UCL

Accountable, Transparent, Responsible AI - Coursework 2

1

The report aims to evaluate the performance of various classification machine learning models on out-of-distribution images generated with the help of **DALLE 2** platform by OpenAI. The models were trained on a **IMAGENET** dataset and tested on images that were outside of that dataset (generated by DALLE with some imperfections). The purpose of this experiment was to determine how well the models can generalise and make accurate predictions on previously unseen data.

1. Collect a test set with epistemic uncertainty and pass it through an image classifier

The task involves selecting five different images that belong to a defined-class in the IMAGENET dataset, but have some peculiarity and are assumed not to be contained in the IMAGENET dataset. We then use these out-of-distribution images with various classification models to check the effect of the peculiar element added to the normal images when they were generated. The class chosen is 'turtle', but there are different species of turtles in the IMAGENET dataset, so a range of classes has been taken as the 'reference' class which is [33,34,35,36,37] in IMAGENET dataset. These classes contain almost similar looking species of turtles which we are using as reference throughout our coursework. The images have been twisted by changing the context of images, like changing the surroundings of the main object to an unexpected one, changing the objects appearance by adding accessories on the object, keeping the object in different postures etc. The images are attached at the bottom of the report. Image reference: 1. The images have been pre-processed before feeding them to the neural networks so that image is compatible with the images that the model has trained on. The steps involved are,

- Resizing the image to the default input size of models and each model requires a different input size of images.
- Converting the image to a numpy array to facilitate the images to be used as inputs by the machine learning model.
- Converting the image color channels to compatible ones.
- Subtracting the mean pixel value from each pixel in the image.

The first and second pre-processing steps have been implemented manually, whereas the last two pre-processing steps have been implemented using tensorflow preprocess_input package for each model.

The five models that have been chosen for our experiment are:

- `VGG16`
- `ResNet50`
- `InceptionV3`
- `MobileNetV2`
- `EfficientNetB5`

VGG16 is a popular classification algorithm that has a deep neural network architecture with 16 layers and 138.4 million parameters. A pre-trained VGG16 model has been imported through *tensorflow applications* module and our selected images have been passed to the model. Out of the 5 images, the model is able to correctly classify only 1 image (Image1). Image Reference: 2.
ResNet50 is a very deep neural network with 107 layers and 25.6 million parameters. A pre-trained Resnet50 model has been imported through *tensorflow applications* module and our selected images have been passed to the model. Out of the 5 images, the model is able to correctly classify only 2 images (Image1 and Image5). Image Reference: 3
InceptionV3 is an extremely deep neural network with 189 layers and 23.9 million parameters. A pre-trained InceptionV3 model has been imported through **tensorflow applications** module and our selected images have been passed to the model. Out of the 5 images, the model is able to correctly classify only 1 image (Image1). Image Reference: 5
MobileNetV2 105 layers and 3.5 million parameters and comparatively shallow as comapred to previously mentioned neural networks. The model could identify only 1 image out of the 5 images passed to it. Image Reference: 4
EfficientNetB5 is the deepest model of our model collection and is 312 layers deep with 30.6 million parameters. Interestingly the model could identified the same 2 images as REstNet50 (Image1 and Image5). Image Reference: 6

The model specific information has been collected from official keras documentation. The accuracies of each model on the five images is given below in Table 1,

One interesting observation is that the although the models failed to recognise the object of interest correctly, but the classes predicted by models were in-line with the context of the out-of-distribution images. For example, Image2 has been identified as comic_book which is not entirely incorrect, but does not identify the main object of the image.

Image Classification	
Model	**Accuracy**
VGG16	20%
ResNet50	40%
InceptionV3	20%
MobileNetV2	20%
EfficientNetB5	40%

Table 1 Performance of models on 5 out-of-distribution images

Ensemble Classification		
Image	**Class**	**Probability**
Image1	terrapin	0.6
Image2	comic_book	1
Image3	comic_book	0.6
Image4	laptop	0.4
Image5	pineapple	0.2

Table 2 Maximum Class Probability for each of the image

2. Build an uncertainty quantification ensemble

The five deep learning models have been collectively used as an ensemble to classify the images and the method of selecting the final class for each image is done through **majority voting**. **Ensemble** combines multiple models to make predictions. When a new image is fed to the ensemble, each classifier makes a prediction. The final prediction is be chosen through various methods, like majority voting, averaging, selecting the prediction with maximum probability etc. For our task, we have selected majority voting as the method to make the final prediction. It is a technique that involves each member of the ensemble making a prediction and the prediction that is most common out of all the predictions is taken as the final class. Probabilistic averaging can be used as another way to classify the images by the ensemble. The final classes for each out-of-distribution image selected through majority voting is given below in appendix. Image Reference : 7

