Supplementary material for "GAMA: Generative Adversarial Multi-Object Scene Attacks"

Abhishek Aich[*], **Calvin-Khang Ta**[*], **Akash Gupta, Chengyu Song,**
Srikanth V. Krishnamurthy, M. Salman Asif, Amit K. Roy-Chowdhury
University of California, Riverside, CA, USA

CONTENTS

List of Tables

List of Figures

[*]Equal contribution. Corresponding author: AA (aaich001@ucr.edu). AG is currently with Vimaan AI, USA.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

We present additional analysis of **GAMA** in the following sections to investigate its attack capabilities under various settings, including black-box embedding visualizations w.r.t. TAP [1], impact of different types of CLIP models, performance with ensemble of surrogate models. We also demonstrate **GAMA**'s transfer attack strength in comparison to prior methods under difficult black-box transfer attacks including in different multi-label distribution, object detection, and robustness of perturbations when victim uses defense mechanisms to minimize classifier performance deterioration. All experiments are done with perturbation budget $\ell_\infty \leq 10$.

1 Additional Analysis on GAMA

Figure 1: **Embedding visualization. GAMA** uses the CLIP extracted text and image embeddings to craft highly transferable adversarial examples. This can be seen in above embedding visualizations where **GAMA**'s perturbed images lie convincingly farther away from the clean images with better margins compared to TAP [1]. Left and right plots show perturbed image embeddings (both on random 1000 ImageNet images) when $\mathcal{G}_{\boldsymbol{\theta}}(\cdot)$ is trained with MS-COCO and Pascal-VOC respectively. Surrogate and victim models are given in parenthesis.

Black-box Setting Embedding Visualization. To demonstrate the phenomenon that **GAMA** learns to create potent perturbations compared to prior works, we perform Principal Components Analysis (PCA) of perturbed images extracted from **GAMA** and TAP [1] in Figure 1 when the $\mathcal{G}_{\boldsymbol{\theta}}(\cdot)$ is trained with MS-COCO and Pascal-VOC. We choose PCA visualization as it preserves the global differences of high dimensional data in low dimensional regimes [2, 3]. Clearly, in an unseen distribution (ImageNet [4]), features obtained from **GAMA**'s perturbed images significantly differ from those of clean images in comparison to TAP [1].

Impact of Different CLIP models. We analyze the impact of different open-source pre-trained CLIP models provided by Open-AI in Table 1 (surrogate model as Res152), both for the same domain and different domain transfer attacks. We observe that the CLIP frameworks with vision encoders with image transformers [5] (ViT-L/14, ViT-B/32, ViT-B/16) as their backbone perform better in our proposed setting than those with the vision encoders as convolutional networks (RN50, RN101). We attribute this to the effectual representation capability of transformers [5].

Table 1: Impact of CLIP model on **GAMA**

(a) Pascal-VOC → Pascal-VOC

	VGG16	VGG19	Res50	Res152	Den169	Den121
No Attack	82.51	83.18	80.52	83.12	83.74	83.07
RN50	8.83	15.25	64.37	67.24	70.53	69.13
RN101	21.74	9.45	60.56	68.53	67.01	66.17
ViT-L/14	43.35	49.89	45.08	**43.30**	54.23	51.53
ViT-B/32	10.58	15.18	67.07	70.34	69.14	68.02
ViT-B/16	**6.12**	**5.89**	**41.17**	45.57	**53.11**	**44.58**

(b) Pascal-VOC → ImageNet

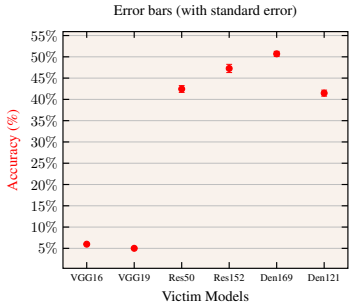
	VGG16	VGG19	Res50	Res152	Den121	Den169
No Attack	70.15	70.94	74.60	77.34	74.22	75.74
RN50	3.13	**2.06**	46.25	52.01	49.33	45.91
RN101	2.93	2.41	42.73	56.16	46.67	45.97
ViT-L/14	16.63	20.04	26.41	**23.18**	31.10	32.53
ViT-B/32	3.90	2.81	49.61	54.41	48.02	46.41
ViT-B/16	**3.07**	3.41	**22.32**	34.04	**24.51**	**30.35**

Random Runs with Error Bars. We report the mean, and standard error in Table 2 along with the error bar plot (with mean and standard error). We can observe that **GAMA** maintains its performance with random seed values over various runs. Here, $\mathcal{G}_{\boldsymbol{\theta}}(\cdot)$ was trained on Pascal-VOC with VGG19 as a surrogate.

