# Fooling Image Classifiers

Henning Heyen

05/04/2023

**Abstract**

The goal of this project is to show how state-of-the-art image classifiers behave when the input images are out of distribution. We use common *Imagenet* pre-trained models to pass the images through. We then build an ensemble model and measure its prediction uncertainty. Then we actively try to fool one of the models using different perturbated images and investigate their saliency maps. Lastly, the findings are discussed within the context of Responsible AI.

## 1 Collecting a test set with epistemic uncertainty and passing it through image classifiers

The chosen class from the *ImageNet* dataset is **umbrella**. I picked one normal umbrella image as a baseline and five *out-of-distribution* images from the internet (see figure 1 in the appendix). The images were found using Google queries like "funny umbrella"[1]. Umbrella3 was generated by *DALL-E 2* using the description "abstract umbrella in Kandinsky style"[2]. All images would presumably be classified as umbrellas from a human perspective. From the Kears API[3] five pre-trained models were chosen: *vgg16, resnet50, inceptionv3, xception* and *mobilenetv2*. Running all images on each model resulted in some interesting predictions (see table 3).

All models confidently classified the baseline image (umbrella0) as an umbrella. Umbrella1 was almost certainly predicted as head cabbage by *resnet, inceptionv3* and *mobilenetv2*, which is unsurprising as the design resembles that of a cabbage. *Vgg16* and *xception* also failed to classify the image correctly. Interestingly, for the bird like image (umbrella2), the models predicted either a hatchet or a cleaver. While *mobilenetv2* correctly predicted umbrella for the *DALL-E* generated image (umbralla3), all other models predicted pinwheel. The painted image (umbrella4) was correctly classified as an umbrellaby *xception* but only with low certainty (0.129). As for umbrella5, *inceptionv3* and *xception* identified an umbrella, whereas the other models found a jersey or shield.

Overall the accuracy of less or equal to 0.5 is pretty low for all classifiers (see table 2), showing the sensitivity with respect to datashifts of pretrained models. The best classifier is *xception* as it identified an umbrella in three of six test images. That finding aligns with the fact that *xceptions* is the best-performing classifier in the top 5 accuracies *ImageNet* benchmark (accuracy=0.949, see table 1). The *vgg16* model performed the worst with only one umbrella occurrence across all top three predictions. As the *vgg16* consists of only 16 layers, one could assume that deeper networks are

[1] https://www.google.com/search?q=funny+umbrella
[2] https://labs.openai.com
[3] https://keras.io/api/applications/

more robust with respect to out-of-distribution instances. Given that only six images were tested, however, the generalizability of the results is questionable.

## 2 Building an uncertainty quantification ensemble

The method used to build the ensemble is **probabilistic averaging**, i.e. averaging over the classifiers' individual predictions. The class with the highest probability is then the final prediction. As an uncertainty measure, I use the **maximum probability** in the ensemble prediction vector, corresponding to how certain the ensemble is in its prediction. We find that apart from the baseline image, only one of the *out-of-distribution* images was correctly classified (umbrella5) with fairly low certainty (0.35). Please find a summary of the ensemble's predictions in table 4. Even though deep ensembles are generally known to be more robust with respect to datashifts [3], we find that the probabilistic averaging approach did not help to design a more robust classifier in this case (accuracy=0.33). Again, the very small test set limits the generalizability of that finding.

## 3 Fooling one of the models

I used two baseline images for this task, a zebra and a flamingo. A wide range of perturbations was applied to these images, including additive noise, rotations, blurring, colour editing, erasing and many more (see figure 2 and 3). Most manipulations were created using the *pixlr* image software[4]. The salt and pepper noise in the last images was implemented using the *cv2* python module[5].

The pretrained *xception* classifier from section 1 was picked for prediction as it seemed more robust on datashifts than the others. While the model correctly identified 12 of 15 zebras (accuracy=0.8), it classified only 9 of 15 flamingos (accuracy=0.6) with similar perturbations. Since the model struggled with the silhouette of the animals (zebra4, flamingo4, zebra12 and flamingo12) it seems like the model relies more on texture than shape. The better performance for zebras is presumably related to the unique stripe pattern. For the flamingo, even light additive noise results in completely different predictions, i.e. window screen (flamingo2) and starfish (flamingo14). The model probably uses the flamingo's characteristic pink to correctly identify the animal. When the colour is manipulated, for instance, flamingo7 and flamingo12, the model predicts American egret, which is a white bird with a similar neck. Removing the head or replacing it with a tiger's head did not suffice to fool the model for both animals (zebra10, flamingo10, zebra13 and flamingo13).

