: Same mistakes are due to AEs having the non-robust features of the class to which a model was misled; on the other hand, AEs may simultane-

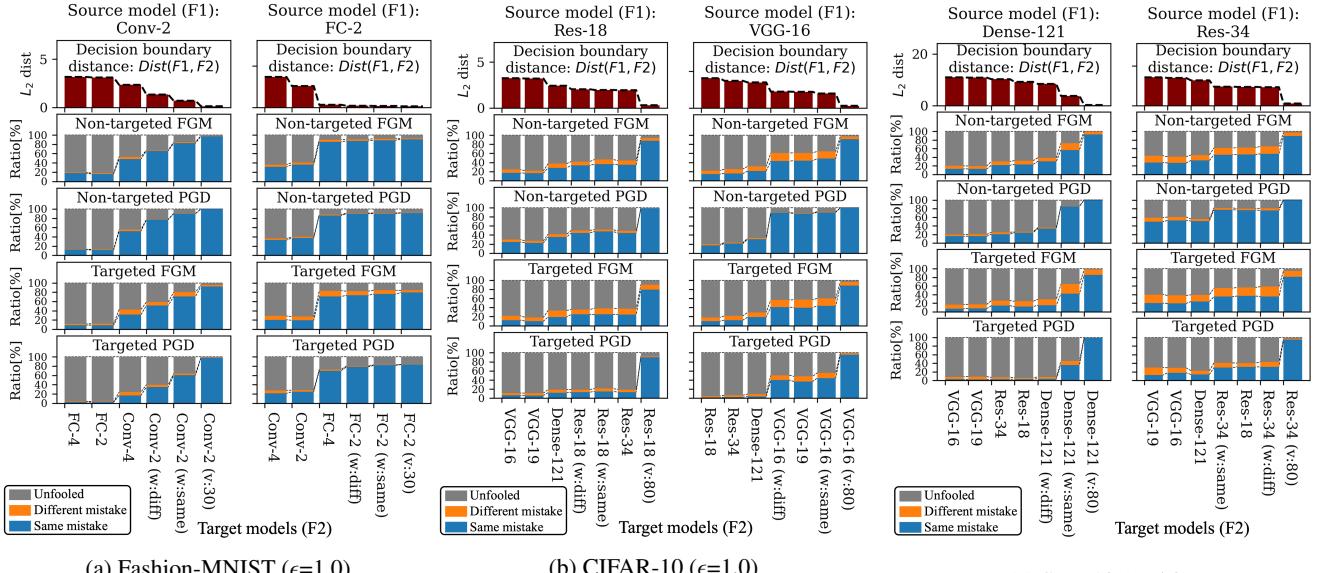


Figure 3. Class-aware transferability of adversarial attacks against various datasets and models. AEs were l_2 -bounded by the specific ϵ . Order of $F2$ is sorted by $Dist(F1, F2)$ (1st row) for each $F1$ so rightmost $F2$ was estimated to be more similar to $F1$.

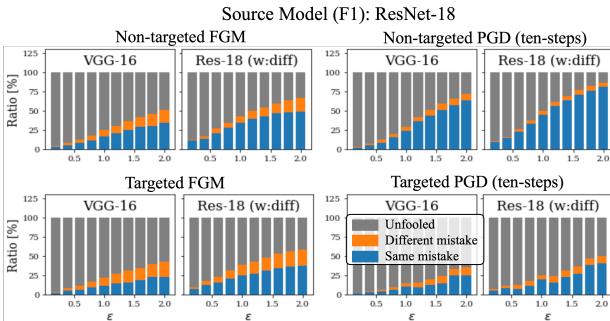


Figure 4. Class-aware transferability of AEs when size of perturbation ϵ was gradually changed (CIFAR-10).

ously have multiple non-robust features that correlate with different classes, and if models use them differently, they may classify the same AEs differently (Figure 2).

4.1.1 Experiment.

We aim to detect non-robust features of two different classes from the AEs that caused different mistakes. Suppose those AEs have non-robust features of two misleading classes simultaneously. In that case, we can assume that the models used the non-robust features differently (as in Figure 2). To demonstrate the presence of non-robust features in AEs, we used the framework for non-robust features described by Ilyas et al. [11]. We aim to detect non-robust features of two different classes from AEs, whereas Ilyas et al. [11] did it with only one class.

The flow of the experiment is illustrated in Figure 6.

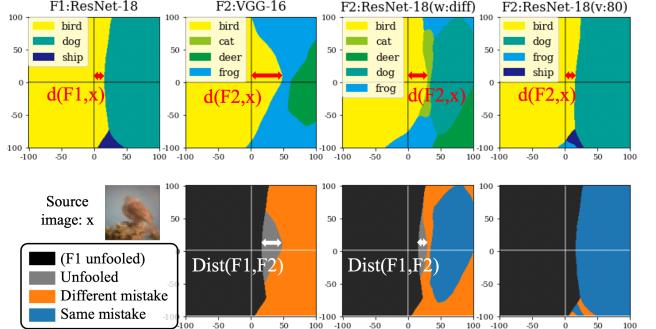


Figure 5. Visualization of decision boundaries for source image of “bird” in CIFAR-10. First row shows classification results; each color represents a certain class. Second row shows to which areas the three cases of class-aware transferability correspond. Distance from (0,0) point to closest decision boundary along x-axis corresponds to metric $d(F1, x)$ described in Section 3.1.3. Unit of each axis is 0.02 in l_2 distance.

First, we generate AEs that can cause different mistakes for models $F1$ and $F2$, on the original training set: each AE x' is generated from the original image x to mislead a model $F1$ to a target class y_1^{tar} and a model $F2$ to a target class y_2^{tar} . Then we created new (non-robust) training sets using two ways of relabeling the whole set of AEs X' , i.e., by either of the corresponding target classes $Y1$ or $Y2$ (note that X , X' , $Y1$, and $Y2$ are the collections of the datapoints x , x' , y_1^{tar} , and y_2^{tar} , respectively). Here, the target classes $Y1$ and $Y2$ are randomly chosen for each data point so that only the non-robust features of specific classes may correlate with the assigned labels, but other features have ap-

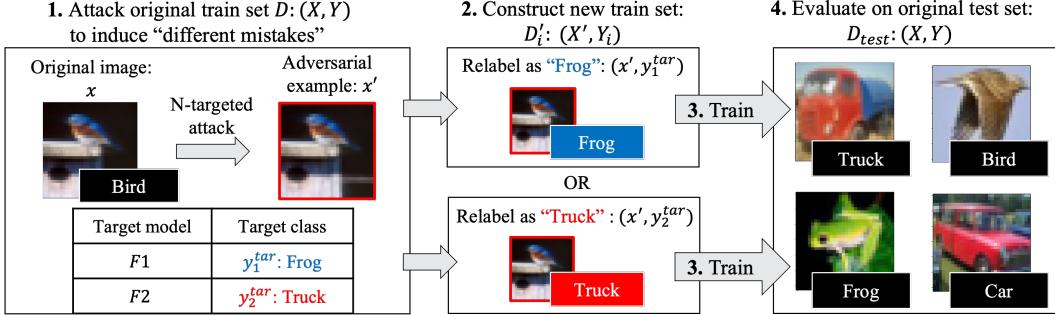


Figure 6. Illustration of experiment to test our hypothesis (Figure 2) that different mistakes can arise from AEs having non-robust features of two different classes to which two different models were misled. First, original training set is attacked by N-targeted attack to generate AEs that induce different mistakes for \$F_1\$ or \$F_2\$. Next, a new (non-robust) dataset is constructed by relabeling generated AEs as either \$Y_1\$ or \$Y_2\$, the target classes for \$F_1\$ or \$F_2\$. Finally, models are trained on new datasets and evaluated on original test set.

proximately zero correlation, as in Ilyas et al. [11]. Finally, we trained a model on the new training set (\$D'_1 : (X', Y_1)\$ or \$D'_2 : (X', Y_2)\$) and evaluated it on the original test set, \$D_{\text{test}} : (X, Y)\$. If both non-robust sets \$D'_1\$ and \$D'_2\$ were useful in generalizing to the original test set \$D_{\text{test}}\$, we can conclude that non-robust features of both classes (\$Y_1\$ and \$Y_2\$) are present in the same AEs at the same time.

We generated AEs that can cause different mistakes for \$F_1\$ and \$F_2\$ by using our extended version of a targeted attack, namely *N-targeted attack*. This attack is aimed at misleading model \$F_i\$ towards each target class \$y_i^{\text{tar}}\$. The objective of an N-targeted attack is represented as

$$\operatorname{argmin}_{x'} \sum_{i=1}^N L(F_i(x'), y_i^{\text{tar}}), \quad \text{s.t. } \|x' - x\|_p < \epsilon. \quad (7)$$

It simply sums up all the loss values for all target models. The optimization problem is solved iteratively using the same algorithm as PGD. The generated AEs for \$\{F_1, F_2\}=\{\text{ResNet-18, VGG-16}\}\$ are shown in Figure 7.

4.2. Experiment Settings

4.2.1 Non-robust Set

We construct non-robust sets for Fashion-MNIST, CIFAR-10, and STL-10, using the models used in Section 3. Non-robust training sets were constructed using an N-targeted attack based on PGD-based optimization with 100 steps with step size \$\alpha=0.1\$ (For STL-10, we generated ten AEs per image to increase the data size from 5,000 to 50,000). AEs were \$l_2\$-bounded by \$\epsilon\$ of 2.0, 1.0, and 5.0 for Fashion-MNIST, CIFAR-10, and STL-10.

Note that AEs generated by N-targeted attack could simultaneously lead predictions of models \$F_1\$ and \$F_2\$ towards different classes \$Y_1\$ and \$Y_2\$ at a high rate: 60% for Fashion-MNIST and over 90% for CIFAR-10 and STL-10. It means that it is easy in a white-box setting to generate AEs that cause different predictions for different models, which is particularly interesting.

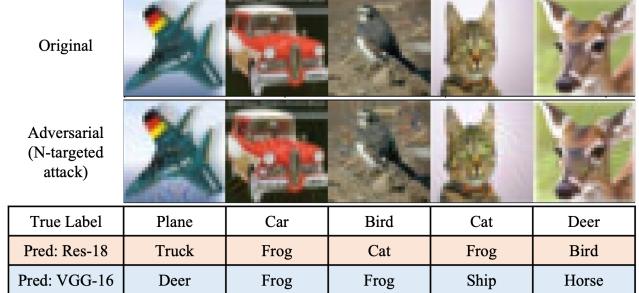


Figure 7. Examples of AEs (lower row) generated for images from CIFAR-10 (upper row) generated by N-targeted attack that was \$l_2\$-bounded by \$\epsilon=1.0\$. ResNet-18 and VGG-16 correctly classified the original images, whereas the AEs misled the models toward two random classes (For the entire set, over 90% were successful). The entire set of generated AEs comprises a non-robust set by relabeling them, as illustrated in Figure 6.

