Natural Color Fool: Towards Boosting Black-box Unrestricted Attacks (Supplementary Material)

Anonymous Author(s)
Affiliation
Address
email

CONTENTS

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

A Color Distribution Library

In this section, we explain how to build our color distribution library. First, all semantic classes are extracted based on the images and annotations of the ADE20K training set [1] (ADE20K contains 150 semantic classes). The semantic classes are clustered into a set of 20 different dominant distribution sets according to their color styles using hierarchical clustering, which is the "distribution of color distribution" (DoD) [2]. Then, since the color distributions of semantic classes in each cluster are similar, we select one object in each cluster to represent that style. Finally, we extract the color distributions to form our color distribution library. The pipeline is shown in Figure 1.

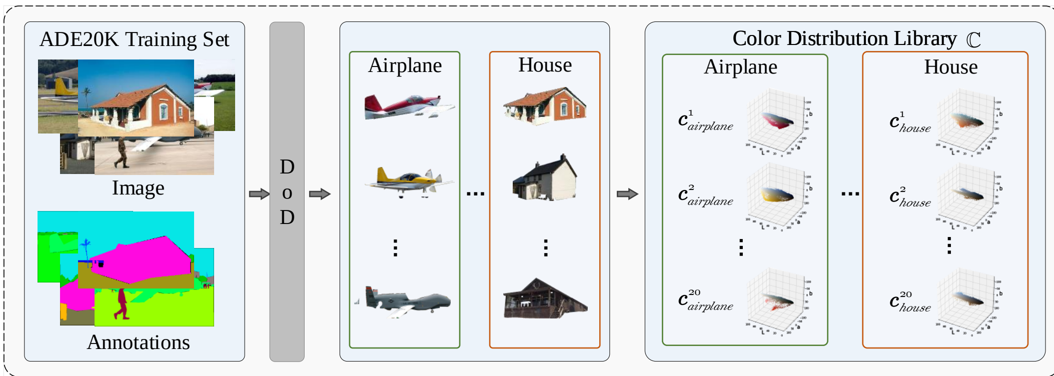

Figure 1: **The pipeline of color distribution library.** We select 20 objects with different color styles for each semantic class and extract their color distributions to build a color distribution library.

B Algorithm

In this section, we provide the algorithm of NCF (see Algorithm 1).

Algorithm 1 Natural Color Fool

Input:
$\boldsymbol{x}$: clean image, y: label, $\tilde{y}$: the semantic class, $\mathcal{S}$: semantic segmentation model,
$\mathcal{F}_\phi(\cdot)$: the substitute model, $\mathbb{C}$: color distribution library, K: the restart number,
N: the iteration of NS, u: the momentum, α: the step size,
Σ: covariance matrix, μ: mean

Output: $\boldsymbol{x}'$: adversarial image

1: Obtain all semantic classes $\tilde{Y}$ of image $\boldsymbol{x}$ by $\mathcal{S}$.
2: Calculate each semantic class's pixel ratio w in the $\boldsymbol{x}$.
3: $\bar{\boldsymbol{x}} \leftarrow rgb2lab(x)$
4: **for** $i \leftarrow 1$ to K **do** ▷ Initialization Reset.
5: **for** $j \leftarrow 1$ to η **do**
6: Randomly sample the color distribution $\boldsymbol{c}_{\tilde{y}}$ of each semantic class $\tilde{y}$ from the library $\mathbb{C}_{\tilde{y}}$
7: $\boldsymbol{H}_j \leftarrow \sum_{\tilde{y}\in\tilde{Y}} w_{\tilde{y}} * \boldsymbol{c}_{\tilde{y}}$
8: $\bar{\boldsymbol{x}}_{\boldsymbol{H}_j} \leftarrow$ Convert $\boldsymbol{H}_j$ to an image without spatial information.
9: $\Sigma_{\bar{\boldsymbol{x}}_{\boldsymbol{H}_j}}, \mu_{\bar{x}_{H_j}} \leftarrow$Calculate covariance matrix and channel means for $\bar{\boldsymbol{x}}_{\boldsymbol{H}_j}$.
10: $\boldsymbol{T} \leftarrow \text{MKSolution}(\Sigma_{\bar{x}}, \Sigma_{\bar{x}_{H_j}})$ (see Eq.(6))
11: $\bar{\boldsymbol{x}}'_{\boldsymbol{H}_j} \leftarrow \boldsymbol{T}(\bar{x} - \mu_{\bar{x}}) + \mu_{\bar{\boldsymbol{x}}_{\boldsymbol{H}_j}}$
12: **end for**
13: $\mathbb{H} \leftarrow \{\boldsymbol{H}_j\}_{j=1}^{\eta}$
14: $\boldsymbol{H}^* \leftarrow \arg\max_{\boldsymbol{H}\in\mathbb{H}} \mathcal{L}_{adv}(\mathcal{F}_\phi(\boldsymbol{x}'_{\boldsymbol{H}}), y)$
15: $\Sigma_{\bar{\boldsymbol{x}}_{\boldsymbol{H}^*}}, \mu_{\bar{x}_{H^*}} \leftarrow$Calculate covariance matrix and channel means for $\bar{\boldsymbol{x}}_{\boldsymbol{H}^*}$.
16: $\boldsymbol{T}^* \leftarrow \text{MKSoulution}(\Sigma_{\bar{x}}, \Sigma_{\bar{x}_{H^*}})$ (see Eq.(6))

```
    μ* ← μ_{x̄_{H*}}
    Initialize T'_0 ← T*
    for n ← 1 to N do                                  ▷ Neighborhood Search.
        x̄'_i ← T'_{n-1}(x̄ − μ_x̄) + μ*
        g ← ∇_{T'} L_adv(F_φ(lab2rgb(x̄'_i)), y)
        g_n ← u · g_{n-1} + g / ||g||_1
        T'_n ← T'_{n-1} + α · sign(g_n)
    end for
    x'_i ← lab2rgb(x̄'_i)
end for
i' ← argmax_{i∈1,2,...,K} L_adv(F_φ(x'_i), y)
x' ← x'_{i'}
return x'
```

