A Appendix

A.1 Training Hyperparameters

DNNs Can Learn Useful Features From Unlearnable Datasets In Section 4.2, we train a number of ResNet-18 (RN-18) [8] models on different unlearnable datasets with cross-entropy loss for 60 epochs using a batch size of 128. We save checkpoints at every epoch of training. For our optimizer, we use SGD with momentum of 0.9 and weight decay of 5×10^{-4}. We use an initial learning rate of 0.1 which decays using a cosine annealing schedule.

For training a new classification layer on feature extractor checkpoints, we use $5,000$ random clean images from the original training set data. Note that $5,000$ images is 10% of CIFAR-10 and CIFAR-100, but 3.9% of Imagenet subset, etc. Following [12], using the feature extractor, we extract embeddings from this clean subset of data and preprocess the embeddings to have mean zero and unit standard deviation. To retrain the last layer, we use the logistic regression implementation from `scikit-learn` (`sklearn.linear_model.LogisticRegression`).

Linearly Separable Perturbations Are Not Necessary In Section 4.3, for every unlearnable dataset, we first gather the set of perturbations by subtracting the clean image from the perturbed image. We zero-one normalize the perturbations before training a linear layer using L-BFGS [15] for 500 steps using a learning rate of 0.5.

Orthogonal Projection for Learning From Datasets with Class-wise, Linear Perturbations In Section 4.4, for CIFAR-10, we train the logistic regression model for 20 epochs using SGD with an initial learning rate of 0.1, which decays by a factor of 10 on epochs 5 and 8 (at epochs which are 0.5 and 0.75 through training). For CIFAR-100, we train the logistic regression model for 60 epochs due to the higher number of classes. After we orthogonally project the unlearnable data using the optimized weights, we train networks using the hyperparameters from checkpoint-training of Section 4.2.

A.2 Additional Section 4.2 Results: DNNs Can Learn Useful Features From Unlearnable Datasets

A.2.1 More Model Architectures for Section 4.2

We consider three more model architectures: VGG-16 [29], GoogLeNet [30], and ViT [4]. Our ViT uses a patch size of 4. For RN-18 and VGG-16, feature vectors are 512-dimensional. Feature vectors for GoogleNet are 1024-dimensional. The ViT class token is 384-dimensional.

Table 4: **Generalizable features can be learned from unlearnable datasets, using a variety of network architectures.** We report Max DFR Test Accuracy for each CIFAR-10 unlearnable dataset. In gray, we indicate test accuracy improvement/deterioration over DFR on the corresponding randomly initialized model architecture.

	Model Architecture		
CIFAR-10 Training Data	VGG-16	GoogLeNet	ViT
None	35.69	48.08	37.40
Unlearnable Examples [10]	37.84 (+2.15)	41.08 (-7.00)	49.57 (+12.17)
Adversarial Poisoning [7]	64.73 (+29.04)	71.70 (+23.62)	68.97 (+31.57)
AR (ℓ_2) [27]	36.98 (+1.29)	40.12 (-7.96)	60.53 (+23.13)
NTGA [37]	56.03 (+20.34)	61.24 (+13.16)	60.53 (+23.13)
Robust Unlearnable [28]	39.13 (+3.44)	40.59 (-7.49)	49.10 (+11.70)
LSP [36]	40.86 (+5.17)	58.22 (+10.14)	50.95 (+13.55)
OPS+EM [35]	31.31 (-4.38)	38.57 (-9.51)	49.73 (+12.33)
∘ OPS [35]	39.63 (+3.94)	52.02 (+3.94)	56.04 (+18.64)
∘ Unlearnable Examples [10]	30.47 (-5.22)	36.32 (-11.76)	44.90 (+7.50)
∘ Regions-4 [26]	43.29 (+7.60)	48.65 (+0.57)	52.60 (+15.20)
∘ Random Noise	72.08 (+36.39)	62.19 (+14.11)	55.58 (+18.18)

In Table 4, we find that across architectures, *Adversarial Poisoning* data is easiest to extract generalizable features from. Surprisingly, ViT is most effective at learning generalizable features from *all* unlearnable datasets, achieving more than 7% test accuracy improvement over a randomly initialized ViT in all cases. For example, using only $5,000$ clean CIFAR-10 samples can be used to achieve nearly 69% test accuracy, while using the same clean samples can only achieve 37.40% test accuracy on a randomly initialized ViT. The GoogleNet architecture weights are seemingly more easily corrupted during training; Max DFR Test Accuracy for *Unlearnable Examples*, *AR*, *Robust Unlearnable*, and other datasets is much lower than test accuracy from a finetuned randomly initialized GoogleNet. Interestingly, the randomly initialized GoogleNet feature extractor achieves the highest DFR test accuracy.