If we compare the performance of individual models with the ensemble, we see that the ensemble gives an overall accuracy of 40% which is same as the accuracy given by the best performing models individually (ResNet50, EfficientNetB5 in Table 1). Image1 and Image5 have been classified correctly. But the other three images have been classified in classes that are not entirely incorrect. If we look at Image2, Image3, Image4 in the appendix, we see that these turtle images have been classified as *comic_book*, *comic_book*, *laptop* respectively. These predicted classes are not wrong, but the images have been considered from a different context in Image2, Image3 and labeled them as 'comic_book' and Image4 identifies a side object in the image as the main object and labels the image as 'laptop'.

Disagreement between members of the ensemble has been taken as the measure to indicate the uncertainty of the ensemble. The probability of the class with highest probability has been calculated from each image and the plot has been attached in the appendix. Image Reference : 13. The probability plots show that for Image2, all the models unanimously vote for 'comic_book' as the class and for Image5, all the models are divided and 5 models have given 5 different classes. Thus, in ensemble the predictions can range from very clear voting to very distributed voting.

The below table showing the maximum probabilities for each image's class is given below in Table 2,

Based on the above uncertainty values, the images have been ranked, and can be seen in appendix. Image Reference : 8

3. Fool one of the models

The model selected for fooling is *RestNet50*. Two new out-of-distributions images (Image6, Image7) have been used for the task as given in appendix. Image Reference : 9. The images were fed to the trained *ResNet50* model and were classified correctly. But when different perturbations were added to these images, the model started to fail recognising the images and classified them incorrectly. Table 3 shows how different alterations impact the classification process.

Classification of Perturbed Images		
Image	**Alteration**	**Classification**
Image1	Original	Correct
Image1	contrast	Correct
Image1	blur	Correct
Image1	color inversion	Incorrect
Image2	Original	Correct
Image2	contrast	Incorrect
Image2	blur	Incorrect
Image2	color inversion	Incorrect

Table 3 Impact of perturbations added to image

As mentioned at the start of the report, the images that lie in the class range [33,34,35,36,37] i.e. different species of turtles have been termed as 'correct' and any other class have been termed as 'incorrect'.

Machine Learning models use features, such as colour, shape, texture, or edges, to make classification decisions. So, the alterations added I decided to add were:

- contrast
- blur
- color inversion

Contrast refers to the difference in brightness between the darkest and lightest areas of an image. The contrast of our sample images was increased by 5 times to manipulate the shape and texture in order to fool the model. The *RestNet50* model was able to identify Image6 correctly, but failed to recognise Image7. The reason could be that the individual features of our object(turtle) could not be interpreted by the model. The incorrect image was

classified as a pool_table which in fact is an object in the background. Image Reference : 10 **Blur** reduces the level of detail or sharpness in an image by averaging out the nearby pixels of the image, resulting in a soft and diffused image. A Gaussian Blur with a radius of 10 was added to the original image. The *RestNet50* model was able to classify Image6 correctly, but Image7 was not recognised by the model correctly. The incorrect image was again classified as a pool_table similar to previous perturbation. Image Reference : 11 **Color Inversion** changes the color in such a way that the brightest colors become the darkest and vice versa. This significantly alters the appearance of objects in the image. Color Inversion was added to the sample images and neither of the images were classified correctly. One of the image was classied as a bubble and the other one as snowmobile which are absolutely unrelated to the original images. Image Reference : 12
The change of shape, features and colour of the images due to perturbations have considerably impacted the classification. Thus, we can say that these aspects of image play a major role in classifying images through machine learning.

4. Analyse saliency maps for your images

There are a total of seven out-of-distribution images that have been used in the cousework and saliency maps for these images have been generated using **VGG16** model. *Saliency Map* is a heatmap that highlights the pixels in an image that contribute the most to classification by the neural network. For our image, a piece of code has been taken from Lab4 of the module (COMP0195) which has further been taken from PAIRWISE github. The function computes the saliency map for an input image by converting images to tensors and using the model's gradients. The maps have been generated in a greyscale format, i.e pixel color ranging only between black and white.