C Initialization Reset

As mentioned in Section 3.3, initialization reset (IR) avoids the optimization of convert matrix to fall into local optimum. Therefore, we study the appropriate value for the number of resets in this section. Our adversarial examples are crafted via Inc-v3 by NCF with different resets, i.e., from 1 to 20. We report the attack success rate on the victim model in Figure 2. As demonstrated in the figure, in the beginning, there is an approximate positive relationship between the attack success rate and the number of resets. Once $K = 10$ or larger, the attack success rate maintains stable. We choose $K = 10$ to reduce the computational overhead, where the attack successes rate is 84.0%, 57.7%, 57.6%, 56.8%, 40.1%, 47.6% for Inc-v3 [3], Res-18 [4], VGG-19 [5], Mobile-v2 [6], Dense-121 [7] and Res-50 [4], respectively.

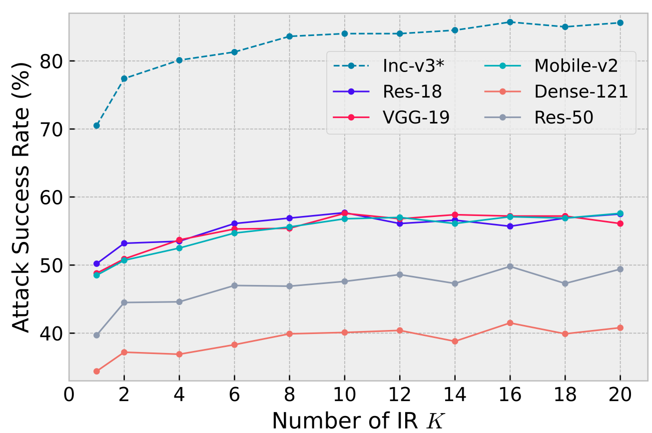

Figure 2: **Attack success rates (%) w.r.t. the number of IR K.** The substitute model is Inc-v3.

D Transferability on ViTs

Recent work [8] points out that the vision transformer models are more concerned with low-frequency information, thus we further evaluate attack success rate of black-box when Transformer models are used as substitute models. Specifically, we generate adversarial examples using ViT-S/16 (ViT-S) [9] and XCiT-N12/16 (XCiT-N12) [10] as substitute models and evaluate the adversarial examples on multiple Transformer models (including ViT-S,XCiT-N12,DeiT-S [9],ViT-B [11],Swin-T [12],PiT-Ti [13]) and multiple CNNs (including Res-18 [4], VGG-19 [5], Mobile-v2 [6]). Table 1 summarizes the results on different black-box models. We can observe that our NCF usually achieves the highest transferability on black-box. In particular, when transferring from ViTs to CNNs, NCF achieves an average success rate of 59.8%, but other attacks only achieve 48.3% (SAE), 29.9% (ReColorAdv), 36.8% (cAdv), 35.1% (ColorFool), 27.3% (ACE).