4.2.2 Training Models on Non-robust Set

The optimizer was SGD with momentum set to 0.9 and weight decay set to 0.0005, with learning rate decay. The initial learning rate, batch size, and data augmentation were optimized using a grid search. We trained FC-2 and Conv-2 (described in Section 3) for Fashion-MNIST, and ResNet-18 and VGG-16.bn (VGG-16 with batch normalization) for CIFAR-10 and STL-10.

4.3. Results and Discussions

Table 1 shows the test accuracies of the models trained on the constructed non-robust sets. For all pairs of attacked models \$F_1\$ and \$F_2\$, the test accuracies on the original test set \$(X, Y)\$ were higher than the random accuracy of 10% for both relabeling cases (\$Y_1\$ or \$Y_2\$). This result shows that the models could learn non-robust features of \$Y_1\$ by training on the non-robust set \$D'_1 : (X', Y_1)\$ and non-robust features of \$Y_2\$ by training on the non-robust set \$D'_2 : (X', Y_2)\$. In other words, it is shown that the generated AEs \$X'\$ had

Dataset	Non-robust set constructed for	Train set	Trained model	Test acc (X, Y)
Fashion-MNIST	F1: Conv-2 F2: FC-2	$D'_1: (X', Y1)$	Conv-2 FC-2	82.9 62.5
		$D'_2: (X', Y2)$	Conv-2 FC-2	80.3 75.4
CIFAR-10	F1: Res-18 F2: VGG-16	$D'_1: (X', Y1)$	Res-18 VGG-16.bn	51.3 53.9
		$D'_2: (X', Y2)$	Res-18 VGG-16.bn	10.2 71.0
	F1: Res-18 F2: Res-18 (w:same)	$D'_1: (X', Y1)$	Res-18 VGG-16.bn	50.1 54.1
		$D'_2: (X', Y2)$	Res-18 VGG-16.bn	59.2 58.9
		$D'_1: (X', Y1)$	Res-18 VGG-16.bn	24.0 25.4
	F1: Res-18 F2: VGG-16	$D'_2: (X', Y2)$	Res-18 VGG-16.bn	53.7 56.0
STL-10	F1: VGG-16 F2: VGG-16 (w:same)	$D'_1: (X', Y1)$	Res-18 VGG-16.bn	38.4 51.8
		$D'_2: (X', Y2)$	Res-18 VGG-16.bn	52.2 52.4

Table 1. Test accuracy on original test set when model was trained on non-robust sets. Non-robust set D'_i contains AEs generated by N -targeted attack and relabeled as Y_i , the target classes for model F_i . Since the random accuracy of 10-class dataset is 10%, we can say that models were generalized to original test set by training on non-robust sets. It shows that the generated AEs that induced different mistakes at a high rate contain multiple non-robust features that correlate with two different classes simultaneously.

multiple non-robust features that correlate with two different classes simultaneously. This result supports our hypothesis: different mistakes can arise when AEs have non-robust features of two classes and when models use them differently. It is interpreted that the ratio of different mistakes did not decrease with larger perturbations (Figure 4) because it did not resolve the misalignment of non-robust feature usage between models.

In addition, Table 1 shows that even models with the same architecture and initial weight parameters learn non-robust features differently. It suggests that learned non-robust features differ only with the stochasticity of updating the weight parameters caused by a shuffled training set or dropout layers, which explains the presence of different mistakes between models with high similarity in Figure 3.

5. Transferability of AEs generated for ensemble models

Section 4 indicates that different mistakes can occur when models use non-robust features differently. Therefore, different mistakes are expected to decrease when AEs contain “general” non-robust features used by many models.

To verify this, we generate targeted AEs for an ensemble of models: those AEs should contain only non-robust features that are “agreed” to be correlated with the target

classes by different models. Liu et al. [15] showed that attacking an ensemble model can improve targeted transferability; we reveal that non-robust features can explain it. In Table 2, we compare targeted AEs generated for a single model (i.e., Vanilla attack) and an ensemble model (i.e., Ensemble attack) using PGD. The source model F1 is ResNet-18, and the target model F2 is VGG-16. We confirmed that different mistakes decrease when Densenet-121 is additionally used in the Ensemble attack, while same mistakes increase. In contrast, the Ensemble attack increased the number of AEs that did not fool F1 (F1 unfooled): since the Ensemble attack tries to inject only non-robust features commonly used by models, it sacrifices the use of model-specific non-robust features used by F1.

Attack	F1: fooled			
	F1: unfooled	F2: unfooled	F2: different mistake	F2: same mistake
Vanilla	414	3910	250	426
Ensemble (+ Dense-121)	1988	1954	216 (-34)	842 (+416)

Table 2. Comparison between Vanilla and Ensemble targeted attack on CIFAR-10. AEs were l_2 -bounded by $\epsilon=1.0$, generated by targeted PGD (ten-step). Source model F1 is ResNet-18, and target model F2 is VGG-16. Each value shows the number of each case for randomly selected 5,000 images. For a fair comparison, we used the same target class for each image for both attacks.

6. Conclusion

We demonstrated that AEs tend to cause same mistakes, which is consistent with the fact that AEs can have non-robust features that correlate with a certain class. However, we further showed that different mistakes could occur between similar models regardless of the perturbation size, raising the question of how AEs cause different mistakes.

We indicate that non-robust features can explain both different and same mistakes. Ilyas et al. [11] showed that AEs can have non-robust features that are predictive but are human-imperceptible, which can cause same mistakes. In contrast, we reveal a novel insight that different mistakes occur when models use non-robust features differently.

Future work includes developing transferable adversarial attacks based on our findings: AEs should transfer when they contain non-robust features commonly used by different DNNs. In addition, since we do not conclude that all same mistakes and different mistakes are due to non-robust features, whether there is another mechanism is an important research question.

Acknowledgements: This work was partially supported by JSPS KAKENHI Grants JP16H06302, JP18H04120, JP20K23355, JP21H04907, and JP21K18023, and by JST CREST Grants JPMJCR18A6 and JPMJCR20D3, Japan.

References

- [1] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. Ieee, 2017.
- [2] Zachary Charles, Harrison Rosenberg, and Dimitris Papailiopoulos. A geometric perspective on the transferability of adversarial directions. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1960–1968. PMLR, 2019.
- [3] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.
- [4] Ambra Demontis, Marco Melis, Maura Pintor, Matthew Jagielski, Battista Biggio, Alina Oprea, Cristina Nita-Rotaru, and Fabio Roli. Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks. In *28th USENIX security symposium (USENIX security 19)*, pages 321–338, 2019.
- [5] Yinpeng Dong, Qi-An Fu, Xiao Yang, Tianyu Pang, Hang Su, Zihao Xiao, and Jun Zhu. Benchmarking adversarial robustness on image classification. In *CVPR*, pages 321–331, 2020.
- [6] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *CVPR*, pages 9185–9193, 2018.
- [7] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.
- [8] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.
- [9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.
- [10] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, pages 4700–4708, 2017.
- [11] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *NeurIPS Reproducibility Challenge*, 2019.
- [12] Jason Jo and Yoshua Bengio. Measuring the tendency of cnns to learn surface statistical regularities. *arXiv preprint arXiv:1711.11561*, 2017.
- [13] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.
- [14] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.
- [15] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *ICLR*, 2017.
- [16] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.
- [17] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.
- [18] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.
- [19] Preetum Nakkiran. A discussion of ‘adversarial examples are not bugs, they are features’: Adversarial examples are just bugs, too. *Distill*, 4(8):e00019–5, 2019.
- [20] Muzammal Naseer, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Fatih Porikli. On generating transferable targeted perturbations. In *ICCV*, 2021.
- [21] Muhammad Muzammal Naseer, Salman H Khan, Muhammad Haris Khan, Fahad Shahbaz Khan, and Fatih Porikli. Cross-domain transferability of adversarial perturbations. *NeurIPS*, 32:12905–12915, 2019.
- [22] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.
- [23] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519, 2017.
- [24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015.
- [25] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.
- [26] Qi Tian, Kun Kuang, Kelu Jiang, Fei Wu, and Yisen Wang. Analysis and applications of class-wise robustness in adversarial training. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1561–1570, 2021.
- [27] Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453*, 2017.
- [28] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *ICLR*, 2019.
- [29] Zhikang Xia, Bin Chen, Tao Dai, and Shu-Tao Xia. Class aware robust training. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3835–3839. IEEE, 2021.
- [30] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashionmnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.
- [31] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *CVPR*, pages 2730–2739, 2019.

A. Details of datasets

In Table 3, we provide the details of the datasets we used in the main paper.

Dataset	Class num.	Image size	Train	Test
Fashion-MNIST	10	(1,28,28)	60,000	10,000
CIFAR-10	10	(3,32,32)	50,000	10,000
STL-10	10	(3,96,96)	5,000	8,000

Table 3. Details of datasets we used in the main paper. Image size represents (channel, height, width) of images.

B. Adversarial Transferability Analysis

In this section, we provide supplementary results for our analysis of the class-aware transferability of AEs.

B.1. Details of evaluated models

All models were trained using the stochastic gradient descent (SGD) optimizer with a momentum of 0.9 and weight decay of 0.0005.

For Fashion-MNIST, we trained models at an initial learning rate of 0.01, which decayed 0.1 times at the 20th epoch with 40 epochs in total. Details of model architectures of FC-2/-4 and Conv-2/-4 are described in Table 4.

For CIFAR-10 and STL-10, we trained models at an initial learning rate of 0.01, which decayed 0.1 times at the 50th epoch with 100 epochs in total. Additionally, we used data augmentation techniques to train models for CIFAR-10 and STL-10 to prevent strong overfit.

B.2. Model Similarity Analysis

Here, we show the supplementary results of class-aware transferability of AEs. The results for Fashion-MNIST, CIFAR-10, and STL-10 are shown in Figure 8, Figure 9, and Figure 10, respectively. These results include the analysis of various models, which are not in the main paper. Furthermore, the analysis of the AEs generated by Momentum Iterative Method (MIM) [6] is added. We confirm the consistency of our findings for various models and attacks: the fact that AEs tend to cause same mistakes, but a non-trivial proportion of different mistakes exist is consistent.