Saliency maps for all seven images have been generated. Image Reference : 14. One important observation is that if the image contains multiple things/objects in the background, the saliency maps create multiple structures which are hard to interpret. For example, except Image1, Image5 and Image6, all other images have quite a few objects in the background. The saliency maps for these images is almost impossible to read.

Also, when the colour of the background is close to the color of the main object (turtle in our case), the saliency maps feel blurry, i.e. the main objects structure seems to get mixed with the background and it is difficult to interpret the shape of the main object.

Saliency maps for our images are somewhat readable for some of the images(Image1, Image3, Image5 and Image6) where the overall structure of the main object(turtle) have a clearer and more readable structures. But the structure are ambiguous and could be interpreted in many ways. For example, Image3 has a turtle standing on a ship with its arms open. The saliency map of this image shows a structure that could easily be read by the model as a chameleon, or a bird or a bat. So, saliency maps might help shortlist the possible classes of the image, but very hard to get clear information, if the image is tricky (i.e. has many objects in the background, colours of background are similar shades etc).

Saliency maps are visual representations to aid image recognition by highlighting regions of an image by separating prominent structures, using specific colours for highlighting objects. But saliency maps have some challenges. Some of them have been discussed above. Summarising the challenges,
1. With multiple objects in the background, the maps fail to highlight the main object of interest.
2. If the colours of the object and background is not contrasting, i.e. object's colour is similar to the background, the saliency maps are not very helpful.
3. There is no standard way to evaluate the quality of saliency maps, so it is hard to compare saliency maps from different models.

5. Discuss within the context of Responsible AI

There are several challenges that upcoming technologies like AI in context of image classification faces. Some of them are,

- **Adversarial attacks:** Machine Learning models depend majorly on the training dataset, which if manipulated with ill-intention can cause the model to break and mis-classify images (sponge-based attacks). For example, amongst our 7 out-of-distribution images, half of them were identified correctly whereas the ones with manipulations caused the model to fail. If more of these images are added to the training dataset, the model's learning can go haywire. For high-risk applications like security, autonomous vehicles, and medical imaging, this can cause serious hazards.
  Also, As discussed in week 7 class notes, there are prediction APIs that can be used by adversaries by tricking models through feeding queries to them and analysing their response, to build approximations of models. This way, learning models can be stolen and cause commercial loss, security breach and other unethical uses of the model
- **Interpretability**: AI applications and their high-end applications deal with a lot of uncertainty. For example, we tried classifying few out-of-distribution images to pre-trained deep learning models. But most of the models failed to classify those images correctly but we do not have any information on how sure the models were when they made these predictions. These neural networks are black-boxes and difficult to interpret. Sometimes the wrong predictions are given, but with a low probability. But it is generally difficult to interpret these probabilities in neural networks. One of the solution to these type of uncertainty is to use Bayesian modeling, or using Interpretation frameworks.
- **Quantifying Uncertainty**: When the model was fed with perturbed images, the model classified them incorrectly. But there is no way the model knew that it was making wrong predictions. For example, while adding contrast to the image, the amount of contrast was varied over a range of values. But it gave right answers to some of the values, but as the value got higher, it classified the image incorrectly. But there is no way the model gave any information on how sure it was about each prediction as the perturbations increased. One solution to be more certain about the predictions is to use ensembles, where not one, but multiple models make the final decision.
- **Regulations on AI**: As explained in week8 guest lecture, the regulations around AI are not very explicit and detailed and adversaries can use the loopholes are work around these regulations without being labelled as unethical that can cause dangers and vulnerabilities.

- **Human-centered AI**: For high-risk applications, AI applications can be designed in a way that a human is always aware and in-control of the decisions the AI system is taking. This way, applications (like our image classifiers) can be secured from adversarial attacks or mis-classifications for high-risk applications.