Lastly, even though not specifically asked, I also tried the other pretrained models from section 1 on the same images to compare the accuracies (see table 5). Interestingly, all models performed equally on the zebras (accuracy=0.8) while struggling with the flamingos. Against the initial expectation, that *xception* is the most robust, *resnet50* and *inceptionv3* performed better (accuracy=0.73). *Mobilenetv2*, however, only identified 7 of 15 flamingos which may be due to its limited number of parameters (see table 1.

[4]https://pixlr.com/
[5]https://pypi.org/project/opencv-python/

# 4 Analysing saliency maps for the images

The saliency map is a method used to determine important features (i.e. pixels) that influenced the model's decision by identifying a sensitivity mask for an image in relation to a specific prediction. The standard approach is based on calculating the gradient of a class prediction with respect to the input pixels, which indicates how changing each pixel would impact the prediction [7]. As this approach typically produces rather noisy maps, the SmoothGrad technique is commonly used [8]. It involves introducing gaussian noise to multiple copies of the image and then averaging the resulting gradients, leading to a smoother and clearer sensitivity mask.

In the notebook, the PAIR saliency library[6] was used to implement SmoothGrad saliency maps on the *xeceptinon* predictions for all images, i.e. umbrellas (figure 4), zebras (figure 5 and 6) and flamingos (figure 7 and 8). Overall, the saliency maps indicate that for all images the classifier successfully focuses on the object of interest rather than the background (see well-known Husky-Wolf case [5]). In general, the noisier the saliency map, the less accurate the prediction (e.g. umbrella1, umbrell4). Interestingly, the model failed to correctly predict umbrella3 even though the saliency map clearly indicates an umbrella. The reason for that is probably related to the fact that the model was trained on real images and not on paintings.

Looking at the zebras, the previous hypothesis that the animal's stripe pattern influences the prediction is further undermined by the saliency maps. For those predictions with high certainty (more than 90%), the stripe pattern is emphasized by the maps (e.g. zebra0, zebra2, zebra3, zebra7, zebra9, zebra10, zebra14). Strangely, the saliency map of zebra13 focuses on the replaced tiger head, but still, zebra was confidently predicted, which shows another limitation of saliency maps. Since the maps only depend on the model's input while ignoring the internal mechanisms, they may not be able to reveal the entire decision-making process of the model.

As far as the flamingos are concerned, we find that for the most accurate predictions, the model focuses mostly on the bird's body and its beak, whereas humans presumably find the neck more characteristic (e.g. flamingo5, flamingo9, flamingo10). Since the maps cannot capture the importance of colour, they cannot help to test the previous hypothesis that the model mostly relies on the flamingo's characteristic pink colour, which poses another limitation of saliency maps. In conclusion, saliency maps don't suffice to fully explain the model's decision and, as discussed in other papers, "reliance, solely on visual assessment can be misleading" [1].

# 5 Discussion within the context of Responsible AI

In this last section, I would like to point out some of the most important challenges related to AI robustness and security and then outline how the AI community aims to tackle them. Note that issues in bias, interpretability, privacy, and accountability are of equal importance, but for the sake of this project, I would like to focus on AI safety.

The project showed that even high-accuracy image classifiers struggle with out-of-distribution examples and are easy to fool. That becomes a serious issue when those models are deployed in high-risk applications. Autonomous vehicles could crash because a stop sign was misclassified through

[6]https://github.com/PAIR-code/saliency

adversarial noise, just like the flamingo earlier. Authentication systems and content filters can potentially be bypassed. On the release day of chatGPT, for instance, users were able to fool the integrated content filter to make the language model incentivize criminal behavior[7]. OpenAI emphasises their work on alignment, the concept of building models that are aligned with human values. To improve alignment, OpenAI incorporates reinforcement learning from human feedback [2], which is a hybrid intelligence technique for language models to learn a reward function based on human feedback using reinforcement learning. But still, the discussion about AI safety remains critical and pressing. Now that GPT-4 has been released, Elon Musk, Steve Wozniak, and other tech leaders urge in an open letter to "immediately pause for at least 6 months the training of AI systems more powerful than GPT-4"[8] to motivate the AI community to focus on implementing a set of shared safety protocols.

Security issues can arise during both the training and deployment of AI models. A common threat in the training phase is data poisoning, where attackers influence the training data to manipulate the results of a predictive model. A countermeasure for regression models was proposed by Jagielski et al. in 2021 using the TRIM method. It uses a trimmed loss function to remove points with large residuals, which can isolate most of the poisoning points and learn a robust regression model. Unfortunately, most adversarial threats happen in the deployment phase. Common attacks include model or data extraction [10], sponge attacks where inputs are designed to maximise energy consumption and latency [6], and inference manipulation, for example, by maximizing the model's prediction error [9]. A traditional approach to tackle adversarial attacks on image classifiers is defensive distillation, a technique that uses the knowledge extracted from a deep neural network to improve its own resilience to adversarial samples [4]. More generally, IBM works on open-sourced robustness toolkits that are model and data-agnostic. For example, the Adversarial Robustness Toolbox (ART)[9] is a Python library for machine learning security that enables developers to defend ML applications against evasion, poisoning, extraction, and inference attacks.

