Supplementary: Do Image Classifiers Generalize Across Time?

Vaishaal Shankar* UC Berkeley
Achal Dave* CMU
Rebecca Roelofs UC Berkeley
Deva Ramanan CMU
Benjamin Recht UC Berkeley
Ludwig Schmidt UC Berkeley

1. Overview

This supplementary provides more details on our datasets and experimental analysis. Section 2 discusses the ImageNet-Vid and Youtube-BB datasets in more detail. Section 3 provides further discussion and analysis of our metric, including the definition of detection PM-k (Section 3.1), a comparison to the metric in [3] (Section 3.2), an analysis of how ℓ_∞ distance correlates with PM-k accuracy (Section 3.3), and how PM-k varies with k (Section 3.4). Section 4 presents further experiments and analysis of main paper experiments, including model robustness per class (Section 4.1), how well PM-k errors transfer across models (Section 4.2), and training on ILSVRC with the `ImageNet-Vid-Robust` vocabulary (Section 4.4). Section 5 presents implementation details and hyperparameters. Finally, Sec. 6 presents the full table of original and PM-k accuracy for all models in our test beds for `ImageNet-Vid-Robust` and `YTBB-Robust`.

2. Source Dataset Overview

2.1. ImageNet-Vid

The 2015 ImageNet-Vid dataset is widely used for training video object detectors [4] as well as trackers [1]. We chose to work with the 2017 ImageNet-Vid dataset because it is a superset of the 2015 dataset. In total, the 2017 dataset consists of 1,181,113 training frames from 4,000 videos and 512,360 validation frames from 1,314 videos. The videos have frame rates ranging from 9 to 59 frames per second (fps), with a median fps of 29. The videos range from 0.44 to 96 seconds in duration with a median duration of 12 seconds. Each frame is annotated with labels indicating the presence or absence of 30 object classes and corresponding bounding boxes for any label present in the frame. The 30 classes are ancestors of 293 of the 1,000 ILSVRC-2012 classes.

2.2. Youtube-BB

The 2017 Youtube-BB is a a large scale dataset with 8,146,143 annotated training frames 253,569 unique videos and with 1,013,246 validation frames from 31,829 videos. The video segments are approximately 19 seconds long on average. Each frame is annotated with exactly one label indicating the presence of 22 object classes, all of which are ancestors of 229 out of the ILSVRC-2012 classes.

*Equal contribution

3. Further metric discussion and analysis

3.1. Detection pm-k

We briefly introduce the mAP metric for detection here and refer the reader to [5] for further details. The standard detection metric proceeds by first determining whether each predicted bounding box in an image is a true or false positive, based on the intersection over union (IoU) of the predicted and ground truth bounding boxes. The metric then computes the per-category average precision (AP, averaged over recall thresholds) across all images. The final metric is reported as the mean of these per-category APs (mAP).

We define the `pm-k` analog of mAP by replacing each anchor frame in the dataset with a nearby frame that minimizes the per-image average precision. Since the category-specific average precision is undefined for categories not present in an image, we minimize the average precision across categories present in each frame rather than the mAP.

3.2. Per-frame conditional robustness metric introduced in [3]

In concurrent work, the authors of [3] considered a different metric of robustness. In this section, we compute this metric on all models in our test bed to compare our findings to [3]. There are two main differences between PM-k and the robustness metric in [3].

1. For two visually similar "neighbor" frames I_0 and I_1 with true label Y and classifier f, [3] studies the conditional probability $P(f(I_1) = y | f(I_0) = y)$
2. While PM-k looks for errors in all neighbor frames in a neighborhood of k frames away from the anchor frame (so this would include frames 1, 2, ..., k frames away), [3] only considers errors from **exactly** k frames away.