We observe that the transferability of adversarial examples between ViTs is weaker than different CNNs (see Section 4.2.1). This is consistent with the observation in [14]. It may be since that ViTs have multiple self-attention blocks that can generate class tokens independently. However, our loss only utilizes the last logit (which is equivalent to utilizing only the last class token). The discriminative information in the previous tokens is not directly utilized, which results the poor transferability of the adversarial examples between different ViTs.

Table 1: **Transferability on ViTs.** We report attack success rates (%) of each method and the leftmost model column denotes the substitute model ("*" means white-box attack results).

Model	Attacks	Transformers						CNNs		
		ViT-S	XCiT-N12	DeiT-S	ViT-B	Swin-T	PiT-Ti	Res-18	VGG-19	Mobile-v2
	Clean	13.3	13.7	5.8	10.7	5.0	11.6	16.1	11.4	12.8
ViT-S	SAE	98.2*	38.7	24.2	37.5	19.6	34.0	49.9	47.6	46.5
	ReColorAdv	96.2*	27.4	20.4	30.6	13.8	24.5	27.6	23.7	26.3
	cAdv	**100.0***	36.5	**32.8**	42.4	19.3	35.4	35.5	27.1	33.4
	ColorFool	99.2*	23.6	11.0	24.3	7.6	21.5	35.0	26.7	29.3
	ACE	99.7*	21.1	10.5	20.0	7.1	19.6	29.8	23.8	28.4
	NCF (Ours)	93.9*	**42.4**	25.9	**62.7**	**20.4**	**39.9**	**57.5**	**55.4**	**54.4**
XCiT-N12	SAE	49.3	86.5*	25.9	36.9	18.9	39.9	51.2	47.5	47.0
	ReColorAdv	21.2	95.9*	20.2	16.4	14.7	37.4	36.3	29.5	35.9
	cAdv	39.4	**100.0***	**38.5**	33.0	26.5	53.9	46.9	36.1	41.7
	ColorFool	47.6	85.9	12.6	34.4	9.7	26.5	43.9	36.5	39.4
	ACE	19.5	98.7	9.8	15.0	6.3	21.0	29.5	24.9	27.3
	NCF (Ours)	**54.1**	89.1*	36.1	**38.5**	**27.3**	**55.8**	**65.1**	**63.7**	**62.6**

E Neighborhood Search Impact on Human Perception

In this section, we study the impact of neighborhood search (NS) on human perception. Figure 3 shows the adversarial examples with and without NS, and we observe that our NS strategy has almost no effect on human perception.

Figure 3: **Visualizations of NCF-IR and NCF-IR-NS.** We use Inc-v3 as an alternative model to generate adversarial examples. We observe from the results that there is no perceptual difference after using NS.

F Visualization of Attention

In this section, we show the Gradient-weighted Class Activation Mapping (Grad-CAM) [15] of different adversarial examples. Figure 4 illustrates that all of SAE, ReColorAdv, cAdv, ColorFool, and ACE have difficulty shifting the attention of the model, while our NCF can dramatically it, i.e., no longer on the target object.

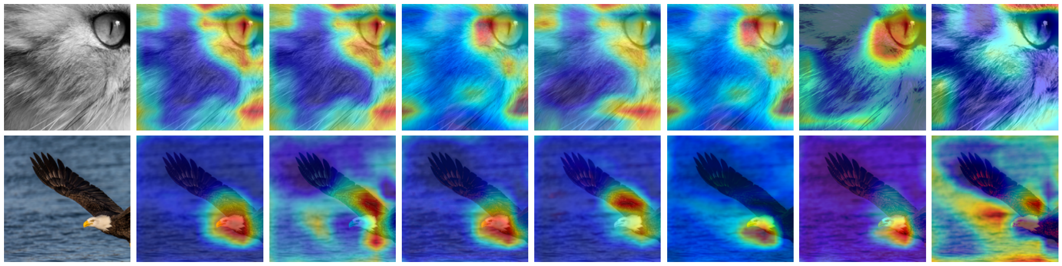

Figure 4: **Visualization of attention.** We use Inc-v3 as the alternative model to generate adversarial examples, and then visualize the attention on VGG-19.

G Visualization of Adversarial Examples

In this section, we show some adversarial examples. In Figure 5, we observe that SAE is prone to global color distortion due to the unbounded range of variation. ReColorAdv limits the perturbation to a small range, thus its adversarial examples are less perceptually different from the clean images. However, some areas of color gradients (e.g., the "clouds" in the second row of Figure 5) may produce some perceptual anomalies. cAdv relies on the colorization model, and some color anomalies are likely to occur locally in the image. ColorFool does not set the perturbation range in non-sensitive areas, resulting in non-sensitive areas prone to color distortion. ACE is similar to ReColorAdv in that sometimes details are lost. (e.g., the "plants" in the third row of Figure 5). NCF transforms the color distribution so that the images' details are preserved.