Effect of Surrogate Ensemble. We analyze the results when all the surrogates (VGG19, Res152, and Den169) are employed together to train the perturbation generator $\mathcal{G}_{\boldsymbol{\theta}}(\cdot)$ using **GAMA**. As can be seen in Table 3 and Table 4 (ensemble denoted as All), we do not observe any significant advantage in results when using multiple surrogates. This same observation has been noted by TAP [1] as well. We

Table 2: Pascal-VOC →Pascal-VOC (s.e. = standard error)

	VGG16	VGG19	Res50	Res152	Den169	Den121
No Attack	82.51	83.18	80.52	83.12	83.74	83.07
Run 1	5.86	5.18	45.43	50.88	52.61	43.44
Run 2	6.00	4.99	42.30	47.54	49.82	40.82
Run 3	6.08	4.88	40.71	46.64	50.31	42.73
Run 4	5.95	5.15	42.28	45.52	51.46	40.92
Run 5	6.01	4.84	41.47	45.77	49.33	39.47
mean	5.98	5.01	42.44	47.27	50.70	41.47
s.e.	0.035	0.067	0.800	0.966	0.590	0.711



hypothesize that the mid-level features from multiple surrogates may not introduce complementary features to learn comparatively powerful perturbations than single classifier based surrogates.

Table 3: Ensemble comparison: VOC→VOC

$f(\cdot)$	VGG16	VGG19	Res50	Res152	Den169	Den121	Average
VGG19	6.11	5.89	41.17	45.57	53.11	44.58	32.74
Res152	33.42	39.42	32.39	20.46	49.76	49.54	37.49
Den169	44.25	52.89	48.83	53.25	45.00	50.96	49.19
All	16.46	21.67	51.97	58.52	54.51	58.20	43.55

Table 4: Ensemble comparison: COCO→COCO

$f(\cdot)$	VGG16	VGG19	Res50	Res152	Den169	Den121	Average
VGG19	3.59	3.75	27.13	30.43	24.60	21.77	18.54
Res152	24.52	27.73	30.62	23.04	31.30	27.31	27.42
Den169	10.40	13.47	19.30	23.46	8.65	10.29	14.26
All	10.08	10.75	23.83	35.23	29.57	30.45	23.32

2 Additional Results w.r.t. Baselines

Black-box Setting (Object Detection). We evaluate a black-box transfer attack with state-of-the-art MS-COCO object detectors (Faster RCNN with Res50 backbone (FRCN) [9], RetinaNet with Res50 backbone (RNet) [10], DEtection TRansformer (DETR) [11], and Deformable DETR (D^{2}ETR) [12]) in Table 5, provided by [13]. It can be observed that **GAMA** beats the baselines when $\mathcal{G}_{\boldsymbol{\theta}}(\cdot)$ is trained with MS-COCO in the majority of scenarios.

Table 5: COCO → COCO Object Detection

$f(\cdot)$	Method	FRCN	RNet	DETR	D^{2}ETR	Average
	No Attack	0.582	0.554	0.607	0.633	0.594
VGG19	GAP [6]	0.347	0.312	0.282	0.304	0.311
	CDA [7]	0.370	0.347	0.312	0.282	0.327
	TAP [1]	**0.130**	**0.120**	**0.099**	**0.104**	**0.113**
	BIA [8]	0.266	0.229	0.185	0.211	0.223
	GAMA	0.246	0.214	0.134	0.155	0.187
Res152	GAP [6]	0.187	0.145	0.097	0.108	0.134
	CDA [7]	0.322	0.301	0.237	0.274	0.283
	TAP [1]	0.167	0.151	0.087	0.123	0.132
	BIA [8]	**0.152**	0.144	0.101	0.121	0.129
	GAMA	0.154	**0.128**	**0.086**	**0.100**	**0.117**
Den169	GAP [6]	0.308	0.261	0.201	0.213	0.245
	CDA [7]	0.325	0.293	0.238	0.255	0.277
	TAP [1]	0.181	0.155	0.126	0.147	0.152
	BIA [8]	0.265	0.236	0.185	0.214	0.225
	GAMA	**0.078**	**0.064**	**0.037**	**0.047**	**0.056**