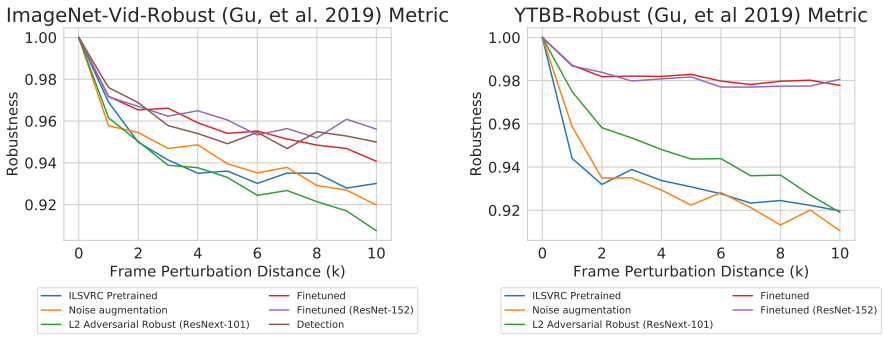

Figure 1: Conditional robustness metric from [3] on perturbed frames as a function of perturbation distance on `ImageNet-Vid-Robust` and `YTBB-Robust`. Model accuracies from five different model types and the best performing model are shown. The model architecture is ResNet-50 unless otherwise mentioned.

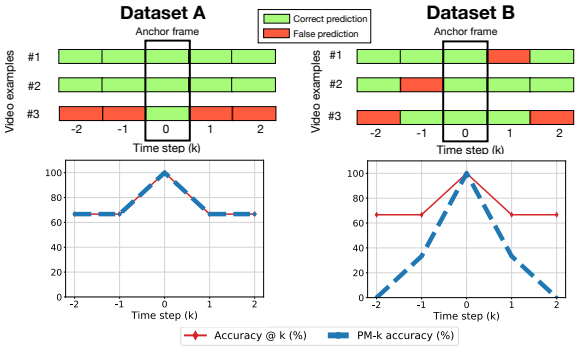

Figure 2: For the two example videos above the score from [3] metric (Accuracy @ K) is identical, but the PM-k metric behaves substantially differently when the errors are spread across many independent videos, as shown in the right example

In Fig. 2 we illustrate simple example where two videos can have the same behavior for the metric introduced by [3] but drastically different behavior for the PM-kmetric.

3.3. ℓ_∞ distance vs PM-k Accuracy

ℓ_∞ adversarial examples are well studied in the robustness community, yet the connection between ℓ_∞ and other forms of more "natural" robustness is unclear. Here, we plot the cumulative distribution of the ℓ_∞ distance between pairs of nearby frames in our datasets. In Figure 3, we show the CDF of ℓ_∞ distance for all pairs, all reviewed pairs, and mistakes made by 3 indicative models. Note the `fbrobust` model is trained specifically to be robust to ℓ_∞ adversaries.

3.4. PM-k Accuracy with varying k

Figure 4 plots the relationship between acc_{pmk} and perturbation distance (i.e., the `k` in the `pm-k` metric). The entire x-axis in Figure 4 corresponds to a temporal distance of at most 0.3 seconds between the original and perturbed frames.

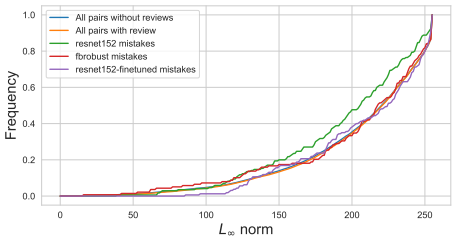

Figure 3: CDF showing the ℓ_∞ distance between pairs of frames from different distributions.

4. Additional model analyses

4.1. Per class analysis

We study the effect of our perturbations on the 30 classes in `ImageNet-Vid-Robust` and `YTBB-Robust` to determine whether the performance drop was concentrated in a few "hard" classes. Figure 5 shows the original and perturbed accuracies across classes for our best performing model (a fine-tuned ResNet-152). Although there are a few particularly difficult classes for perturbed accuracy (e.g., lion or monkey on `ImageNet-Vid-Robust`), the accuracy drop is spread across most classes. On `ImageNet-Vid-Robust`, this model saw a total drop of 14.4% between original and perturbed images and a median drop of 14.0% in per-class accuracy. On `YTBB-Robust`, the total drop was 8.9% and the median drop was 6.7%.