We also evaluated two optimization-based adversarial attacks, CW [1] and Deepfool [18]. Figure 11 shows the results for CIFAR-10. Since these optimization-based attacks try to find minimum perturbations that are enough to fool the source model F1, they hardly transferred between models. Interestingly, AEs generated by the optimization-based attacks do not transfer even to the source models at 80th epochs. Therefore, fooled ratios were too small to analyze the class-aware transferability of the AEs.

B.3. Correlation Between Decision Boundaries' Distance and Class-aware Transferability

The correlations between $Dist(F1, F2)$, which is the quantitative measurement of the distance between models' decision boundaries, and the fooled ratio in Figure 12a. We confirmed that the distance metric of decision boundaries $Dist(F1, F2)$ is directly related to the non-target transferability, as stated by Tramer et al.[27]. In addition, the correlations between $Dist(F1, F2)$ and the same mistake ratio for all evaluated models and adversarial attacks are shown in Figure 12b. These results show that non-targeted transferability and same mistakes are strongly associated with each other.

B.4. Perturbation Size Analysis

The class-aware transferability of AEs when the perturbation size was gradually changed is shown in Figure 13a for the ResNet-18 source model and Figure 13b for the VGG-16 source model.

B.5. Decision Boundary Analysis

The visualization of the decision boundary for several different images is shown in Figure 14 for the ResNet-18 source model and Figure 15 for the VGG-16 source model.

B.6. Additional Adversarial Transferability Analysis on FGVC-Aircraft dataset

We additionally analyze the class-aware transferability of AEs generated for FGVC-Aircraft dataset [17] to understand the effect of class similarity and the number of classes. FGVC Aircraft dataset contains 10,000 images, which are split into 6,667 images for train set and 3,333 images for test set. It is composed of only images of aircrafts, which are labeled hierarchically: For example, the label level of “variant”, e.g. “Boeing 737-700”, has 100 classes which are finest visually distinguishable classes. The label level of “manufacturer”, e.g. “Boeing”, has 40 classes of different manufacturers. We trained all models at the initial learning rate of 0.01, which decayed 0.1 times at the 100th and 150th epoch with 200 epochs in total.

Since FGVC-Aircraft contains only aircraft images, images for different classes are visually more similar than, e.g., “cat” and “truck” images in CIFAR-10. Therefore, it is more likely that AEs cause different mistakes unless the AEs have a substantial effect on fooling models towards a specific class.

Figure 16 shows the class-aware transferability of AEs generated for FGVC-Aircraft (“variants”) dataset. Note that if AEs fool target models towards random directions, the proportion of same mistake ratio out of fooled ratio is 1% for a 100-class dataset.

We observe that non-targeted attacks caused same mistakes at a high rate. On the other hand, targeted attacks did not cause same mistakes as many as non-targeted attacks; however, still the proportions of same mistake ratio out of fooled ratio were more than 1%.

It is intriguing that, although FGVC-Aircraft (“variant”) has 100 classes, the same mistake ratio is high with non-targeted attacks. It indicates that the AEs generated by non-targeted attacks had strong effects on fooling models towards specific classes, which suggests the existence of non-robust features of the specific classes.

For targeted attacks, although targeted attacks still cause a moderate number of same mistakes, they are not as much as non-targeted attacks. For example, the proportions of same mistake ratio out of fooled ratio when {F1, F2}={ResNet-18, VGG-16} were 9.4%, 6.0%, and 9.8% for targeted FGM, PGD, and MIM, respectively (the leftmost column of Figure 16).

To understand how different mistakes occur with the FGVC-Aircraft dataset, we further analyzed different mistakes at a class-wise level (Figure 17). For non-targeted FGM (Figure 17a), it is observed that different mistakes tend to occur within the same “manufacturer”. It indicates that in the different mistake cases in non-targeted AEs, the non-robust features of a specific class were recognized as a different but similar class. On the other hand, targeted FGM (Figure 17b) caused different mistakes for other “manufacturers” more than non-targeted FGM. In general, targeted attacks are harder to perform than non-targeted attacks since targeted attacks are forced to aim at a specific class. Therefore, we think that this difficulty of targeted attacks can result in targeted attacks generating AEs with model-specific non-robust features for the source model, which are not likely to be perceived similarly by a target model. However, the differences between the mechanism and nature of targeted and non-targeted attacks are still not fully understood, which should be future work.

C. Non-robust Feature Analysis

C.1. Theory: The Difference in Learned Features Causes Different Behavior on Adversarial Attacks

In the paper, we showed that different models might classify AEs differently due to the different usage of non-robust features. In this section, we show a mathematical example of this phenomenon using a simple mathematical model proposed by Tspiras et al. [28].

C.1.1 Setup

As in Tspiras et al. [28], we consider a binary classification task in which the data consists of input-label pairs (x, y)

sampled from a distribution D as follows:

$$y \stackrel{u.a.r}{\sim} \{-1, 1\}, \quad x_1 = \begin{cases} +y & w.p. \ p \\ -y & w.p. \ 1-p \end{cases}, \\ x_2, \dots, x_{d+1} \stackrel{i.i.d}{\sim} N(\eta y, 1),$$

where $N(\mu, \sigma^2)$ is a normal distribution with mean μ and variance σ^2 , and $p \leq 0.5$. Features x include strongly correlated feature x_1 and weakly correlated features x_2, \dots, x_{d+1} with small coefficient η . Here, the features x_2, \dots, x_{d+1} are non-robust to perturbations with size η .

C.1.2 Weakly-correlated features suffice standard classification accuracy

Although x_1, \dots, x_{d+1} only weakly correlate, and each cannot be predictive individually, they can be used to acquire good standard accuracy. As shown in [28], a simple linear classifier

$$f_{avg}(x) := sign(w_{unif}^T x), \\ \text{where } w_{unif} := \left[0, \frac{1}{d}, \dots, \frac{1}{d} \right]$$

can achieve standard accuracy over 99% when $\eta \geq 3/\sqrt{d}$ (e.g. if $d=1000$, $\eta \geq 0.095$). Proof is shown below.

$$\begin{aligned} Pr[f_{avg}(x) = y] &= Pr[sign(w_{unif}^T x) = y] \\ &= Pr\left[\frac{y}{d} \sum_{i=1}^d N(\eta y, 1) > 0\right] \\ &= Pr\left[N\left(\eta, \frac{1}{d}\right) > 0\right] \\ &> 99\% \text{ (when } \eta \geq 3/\sqrt{d}) \end{aligned}$$

This means that even when features are weakly correlated, their collection could be predictable enough for classification.

C.1.3 Different usage of weakly-correlated features can cause different predictions

Next we think of classifiers f_A, f_B which have weights w_A, w_B as below.

$$\begin{aligned} f_A(x) &:= sign(w_A^T x), \\ \text{where } w_A &:= \frac{2}{d(d+1)} [0, 1, 2, \dots, d] \\ f_B(x) &:= sign(w_B^T x), \\ \text{where } w_B &:= \frac{2}{d(d+1)} [0, d, d-1, \dots, 1] \end{aligned}$$

These classifiers only use the weakly-correlated features, but they have a bias on weights, different from w_{unif} . The difference between these two classifiers is that the preference for using weakly correlated features is the opposite. These classifiers achieve a standard accuracy of over 99% when $\eta \geq \sqrt{\frac{6(2d+1)}{d(d+1)}}$ (e.g. if $d=1000$, $\eta \geq 0.11$). The proof for f_A is shown below (the same calculation also proves for f_B).

$$\begin{aligned} \Pr[f_A(x) = y] &= \Pr[\text{sign}(w_A x) = y] \\ &= \Pr\left[\frac{2y}{d(d+1)} \sum_{i=1}^d i \cdot N(\eta y, 1) > 0\right] \\ &= \Pr\left[N\left(\eta \frac{d(d+1)}{2}, \frac{d(d+1)(2d+1)}{6}\right) > 0\right] \\ &> 99\% \quad \left(\text{when } \eta \geq \sqrt{\frac{6(2d+1)}{d(d+1)}}\right) \end{aligned}$$

Now we think of an adversarial attack that perturbs each feature x_i by a moderate ϵ . For instance, if $\epsilon = 2\eta$, adversary can shift each weakly-correlated feature towards $-y$. Here, we consider the case in which only the first half of the weakly-correlated features are perturbed by $\epsilon = 2\eta$: we consider perturbed features x'_2, \dots, x'_{k+1} are sampled i.i.d. from $N(-\eta y, 1)$, where $k = d/2$ (for simplicity, suppose d is an even number and $d \gg 2$). Then the probability of f_A correctly predicting y is over 90% when $\eta \geq \sqrt{\frac{6(2d+1)}{d(d+1)}}$ (e.g. if $d=1000$, $\eta \geq 0.11$).

$$\begin{aligned} \Pr[f_A(x') = y] &= \Pr[\text{sign}(w_A x') = y] \\ &= \Pr\left[\frac{2y}{d(d+1)} \left(\sum_{i=1}^k i \cdot N(-\eta y, 1) + \sum_{i=k+1}^{2k} i \cdot N(\eta y, 1) \right) > 0\right] \\ &= \Pr\left[N\left(\eta k^2, \frac{d(d+1)(2d+1)}{6}\right) > 0\right] \\ &= \Pr\left[N\left(\eta \frac{d^2}{4} \sqrt{\frac{6}{d(d+1)(2d+1)}}, 1\right) > 0\right] \\ &> 90\% \quad \left(\text{when } \eta \geq \sqrt{\frac{6(2d+1)}{d(d+1)}}\right) \end{aligned}$$

In the same way, the probability of f_B correctly predicting y is less than 10% when $\eta \geq \sqrt{\frac{6(2d+1)}{d(d+1)}}$ (e.g. if $d=1000$, $\eta \geq 0.11$).

$$\begin{aligned} \Pr[f_B(x') = y] &= \Pr[\text{sign}(w_B x') = y] \\ &= \Pr\left[N\left(-\eta \frac{d^2}{4} \sqrt{\frac{6}{d(d+1)(2d+1)}}, 1\right) > 0\right] \\ &< 10\% \quad \left(\text{when } \eta \geq \sqrt{\frac{6(2d+1)}{d(d+1)}}\right) \end{aligned}$$

Therefore, it is proved that there exists a case in which the perturbed input x' is correctly predicted by f_A while

incorrectly predicted by f_B . This analysis shows that how each model puts weights on weakly-correlated features can determine the transferability of adversarial examples. Similarly, simply extending this analysis to a multi-class setting can theoretically show that there is a possibility to attack different models to cause different mistakes when the models use features differently.

C.2. Supplementary Results

Here, we provide supplementary results and details of the non-robust feature analysis.

