A. Implementation Details

Model Architectures Each model consists of 3-4 residual layers with pre-activation residual blocks, which use an identity mapping as the shortcut connection. ResNet-34 consists of 4 residual layers with 2 convolution operations per residual block. ResNet-50 and ResNeXt-29 consist of 4 and 3 residual layers respectively, and both have 3 convolution operations per residual block. Additionally, ResNeXt-29 has a bottleneck width of 4 and cardinality of 32. In this case, `ReScaler` is adapted to instead work with the grouped convolution operations of ResNeXt, by instead applying a scalar weight to each group. As a result, the number of `ReScaler` parameters for ResNeXt scales with cardinality.

Attack Setup For data poisoning attacks, we assume that the attacker has a poisoning budget of 1% of the size of the whole training set and a the perturbation budget $\epsilon = 16$ as measured by l^∞-norm. For clean-label backdoor attacks, we allow a poisoning budget of 1% with trigger patches of size 6×6 with a perturbation budget of $\epsilon = 16$ for ResNet-18, ResNet-34 and RexNeXt-29, and $\epsilon = 32$ for ResNet-50.

Moreover, we also specify the attack settings here in order to better clarify the attacker's capacity. That is, for CP and BP attacks, the attacker is allowed to extract features from several layers of the network when crafting poisons, as we find that it benefits the most number of successful attacks. When poisoning these models, all batch-normalization layers are frozen and only the last layer is finetuned on the poisoning data set. We run these crafting algorithms for the default number of iterations as specified by the authors. Additionally, we exclude the poison images from data augmentations, as we find this reduces the overall attack efficacy. When finetuning the last layer, we use SGD with a step size of 0.01 for 35 epochs.

For clean-label attacks in our evaluation, we focus on the transfer learning setting in which existing attacks are most effective. Specifically, we assume that the attacker has the full knowledge of a model's pre-trained weights when crafting poisons. Afterward, when the defender finetunes the model on the poisoned data set, only the last fully connected layer is trained, and all other model weights are frozen.

We apply `ReScaler` onto each successfully poisoned model, and perform the 1-gradient step update described in Section 3. We set the step size $\alpha = 1.0$, and set ϵ as 0.15 and 0.03 for ResNet-34/ResNet-50 and ResNeXt-29 respectively unless otherwise specified.

Figure 2 setup For all models, the ϵ values increase linearly from left to right. For ResNeXt-29, ϵ increases between the values of 0.01 and 0.05, with a step size of 0.01. For ResNet-34 and ResNet-50, ϵ increases between the values of 0.05 and 0.25, with a step size of 0.05. For each value of ϵ, the ASR and validation error are calculated on 3 randomly sampled models from each attack setting applying with `ReScaler`. Points are removed if there is no change in ASR or validation error.

B. Related Work

Poisoning and Backdoor Attacks. Existing poisoning and backdoor attacks alter models by injecting poisoning samples into the training data. Poisoning attacks optimize perturbations on a set of poisoning samples in order to mislead the model on some specific target images during inference. Besides Feature Collision (FC), Convex Polytope (CP) and Bullseye Polytope (BP) introduced in the former section, there are some other poisoning attack strategies formulating it into a bilevel optimization problem [9, 14]. Backdoor attacks alter the model on some specific trigger patterns at test time. They can be roughly divided into clean-label [20, 24] and dirty-label attack [11] ones, according to whether or not the maliciously poisoning samples keep label-consistency with their original clean ones. Besides CLBD and HTBD mentioned before, there exist more dirty-label attacks. Typical trigger patterns can be either a fixed tiny patch or other dynamic patterns. BadNet [11] and Blend [5] propose to apply a typical tiny patch. SIG [3] adopts a more complex form, e.g., sinusoidal strips. Recent study shows that natural reflection [19] can also be conducted as a backdoor attack.

Existing defense methods work either post-poisoning or during training. The post-poisoning defenses usually detect and remove the poisoned data from training set [4, 23]. The repair-based methods adopt fine-tuning, fine-pruning [17] or other training strategies [18, 26] to directly erase the malicious impacts caused by the backdoor triggers, while simultaneously maintain the model's overall performance on clean data. Mode connectivity repair is also proposed to remove backdoor related neural paths [32]. However, these defense methods can hardly defend against clean-label attacks. Adversarial poisoning [8] proposes to extend the adversarial training framework to defend against (during training) poisoning and backdoor attacks, yet results in a substantial computational cost.

Residual Connections. Residual modules are widely applied along with the increasing depth of DNNs, e.g., ResNet, WideResNet [30], DenseNet and ResNeXt. It often uses an identity shortcut to prevent gradient vanishing. However, recent studies find that the information flow between residual blocks and their identity shortcuts (during forward and backward propagation) hints some insightful phenomenon. Bachlechner et al. propose the ReZero [2] operation to re-assign the weight of residual block as zero, which could further improve the model convergence. Wu et

al. [27] show that downweighting residual block gradients during backward propagation could improve the transferability of generated adversarial examples. The above studies motivate us to explore the residual modules in poisoning and backdoor attacks.