4.2. Model independent perturbed frame selection

We have so far considered *model dependent* perturbations, as we selected the worst neighbor frame *for each model*. Here, we study the same problem but impose a static set of perturbed frames across all models. In Figure 6, we study a static set of perturbations across all models and still see a substantial (but smaller) drop in accuracy for both models.

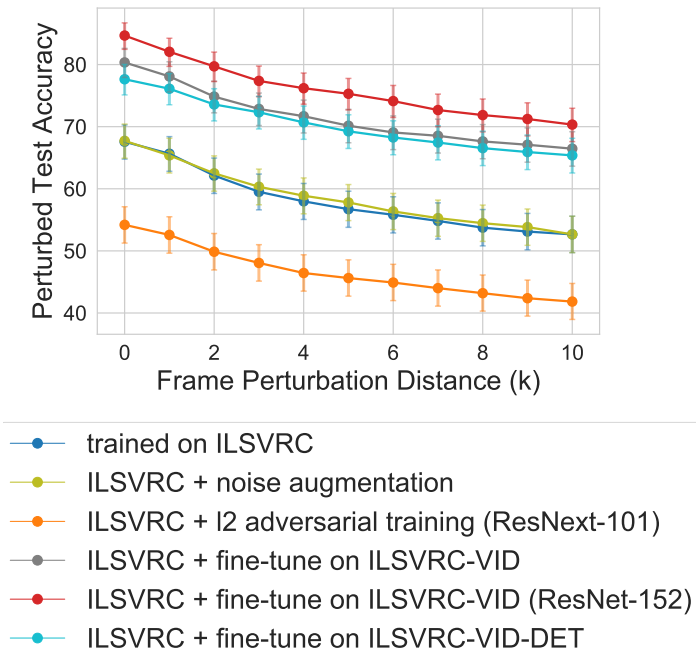

Figure 4: Model classification accuracy on perturbed frames as a function of perturbation distance (shown with 95% Clopper-Pearson confidence intervals). Model accuracies from five different model types and the best performing model are shown. The model architecture is ResNet-50 unless otherwise mentioned.

	Accuracy			
FPS	Original	Perturbed	Δ	# Videos
25	87.3	73.3	14.0	292
29	87.7	74.9	12.8	383
30	78.3	61.7	16.6	313

Table 1: `ImageNet-Vid-Robust` subsets with fixed FPS.

The static set of perturbations were chosen by choosing the neighbor frame that the largest number of models misclassified.

4.3. FPS analysis

Next, we analyze how video frame rate impacts model accuracy. At low frame rates, nearby frames may be more likely to be dissimilar, or exhibit artifacts such as motion blur. We show in Table 1 that videos in `ImageNet-Vid-Robust` range from 25 to 30 FPS. We evaluate a fine-tuned ResNet-152 model on subsets of the dataset corresponding to different frame rates, and find that the *gap* between original and perturbed accuracy is similar across these subsets, and similar to the gap for the entire dataset. This suggests that low frame rates do not account for the drop in accuracy, and different frame rates do not significantly impact the results.