C.2.1 N-targeted attack

Figure 18 describes the difference between vanilla targeted attack and the N-targeted attack. N-targeted attack aims to fool multiple models towards each specified target class. It simply adds up the gradients for all target models.

Table 5 shows the accuracy of models on the AEs generated by the N-targeted attack, which constructs non-robust sets.

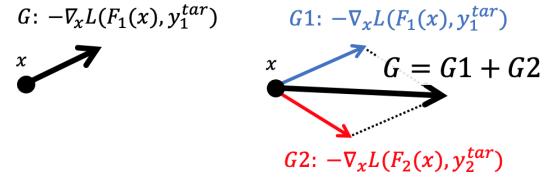


Figure 18. Difference between (a) targeted attack and (b) proposed N-targeted attack ($N=2$), which sums up all gradients for all target models ($G = G1 + G2$) and aims to mislead model $F1$ towards class y_1^{tar} and model $F2$ towards class y_2^{tar} .

C.2.2 Full Results and Optimized Hyperparameters

For CIFAR-10 and STL-10, we conducted a grid search to obtain the best hyperparameters for training models on the constructed non-robust sets. The grid search area of hyperparameters is shown in Figure 19. Initial learning rate, batch size, and level of data augmentation were optimized. The results and corresponding hyperparameters are shown in Figure 6, Figure 7, and Figure 8 for Fashion-MNIST, CIFAR-10, and STL-10, respectively.

C.2.3 Accuracy Curves

Train and test accuracy curves for training models on the constructed non-robust sets are shown in Figure 20 (CIFAR-10). Note that train accuracy represents accuracy on the constructed non-robust sets, which seem completely mislabeled for humans, and test accuracy represents accuracy on the original test set that is correctly labeled.

Following the experiment from Ilyas et al. [11], the accuracy numbers reported correspond to the last iteration since we cannot do meaningful early-stopping as the validation set itself comes from the constructed non-robust set and not from the true data distribution.

D. Potential Application of Our Findings

In this paper, we have mainly focused on the theoretical understanding of adversarial transferability. This section lists some potential applications to use our main findings.

D.1. Attack-side perspective

Using the N-targeted attack concept is one potential application. It can be used to attack systems with the primary classifier model and an AE detection model. Experiments showed that it might be possible to generate AEs with non-robust features that are recognized by the primary classifier but not by the AE detection model. Another potential application of our paper is to generate transferable AEs. Our paper suggests that AEs transfer when they have non-robust features that DNNs commonly recognize. Therefore, the promising direction to generate transferable AEs is to investigate how to find “commonly perceived” non-robust features by different DNNs.

D.2. Defense-side perspective

In general, our work further supports viewing adversarial vulnerability as a feature learning problem, as asserted by Ilyas et al. [11]: to reduce the adversarial vulnerability of DNNs, it is necessary to restrict DNNs from learning non-robust features that humans do not use. Our contribution is to support this view by showing that non-robust features can explain the transferability of AEs, even from the more detailed perspective of class-aware transferability. One specific approach our paper suggests is to ensemble models: it can alleviate the sensitivity to non-robust features learned by a particular model and become only sensitive to the non-robust features commonly learned by all models to be ensembled. In other words, the ensemble model may rely less on specific non-robust features than a single model, which can reduce adversarial vulnerability.

FC-2	FC-4	Conv-2	Conv-4
Linear: (784, 500) ReLU Linear: (500, 10)	Linear: (784, 500) ReLU Linear: (500, 200) ReLU Linear: (200, 100) ReLU Linear: (100, 10)	Conv2d: (1, 32, 3, 1) ReLU Conv2d: (32, 64, 3, 1) ReLU Conv2d: (64, 128, 3, 1) ReLU Maxpool Conv2d: (128, 128, 3, 1) ReLU Maxpool Linear: (9216, 128) ReLU Dropout Linear: (128, 10)	Conv2d: (1, 32, 3, 1) ReLU Conv2d: (32, 64, 3, 1) ReLU Conv2d: (64, 128, 3, 1) ReLU Maxpool Conv2d: (128, 128, 3, 1) ReLU Maxpool Linear: (9216, 128) ReLU Dropout Linear: (128, 10)

Table 4. Model architectures used for Fashion-MNIST. “Linear: (i, j) ” is a fully connected layer with input size i and output size j . “Conv2d: (C_i, C_o, k, s) ” is a convolution layer with input channel size C_i , output channel size C_o , kernel size k , and stride s .

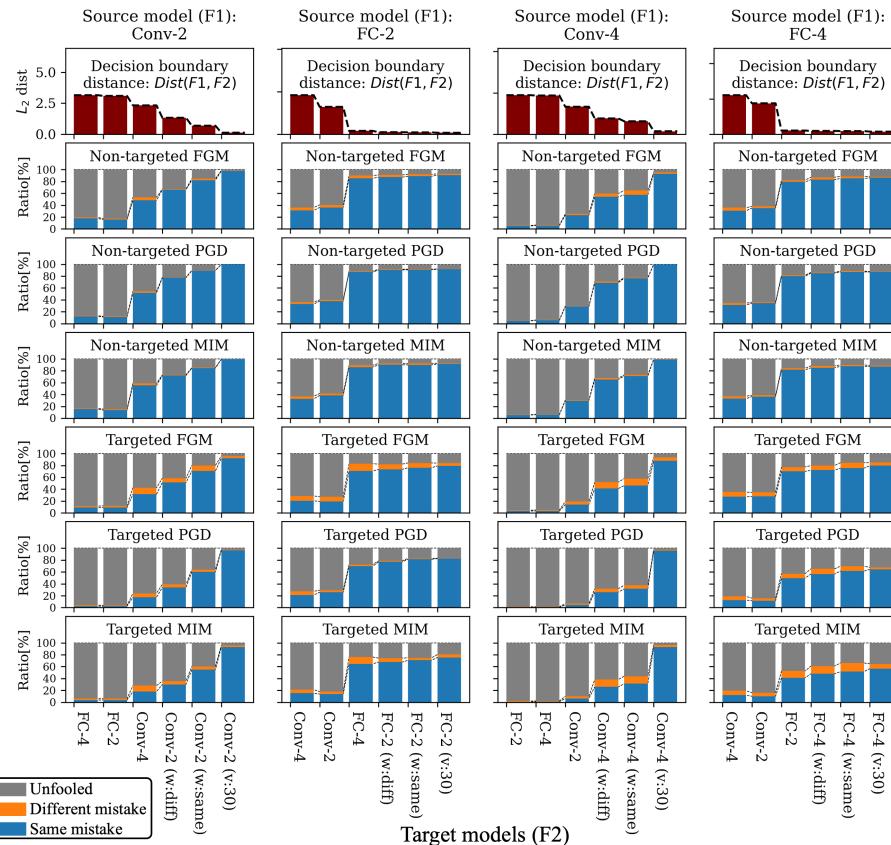


Figure 8. Class-aware transferability of adversarial attacks for Fashion-MNIST. We evaluate FGM [8], PGD [16], and MIM [6] with both non-targeted and targeted objectives. AEs were l1-bounded by $\epsilon=1.0$. Order of F2 is sorted by $Dist(F1, F2)$ (1st row) for each F1 so rightmost F2 was estimated to be more similar to F1.

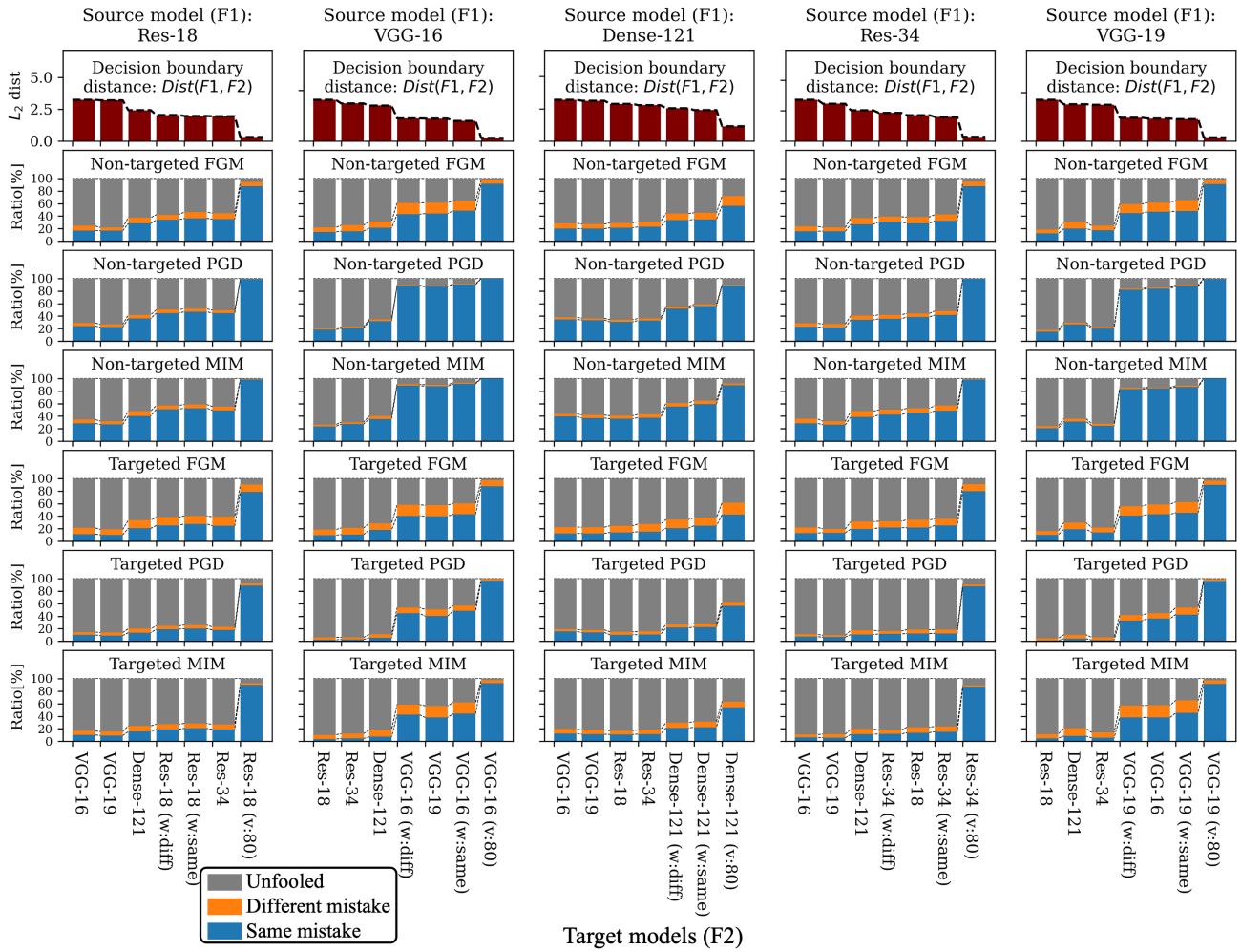


Figure 9. Class-aware transferability of adversarial attacks for CIFAR-10. We evaluate FGM [8], PGD [16], and MIM [6] with both non-targeted and targeted objectives. AEs were ℓ_2 -bounded by $\epsilon=1.0$. Order of F2 is sorted by $Dist(F_1, F_2)$ (1st row) for each F1 so rightmost F2 was estimated to be more similar to F1..