H The Influence of Segmentation Models

In this section, we compare the performance of our NCF under different semantic segmentation models pre-trained on ADE20K (including Swin-T [12], OCRNet [16] and Deeplabv3+ [17]). As indicated in Table 2, segmentation models have impact on the attack success rates of resulting adversarial examples. Among these models, Swin-T is usually the best choice for our NCF. Therefore, in our paper, we choose it to segment inputs. Note that even if the segmentation model affects our method, the lowest black-box attack success rate of NCF is still much higher than the existing methods.

Table 2: The influence of different segmentation models on attack success rates. ("*" denotes the white-box attack)

Segm	Res-18*	VGG-19	Mobile-v2	Inc-v3	Dense-121	Res-50	ViT-S
Swin-T	**92.9***	**72.1**	**72.7**	**48.3**	**55.3**	**66.7**	53.0
OCRNet	89.9*	69.1	67.1	44.2	50.6	61.1	**56.5**
Deeplabv3+	91.0*	68.0	68.6	45.3	49.2	62.0	54.0

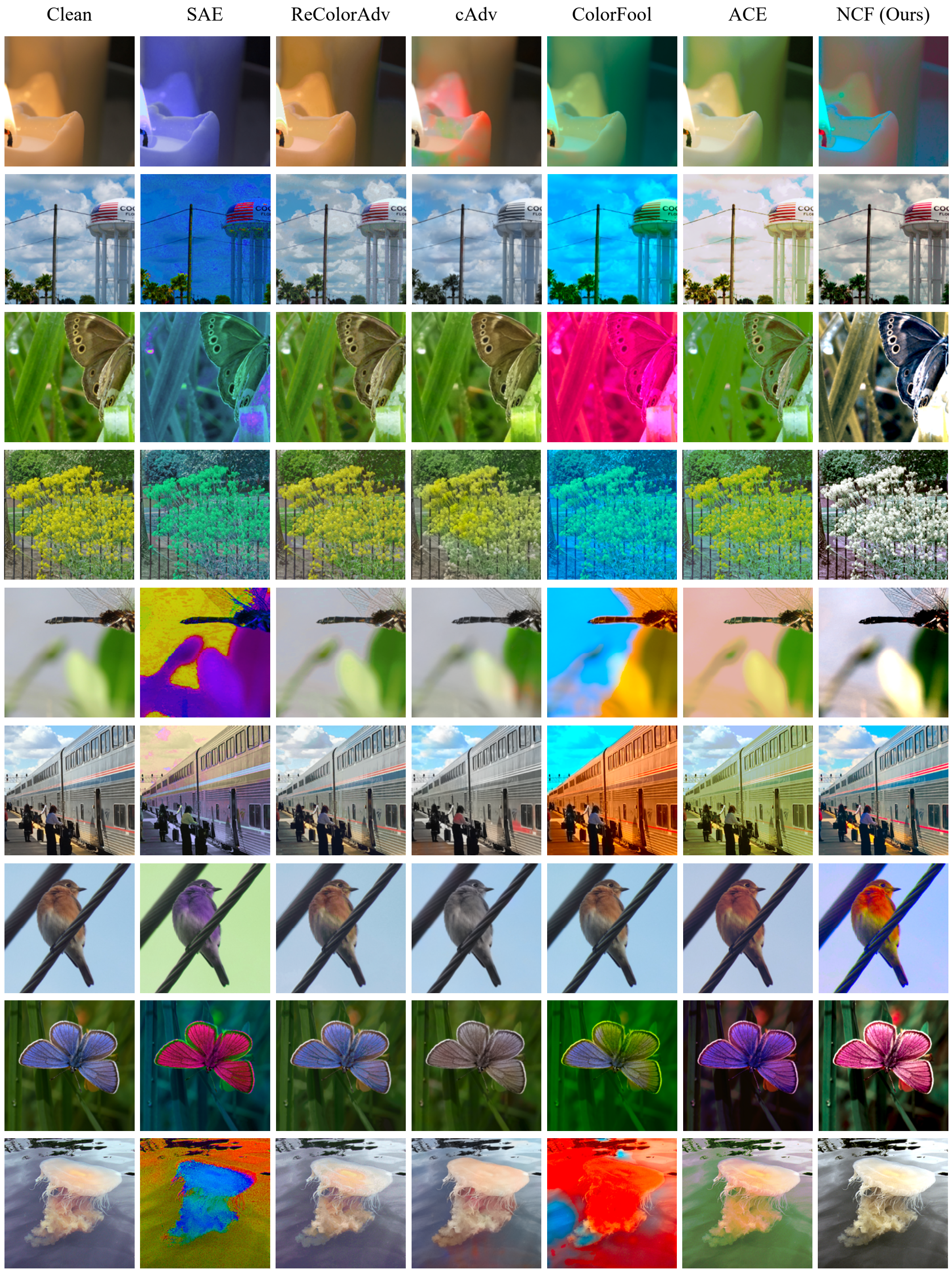

Figure 5: **Visualization of adversarial examples.** All adversarial examples are generated on Inc-v3.

I The Effect of Ensemble Attack

As for ensemble model attack (fusing the logits of multiple models like [18]), here we report the result of NCF. As indicated in Table 3, the attack success rate of NCF can be further improved when crafting via an ensemble of models.

Table 3: Comparison of ensemble attack and single model attack. We report attack success rates (%) of NCF and the leftmost model column denotes the substitute model, where Ensemble means an ensmeble of Res-18, VGG-19 and Mobile-v2.

Model	Dense-121	Res-50	ViT-S	XCiT-N12	DeiT-S
Clean	7.9	7.5	13.3	13.7	5.8
Res-18	55.3	66.7	53.0	55.3	32.8
VGG-19	53.6	64.3	56.5	53.5	30.7
Mobile-V2	54.4	66.2	55.4	56.4	32.6
Ensemble	**63.5**	**71.6**	**59.7**	**61.7**	**37.0**

J Attack by Selecting Random Colors

In this section, we discuss the difference between NCF-IR-NS and random color attack. Formally, NCF-IR-NS does not mean selecting random colors to attack. Specifically, it first generates a set of adversarial examples with different color distributions and then selects the best example from them based on the loss of the white-box model to attack. Therefore, NCF-IR-NS is close to a white-box attack.