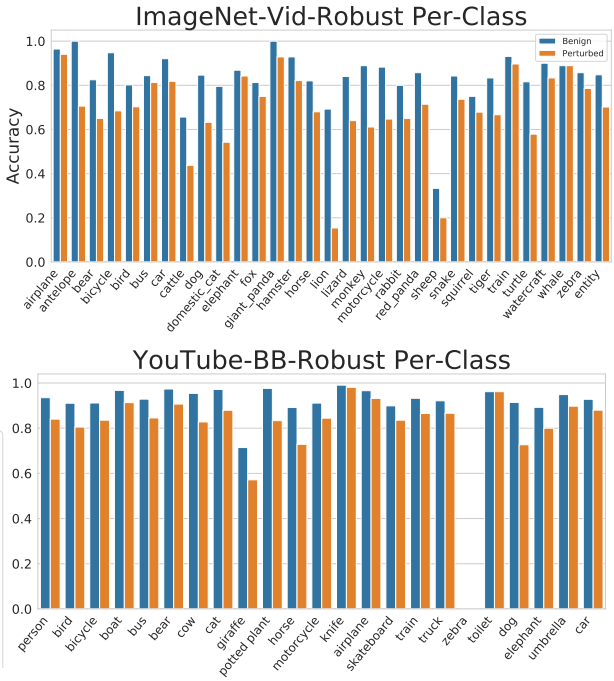

Figure 5: Per-class accuracy statistics for our best performing classification model (fine-tuned ResNet152) on `ImageNet-Vid-Robust` and `YTBB-Robust`. For Youtube-BB, note that 'zebra' is the least common label, present in only 24 anchor frames sampled by [3], of which 4 are included in our dataset.

Table 2: Results of training ResNet-50 on ILSVRC with 30 classes from `ImageNet-Vid-Robust`.

Model	Acc. Orig.	Acc. Perturbed	Drop
ILSVRC-30	61.0	44.9	15.1
ILSVRC-30 + FT	77.8	59.9	17.9

4.4. ILSVRC training with `ImageNet-Vid-Robust` classes

We trained ResNet-50 from scratch on ILSVRC using the 30 ImageNet-Vid classes. We also fine-tuned the model on ImageNet-Vid. In Table 2, we show the accuracy drops are consistent with models in our submission. We hypothesize that the lower accuracy is due to coarser supervision on ILSVRC.

5. Experimental Details & Hyperparameters

All classification experiments were carried out using PyTorch version 1.0.1 on an AWS p3.2xlarge with the NVIDIA V100 GPU. All pretrained models were downloaded from [2] at commit hash `021d97897c9aa76ec759deff43d341c4fd45d7ba`.

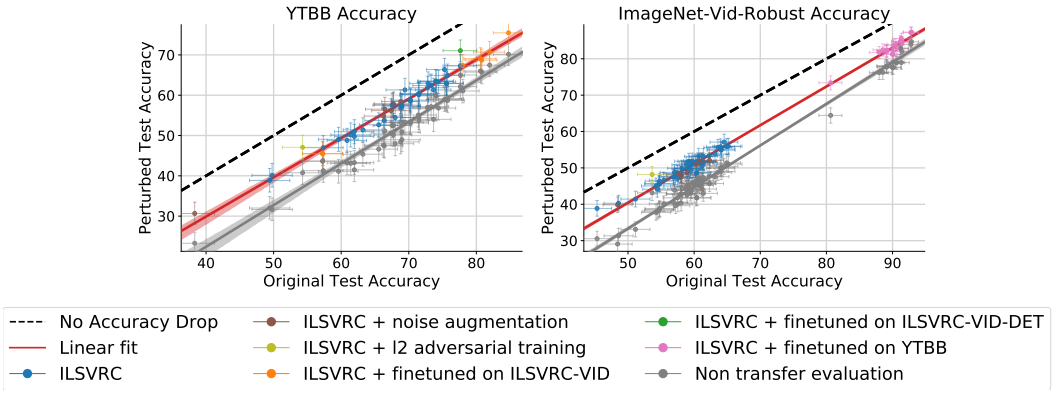

Figure 6: Model accuracy on original vs. perturbed images for a static, model-*independent* set of perturbed frames. The grey points and grey linear fit correspond to the perturbed accuracies of models evaluated on per model perturbations studied in main text. See Section 4.2 for details.

Table 3: Hyperparameters for models finetuned on ImageNet-Vid. Plateau indicates that the LR is automatically decreased when the training loss plateaus.

Model	Base LR	LR Schedule	Batch Size	Epochs
resnet152	10^{-4}	Plateau	32	10
resnet50	10^{-4}	Plateau	32	10
alexnet	10^{-5}	Plateau	32	10
vgg16	10^{-5}	Plateau	32	10

Table 4: Hyperparameters for Faster R-CNN detection models. R50 and R101 are ResNet-50 and ResNet-101, respectively.