C. Additional Analysis

C.1. Transferability Discussion

We explore the transferability of learned `ReScaler` weights between different poisoned models as well as between different attacks. Specifically, we first learn `ReScaler` weights for a particular target image and attack, then we apply these weights onto another poisoned model which corresponds to a different target image or attack. If these weights are transferable, we would expect the transfer attack success rate, or the attack success rate of a poisoned model after these learned weights have been applied, to be low. In order to effectively transfer these weights across models, we modify the 1-gradient step method discussed in Section 3.2. Specifically, we change the objective to instead minimize the cross-entropy loss between the target image and its ground-truth class, thus the learned `ReScaler` weights will have the effect of sending the target image to its ground-truth class. Additionally, we also increase ϵ to 0.3 and increase the number of optimization steps to 100, which leads to consistently better results.

Transferability among different poisoning sets. In Figure 3, we first train the `ReScaler` weights for a subset of ResNet-34 FC poisoned models, then transfer them to other FC poisoned models (denoted as FC $\rightarrow$ FC), as well as for BP (denoted as BP $\rightarrow$ BP). We see that a majority of these weights do transfer in both settings, resulting in a significant decrease in the transfer attack success rate. Moreover, we observe that the transferability among BP poisoned models is generally more effective and the transfer attack success rate is lower. We also notice a correlation between the transferability of the weights learned on a poisoned model and the poison-class confidence of that poisoned model. In Figure 4, we see that models with target images that have the highest initial poison-class confidences tend to yield the lowest transfer attack success rates in both the FC and BP settings. This shows that training on models which have target images more strongly embedded in the poison class generally yields `ReScaler` weights with the more effective transferability to other poisoned models.

Transferability among different attack strategies. In Figure 5, we train `ReScaler` weights on FC poisoned models, then measure the transfer attack success rate on BP poisoned models (denoted FC $\rightarrow$ BP), and also from BP to FC (denoted BP $\rightarrow$ FC). We see that in both cases, the transfer attack success rates are comparable to the setting in Figure 3, suggesting that `ReScaler` weights are effective at providing a decent defense even when applied onto models poisoned with a different attack strategy.

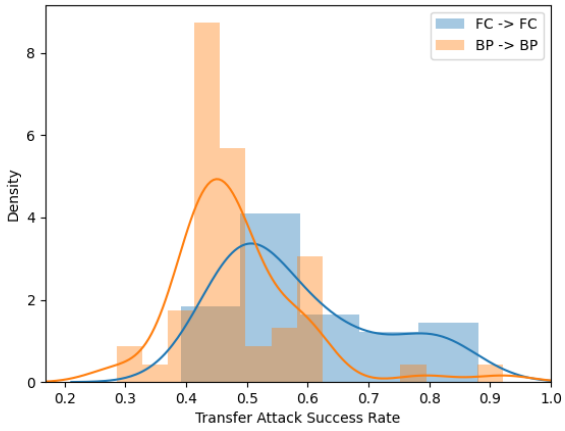

Figure 3. Transferability of `ReScaler` among the FC and BP poisoning sets. The transfer attack success rates are evaluated on the larger set of FC/BP poisoned models after applying with `ReScaler`, which are trained on their randomly sampled 50 FC/BP attacked models respectively. `ReScaler` transfers well across different poisoning sets.

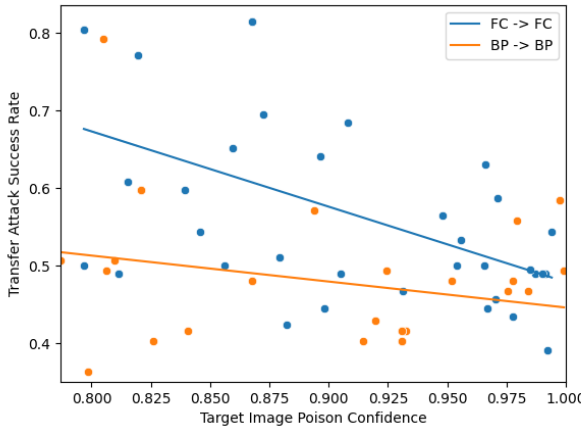

Figure 4. Correlation between transfer attack success rate and target image poison-class confidence for FC and BP poisoned ResNet-34 models. The higher the poison-class confidence, the lower the transfer attack success rate. The correlation indicates that training `ReScaler` on more strongly poisoned models enhances the transferability.