A.2.2 More Datasets for Section 4.2

We consider three additional base datasets for four unlearnable dataset methods. We use an ImageNet [25] subset of the first 100 classes, following [10]. The train split consists of $129,395$ images, while the test split consists of $5,000$ images.

Our SVHN [18], CIFAR100 [13], and *Adversarial Poisoning* ImageNet subset datasets contain perturbations of size $\|\delta\|_\infty \leq \frac{8}{255}$. *Unlearnable Examples* ImageNet [25] subset contains perturbations of size $\|\delta\|_\infty \leq \frac{16}{255}$, following their open-source repository. We generate the *Adversarial Poisoning* [7] ImageNet subset from published source code using 1 PGD restart, as opposed to 8 due to computation time. Our SVHN and CIFAR-100 *Adversarial Poisoning* datasets use 3 PGD restarts. On clean SVHN, CIFAR-100, and ImageNet subset, RN-18 achieves 96.33%, 74.14%, 78.92% test accuracy respectively.

Table 5: **Generalizable features can be learned from unlearnable datasets of different underlying distributions.** We report Max DFR Test Accuracy for each unlearnable dataset. RN-18 checkpoints are trained on SVHN, CIFAR-100, and ImageNet subset unlearnable datasets, and DFR is performed using $5,000$ clean samples from the corresponding base dataset.

	Finetune Data		
Training Data	SVHN	CIFAR-100	ImageNet
None	32.05	8.13	3.64
Unlearnable Examples [10]	26.76	16.12	8.44
Adversarial Poisoning [7]	87.06	44.37	20.22
∘ Unlearnable Examples [10]	22.48	10.32	7.88
∘ Random Noise	27.35	47.41	23.30

In Table 5, we again show that *Adversarial Poisoning* unlearnable data can be easily used to extract generalizable features regardless of the underlying distribution (base dataset of SVHN, CIFAR-100, or ImageNet). For SVHN, *Adversarial Poisoning* is the only dataset from which the trained feature extractor performs better in Max DFR Test Accuracy (87.06%) over a randomly initialized RN-18 (32.05%). As mentioned in Section 4.2, error-minimizing perturbations of *Unlearnable Examples* tend to be most effective at corrupting weights during training, regardless of underlying finetune data.

A.2.3 Additional Plots for Section 4.2

We add results to the experiment from Figure 2. In Figure 4, unlearnable datasets sufficiently corrupt RN-18 weights during training and prevent DFR from recovering test accuracy.