Dataset	Non-robust set constructed for		$F_1(X') = Y_1$	$F_2(X') = Y_2$	$F_1(X') = Y_1 \& F_2(X') = Y_2$
	F1	F2			
Fashion-MNIST	Conv-2	FC-2	92.3	60.0	60.0
	Conv-2	Conv-2 (w:same)	94.6	93.8	93.0
	FC-2	FC-2 (w:same)	58.7	58.5	46.0
CIFAR-10	ResNet-18	VGG-16	95.6	99.0	95.4
	ResNet-18	ResNet-18 (w:same)	94.1	94.1	92.0
	VGG-16	VGG-16 (w:same)	99.5	99.5	99.2
STL-10	ResNet-18	VGG-16	99.2	99.7	99.0
	ResNet-18	ResNet-18 (w:same)	99.2	99.3	99.0
	VGG-16	VGG-16 (w:same)	99.8	99.5	99.2

Table 5. Accuracy of models attacked using AEs X' generated by N-targeted attack, which constructs non-robust sets. Y_1 is target classes for N-targeted attack for model F_1 , and Y_2 is that for model F_2 . These results are particularly interesting: in a white-box setting, it is easy to generate AEs that lead to different sequences of classes Y_1 and Y_2 (success rate > 90% for CIFAR-10 and STL-10).

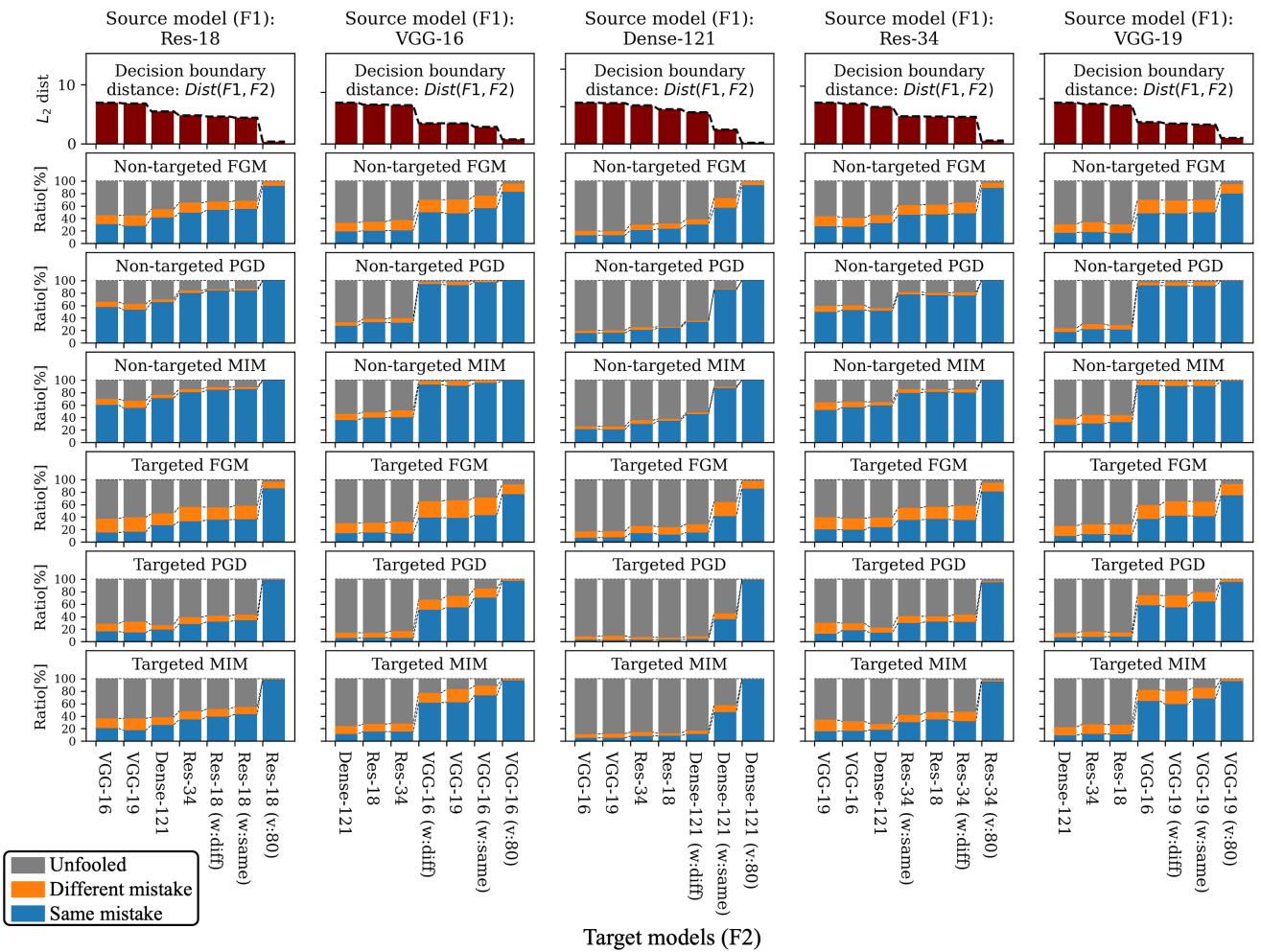


Figure 10. Class-aware transferability of adversarial attacks for STL-10. We evaluate FGM [8], PGD [16], and MIM [6] with both non-targeted and targeted objectives. AEs were l_2 -bounded by $\epsilon=5.0$. Order of F2 is sorted by $Dist(F_1, F_2)$ (1st row) for each F1 so rightmost F2 was estimated to be more similar to F1.

Dataset	Non-robust set constructed for	Train set	Trained model	Test acc (X, Y)	Initial learning rate	Batch size	Data aug.
Fashion-MNIST	F1: Conv-2 F2: FC-2	$D'_1 : (X', Y_1)$	Conv-2 FC-2	82.9 62.1	0.01	256	None
		$D'_2 : (X', Y_2)$	Conv-2 FC-2	80.3 75.4			
	F1: Conv-2 F2: Conv-2 (w:same)	$D'_1 : (X', Y_1)$	Conv-2 FC-2	81.9 66.2			
		$D'_2 : (X', Y_2)$	Conv-2 FC-2	82.4 67.1			
	F1: FC-2 F2: FC-2 (w:same)	$D'_1 : (X', Y_1)$	Conv-2 FC-2	79.0 80.5			
		$D'_2 : (X', Y_2)$	Conv-2 FC-2	77.6 81.4			

Table 6. Non-robust features analysis for Fashion-MNIST. Initial learning rate, batch size, and data augmentations were fixed.

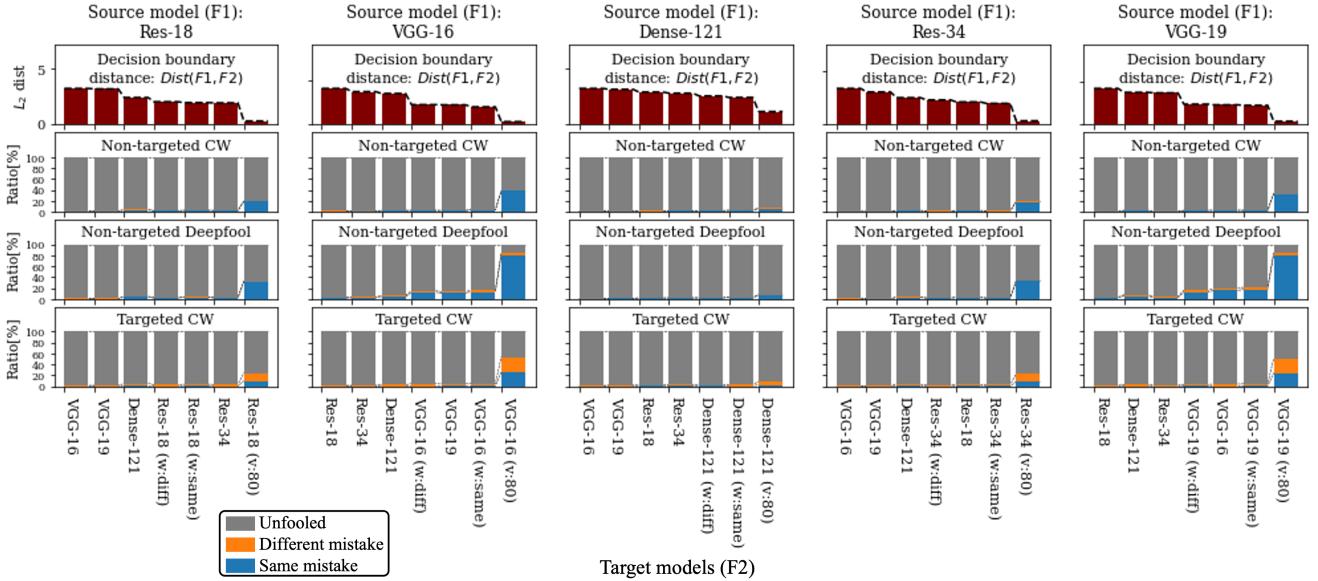
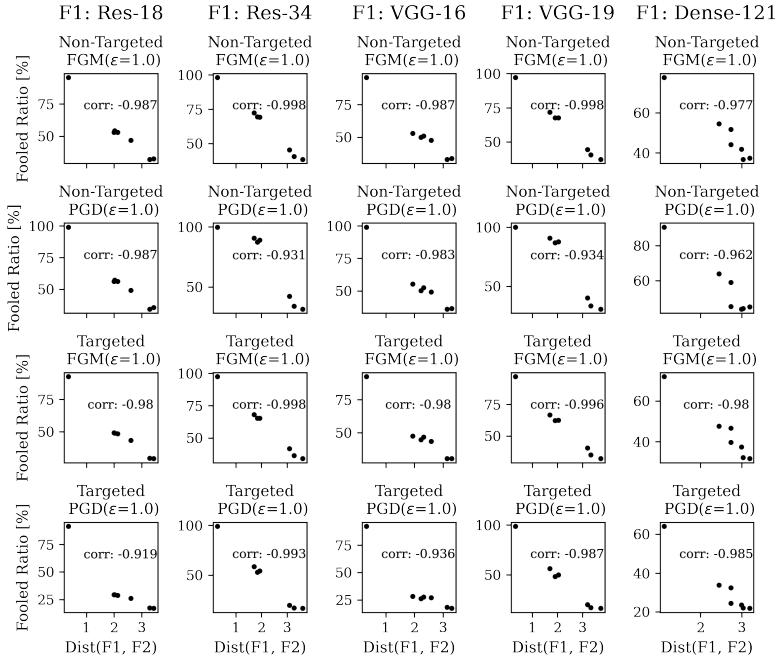