Model	Base LR	LR Sched	Batch Size	Iterations
R50	10^{-2}	Step 20k, 30k	8	40k
R101	10^{-2}	Step 20k, 30k	8	40k

Evaluations in Tables 6.1 and 6.2 all use the default settings for evaluation. The hyperparameters for the *fine-tuned* models are presented in Table 3. We searched for learning rates between 10^{-3} and 10^{-5} for all models.

We additionally detail hyperparameters for detection models in Table 4. Detection experiments were conducted with PyTorch version 1.0.1 on a machine with 4 Titan X GPUs, using the Mask R-CNN benchmark repository[6]. We used the default learning rate provided in [6]. For R-FCN, we used the model trained by [7].

For the classification model trained with a pm-k loss in Section 4.1, we use the following setup. We initialize the model from ImageNet (ILSVRC) pre-training, and fine-tune on ImageNet-Vid using the same setup as for regular fine-tuning, where we train for 1 epoch. We only train on a set of k frames if all k frames share the same label, and use this label as the training target.

6. Full Original vs Perturbed Accuracies

6.1. ImageNet-Vid-Robust

Model	Accuracy Original	Accuracy Perturbed	Δ
clip_zeroshot	95.3 [93.8, 96.4]	89.2 [87.2, 91.0]	6.1
clip_linearprobe	89.1 [87.1, 90.9]	77.2 [74.6, 79.6]	11.9
resnet152_finetuned	84.8 [82.5, 86.8]	70.2 [67.4, 72.8]	14.6
resnet50_finetuned	80.8 [78.3, 83.1]	65.7 [62.9, 68.5]	15.1
vgg16bn_finetuned	78.0 [75.4, 80.4]	61.0 [58.1, 63.9]	17.0
nasnetalarge_imagenet_pretrained	77.6 [75.1, 80.1]	62.1 [59.2, 65.0]	15.5
resnet50_detection	77.6 [75.1, 80.1]	65.0 [62.1, 67.8]	12.6
inceptionresnetv2_imagenet_pretrained	75.7 [73.1, 78.2]	58.7 [55.7, 61.6]	17.0
dpn107_imagenet_pretrained	75.6 [72.9, 78.1]	59.1 [56.1, 62.0]	16.5
inceptionv4_imagenet_pretrained	75.3 [72.6, 77.8]	59.0 [56.0, 61.9]	16.3
dpn92_imagenet_pretrained	74.4 [71.7, 76.9]	56.8 [53.8, 59.7]	17.6
dpn131_imagenet_pretrained	74.0 [71.3, 76.6]	59.9 [56.9, 62.8]	14.1
dpn68b_imagenet_pretrained	73.7 [71.0, 76.2]	54.0 [51.0, 57.0]	19.7
resnext101_32x4d_imagenet_pretrained	73.3 [70.6, 75.9]	57.2 [54.2, 60.1]	16.1
resnext101_64x4d_imagenet_pretrained	72.9 [70.1, 75.5]	56.6 [53.7, 59.6]	16.3
resnet152_imagenet_pretrained	72.8 [70.0, 75.4]	57.0 [54.0, 59.9]	15.8