APPENDIX

Figure 1 5 Out-of-Distribution images generated from DALLE 2

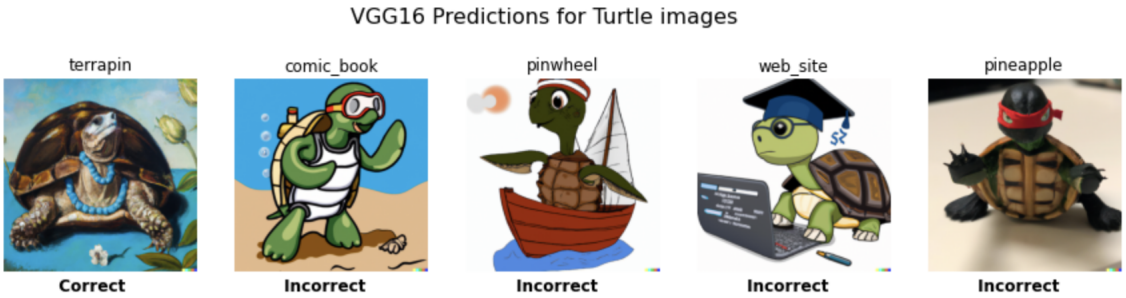

Figure 2 Images Classified by pre-trained VGG16 model

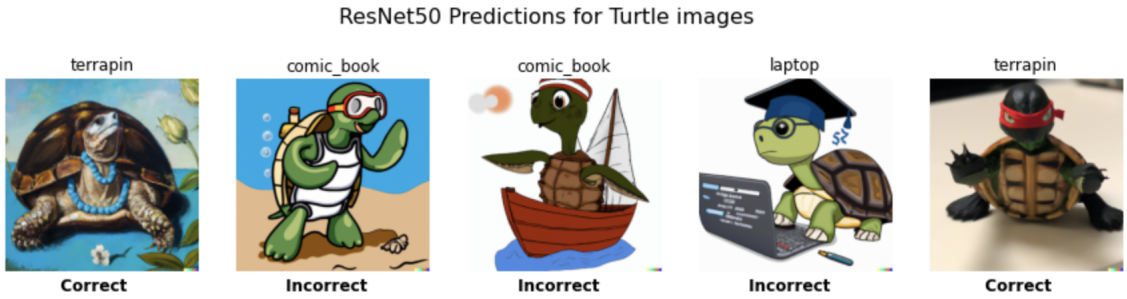

Figure 3 Images Classified by pre-trained ResNet50 model

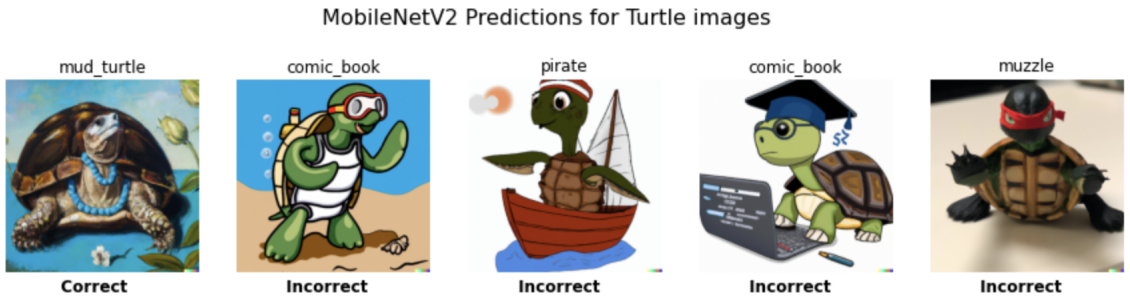

Figure 4 Images Classified by pre-trained MobileNetV2 model

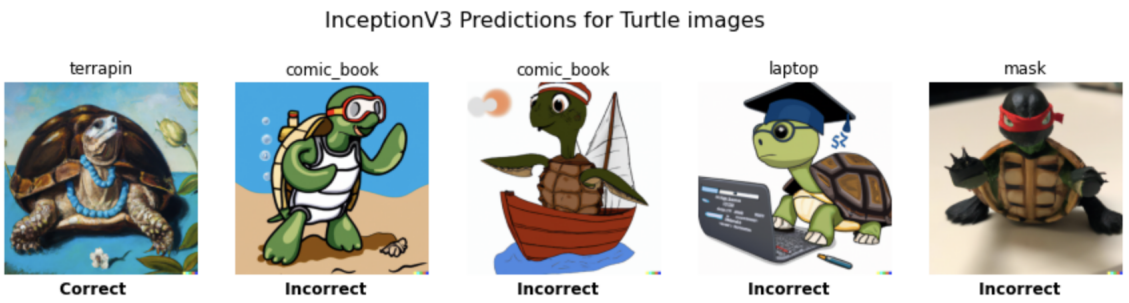