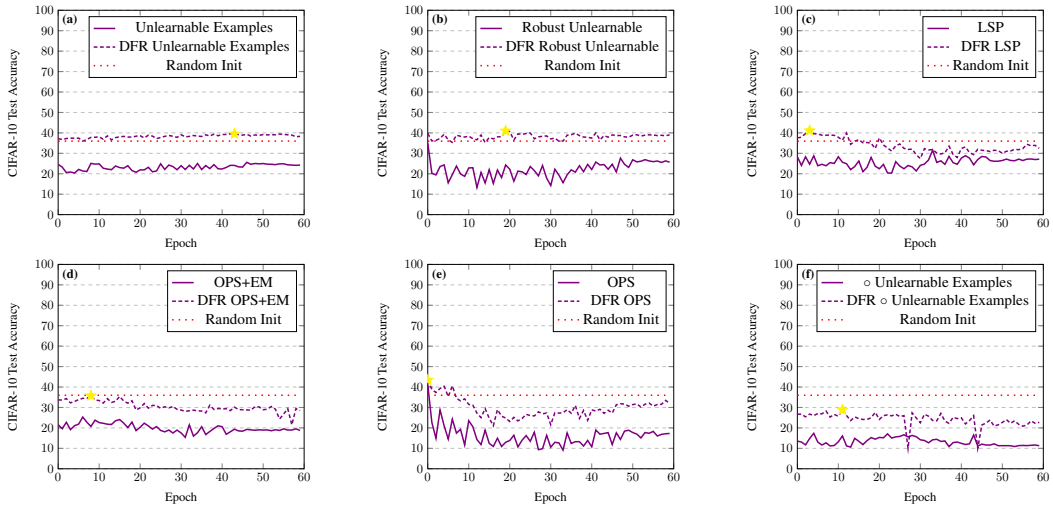

Figure 4: **Representations learned by other poisons we consider are no better than random. (a-b)** Reweighting deep features from sample-wise error minimizing noises provide no benefit over random features. DFR on a randomly initialized RN-18 only achieves 40% test accuracy (dotted line). **(c-f)** Other class-wise perturbations are very effective at corrupting network representations during training – so effective that even DFR is unable to recover test accuracy from random features (red dotted line).

A.3 Additional Results for Section 4.3: Linearly Separable Perturbations Are Not Necessary

A.3.1 Other Background

A related unlearnable dataset introduces entangled features (EntF) [34] which is motivated by the separability of recent poisoning perturbations. However, separability is qualitatively evaluated through t-SNE visualizations, which is different from the separability experiment we perform. More specifically, t-SNE cluster separability should not be equated to the linear separability we measure in Table 2 because it is possible to have linearly separable data that, when plotted using t-SNE, appears not separable. In other words, the EntF poison from [34] could still contain linearly separable perturbations.

A.3.2 Evaluating Linear Separability of Poison Images

In Section 4.3, we document the linear separability of perturbations from various poisons, as in [36]. Poison images, on the other hand, behave slightly differently. In Table 6, we report logistic regression train accuracy on various CIFAR-10 poison images. We find that *Unlearnable Examples, LSP, OPS+EM*, and class-wise poisons have linearly separable poison images, but the remaining poisons we consider do not.

A.4 Additional Section 4.4 Results: Orthogonal Projection for Learning From Datasets with Class-wise, Linear Perturbations

A.4.1 ViT for Section 4.4

To evaluate recovered data from Orthogonal Projection, we consider an additional architecture: ViT. In Table 7, we train ViT with patch size of 4 on CIFAR-10 unlearnable datasets using different attacks. Our Orthogonal Projection method is competitive with adversarial training for all class-wise perturbed unlearnable datasets and most sample-wise perturbed datasets. Orthogonal Projection is the best performing method for *OPS+EM* and *OPS* (ICLR 2023). Note that *OPS+EM* and *OPS* are most difficult for adversarial training.

Table 6: **NTGA contains the least linearly separable images, while AR(ℓ_2) images are most comparable to the clean distribution. Other unlearnable datasets become *more* linearly separable.** We train a linear logistic regression model on poison images and report train accuracy. High train accuracy indicates linear separability of poison images. Mean and one standard deviation computed from 10 independent runs.

Training Data	Train Accuracy
Clean	53.94 ± 0.02
Unlearnable Examples [10]	100.00 ± 0.00
Adversarial Poisoning [7]	62.40 ± 0.01
AR (ℓ_2) [27]	53.97 ± 0.02
NTGA [37]	31.48 ± 0.02
Robust Unlearnable [28]	77.21 ± 0.01
LSP [36]	100.00 ± 0.00
OPS+EM [35]	100.00 ± 0.00
∘ OPS [35]	100.00 ± 0.00
∘ Unlearnable Examples [10]	100.00 ± 0.00
∘ Regions-4 [26]	100.00 ± 0.00
∘ Random Noise	100.00 ± 0.00

Table 7: **Orthogonal Projection can make class-wise unlearnable data learnable for ViT.** Especially for unlearnable datasets with class-wise, linearly separable perturbations, our Orthogonal Projection attack is competitive with ℓ_∞ Adversarial Training at a fraction of the computational cost.

	Attack		
CIFAR-10 Training Data	None	Adv Training	Ortho Proj (Ours)
Clean	84.99	76.38	74.14
Unlearnable Examples [10]	25.39	75.44	60.15
Adversarial Poisoning [7]	31.33	75.15	41.49
AR (ℓ_2) [27]	17.13	75.12	35.11
NTGA [37]	32.67	71.95	66.19
Robust Unlearnable [28]	28.24	78.03	37.34
LSP [36]	29.40	75.45	74.77
OPS+EM [35]	20.73	11.79	51.94
∘ OPS [35]	21.58	10.17	72.80
∘ Unlearnable Examples [10]	12.19	76.35	76.01
∘ Regions-4 [26]	15.00	75.96	67.20
∘ Random Noise	29.66	76.23	73.05

A.4.2 CIFAR-100 Dataset for Section 4.4

We consider an additional dataset, CIFAR-100, to evaluate Orthogonal Projection. We train a RN-18 on four unlearnable dataset methods. During the first step of Orthogonal Projection, we train the logistic regression model for 60 epochs using SGD with an initial learning rate of 0.1, which decays by a factor of 10 on epochs 30 and 45.