In conclusion, it is crucial to acknowledge that when AI surpasses human performance, accuracy cannot be the sole metric to evaluate AI models anymore, especially when deployed in high-risk applications. Researchers must focus on developing safe, interpretable, robust and aligned AI models while working with policymakers to accelerate AI governance systems and regulations.

[7] https://www.lesswrong.com/posts/RYcoJdvmoBbi5Nax7/jailbreaking-chatgpt-on-release-day
[8] https://futureoflife.org/open-letter/pause-giant-ai-experiments/
[9] https://github.com/Trusted-AI/adversarial-robustness-toolbox

# Appendix

(a) umbrella0

(b) umbrella1

(c) umbrella2

(d) umbrella3

(e) umbrella4

(f) umbrella5

Figure 1: Images from section 1

| model | top 1 accuracy | top 5 accuracy | parameters | depth | special characteristics |
|---|---|---|---|---|---|
| vgg16 | 0.713 | 0.901 | 138.4M | 16 | simple and uniform architecture |
| resnet50 | 0.749 | 0.921 | 25.6M | 50 | residual blocks, state of the art |
| inception_v3 | 0.779 | 0.938 | 23.9M | 189 | multiple parallel convolutional operations |
| xception | 0.79 | 0.949 | 22.9M | 81 | depthwise separable convolutions |
| mobilenetv2 | 0.713 | 0.901 | 3.5M | 105 | efficient on mobile devices, fewer parameters |

Table 1: Overview on models

| model | accuracy |
|---|---|
| vgg16 | 0.16 |
| resnet50 | 0.16 |
| inception_v3 | 0.33 |
| xception | 0.5 |
| mobilenetv2 | 0.33 |
| ensemble | 0.33 |

Table 2: Accuracies on all 6 test images for each model

| model | image | Top 3 predictions |
|---|---|---|
| vgg16 | umbrella0 | umbrella 0.752 parachute 0.244 flagpole 0.003 |
| resnet50 | umbrella0 | umbrella 0.998 parachute 0.002 flagpole 0.0 |
| inception_v3 | umbrella0 | umbrella 0.992 parachute 0.0 mountain_tent 0.0 |
| xception | umbrella0 | umbrella 0.992 mountain_tent 0.0 parachute 0.0 |
| mobilenetv2 | umbrella0 | umbrella 0.87 parachute 0.104 flagpole 0.009 |
| vgg16 | umbrella1 | shopping_basket 0.38 head_cabbage 0.15 pot 0.106 |
| resnet50 | umbrella1 | head_cabbage 0.986 cucumber 0.004 cauliflower 0.002 |
| inception_v3 | umbrella1 | head_cabbage 0.745 sleeping_bag 0.009 cauliflower 0.006 |
| xception | umbrella1 | sleeping_bag 0.086 head_cabbage 0.081 plastic_bag 0.059 |
| mobilenetv2 | umbrella1 | head_cabbage 0.928 sleeping_bag 0.011 cauliflower 0.006 |
| vgg16 | umbrella2 | hatchet 0.493 cleaver 0.283 paintbrush 0.029 |
| resnet50 | umbrella2 | cleaver 0.544 hatchet 0.278 paintbrush 0.049 |
| inception_v3 | umbrella2 | cleaver 0.432 hatchet 0.088 stretcher 0.045 |
| xception | umbrella2 | hatchet 0.136 cleaver 0.1 ocarina 0.027 |
| mobilenetv2 | umbrella2 | cleaver 0.735 hatchet 0.105 ocarina 0.033 |
| vgg16 | umbrella3 | pinwheel 0.96 umbrella 0.02 wall_clock 0.008 |
| resnet50 | umbrella3 | pinwheel 0.984 umbrella 0.008 wall_clock 0.004 |
| inception_v3 | umbrella3 | pinwheel 0.52 hook 0.052 umbrella 0.038 |
| xception | umbrella3 | pinwheel 0.403 umbrella 0.041 hook 0.024 |
| mobilenetv2 | umbrella3 | umbrella 0.307 pinwheel 0.245 mortarboard 0.155 |
| vgg16 | umbrella4 | spoonbill 0.453 American_lobster 0.053 conch 0.049 |
| resnet50 | umbrella4 | shovel 0.435 barrow 0.19 wreck 0.067 |
| inception_v3 | umbrella4 | barrow 0.113 wreck 0.046 canoe 0.045 |
| xception | umbrella4 | umbrella 0.129 conch 0.05 coffeepot 0.023 |
| mobilenetv2 | umbrella4 | stingray 0.195 umbrella 0.144 canoe 0.066 |
| vgg16 | umbrella5 | jersey 0.773 sweatshirt 0.052 plastic_bag 0.044 |
| resnet50 | umbrella5 | shield 0.311 packet 0.106 maillot 0.066 |
| inception_v3 | umbrella5 | umbrella 0.756 jersey 0.016 maillot 0.016 |
| xception | umbrella5 | umbrella 0.935 carton 0.006 mountain_tent 0.006 |
| mobilenetv2 | umbrella5 | jersey 0.585 packet 0.229 maillot 0.035 |