Overlap similarity as an indicator for transferability. We show that the degree of transferability from a poisoned model to some other poisoned model can be reasonably attributed to the similarity of their individual `ReScaler` weights. Given some poisoned model m_i and target image

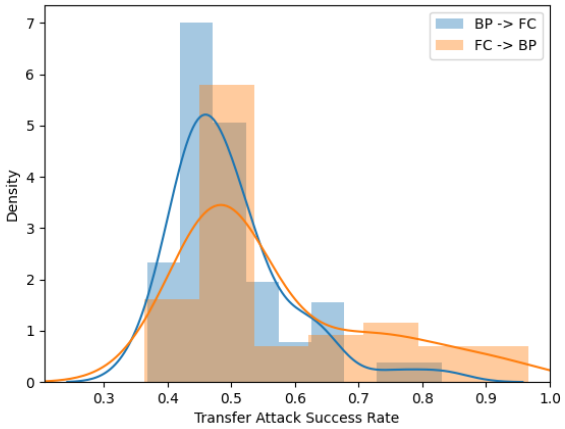

Figure 5. Transferability of ReScaler between FC and BP attacks, which is evaluated on a subset of 50 ResNet-34 poisoned models for each setting. ReScaler transfers well across different attack strategies.

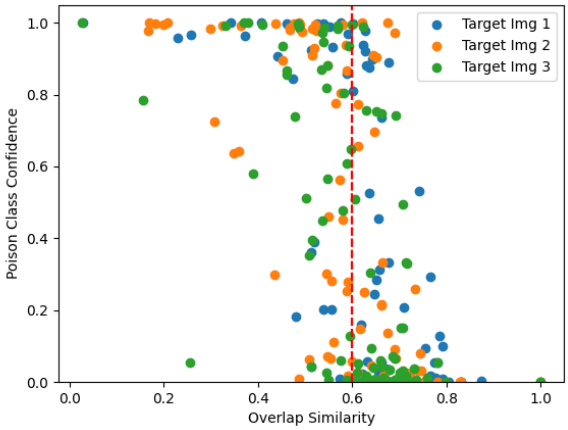

Figure 6. Overlap similarity VS poison confidence for a subset of target images, where the models apply the ResNet-34 architecture and are poisoned using the FC attack. We observe that the overlap similarity is correlated with the poison-class confidence.

$\boldsymbol{t}_i$, ReScaler yields some set of scalar weights $\boldsymbol{w}_i$. We measure the overlap similarity between $\boldsymbol{w}_i$ and some other $\boldsymbol{w}_j$ in the following manner:

$$\text{sim}(\boldsymbol{w}_i, \boldsymbol{w}_j) = \frac{\|\min(\boldsymbol{w}_i, \boldsymbol{w}_j)\|}{\|\max(\boldsymbol{w}_i, \boldsymbol{w}_j)\|}, \text{for } \boldsymbol{w}_i, \boldsymbol{w}_j \neq \boldsymbol{0}, \quad (4)$$

where min and max are element-wise minimum and maximum operators respectively. This similarity achieves a maximum value of 1 when $\boldsymbol{w}_i = \boldsymbol{w}_j$, and a minimum value of 0 if $\min(\boldsymbol{w}_i, \boldsymbol{w}_j) = 0$, i.e., either $\boldsymbol{w}_{i,k}$ or $\boldsymbol{w}_{j,k}$ equals 0 for all element-wise values $\boldsymbol{w}_{*,k}$. We first compute ReScaler weights $\boldsymbol{w}_j$ for each ResNet-34 FC poisoned model m_j. For plotting purposes, we choose ReScaler weights $\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3$ for 3 random poisoned FC models such that their corresponding target images $\boldsymbol{t}_1, \boldsymbol{t}_2, \boldsymbol{t}_3$ have a high poison class confidence. In Figure 6, we plot the overlap similarity for each of $\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3$ and all other models with weights $\boldsymbol{w}_j$ on the x-axis. On the y-axis, we plot the poison class confidence for target images $\boldsymbol{t}_j$ after each of $\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3$ has been applied to m_j. We see that ReScaler weights $\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3$ are much more likely to reduce the poison-class confidence of a poisoned model m_j with weights $\boldsymbol{w_j}$ if $\text{sim}(\boldsymbol{w_i}, \boldsymbol{w_j})$ is sufficiently high. In particular, we see that overlap similarities below 0.6 are generally correlated with higher poison class confidences, while those above 0.6 are correlated with lower poison class confidences. This provides an explanation for the transferability of learned ReScaler weights among poisoned models, and also suggests a stronger point – that our defense is capable of detecting and dampening a common set of features shared among many poisoned models.

C.2. Analysis

In this section, we analyze properties of ReScaler. We first show that learned ReScaler weights are similar even when learning to send a particular target image into arbitrary classes. We then discuss how learned ReScaler weights for the target image of some poisoned model can effectively transfer to its set of training poisoning images.