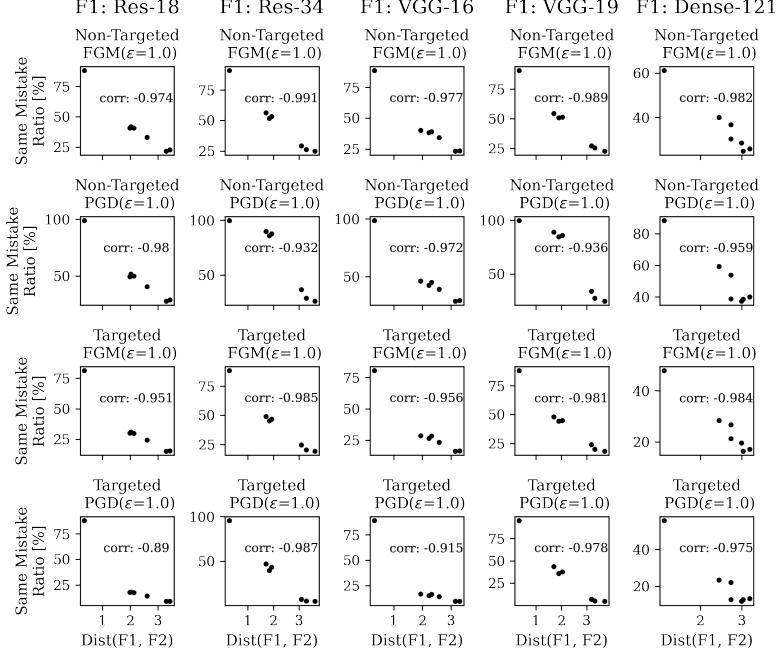
Figure 11. Class-aware transferability of optimization-based adversarial attacks for CIFAR-10. We evaluate CW [1] and Deepfool [18] (Deepfool is defined only for a non-targeted objective). Order of F2 is sorted by $Dist(F_1, F_2)$ (1st row) for each F1 so rightmost F2 was estimated to be more similar to F1. Since these optimization-based attacks try to find minimum perturbations that are enough to fool the source model F1, they hardly transfer between models.

Dataset	Non-robust set constructed for	Train set	Trained model	Test acc (X,Y)	Initial learning rate	Batch size	Data aug.
CIFAR-10	F1: ResNet-18 F2: VGG-16	$D'_1 : (X', Y_1)$	ResNet-18 VGG-16.bn	51.3 53.9	0.005 0.001	128 128	Level 3 Level 2
		$D'_2 : (X', Y_2)$	ResNet-18 VGG-16.bn	10.2 71.0	0.05 0.01	128 128	Level 1 Level 1
	F1: ResNet-18 F2: ResNet-18 (w:same)	$D'_1 : (X', Y_1)$	ResNet-18 VGG-16.bn	50.1 54.1	0.05 0.005	128 256	Level 3 Level 3
		$D'_2 : (X', Y_2)$	ResNet-18 VGG-16.bn	59.2 58.9	0.05 0.005	128 128	Level 1 Level 3
	F1: VGG-16 F2: VGG-16 (w:same)	$D'_1 : (X', Y_1)$	ResNet-18 VGG-16	63.5 68.8	0.05 0.01	128 256	Level 2 Level 3
		$D'_2 : (X', Y_2)$	ResNet-18 VGG-16	11.0 73.1	0.01 0.01	128 128	Level 1 Level 1

Table 7. Non-robust features analysis for CIFAR-10. Optimized hyperparameters are shown besides the test accuracy.

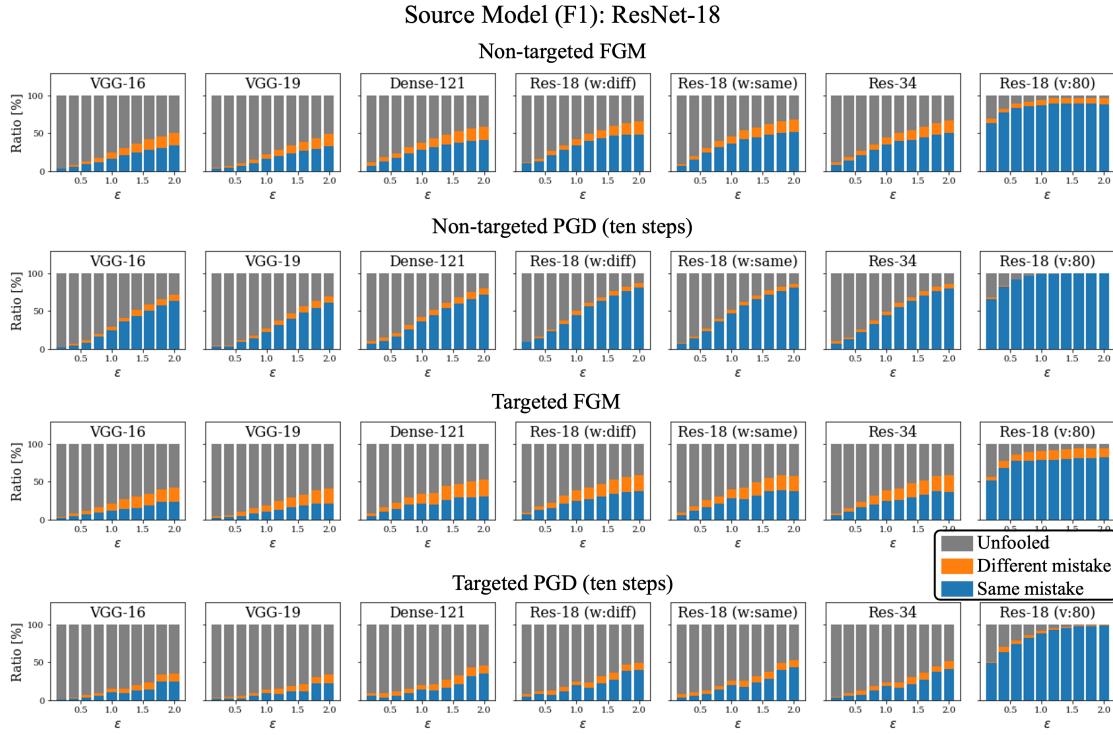


(a) Correlation between $Dist(F1, F2)$ and fooled ratio for CIFAR-10.

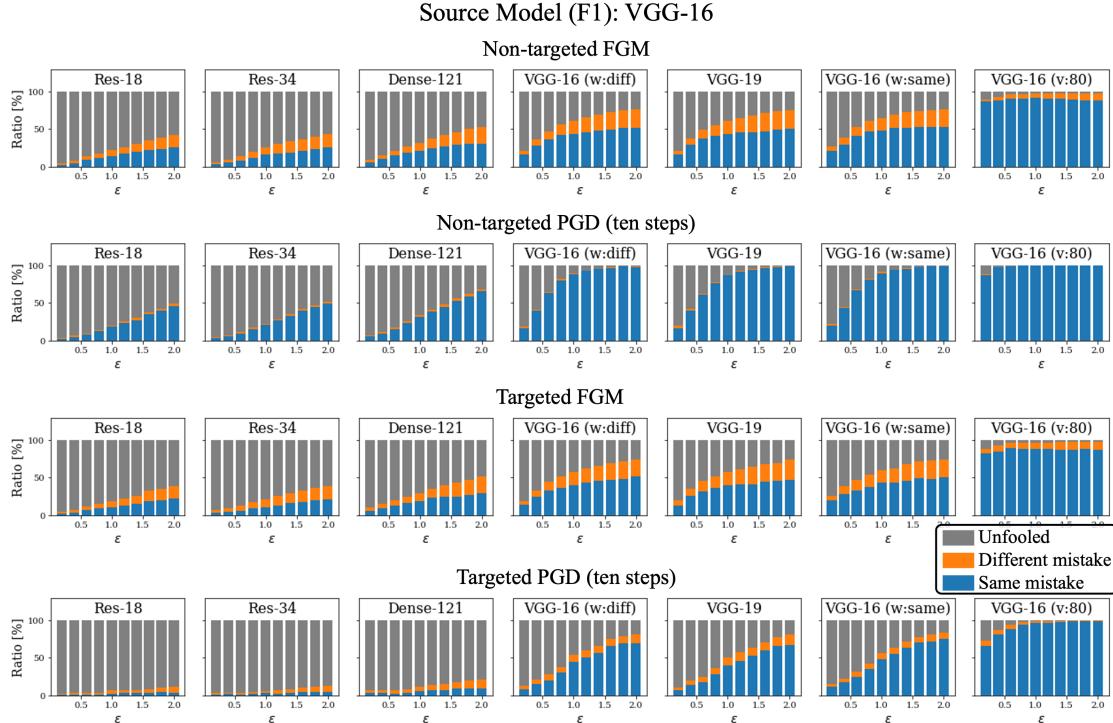


(b) Correlation between $Dist(F1, F2)$ and same mistake ratio for CIFAR-10.

Figure 12. Correlations (1) between $Dist(F1, F2)$ and fooled ratio ratio (Figure 12a), and (2) between $Dist(F1, F2)$ and same mistake (Figure 12b) for CIFAR-10. ResNet-18, ResNet-34, VGG-16, VGG-19, and DenseNet-121 source models (1st to 5th columns, respectively) were attacked by non-targeted and targeted attack (1-2nd and 3-4th rows, respectively) using FGM and PGD (ten-step) (1,3rd and 2,4th rows, respectively) methods. For each source model, the results of the corresponding seven target models (shown in Figure 9) are displayed in a scatter plot.



(a) Class-aware transferability of AEs generated for ResNet-18 source model.



(b) Class-aware transferability of AEs generated for VGG-16 source model.