Black-box Setting (Multi-Label Classification). We perform a black-box transfer attack on different multi-label domain than that of $\mathcal{G}_{\boldsymbol{\theta}}(\cdot)$'s training set: Pascal-VOC → MS-COCO in Table 6 and MS-COCO → Pascal-VOC in Table 7. We outperform all baselines in the majority of cases, with an average absolute difference (w.r.t. closest method) of ∼5 percentage points (pp) for Pascal-VOC → MS-COCO, and ∼13.5pp for MS-COCO → Pascal-VOC.

Table 6: Pascal-VOC → MS-COCO

$f(\cdot)$	Method	VGG16	VGG19	Res50	Res152	Den169	Den121	Average
	No Attack	65.80	66.49	65.64	67.94	67.60	66.39	66.64
VGG19	GAP [6]	20.14	20.61	54.12	58.71	53.68	50.87	43.02
	CDA [7]	18.87	15.93	41.96	48.09	47.62	42.74	35.86
	TAP [1]	7.84	10.03	45.96	48.46	43.40	39.76	32.57
	BIA [8]	8.56	10.06	41.32	49.07	46.03	40.60	32.60
	GAMA	**2.92**	**3.83**	**23.37**	**28.26**	**22.07**	**17.69**	**16.35**
Res152	GAP [6]	32.90	33.63	46.70	54.18	53.71	51.40	45.41
	CDA [7]	27.28	32.25	41.32	44.59	48.33	45.10	39.81
	TAP [1]	31.68	37.33	36.09	36.85	47.77	45.59	39.22
	BIA [8]	26.99	29.83	33.86	35.35	45.87	41.70	35.59
	GAMA	**21.43**	**28.59**	**29.54**	**24.95**	**32.92**	**29.89**	**27.89**
Den169	GAP [6]	42.64	44.07	50.14	57.48	57.01	53.16	50.75
	CDA [7]	39.60	39.13	44.85	53.07	50.01	47.52	45.69
	TAP [1]	38.96	40.87	40.86	47.01	28.67	40.62	39.50
	BIA [8]	31.86	37.59	37.98	44.93	28.25	36.15	36.13
	GAMA	**26.43**	**32.64**	**32.30**	**38.88**	**22.06**	**30.62**	**30.49**

Table 7: MS-COCO → Pascal-VOC

$f(\cdot)$	Method	VGG16	VGG19	Res50	Res152	Den169	Den121	Average
	No Attack	82.51	83.18	80.52	83.12	83.74	83.07	82.69
VGG19	GAP [6]	17.07	15.01	61.14	67.17	69.30	63.04	48.78
	CDA [7]	15.23	13.19	58.81	63.80	67.43	62.23	46.78
	TAP [1]	15.35	12.74	**42.12**	**42.52**	**48.61**	**42.23**	**33.93**
	BIA [8]	8.10	8.82	52.85	55.82	63.05	56.58	40.87
	GAMA	**6.60**	**7.08**	44.16	49.20	57.49	52.52	36.17
Res152	GAP [6]	**27.09**	**28.45**	45.91	37.28	58.07	51.28	41.34
	CDA [7]	53.45	55.82	64.68	64.12	70.74	65.04	62.31
	TAP [1]	42.21	41.26	41.02	35.35	58.99	54.77	45.60
	BIA [8]	37.04	36.46	44.91	36.12	54.60	49.95	43.18
	GAMA	36.86	40.62	**38.23**	**23.52**	**48.56**	**48.03**	**39.30**
Den169	GAP [6]	48.37	46.35	58.04	60.73	52.89	57.83	54.03
	CDA [7]	58.51	58.20	67.61	69.73	67.26	65.88	64.53
	TAP [1]	46.83	47.88	46.98	57.68	44.95	43.99	48.05
	BIA [8]	42.14	49.84	54.47	62.05	48.75	50.91	51.34
	GAMA	**19.68**	**20.29**	**23.22**	**33.57**	**26.33**	**16.37**	**23.25**