Table 8: **Orthogonal Projection is competitive on CIFAR-100 class-wise unlearnable data.**

	Attack		
CIFAR-100 Training Data	None	Adv Training	Ortho Proj (Ours)
Clean	74.14	59.23	26.78
Unlearnable Examples [10]	8.11	58.29	28.44
Adversarial Poisoning [7]	5.93	57.60	25.24
∘ Unlearnable Examples [10]	1.72	60.31	41.83
∘ Random Noise	1.30	58.83	51.23

In Table 8, we find that Orthogonal Projection performs better on class-wise unlearnable data than sample-wise perturbed data, as expected. At approximately the cost of standard training (Attack: None), Orthogonal Projection achives gains of more than 20% test accuracy for sample-wise perturbed data and more than 40% test accuracy for class-wise perturbed data.

A.4.3 Additional Intuition for Orthogonal Projection

Assume CIFAR-10 images of shape $(3, 32, 32)$. Each column i of W (optimized in Alg 1, Lines 1-4) is a 3072-dimensional vector that represents the most predictive image feature for class i. This step serves as recovery of the perturbation. After the QR decomposition of W, Q consists of orthonormal columns that form a basis for the column space of W. When we say Orthogonal Projection "ensures that the dot product of a row of X with every column of Q is zero," (i.e., $X_r \cdot Q = 0$) this means that every recovered image vector does not contain any linearly separable component (i.e., does not contain any column of Q as a component). Alg. 1, Line 6 ensures image vectors and columns of Q are orthogonal and so the dot product is 0. The "recovered" data thus has 10 dimensions (approximations of the 10 perturbations) removed.

A.4.4 Subtracting a Class-wise Image

Given that the goal of Orthogonal Projection is to extract perturbations from poison images, it is reasonable to consider visualizing the average image of a class for class-wise poisons like *LSP* and *OPS*. In Figure 5, we see that class-wise average images somewhat reveal class-wise perturbations, but the results are not clear enough to be useful. In contrast, class-wise patters and clearly present in learned weights from logistic regression.

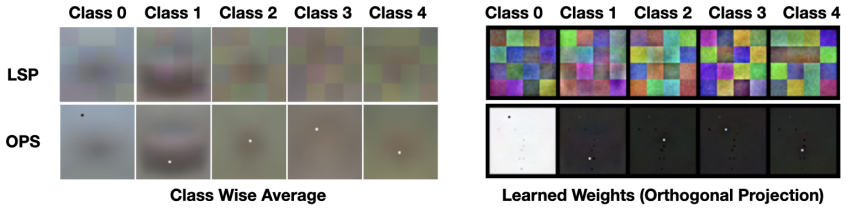

Figure 5: **Average class images display class-wise perturbations, but not as clearly as learned weights from logistic regression.** We compare the average image of a class **(Left)** and learned weights of a logistic regression classifier **(Right)** trained on image pixels (the first step of Orthogonal Projection method) to for *LSP* and *OPS* Poisons. While average image of a class does reveal the class-wise perturbation (block pattern for *LSP* and one highlighted pixel for *OPS*), the result is blurry and contains other semantic image features. Learned weights from Orthogonal Projection properly isolate the perturbation.

For training models on class-wise perturbed data, one might consider subtracting the class-wise average image from each class. However, simply subtracting this average class image from each image does not remove the poisoning effect. Additionally, because we do not know the true class at inference time, we cannot subtract the class image, resulting in a distribution mismatch between train and test sets. This trivial method of subtracting average class images is compared to Orthogonal Projection in Table 9.

A.4.5 Visualizing Additional Logistic Regression Weights

We visualize additional linear model weights (from the first step of Orthogonal Projection) for sample-wise perturbed unlearnable datasets in Figure 6, and for class-wise perturbed unlearnable datasets in Figure 7. We find that for *Adversarial Poisoning*, *AR*, and *Robust Unlearnable* the linear model learns features comparable to when trained on clean data. We posit that because the diversity of perturbations in these datasets is higher, the linear model struggles to find predictive features to project away. In contrast, for class-wise perturbed data, Figure 7, demonstrates that the linear model can recover features that resemble the original class-wise perturbation.