Figure 13. Class-aware transferability of AEs when the perturbation size ϵ is gradually changed (CIFAR-10). Here we show the results of attacking the source model of ResNet-18 (Figure 13a) and VGG-16 (Figure 13b) with non-targeted attack (1st and 2nd rows) and targeted attack (3rd and 4th rows), using FGM (1st and 3rd rows) and PGD (ten-step) (2nd and 4th rows) methods.

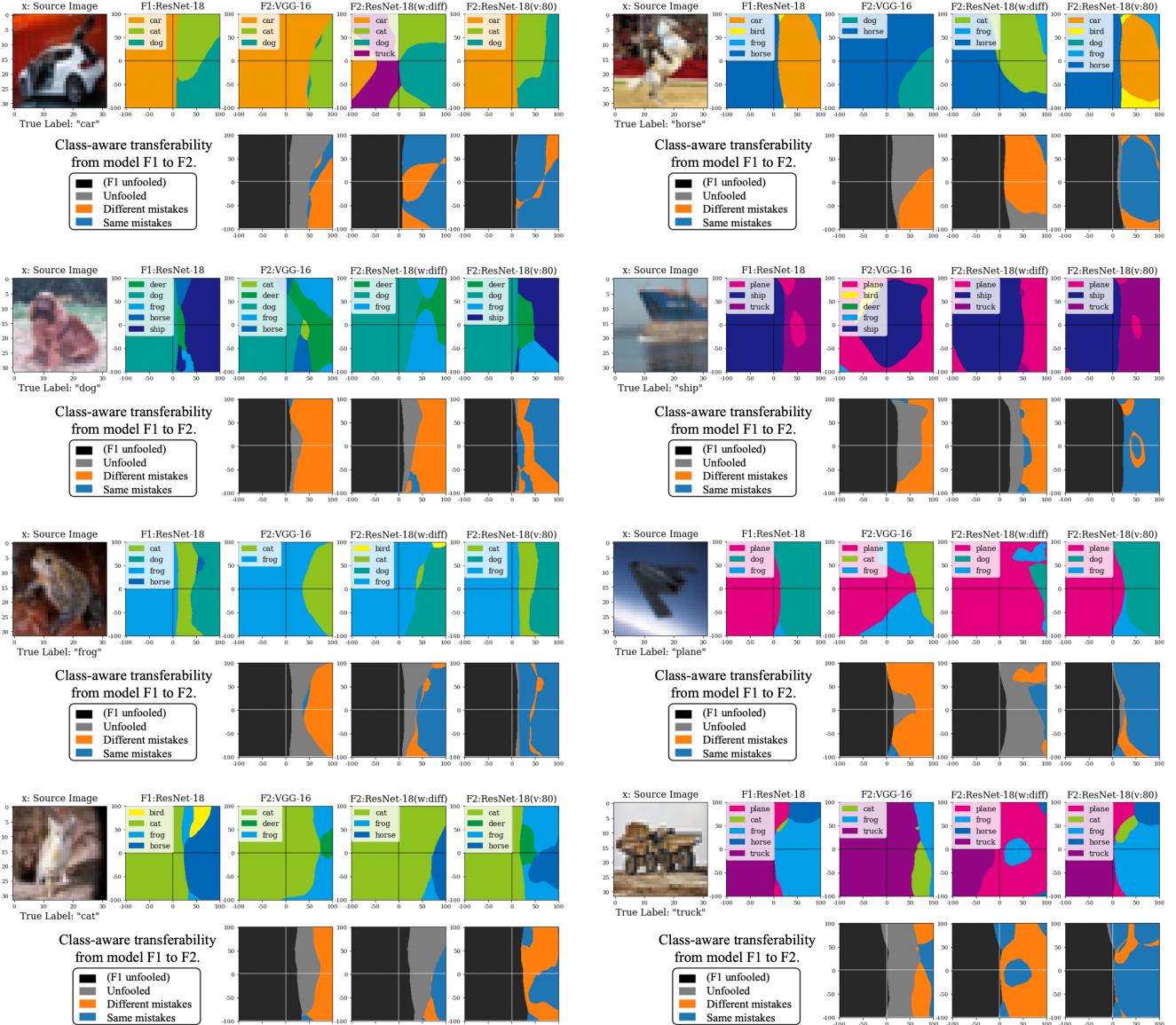


Figure 14. Visualization of decision boundaries when the source model is ResNet-18 (CIFAR-10). For each image, the first row shows the classification results, where each color represents a certain class. The second row shows to which area the three cases of class-aware transferability correspond. The distance from $(0, 0)$ point to the closest decision boundary along the x-axis corresponds to the metric $d(F1, x)$ for each image x . The unit of each axis is 0.02 in L2 distance.

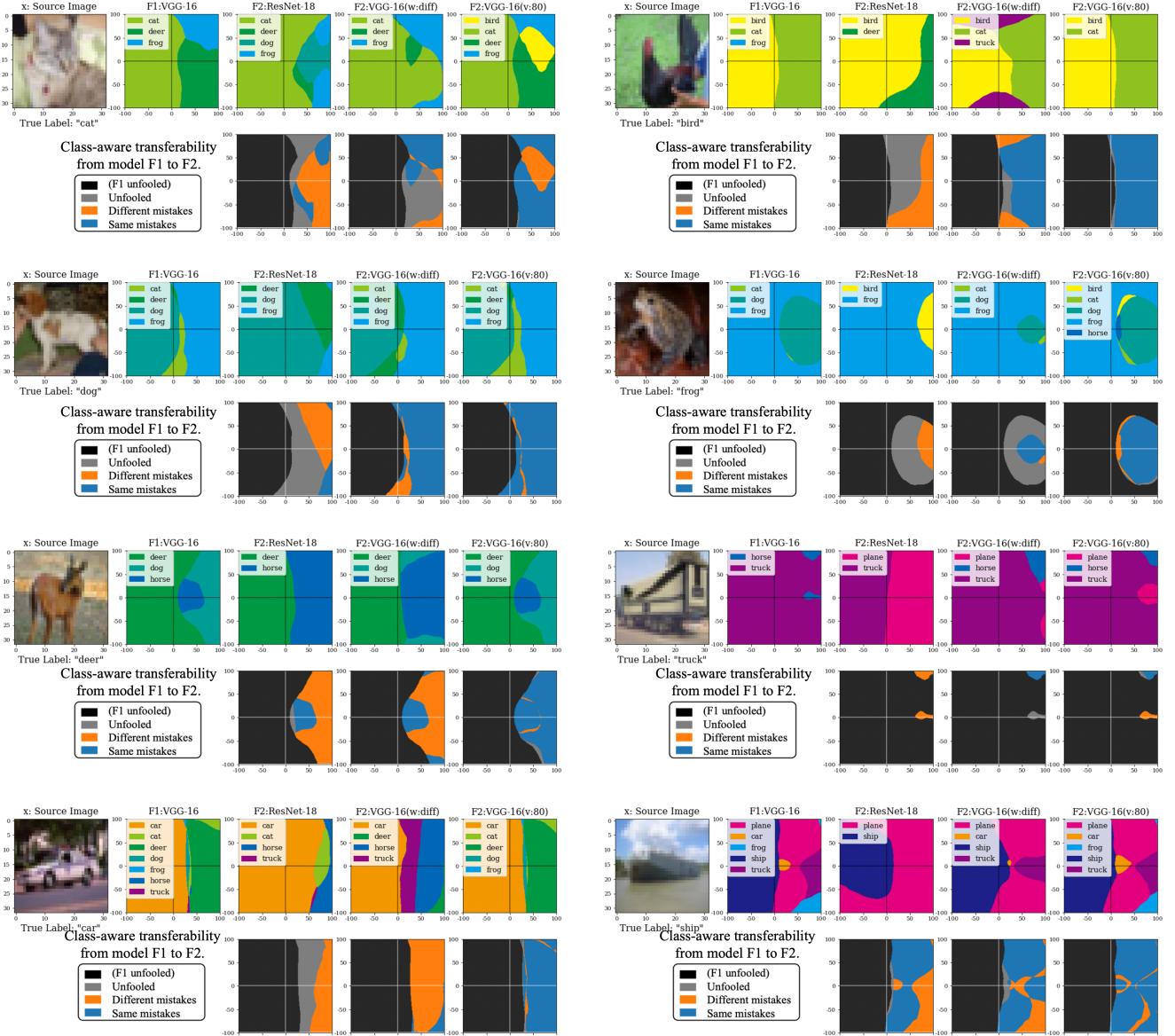


Figure 15. Visualization of decision boundaries when the source model is VGG-16 (CIFAR-10). For each image, the first row shows the classification results, where each color represents a certain class. The second row shows to which area the three cases of class-aware transferability correspond. The distance from $(0, 0)$ point to the closest decision boundary along the x -axis corresponds to the metric $d(F1, x)$ for each image x . The unit of each axis is 0.02 in L_2 distance.