ReScaler weights are independent of class. In Figure 7, we use overlap similarity to compare the ReScaler weights for sending a particular target image into multiple different classes (denoted as one target image $\rightarrow$ different classes). In particular, we see that the weights for sending a target image into arbitrary classes are relatively similar, suggesting that ReScaler learns a common set of weights needed to 'un-poison' a target image and send it into any class. We also show that ReScaler weights do not depend on the poison class of the target images they are applied onto. In Figure 7, we use overlap similarity to compare the ReScaler weights for sending different target images from the same poisoning class back to their own ground truth class (denoted as different target images $\rightarrow$ same class). We see that there is generally less overlap among these ReScaler weights, further suggesting that ReScaler is independent of the poison class of target images and learns critical weights that are needed to 'un-poison' these models.

ReScaler transfers effectively to training poisoning images. In cases where ReScaler is able to successfully defend against against a poisoning attack, we find that these weights are also successful at reducing the poisoned

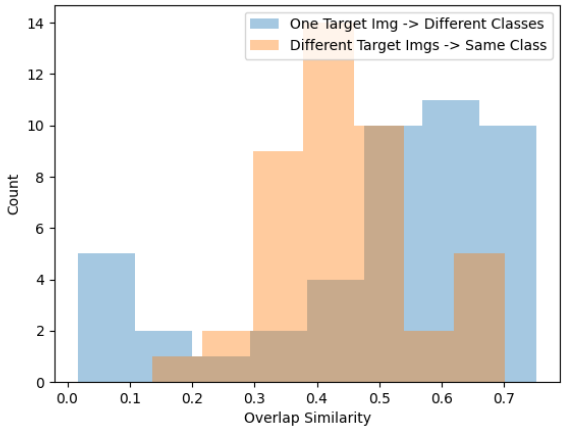

Figure 7. Overlap similarity among ReScaler for sending the same target image into different classes and sending different target images into the same class. For the first setting, we choose a subset of 5 FC poisoned models and send each model's target image into the remaining 9 classes. In the second setting, we sample 9 FC poisoned models with target images from each of 5 random poison classes, and learn weights to send those target images back to their ground truth class.

model's accuracy on the clean-label set of poison training images to below 10%, while the accuracy on the clean subset of the training set remains largely unaffected. This provides an additional explanation for the efficacy of our approach in reversing the effects of poisoning attacks across numerous settings.

C.3. Saliency Maps of Target and Poisoning Images

We use saliency maps to investigate the effect of ReScaler on the prediction of both a target image and its corresponding poisoning training images. In particular, we visualize the salient features for the prediction of the target image using the base model (i.e., a standard unpoisoned model), the poisoned model, and the defended model (i.e., the poisoned model with ReScaler applied onto it). We present these results in the first rows of Figures 8, 9 and 10, respectively. We observe that the salient features for the prediction of the target image using the poisoned model tend to differ significantly from those of the base model. Specifically, we notice that after poisoning, the target image's salient features tend to shift towards a small number of highly influential points. We also see that after ReScaler has been applied, the target image's salient features more closely resemble that of the base model. This provides an explanation for how our defense is able to successfully counteract the effect of poisoning and allow the model to correctly classify the target image.

We also expand upon the finding in Section C.2, which discusses how ReScaler weights effectively transfer to the poisoning training images that correspond to a particular target image. In particular, we randomly sample some poisoning training images that correspond to each target image, and investigate the salient features for the prediction of these images using both the poisoned and defended model. We present these results for each target image in the last three rows of Figures 8, 9, and 10. We see that the salient features for the prediction of the poisoning images using the defended model are much different from those of the poisoned model. Additionally, we observe that the salient features tend to shift towards a larger number of highly influential points after the defense has been applied. We also note that these poisoning images are incorrectly classified by the defended model, and in fact are returned to the ground truth class of the target image. These findings demonstrate how our defense can transfer to identifying poisoning training images, and provide further explanation to support the ability of ReScaler to effectively "un-poison" these models.

C.4. Per-block ReScaler Analysis

In Figure 11, we provide a visualization of learned ReScaler weights applied to each residual block for various poisoned ResNet models, and we also investigate the effect of these weights on the cosine similarity between the output of each block and the output of the shortcut connection for each residual block in the poisoned and defended models. We notice that the magnitude and placement of the learned ReScaler weights, applied onto both the first and second convolutions of each residual block, vary dramatically across each poisoned model. In fact, we expand upon the findings in Section C.2, and observe that the distribution of these weights tend to depend heavily on the particular target image of each poisoned model, rather than the target or poison class. We also observe that for some residual blocks, the ReScaler weights for either the first or second convolutional layer are learned, rather than both.