resnet101_imagenet_pretrained	71.5 [68.7, 74.1]	53.7 [50.8, 56.7]	17.8
fbresnet152_imagenet_pretrained	71.5 [68.7, 74.1]	54.5 [51.5, 57.4]	17.0
densenet161_imagenet_pretrained	71.4 [68.7, 74.1]	55.1 [52.1, 58.1]	16.3
densenet169_imagenet_pretrained	70.2 [67.5, 72.9]	53.1 [50.1, 56.1]	17.1
densenet201_imagenet_pretrained	70.2 [67.5, 72.9]	53.4 [50.4, 56.4]	16.8
dpn68_imagenet_pretrained	69.4 [66.6, 72.1]	53.3 [50.3, 56.3]	16.1
bninception_imagenet_pretrained	69.0 [66.2, 71.7]	49.0 [46.0, 51.9]	20.0
densenet121_imagenet_pretrained	69.0 [66.2, 71.7]	50.9 [47.9, 53.8]	18.1
nasnetamobile_imagenet_pretrained	68.8 [66.0, 71.5]	48.4 [45.4, 51.4]	20.4
resnet50_augment___jpeg_compression	68.8 [66.0, 71.5]	53.2 [50.2, 56.2]	15.6
resnet34_imagenet_pretrained	68.0 [65.2, 70.7]	48.0 [45.0, 51.0]	20.0
resnet50_augment___impulse_noise	67.7 [64.9, 70.5]	50.2 [47.2, 53.2]	17.5
resnet50_augment__gaussian_blur	67.7 [64.9, 70.5]	52.5 [49.5, 55.5]	15.2
resnet50_imagenet_pretrained	67.5 [64.7, 70.3]	52.5 [49.5, 55.5]	15.0
resnet50_augment___gaussian_noise	67.4 [64.5, 70.1]	50.6 [47.6, 53.6]	16.8
resnet50_augment___shot_noise	66.5 [63.6, 69.2]	51.1 [48.1, 54.1]	15.4
vgg16_bn_imagenet_pretrained	66.4 [63.5, 69.1]	47.4 [44.5, 50.4]	19.0
resnet50_augment___defocus_blur	66.3 [63.4, 69.1]	47.6 [44.6, 50.6]	18.7
vgg19_bn_imagenet_pretrained	65.6 [62.7, 68.4]	46.6 [43.6, 49.6]	19.0
vgg19_imagenet_pretrained	63.2 [60.3, 66.1]	45.4 [42.4, 48.3]	17.8
resnet18_imagenet_pretrained	61.9 [59.0, 64.8]	41.5 [38.6, 44.4]	20.4
vgg13_bn_imagenet_pretrained	61.9 [59.0, 64.8]	43.3 [40.3, 46.3]	18.6
vgg16_imagenet_pretrained	61.4 [58.5, 64.3]	43.1 [40.2, 46.1]	18.3
vgg11_bn_imagenet_pretrained	60.9 [57.9, 63.8]	43.2 [40.3, 46.2]	17.7
vgg13_imagenet_pretrained	59.6 [56.6, 62.5]	41.1 [38.2, 44.1]	18.5
vgg11_imagenet_pretrained	57.3 [54.4, 60.3]	41.3 [38.4, 44.3]	16.0
alexnet_finetuned	57.3 [54.3, 60.2]	43.6 [40.7, 46.6]	13.7
ResNeXtDenoiseAll-101_robust_pgd	54.3 [51.3, 57.2]	40.8 [37.8, 43.7]	13.5
squeezenet1_1_imagenet_pretrained	49.8 [46.8, 52.8]	31.7 [28.9, 34.5]	18.1
alexnet_imagenet_pretrained	49.4 [46.4, 52.4]	32.0 [29.3, 34.8]	17.4
resnet50_augment___contrast_change	38.3 [35.5, 41.3]	23.3 [20.8, 25.9]	15.0