Table 3: Top 3 predictions and uncertainties using different models and images

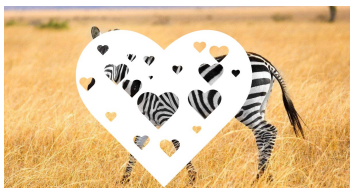
| image | prediction | max probability |
|---|---|---|
| umbrella0 | umbrella | 0.922 |
| umbrella1 | head_cabbage | 0.578 |
| umbrella2 | cleaver | 0.419 |
| umbrella3 | pinwheel | 0.623 |
| umbrella4 | spoonbill | 0.095 |
| umbrella5 | umbrella | 0.353 |

Table 4: Average probability ensemble predictions with maximum probability uncertainty

| model | accuracy zebra | accuracy flamingo |
|---|---|---|
| vgg16 | 0.8 | 0.6 |
| resnet50 | 0.8 | 0.733 |
| inception_v3 | 0.8 | 0.733 |
| xception | 0.8 | 0.6 |
| mobilenetv2 | 0.8 | 0.467 |

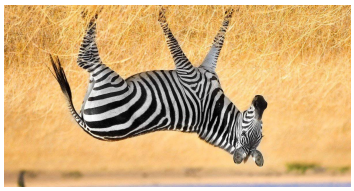
Table 5: Accuracy on perturbation images for each model

(a) zebra0
$pred = zebra$
$prob = 0.960$

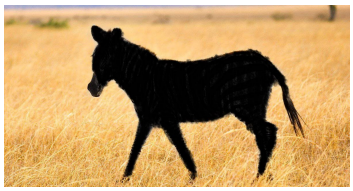


(b) zebra1
$pred = zebra$
$prob = 0.741$

(c) zebra2
$pred = zebra$
$prob = 0.928$

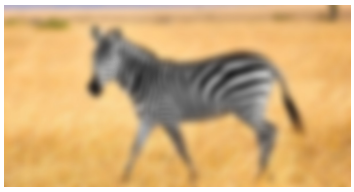


(d) zebra3
$pred = zebra$
$prob = 0.937$



(e) zebra4
$pred = groenendael$
$prob = 0.060$

(f) zebra5
$pred = zebra$
$prob = 0.959$



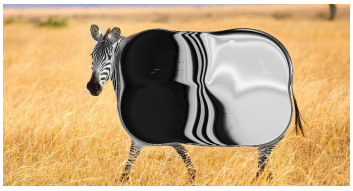
(g) zebra6
$pred = zebra$
$prob = 0.748$

(h) zebra7
$pred = zebra$
$prob = 0.937$

(i) zebra8
$pred = zebra$
$prob = 0.315$

(j) zebra9
$pred = zebra$
$prob = 0.939$

(k) zebra10
$pred = zebra$
$prob = 0.977$



(l) zebra11
$pred = backpack$
$prob = 0.309$

(m) zebra12
$pred = gazelle$
$prob = 0.473$

(n) zebra13
$pred = zebra$
$prob = 0.937$

(o) zebra14
$pred = zebra$
$prob = 0.985$

Figure 2: Zebra perturbations

(a) flamingo0
$pred = flamingo$
$prob = 0.953$

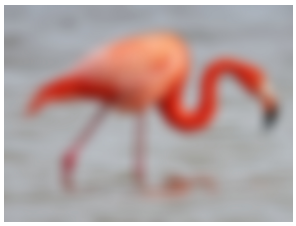
(b) flamingo1
$pred = paddle$
$prob = 0.052$

(c) flamingo2
$pred = window\ screen$
$prob = 0.039$

(d) flamingo3
$pred = flamingo$
$prob = 0.944$

(e) flamingo4
$pred = littleblue\ heron$
$prob = 0.190$

(f) flamingo5
$pred = flamingo$
$prob = 0.943$



(g) flamingo6
$pred = flamingo$
$prob = 0.980$

(h) flamingo7
$pred = American\ egret$
$prob = 0.270$

(i) flamingo8
$pred = flamingo$
$prob = 0.951$

(j) flamingo9
$pred = flamingo$
$prob = 0.922$

(k) flamingo10
$pred = flamingo$
$prob = 0.902$

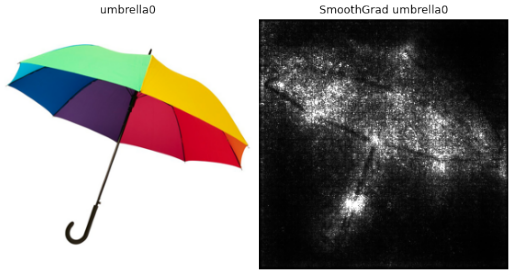
(l) flamingo11
$pred = flamingo$
$prob = 0.445$