After the defense has been applied onto the poisoned model, we notice that for blocks where ReScaler weights for both the first and second convolutions are applied, the similarity between the block output and the residual output tends to increase. This suggests that ReScaler weights help downweight these convolutional transformations in favor of those propagated through the residual skip connections. We also observe that the cosine similarity between these representations is generally higher in deeper residual blocks and much lower for the first blocks of each residual layer, suggesting that deeper blocks tend to propagate representations much more through skip connections. Additionally, as discussed in Section 3.1, we note that strictly applying ReScaler weights onto residual blocks with higher cosine similarities tends to yield a much higher benign validation accuracy for the defended model.

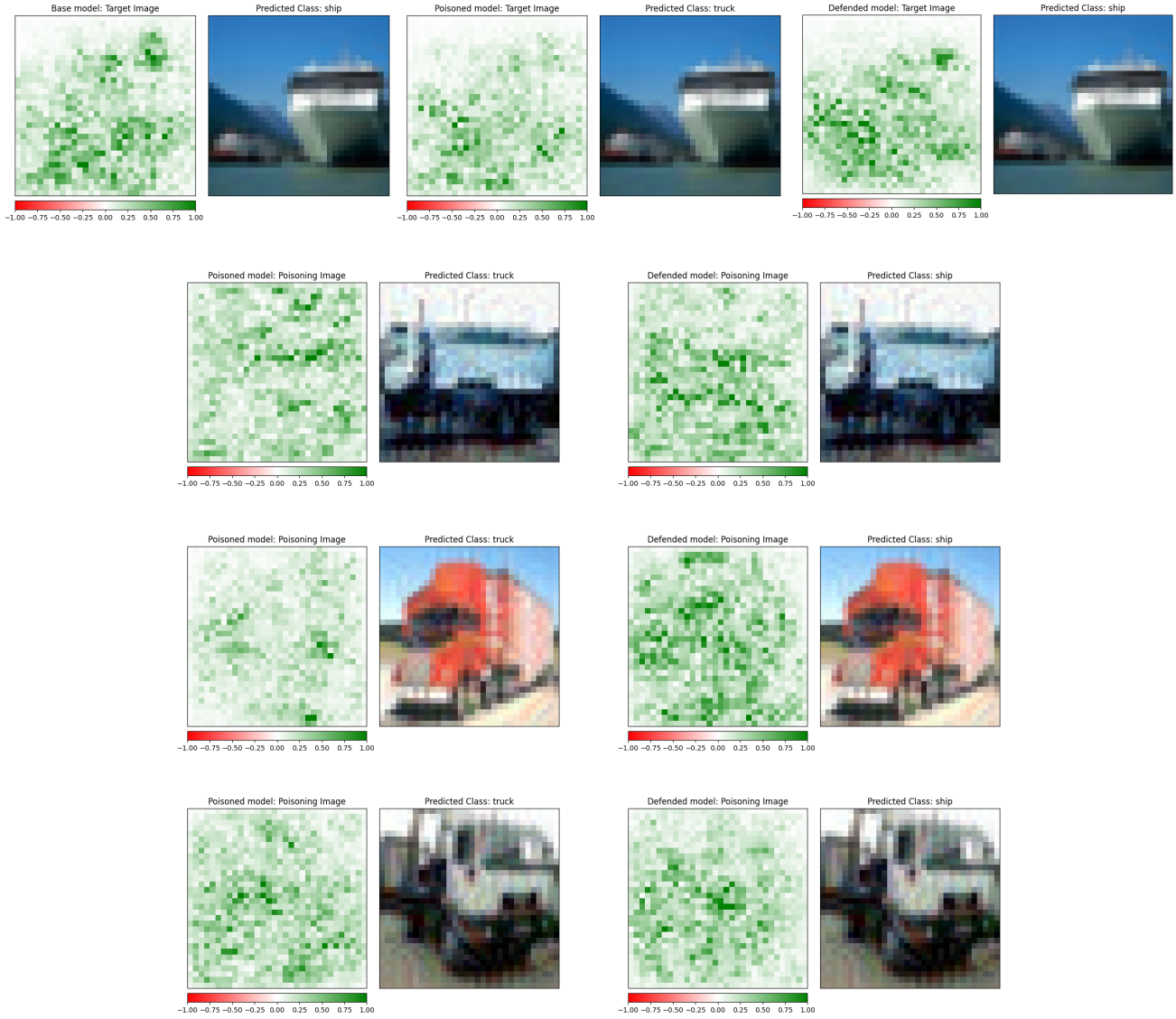

Figure 8. Saliency maps of target and poisoning images for a ResNet-34 model in the FC attack setting with $\epsilon = 0.3$. We consider a target image with ground truth class "ship" and poison class of "truck". More influential features for a particular prediction are colored dark green, and similarly less influential features are lighter shades of green. In the first row, we see that the poisoned model considers a small number highly influential features when making the prediction into the "truck" class. We also see that after applying `ReScaler`, the prediction for the target image is sent back into the "ship" class, and that the influential features for this prediction more closely resemble that of the base model. In the bottom 3 rows, we randomly sample poisoning images from the "truck" class, and see that the the number of salient features tends to increase after the defense has been applied. We also see that the defense sends these poisoning images, with ground truth "truck", into to the "ship" class.