Table 5: Classification model perturbed and original accuracies for all models in our test bed evaluated on the ImageNet-Vid-Robust dataset.

6.2. YTBB-Robust

Model	Accuracy Original	Accuracy Perturbed	Δ
clip_zeroshot	95.2 [93.9, 95.8]	88.5 [87.0, 89.8]	6.7
clip_linearprobe	68.7 [66.6, 70.7]	63.1 [61.0, 65.2]	5.6
resnet152_finetuned	92.9 [91.2, 94.3]	84.7 [82.4, 86.8]	8.2
resnet50_finetuned	91.4 [89.6, 93.0]	82.0 [79.6, 84.2]	9.4
inceptionresnetv2_finetuned	91.3 [89.5, 92.9]	79.0 [76.4, 81.3]	12.3
vgg19_finetuned	90.5 [88.6, 92.2]	79.1 [76.5, 81.4]	11.4
vgg16_finetuned	89.1 [87.1, 90.8]	78.0 [75.4, 80.4]	11.1
inceptionv4_finetuned	88.5 [86.5, 90.3]	76.3 [73.6, 78.7]	12.2
resnet18_finetuned	88.0 [85.9, 89.8]	76.2 [73.6, 78.7]	11.8
alexnet_finetuned	80.6 [78.2, 82.9]	64.4 [61.5, 67.3]	16.2
pnasnet5large_imagenet_pretrained	65.2 [62.3, 68.0]	51.0 [48.0, 54.0]	14.2
nasnetalarge_imagenet_pretrained	64.9 [62.0, 67.7]	51.4 [48.4, 54.4]	13.5
inceptionresnetv2_imagenet_pretrained	64.5 [61.6, 67.4]	50.4 [47.5, 53.4]	14.1
dpn98_imagenet_pretrained	64.1 [61.2, 66.9]	49.0 [46.0, 52.0]	15.1
dpn107_imagenet_pretrained	64.1 [61.2, 66.9]	50.1 [47.2, 53.1]	14.0
dpn131_imagenet_pretrained	64.0 [61.1, 66.8]	49.9 [46.9, 52.9]	14.1
inceptionv4_imagenet_pretrained	63.6 [60.7, 66.4]	48.8 [45.8, 51.8]	14.8
xception_imagenet_pretrained	63.2 [60.2, 66.0]	47.6 [44.6, 50.6]	15.6
dpn92_imagenet_pretrained	62.3 [59.3, 65.1]	47.7 [44.8, 50.7]	14.6
resnet50_augment__jpeg_compressioon	62.3 [59.4, 65.2]	45.7 [42.8, 48.7]	16.6
polynet_imagenet_pretrained	61.4 [58.4, 64.3]	47.3 [44.4, 50.3]	14.1
nasnetamobile_imagenet_pretrained	61.4 [58.4, 64.3]	43.0 [40.1, 46.0]	18.4
resnet50_augment__shot_noise	61.3 [58.3, 64.2]	46.4 [43.4, 49.3]	14.9
dpn68_imagenet_pretrained	61.2 [58.3, 64.1]	44.2 [41.2, 47.2]	17.0
fbresnet152_imagenet_pretrained	61.1 [58.1, 64.0]	45.9 [42.9, 48.8]	15.2
resnet152_imagenet_pretrained	60.8 [57.8, 63.7]	46.5 [43.5, 49.5]	14.3
resnet101_imagenet_pretrained	60.8 [57.8, 63.7]	45.2 [42.2, 48.2]	15.6
senet154_imagenet_pretrained	60.7 [57.7, 63.6]	47.2 [44.3, 50.2]	13.5
resnet50_augment__impulse_noise	60.6 [57.7, 63.5]	45.5 [42.6, 48.5]	15.1
se_resnet101_imagenet_pretrained	60.5 [57.6, 63.4]	45.6 [42.6, 48.6]	14.9
bninception_imagenet_pretrained	60.4 [57.4, 63.3]	41.8 [38.9, 44.7]	18.6
densenet161_imagenet_pretrained	60.2 [57.3, 63.1]	46.4 [43.4, 49.4]	13.8
resnet50_augment__gaussian_noise	60.2 [57.3, 63.1]	45.7 [42.8, 48.7]	14.5
se_resnext50_32x4d_imagenet_pretrained	59.9 [56.9, 62.8]	45.7 [42.7, 48.6]	14.2
dpn68b_imagenet_pretrained	59.7 [56.7, 62.6]	45.9 [42.9, 48.8]	13.8
inceptionv3_imagenet_pretrained	59.6 [56.6, 62.5]	43.8 [40.8, 46.8]	15.8
densenet121_imagenet_pretrained	59.5 [56.5, 62.4]	43.1 [40.1, 46.0]	16.4
se_resnext101_32x4d_imagenet_pretrained	59.2 [56.3, 62.1]	45.2 [42.3, 48.2]	14.0
densenet201_imagenet_pretrained	59.2 [56.2, 62.1]	44.8 [41.8, 47.8]	14.4
densenet169_imagenet_pretrained	59.2 [56.2, 62.1]	44.6 [41.7, 47.6]	14.6
resnet50_augment__brightness_change	58.9 [56.0, 61.8]	42.6 [39.6, 45.5]	16.3
se_resnet50_imagenet_pretrained	58.8 [55.9, 61.7]	44.1 [41.1, 47.1]	14.7
se_resnet152_imagenet_pretrained	58.8 [55.9, 61.7]	44.8 [41.9, 47.8]	14.0
cafferesnet101_imagenet_pretrained	58.2 [55.2, 61.1]	44.3 [41.3, 47.3]	13.9
resnet50_augment__regular	58.0 [55.1, 61.0]	42.9 [39.9, 45.8]	15.1