Table 9: **Subtracting the average class image from training images is not effective.** For *LSP* and *OPS*, datasets with class-wise perturbations, our Orthogonal Projection attack improves CIFAR-10 test accuracy over simply subtracting the average class image at training time.

	Attack	
Training Data	Class-Avg Subtract	Ortho Proj (Ours)
LSP [36]	13.05	87.99
○ OPS [35]	12.62	87.94

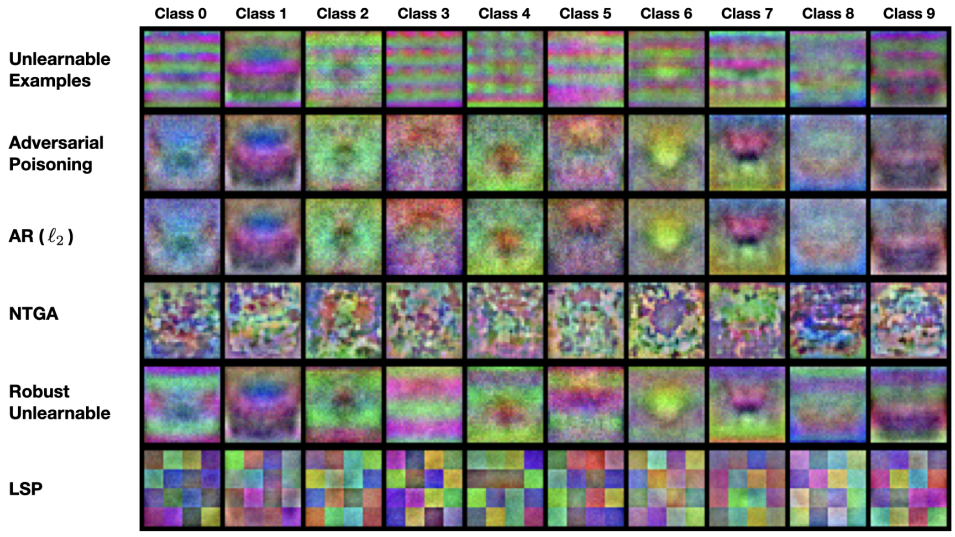

Figure 6: **Learned Weights from first step of Orthogonal Projection on sample-wise perturbed CIFAR-10 unlearnable datasets.** We visualize learned weights (W in Algorithm 1) after training a linear model on unlearnable datasets. Learned weights from *Adversarial Poisoning*, *AR*, and *Robust Unlearnable* resemble the learned weights from clean data (See Figure 3). In our Orthogonal Projection attack, we project each perturbed image to be orthogonal to each of these learned weights (Algorithm 1, Line 6).

A.5 Samples from Unlearnable Datasets

We visualize samples from sample-wise perturbed CIFAR-10 unlearnable datasets in Figure 8, and from class-wise perturbed unlearnable datasets (prefixed by ○ throughout results) in Figure 9. NTGA is omitted due to data ordering of the publicly available poison. We also visualize SVHN samples in Figure 10 and CIFAR-100 in Figure 11.

A.6 Broader Impact Statement

Our findings test prevailing hypothesis about unlearnable datasets and our results have practical implications for their use. Two of our three main conclusions relate to privacy vulnerabilities when employing unlearnable datasets for data protection. In one experiment, we demonstrate useful features can be learned from unlearnable data. In another, we demonstrate how one can effectively remove a class-wise perturbation. Our findings highlight the need for extra caution when it comes to using unlearnable datasets. By making this information available to the public, the capabilities and vulnerabilities of unlearnable datasets can be better understood.