(m) flamingo12
$pred = American\ egret$
$prob = 0.760$

(n) flamingo13
$pred = flamingo$
$prob = 0.273$

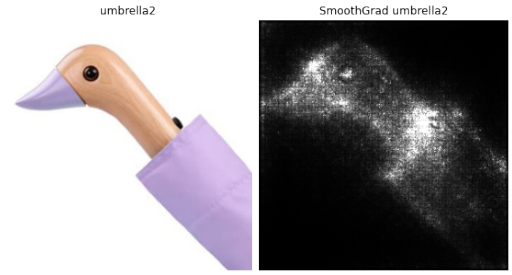
(o) flamingo14
$pred = starfish$
$prob = 0.798$

Figure 3: Flamingo perturbations

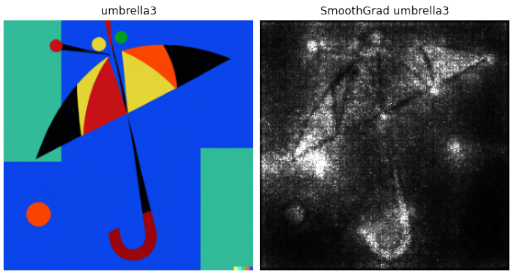



(a) $pred = umbrella$, $prob = 0.992$

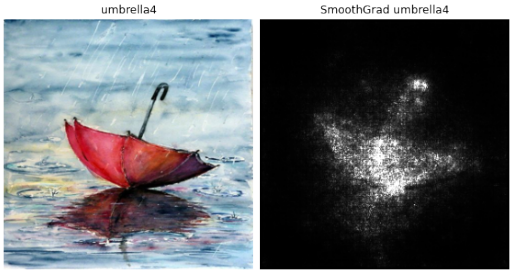

(b) $pred = sleepingbag$, $prob = 0.086$



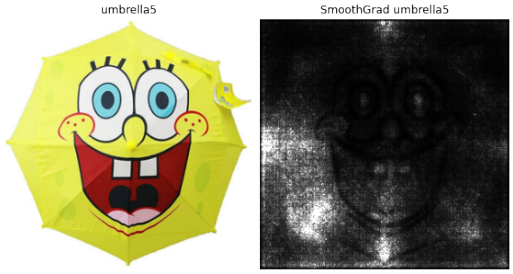

(c) $pred = hatchet$, $prob = 0.136$




(d) $pred = pinwheel$, $prob = 0.403$

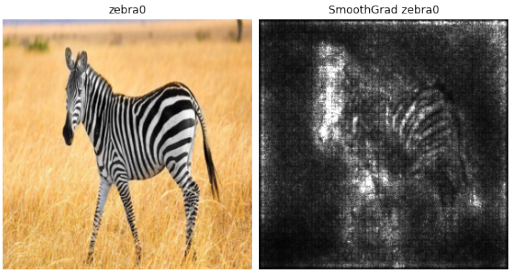



(e) $pred = umbrella$, $prob = 0.129$

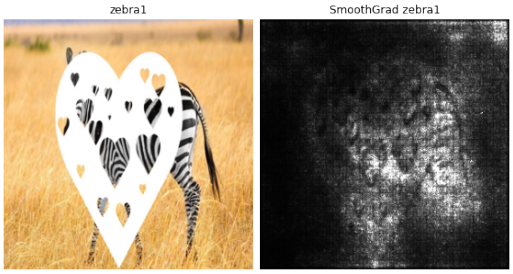



(f) $pred = umbrella$, $prob = 0.935$

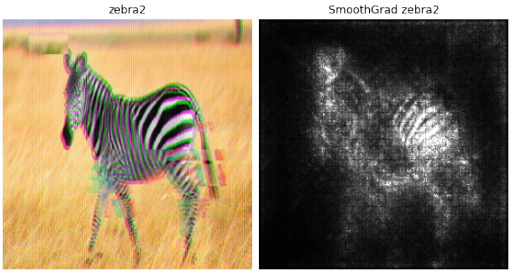
Figure 4: Umbrella saliency maps using the *xception* classifier