Figure 5 Images Classified by pre-trained InceptionV3 model

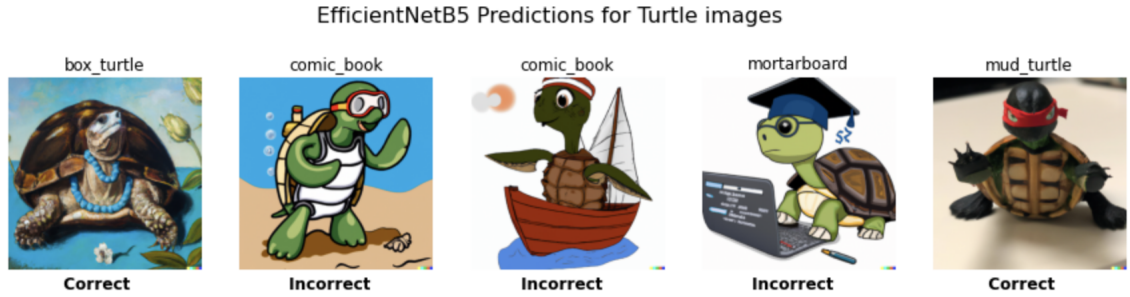

Figure 6 Images Classified by pre-trained EfficientNetB5 model

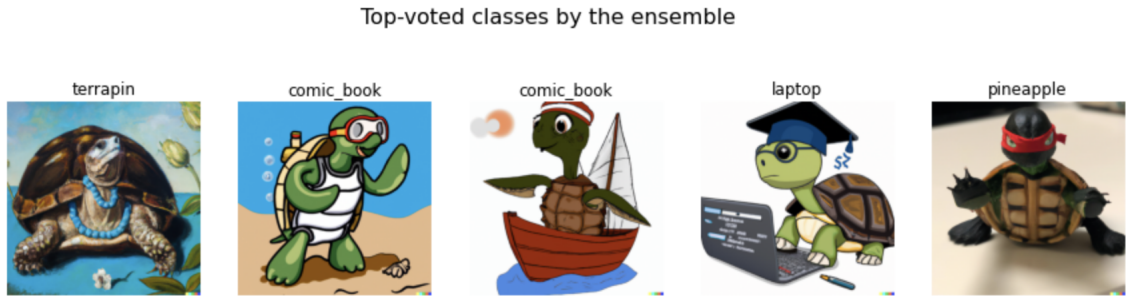

Figure 7 Majority-voted classes for each of the images by the Ensemble

Figure 8 Images ranked on the basis of uncertainty in predictions with less uncertainty being high ranked