D. Limitations and Social Impact.

Efficient defense strategy against poisoning and backdoor attacks is urgently needed especially for larger DNNs, which are more likely to scrape enormous data for training. As for the limitations of `ReScaler`, we will consider combining attack detection and test-time adaptation to reduce the required gradient step updates in the future. In addition, if more information of the training and test distributions are available, such test-time adaptations can be designed more effectively as well.

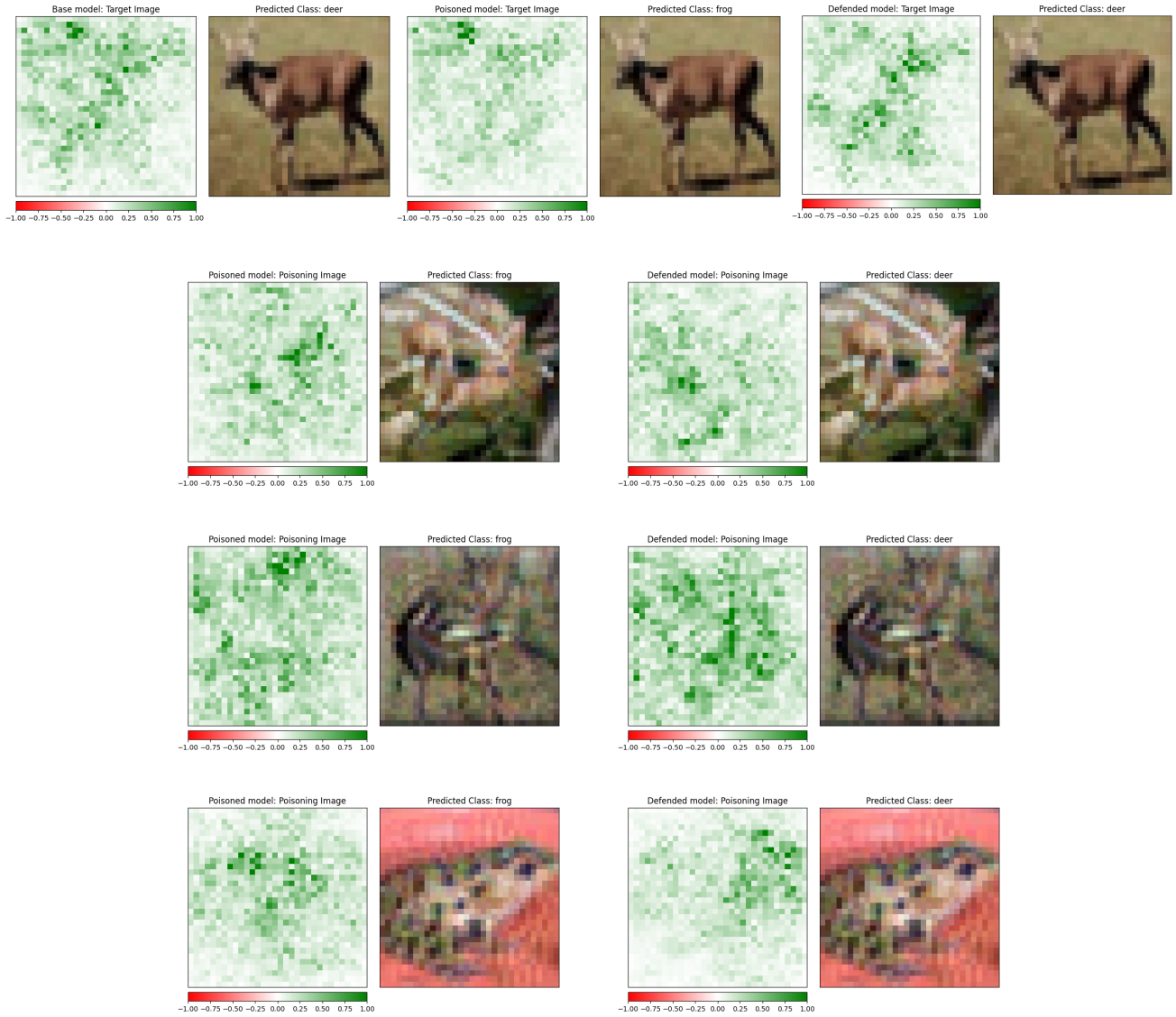

Figure 9. Saliency maps of target and poisoning images for a ResNet-34 model in the FC attack setting with $\epsilon = 0.3$. We consider a target image with ground truth class "deer" and poison class of "frog". More influential features for a particular prediction are colored dark green, and similarly less influential features are lighter shades of green. In the first row, we see that the poisoned model considers a small number highly influential features when making the prediction into the "frog" class. We also see that after applying `ReScaler`, the prediction for the target image is sent back into the "deer" class, and that the influential features for this prediction more closely resemble that of the base model. In the bottom 3 rows, we randomly sample poisoning images from the "frog" class, and see that the the number of salient features tends to increase after the defense has been applied. We also see that the defense sends these poisoning images, with ground truth "frog", into to the "deer" class.