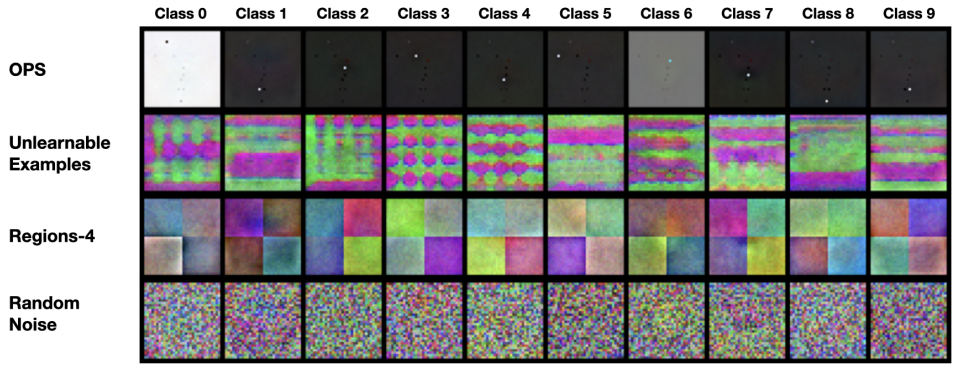

Figure 7: **Learned Weights from first step of Orthogonal Projection on class-wise perturbed CIFAR-10 unlearnable datasets.** We visualize learned weights (W in Algorithm 1) after training a linear model on unlearnable datasets. Learned weights appear to recover the added class-wise perturbation for all datasets. In our Orthogonal Projection attack, we project each perturbed image to be orthogonal to each of these learned weights (Algorithm 1, Line 6).

Figure 8: **Samples from CIFAR-10 unlearnable datasets.** We visualize the first 10 images from each sample-wise perturbed unlearnable dataset.

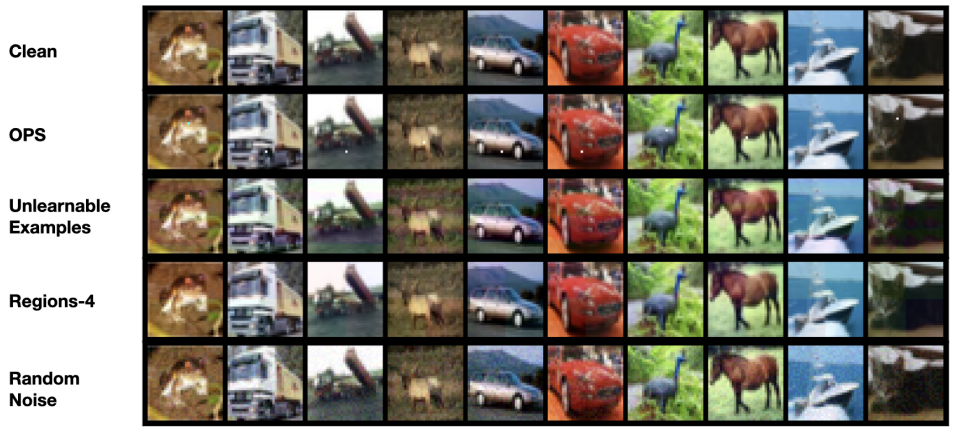

Figure 9: **Samples from CIFAR-10 unlearnable datasets.** We visualize the first 10 images from each class-wise perturbed unlearnable dataset.

Figure 10: **Samples from SVHN unlearnable datasets.**

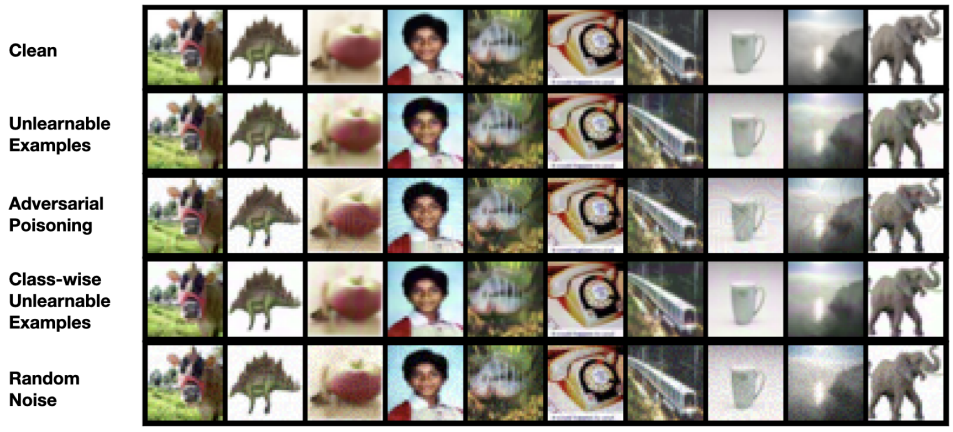

Figure 11: **Samples from CIFAR-100 unlearnable datasets.**