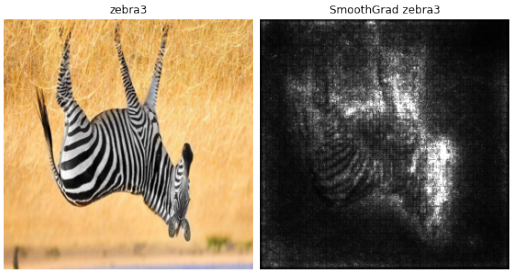

(a) $pred = zebra$, $prob = 0.960$



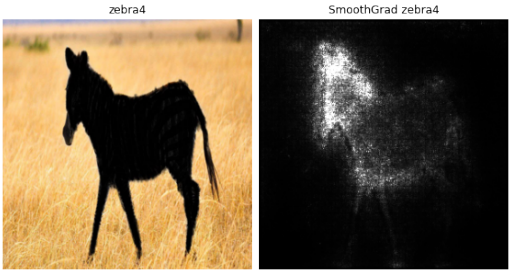

(b) $pred = zebra$, $prob = 0.741$



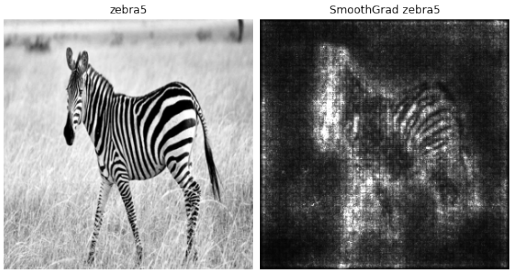

(c) $pred = zebra$, $prob = 0.928$



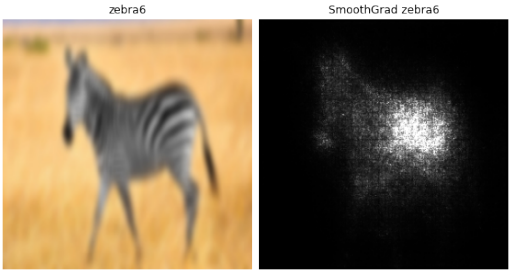

(d) $pred = zebra$, $prob = 0.937$



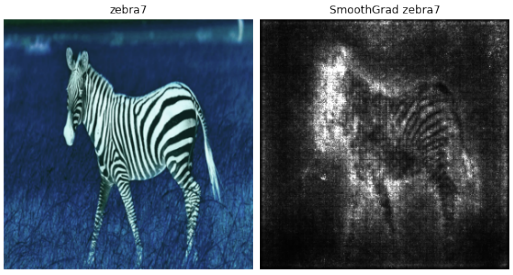

(e) $pred = groenendael$, $prob = 0.060$




(f) $pred = zebra$, $prob = 0.959$

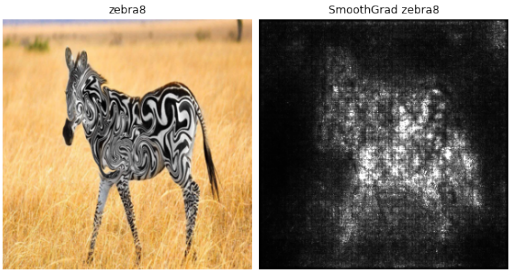



(g) $pred = zebra$, $prob = 0.748$

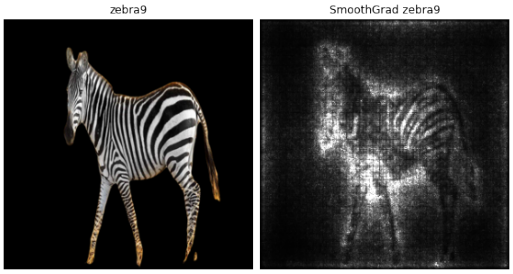



(h) $pred = zebra$, $prob = 0.937$

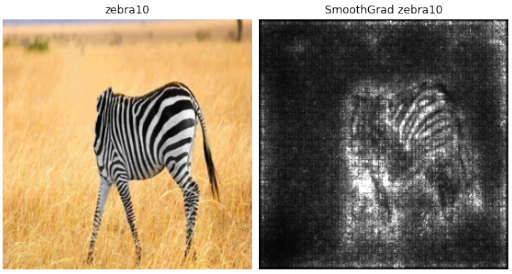
Figure 5: zebra0 - zebra7 saliency maps using the *xception* classifier