Robustness Analysis. We launch misclassification attacks ($\mathcal{G}_{\boldsymbol{\theta}}(\cdot)$ trained on Pascal-VOC with the surrogate model as Den169) when the victim uses input processing based defense such as median blur with window size as 3×3 (Table 8(a)), and Neural Representation purifier (NRP) [14] (Table 8(b)) on three ImageNet models (VGG16, Res152, Den169). We can observe that the attack success of **GAMA** is better than prior methods even when the victim pre-processes the perturbed image. Further,

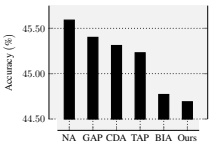
in Figure 8(c), we see that Projected Gradient Descent (PGD) [15] assisted Res50 is difficult to break with **GAMA** performing slightly better than other methods.

Method	VGG16	Res152	Den121	Average
No Attack	64.57	74.04	71.68	69.92
GAP [6]	47.91	65.64	61.69	58.41
CDA [7]	33.62	58.70	50.12	47.48
TAP [1]	23.92	**48.89**	43.66	38.82
BIA [8]	24.49	50.96	40.29	38.58
GAMA	**22.84**	52.10	**36.19**	**37.04**

(a) Median Blur

Method	VGG16	Res152	Den121	Average
No Attack	56.26	62.37	68.62	62.41
GAP [6]	33.09	50.50	53.80	45.79
CDA [7]	33.48	48.28	49.74	43.83
TAP [1]	27.45	42.98	42.66	37.69
BIA [8]	24.62	41.81	37.91	34.78
GAMA	**18.61**	**34.66**	**24.93**	**26.06**

(b) NRP



(c) PGD ($\epsilon = 8$)

Table 8: **Robustness Analysis against various defenses**. **GAMA** consistently shows better robustness in cases where victim uses attack defenses ($\mathcal{G}_\theta(\cdot)$ trained on Pascal-VOC). 'NA' in Figure 8(c) denotes 'No Attack'.

Evaluation of adversarial images on CLIP. We evaluated CLIP (as a "zero-shot prediction" model) on the perturbed images from Pascal-VOC and computed the top two associated labels in Figure 2 using CLIP's image-text aligning property. Specifically, we used the whole class list of Pascal-VOC and computed the top-2 associated labels both for clean and perturbed images. We can observe the perturbations change the labels associated with the clean image.

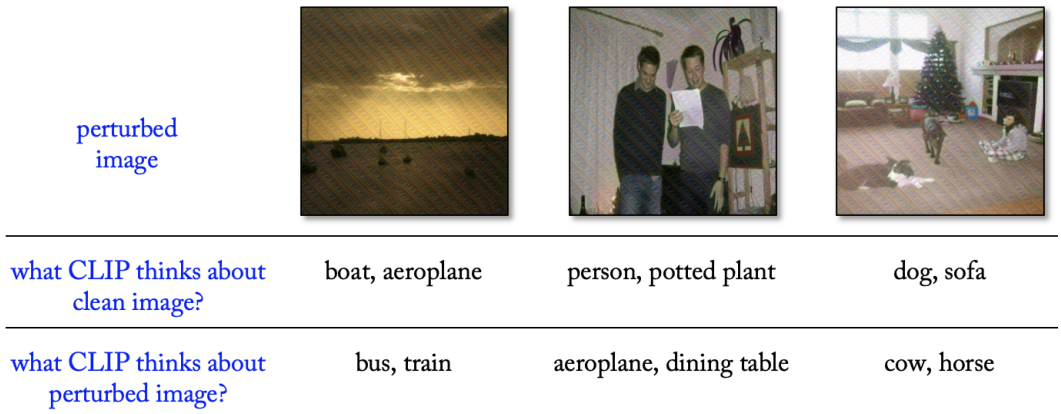


Figure 2: Evaluation of adversarial images on CLIP. Surrogate model is VGG19 trained on Pascal-VOC.