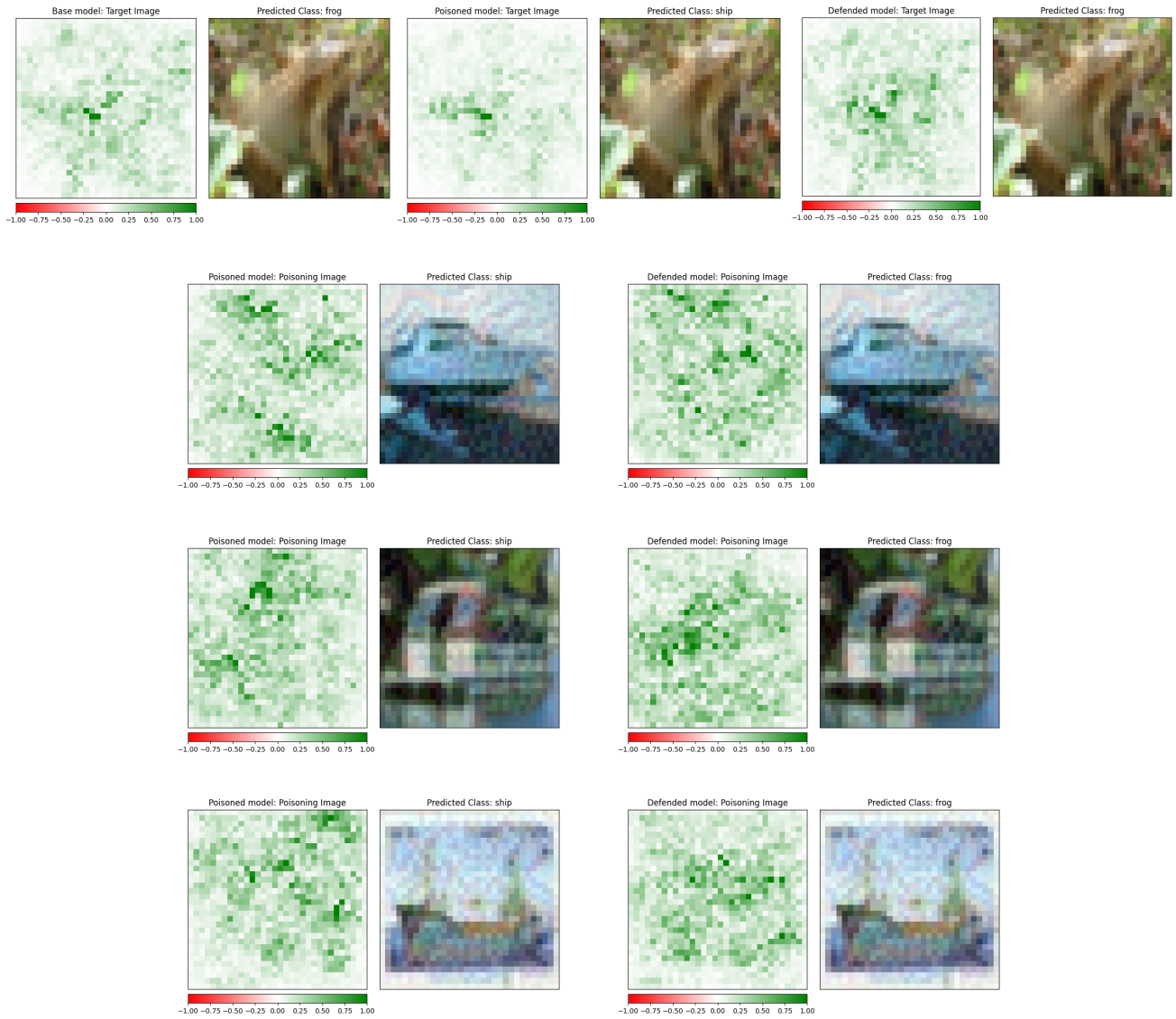

Figure 10. Saliency maps of target and poisoning images for a ResNet-34 model in the FC attack setting with $\epsilon = 0.3$. We consider a target image with ground truth class "frog" and poison class of "ship". More influential features for a particular prediction are colored dark green, and similarly less influential features are lighter shades of green. In the first row, we see that the poisoned model considers a small number highly influential features when making the prediction into the "ship" class. We also see that after applying `ReScaler`, the prediction for the target image is sent back into the "frog" class, and that the influential features for this prediction more closely resemble that of the base model. In the bottom 3 rows, we randomly sample poisoning images from the "ship" class, and see that the the number of salient features tends to increase after the defense has been applied. We also see that the defense sends these poisoning images, with ground truth "ship", into to the "frog" class.