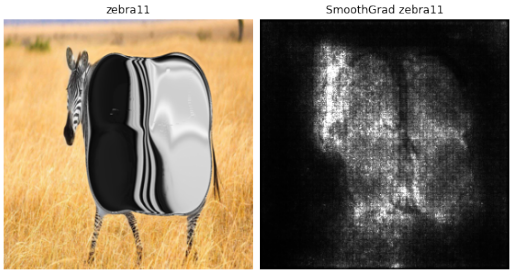

(a) $pred = zebra$, $prob = 0.315$



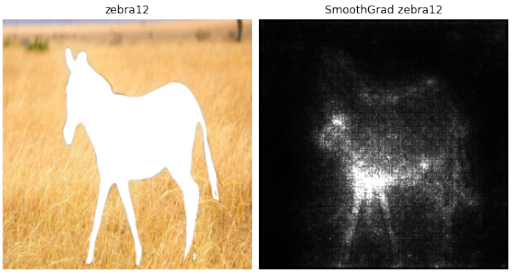

(b) $pred = zebra$, $prob = 0.939$



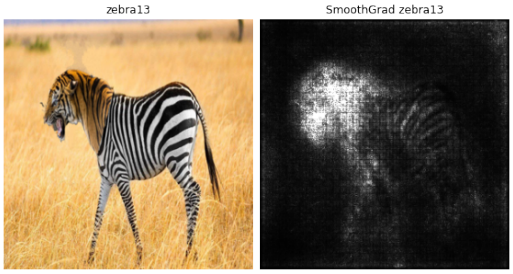

(c) $pred = zebra$, $prob = 0.977$



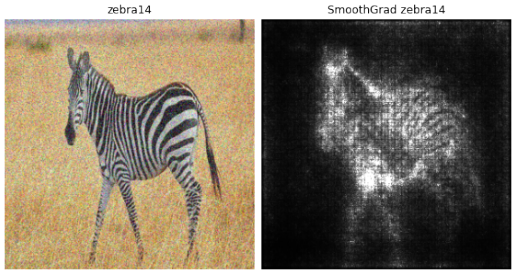

(d) $pred = backpack$, $prob = 0.309$




(e) $pred = gazelle$, $prob = 0.473$

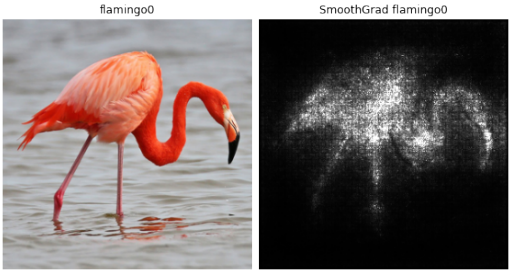



(f) $pred = zebra$, $prob = 0.937$



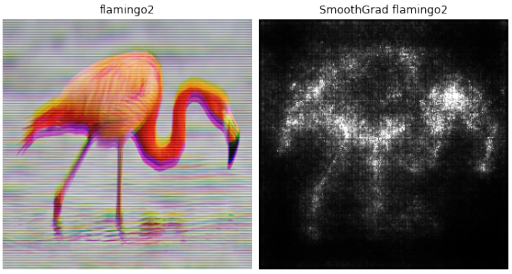

(g) $pred = zebra$, $prob = 0.985$

Figure 6: zebra8 - zebra14 saliency maps using the *xception* classifier

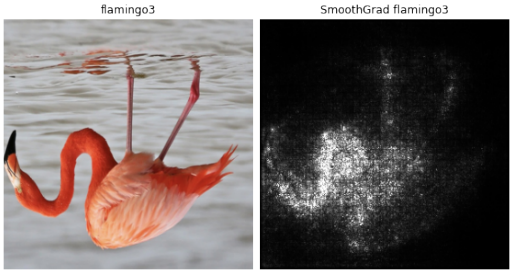



(a) $pred = flamingo$, $prob = 0.953$

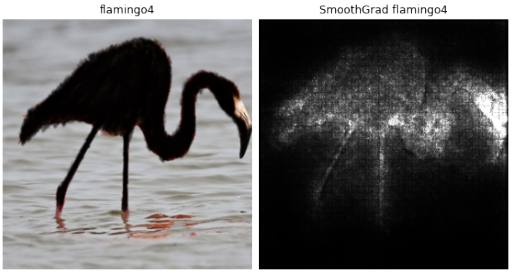

(b) $pred = paddle$, $prob = 0.052$



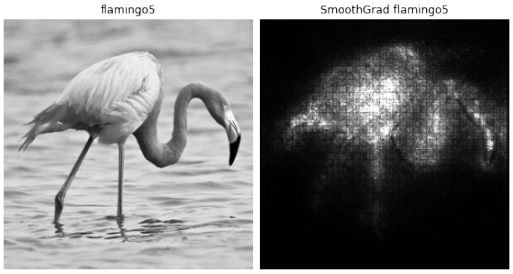

(c) $pred = window\ screen$, $prob = 0.039$



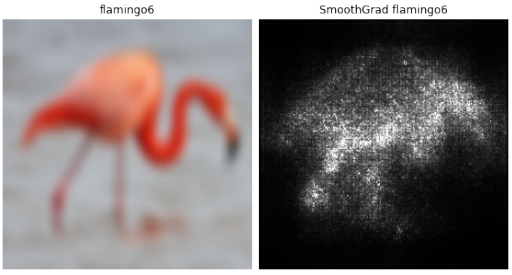

(d) $pred = flamingo$, $prob = 0.944$



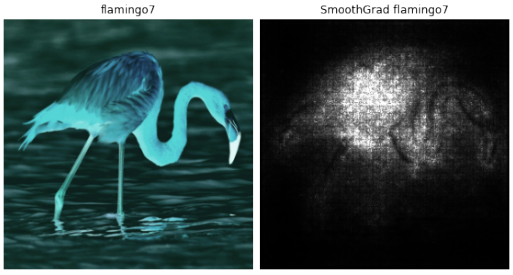

(e) $pred = littleblue\ heron$, $prob = 0.190$




(f) $pred = flamingo$, $prob = 0.943$

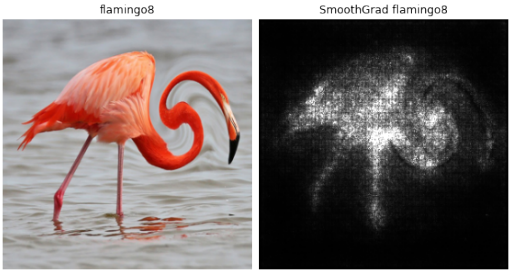



(g) $pred = flamingo$, $prob = 0.980$



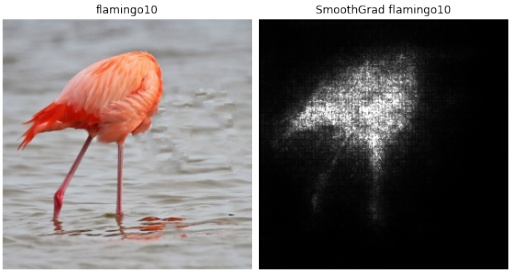

(h) $pred = American\ egret$, $prob = 0.270$

Figure 7: flamingo0 - flamingo7 saliency maps using the *xception* classifier

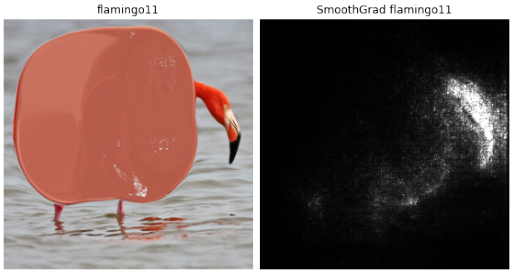