To support our claim, we evaluate the performance of random color attack (NCF-IR-NS-), i.e., randomly select colors for each semantic class and use the resulting adversarial examples to attack. As demonstrated in Table 4, the performance of NCF-IR-NS- is much lower than NCF-IR-NS. For example, NCF-IR-NS- only achieves a 27.3% (degraded from 51.6%) success rate on Inc-v3. Thus, directly using random colors to generate adversarial examples is ineffective.

Table 4: The attack success rate of using white-box information and not using it. NCF-IR-NS using Inc-v3 as the substitute model. ("*" denotes the white-box attack)

Methods	Inc-v3*	Res-18	VGG-19	Mobile-v2	Dense-121	Res-50	ViT-S	XCiT-N12	DeiT-S
Clean	19.2*	16.1	11.4	12.8	7.9	7.5	13.3	13.7	5.8
NCF-IR-NS	**51.6***	**43.8**	**42.2**	**42.4**	**28.0**	**33.0**	**38.3**	**32.0**	**14.8**
NCF-IR-NS-*	27.3	34.8	30.9	31.1	20.5	24.2	32.7	25.0	11.6

References

[1] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *CVPR*, 2017. 2

[2] Mahmoud Afifi, Brian L. Price, Scott Cohen, and Michael S. Brown. Image recoloring based on object color distributions. In *Eurographics*, 2019. 2

[3] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 3

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3

[5] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 3

[6] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018. 3

[7] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 3

[8] Namuk Park and Songkuk Kim. How do vision transformers work? In *ICLR*, 2022. 3

[9] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021. 3

[10] Alaaeldin Ali, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, and Hervé Jégou. Xcit: Cross-covariance image transformers. In *NeurIPS*, 2021. 3

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 3

[12] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 9992–10002. IEEE, 2021. 3, 5

[13] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In *ICCV*, 2021. 3

[14] Muzammal Naseer, Kanchana Ranasinghe, Salman Khan, Fahad Khan, and Fatih Porikli. On improving adversarial transferability of vision transformers. In *ICLR*, 2022. 4

[15] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017. 4

[16] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. 2020. 5

[17] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Zhi Zhang, Haibin Lin, Yue Sun, Tong He, Jonas Muller, R. Manmatha, Mu Li, and Alexander Smola. Resnest: Split-attention networks. *arXiv preprint arXiv:2004.08955*, 2020. 5

[18] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *CVPR*, 2018. 6