Mid-layer selection from surrogate model for training perturbation generator. Our mid-layer from surrogate model is chosen based on the embedding size of CLIP: *e.g.* if the embedding size of the CLIP encoder is 512, we select the layer from the surrogate model that outputs 512 dimension features. In comparison, prior state-of-the-art generative attack TAP [1] *manually* searches for the optimal layer from the surrogate model to train the perturbation generator (see *Limitations* in Section 4.6 in their paper [1]). In particular, finding the optimal mid-layer (that gives the best attack results) requires searching over each block of M layers (which is around an average of $M = 5$ layers [1]) for each surrogate model. Hence to find the best layer to train a perturbation generator for a particular model, the computation time cost for such an exhaustive search will be MN GPU hours where N is the total training time (in GPU hours) per layer. Moreover, our analysis shows this layer might not result in best attack results when the training data distribution varies and would require a manual search for all the different combinations of surrogate model and data distributions. Such a search is very time-consuming, impractical, and clearly not scalable. Finally, directly using TAP's suggested layer is not possible because the embedding size doesn't match that of CLIP, and would require us to introduce embedding modifications mechanisms (e.g. Principal Component Analysis (PCA), t-distributed stochastic neighbor embedding (t-SNE)) leading to an unreasonable increase in training time for every epoch. Note that if we do not consider the manual search of an optimal layer from the surrogate model to train the perturbation generator, then the proper baseline on ImageNet would be CDA [7]. As evident throughout our analysis, we convincingly outperform them on all settings.

3 Implementation Details

We use two multi-label datasets to stimulate the scenario of multi-object scenes: Pascal-VOC (training set: *trainval* from 'VOC2007' and 'VOC2012', testing set: 'VOC2007_test') and MS-COCO (training set: *train2017*, testing set: *val2017*). We follow prior works [7, 8] for the generator network for $\mathcal{G}_{\boldsymbol{\theta}}(\cdot)$. To stabilize the training, we replace all the ReLU [16] activation functions with Fused Leaky ReLU [17] activation function (negative slope = 0.2, scale = $\sqrt{2}$). We use a margin $\alpha = 1.0$ for the contrastive loss. All our training setup uses ViT-B/16 as the CLIP model. We use Adam optimizer [18] with a learning rate 0.0001, batch size 16, and exponential decay rates between 0.5 and 0.999. All images were resized to 224×224. Training time was observed to be $\sim$1 hr for Pascal-VOC dataset (10 epochs) and $\sim$10 hrs for MS-COCO dataset (5 epochs) on one NVIDIA GeForce RTX 3090 GPUs. PyTorch is employed [19] in all code implementations.

References

[1] Mathieu Salzmann et al. Learning transferable adversarial perturbations. *Advances in Neural Information Processing Systems*, 34, 2021.

[2] Abubakar Abid, Martin J Zhang, Vivek K Bagaria, and James Zou. Exploring patterns enriched in a dataset with contrastive principal component analysis. *Nature communications*, 9(1):1–7, 2018.

[3] Takanori Fujiwara, Oh-Hyun Kwon, and Kwan-Liu Ma. Supporting analysis of dimensionality reduction results with contrastive learning. *IEEE transactions on visualization and computer graphics*, 26(1):45–55, 2019.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE, 2009.

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[6] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative Adversarial Perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4422–4431. IEEE, 2018.

[7] Muzammal Naseer, Salman H Khan, Harris Khan, Fahad Shahbaz Khan, and Fatih Porikli. Cross-Domain Transferability of Adversarial Perturbations. *arXiv preprint arXiv:1905.11736*, 2019.

[8] Qilong Zhang, Xiaodan Li, YueFeng Chen, Jingkuan Song, Lianli Gao, Yuan He, and Hui Xue'. Beyond imagenet attack: Towards crafting adversarial examples for black-box domains. In *International Conference on Learning Representations*. International Conference on Learning Representations (ICLR), 2022.

[9] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2015.

[10] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2017.

[11] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[12] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=gZ9hCDWe6ke.

[13] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[14] Muzammal Naseer, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Fatih Porikli. A self-supervised approach for adversarial robustness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 262–271, 2020.

[15] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[16] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*. ICML, 2010.

[17] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410. IEEE, 2019.

[18] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, pages 8026–8037. NeurIPS, 2019.