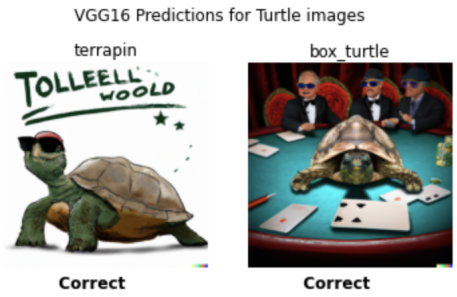

Figure 9 Original Images for adversarial perturbations

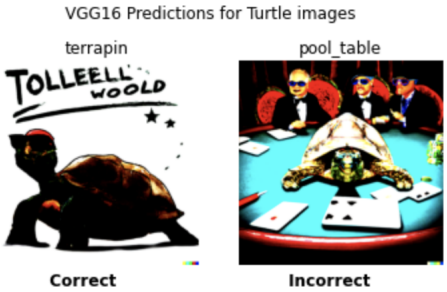

Figure 10 Images perturbed by increasing the contrast

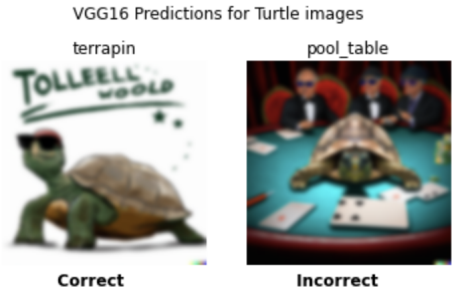

Figure 11 Images perturbed by adding blur to the images

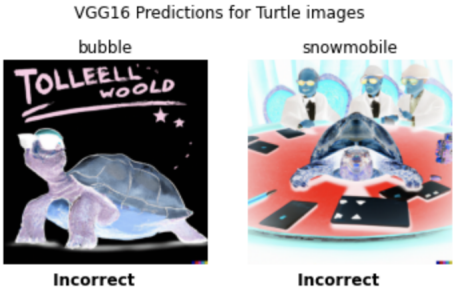

Figure 12 Images perturbed by inverting the colours

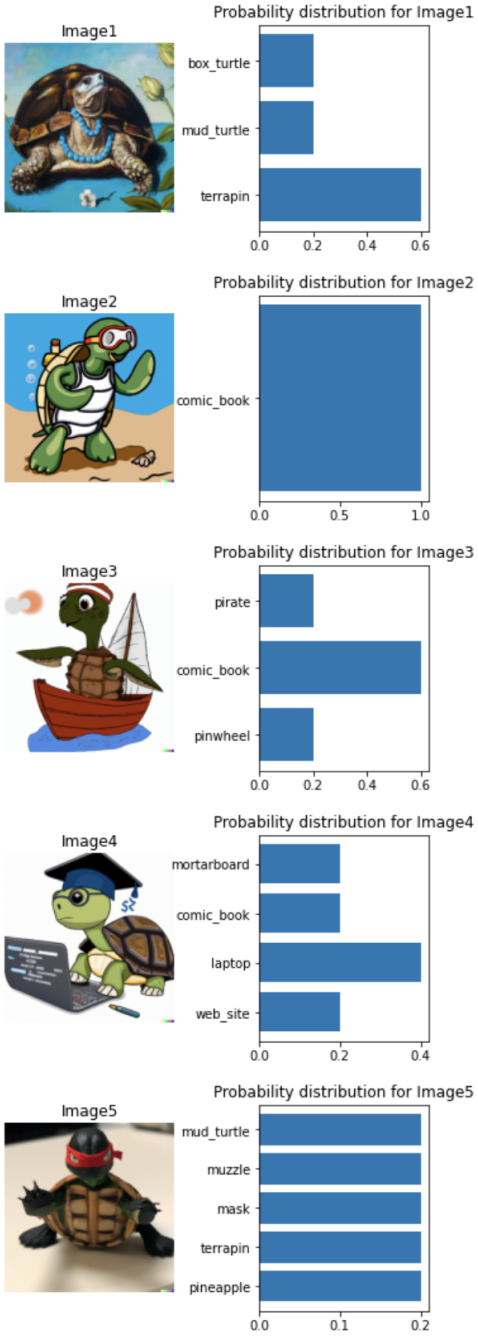

Figure 13 Class Probabilites for each image

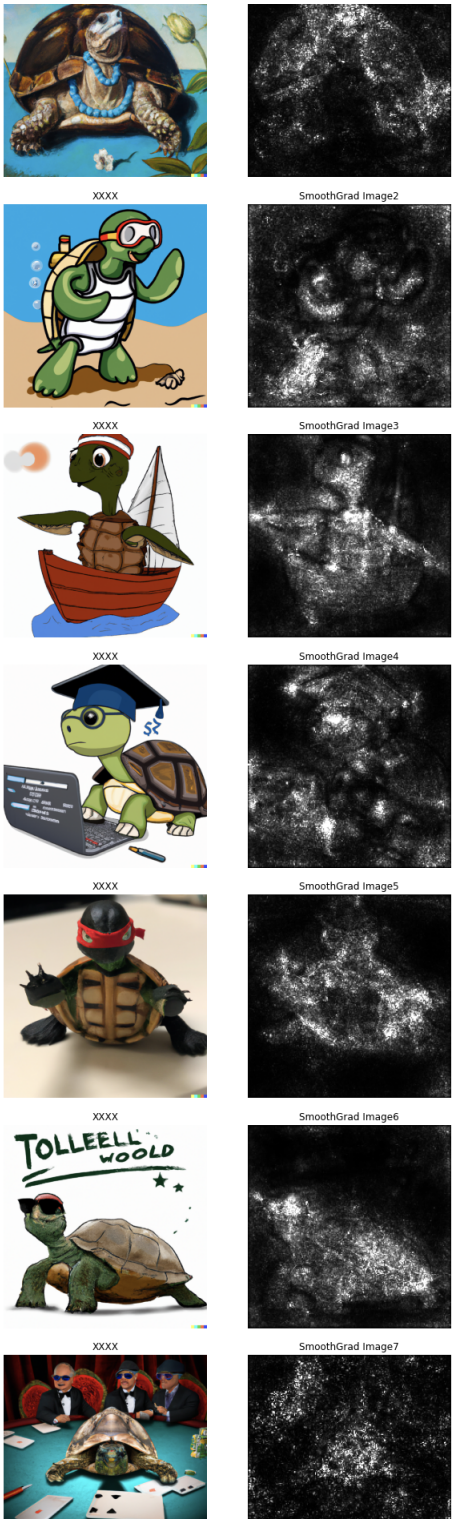

Figure 14 Saliency Maps for all images