(a) $pred = flamingo$, $prob = 0.951$

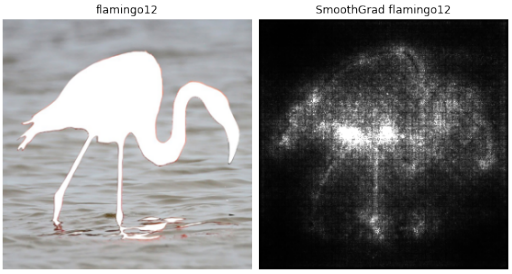

(b) $pred = flamingo$, $prob = 0.922$



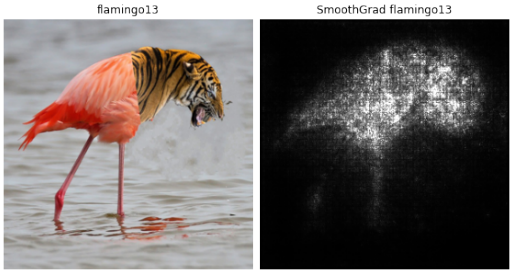

(c) $pred = flamingo$, $prob = 0.902$



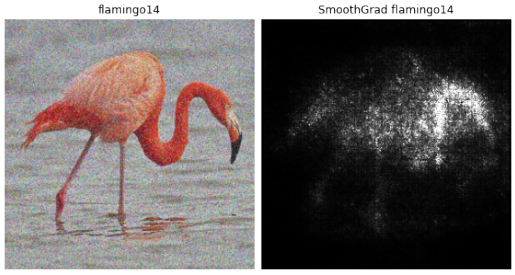

(d) $pred = flamingo$, $prob = 0.445$




(e) $pred = American\ egret$, $prob = 0.760$




(f) $pred = flamingo$, $prob = 0.273$




(g) $pred = starfish$, $prob = 0.798$

Figure 8: flamingo8 - flamingo14 saliency maps using the *xception* classifier

# References


[1] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *Advances in neural information processing systems*, 31, 2018.

[2] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[3] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles, 2017.

[4] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks, 2016.

[5] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[6] Ilia Shumailov, Yiren Zhao, Daniel Bates, Nicolas Papernot, Robert Mullins, and Ross Anderson. Sponge examples: Energy-latency attacks on neural networks, 2021.

[7] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[8] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

[9] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2014.

[10] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction apis. In *USENIX security symposium*, volume 16, pages 601–618, 2016.