resnet34_imagenet_pretrained	57.9 [55.0, 60.9]	42.8 [39.8, 45.7]	15.1
vgg19_imagenet_pretrained	57.5 [54.6, 60.5]	40.1 [37.2, 43.1]	17.4
resnet50_augment__gaussian_blur	57.5 [54.5, 60.4]	41.8 [38.9, 44.7]	15.7
vgg16_bn_imagenet_pretrained	57.2 [54.2, 60.1]	39.6 [36.7, 42.6]	17.6
resnet50_imagenet_pretrained	57.0 [54.1, 60.0]	43.8 [40.9, 46.8]	13.2
vgg19_bn_imagenet_pretrained	56.8 [53.9, 59.8]	40.6 [37.7, 43.5]	16.2
vgg16_imagenet_pretrained	55.4 [52.4, 58.4]	40.1 [37.2, 43.1]	15.3
vgg13_bn_imagenet_pretrained	54.8 [51.8, 57.7]	38.6 [35.7, 41.6]	16.2
vgg11_bn_imagenet_pretrained	54.8 [51.8, 57.7]	38.8 [35.9, 41.8]	16.0
vgg11_imagenet_pretrained	54.7 [51.7, 57.6]	38.4 [35.5, 41.3]	16.3
resnet18_imagenet_pretrained	54.4 [51.4, 57.4]	38.1 [35.2, 41.0]	16.3
vgg13_imagenet_pretrained	54.2 [51.3, 57.2]	37.7 [34.9, 40.7]	16.5
ResNeXtDenoiseAll-101_robust_pgd	53.6 [50.7, 56.6]	43.2 [40.2, 46.1]	10.4
squeezenet1_0_imagenet_pretrained	51.1 [48.1, 54.1]	33.1 [30.3, 36.0]	18.0
squeezenet1_1_imagenet_pretrained	48.6 [45.6, 51.6]	31.3 [28.6, 34.2]	17.3
resnet50_augment__defocus_blur	48.4 [45.4, 51.4]	29.1 [26.4, 31.8]	19.3
alexnet_imagenet_pretrained	45.3 [42.4, 48.3]	30.5 [27.8, 33.3]	14.8

Table 6: Classification model perturbed and original accuracies for all models in our test bed evaluated on the YTBB-robust dataset..

References

[1] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *ECCV*, 2016. 1

[2] Remi Cadene. Pretrained models for pytorch. https://github.com/Cadene/pretrained-models.pytorch. Accessed: 2019-05-20. 3

[3] Keren Gu, Brandon Yang, Jiquan Ngiam, Quoc Le, and Jonathan Shlens. Using videos to evaluate image model robustness. *arXiv preprint arXiv:1904.10076*, 2019. 1, 2, 3

[4] Wei Han, Pooya Khorrami, Tom Le Paine, Prajit Ramachandran, Mohammad Babaeizadeh, Honghui Shi, Jianan Li, Shuicheng Yan, and Thomas S Huang. Seq-nms for video object detection. *arXiv preprint arXiv:1602.08465*, 2016. 1

[5] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. MS COCO detection evaluation. Accessed: 2019-05-16. 1

[6] Francisco Massa and Ross Girshick. maskrcnn-benchmark: Fast, modular reference implementation of Instance Segmentation and Object Detection algorithms in PyTorch. https://github.com/facebookresearch/maskrcnn-benchmark, 2018. Accessed: 2019-05-20. 4

[7] Fanyi Xiao and Yong Jae Lee. Video object detection with an aligned spatial-temporal memory. In *ECCV*, 2018. 4