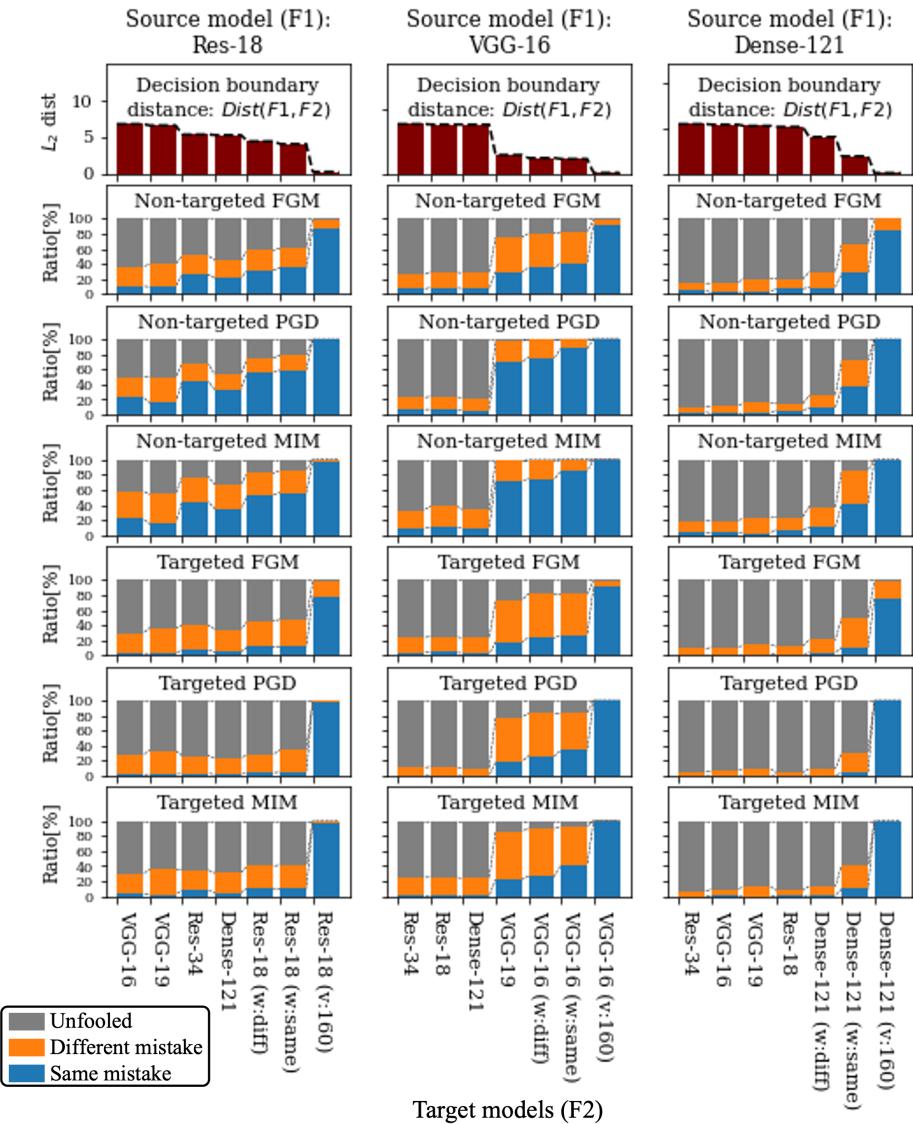
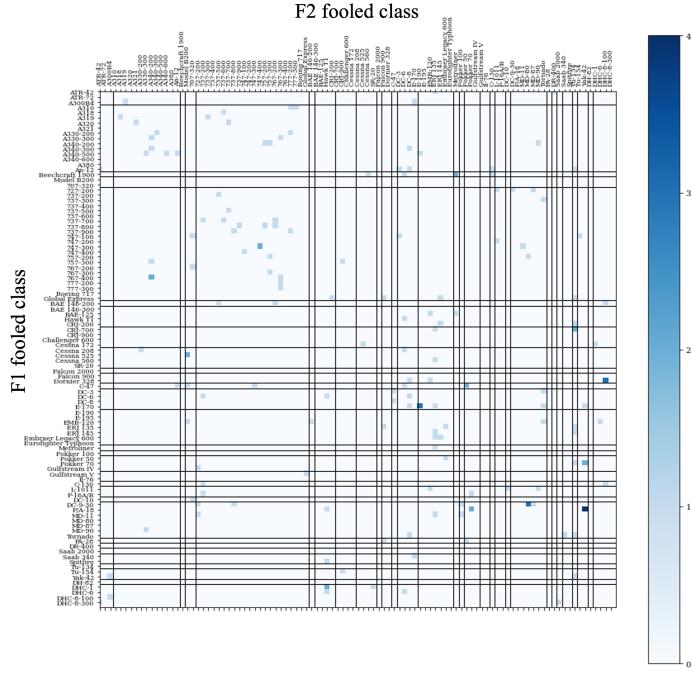
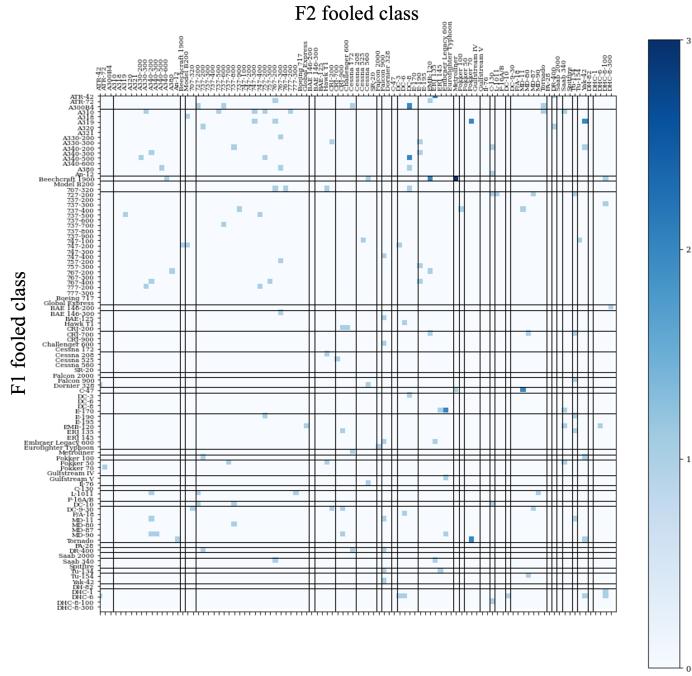


Figure 16. Class-aware transferability of adversarial attacks for FGVC-Aircraft (“variant”). AEs were l_2 -bounded by $\epsilon=5.0$. Order of F2 is sorted by $Dist(F1, F2)$ (1st row) for each F1 so rightmost F2 was estimated to be more similar to F1.



(a) Class-wise analysis of different mistakes caused by AEs generated by non-targeted FGM.



(b) Class-wise analysis of different mistakes caused by AEs generated by targeted FGM.

Figure 17. Class-wise analysis of “different mistakes” for FGVC-Aircraft (“variant”). The y-axis shows the classes to which the source model F1 misclassified the AEs and the x-axis shows the classes to which the target model F2 misclassified the AEs. Each value represents the number of each case. The source model F1 is ResNet-18 and the target model F2 is VGG-16. The classes are sorted by the “manufacturer” labels and the black lines separates “variant” classes for each “manufacturer”. For non-targeted FGM (Figure 17a), it is observed that different mistakes tend to occur within the same “manufacturer”. On the other hand, targeted FGM (Figure 17b) caused different mistakes for other “manufacturers” more than non-targeted FGM.

Initial learning rate	Batch size	Data augmentation
0.05, 0.01, 0.005, 0.001	128, 256	<p>Level 1: Random crop by padding=4</p> <p>Level 2: Random crop by padding=4 + Random horizontal flip</p> <p>Level 3: Random crop by padding=4 + Random horizontal flip + Random affine augmentation</p>

Figure 19. Grid search area to obtain hyperparameters for training models on the constructed non-robust sets (used for CIFAR-10 and STL-10).

Dataset	Non-robust set constructed for	Train set	Trained model	Test acc (X, Y)	Initial learning rate	Batch size	Data aug.
STL-10	F1: ResNet-18 F2: VGG-16	$D'_1 : (X', Y_1)$	ResNet-18 VGG-16_bn	24.0 25.4	0.001 0.001	256 256	Level 3 Level 2
		$D'_2 : (X', Y_2)$	ResNet-18 VGG-16.bn	53.7 57.2	0.005 0.01	128 128	Level 1 Level 2
	F1: ResNet-18 F2: ResNet-18 (w:same)	$D'_1 : (X', Y_1)$	ResNet-18 VGG-16.bn	18.6 20.1	0.001 0.001	256 256	Level 3 Level 3
		$D'_2 : (X', Y_2)$	ResNet-18 VGG-16.bn	32.5 32.4	0.01 0.01	128 256	Level 1 Level 1
	F1: VGG-16 F2: VGG-16 (w:same)	$D'_1 : (X', Y_1)$	ResNet-18 VGG-16.bn	38.5 51.8	0.001 0.01	256 256	Level 3 Level 3
		$D'_2 : (X', Y_2)$	ResNet-18 VGG-16.bn	52.2 52.2	0.01 0.01	128 256	Level 2 Level 2

Table 8. Non-robust features analysis for STL-10. Optimized hyperparameters are shown besides the test accuracy.

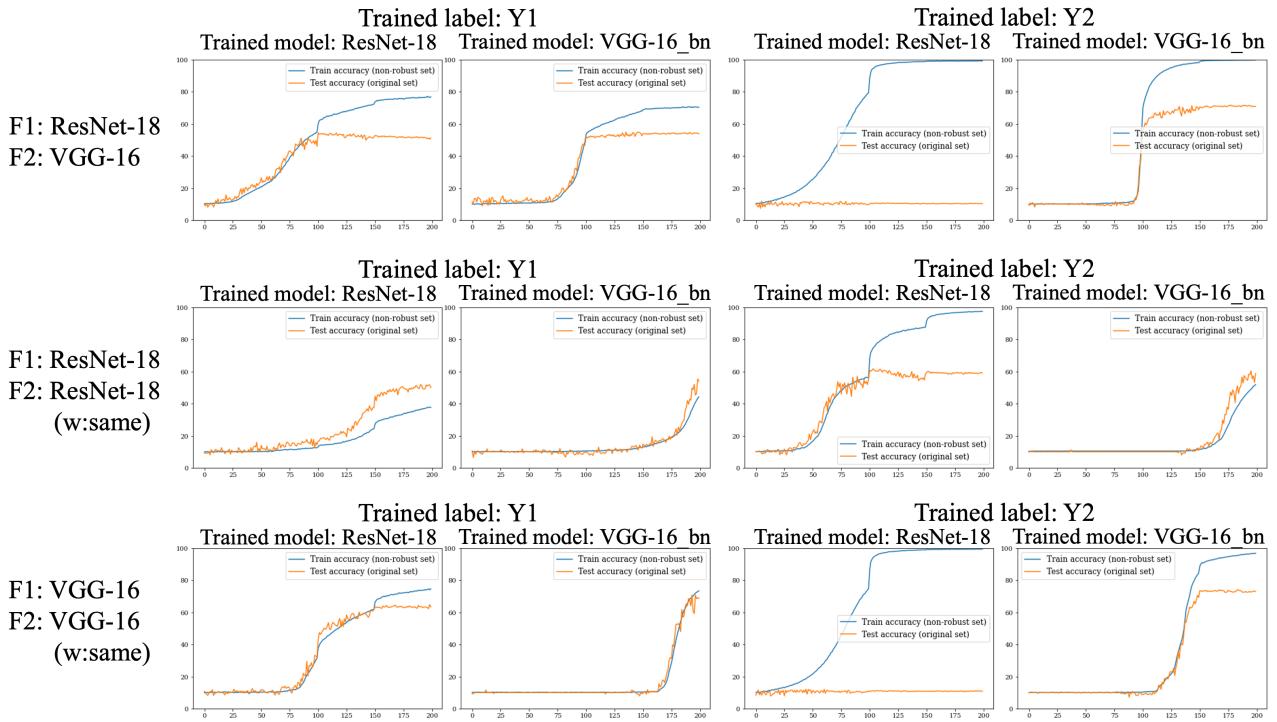


Figure 20. Accuracy curves when models were trained on the constructed non-robust sets (CIFAR-10). Each figure plots training accuracy on the constructed non-robust set (blue line) and test accuracy on the original test set (orange line).