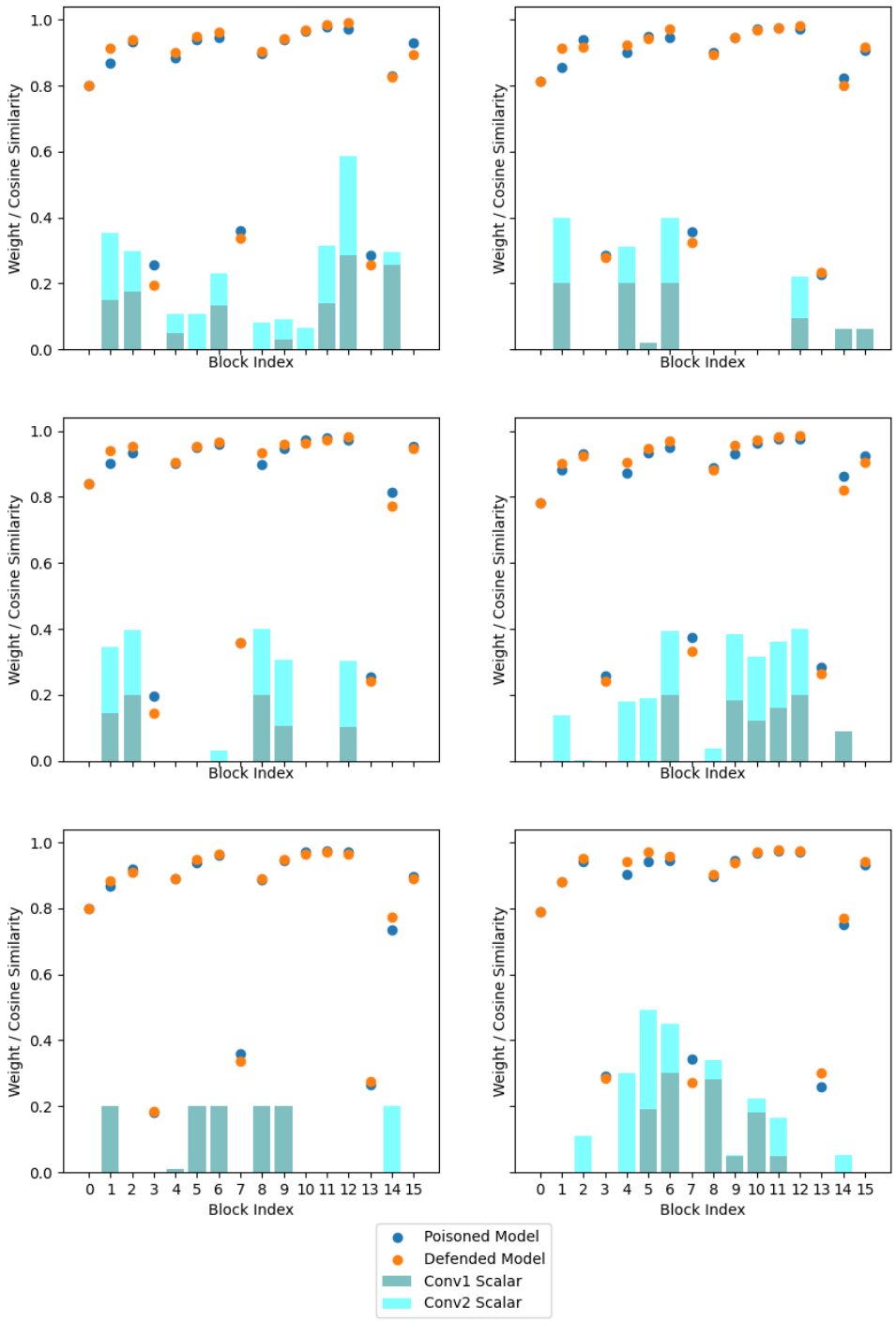

Figure 11. Effect of `ReScaler` on the cosine similarity between block and shortcut output. We randomly sample 6 `ReScaler` weights and successfully defended ResNet-34 models in the FC attack setting, and plot each one in the figures above. Each `ReScaler` is learned with an ϵ bound of 0.3, and the learned weights are displayed using a stacked bar plot to represent the scalar multiples for both the first and second convolutional layers (ie. "Conv1 Scalar" and "Conv2 Scalar" respectively). We also plot the cosine similarity between the output of each residual block $f(z)$ and the output of its skip connection z. Higher cosine similarities imply that the residual block propagates the input representation largely through its skip connection. We see that the defended model generally has higher cosine similarities than the poisoned model. We also observe that the first blocks of each residual layer (i.e., indices 0, 3, 7, and 13) tend to have much lower similarities than deeper blocks in the layer.