Performance Analysis of Deep Neural Networks on Objects with Occlusion

Final Project

6.861 Aspects of a Computational Theory of Intelligence

Massachusetts Institute of Technology

Enrique Fernandez
[redacted]@mit.edu

December 9th, 2016

Contents

1 Introduction

Over the last few years Deep Convolutional Neural Networks have demonstrated a great success in image classification tasks. Since 2012, when the commonly known 'AlexNet' [5] Deep Convolutional Neural Network won the Large Scale Visual Recognition Challenge (ILSRVC) [8], deep CNNs have been the state of the art in object recognition and have achieved a level of performance that has exceeded previous approaches by a large margin. After 'AlexNet', the community has seen significant breakthroughs in recognition performance year after year due to deeper neural network architectures (e.g. VGG [9], Inception [10]), more efficient training techniques (e.g. Batch Normalization [4]) or clever insights, such as those present in the ResNet [3] network, in which only the differences between layers are learned, instead of learning everything from scratch.

However, due to the vast amount of parameters and connections involved in these deep neural networks, it is hard to understand how they work internally, especially in their deepest layers. It is well know that while these networks can achieve near human-level performance in normal images, it is possible to perturb images slightly in a way that these networks cannot recognize them anymore, but that does not affect how humans perceive them [7].

In this project I aim to study the effect of occlusions in the performance of state of the art Deep Convolutional Neural Networks. In particular, I look for answers to the following questions:

1. How much does the presence of occlusions affect image recognition performance?
2. Are different models affected in a different way?
3. What image categories are more or less resilient to occlusions with respect to classification?
4. How important is the background of an image in classification?

In the following sections I explain the method that I used to seek answers to these questions, the data and the models that I employed and the results that I obtained.

2 Method

We assess the performance of the following state of the art models under the presence of occlusions:

1. AlexNet [5]
2. VGG16 and VGG19 [9]
3. Inception v3 [10]
4. ResNet 50 [3]

I found the weights of these pre-trained models online. These models were all trained on the ImageNet dataset. In order to make predictions with these models, I used the Keras[2] library with the Tensorflow [1] backend.

The method for assessing the performance of these models under occlusions is shown in Figure 1. In a nutshell, images from a database that contains ground truth labels (described in Section 3) are fed into the models, and the top-1 and top-5 predictions are compared to the ground truth in order to establish the performance.

3 Image Dataset

The image dataset used in this project is the ImageNet dataset from the ILSVRC [8], since it is a high quality large dataset that is very well annotated. There are two ILSRVC datasets that could have been used in this project:

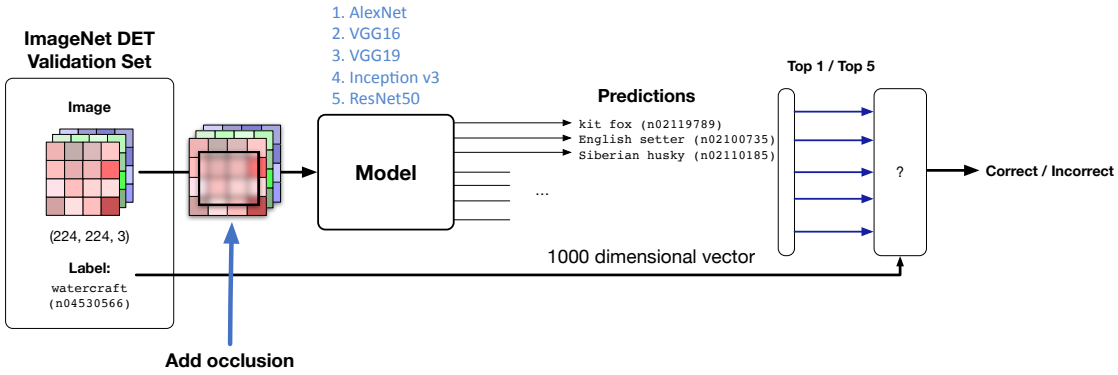


Figure 1: Method

- The Classification-Location Dataset (**CLS-LOC**)
  This dataset consists of 1.3 million training, 50,000 validation and 100,000 test images. The test and validation images are annotated with 1000 categories that are very specific (such as different dog breeds). All models used in this project were trained with this dataset.

- The Object Detection Dataset (**DET**)
  This dataset consists of about 450,000 training, 20,000 validation and 50,000 test images, annotated with 200 different categories, that are much more general than the CLS-LOC ones. Also, all training and validation images are annotated with the bounding boxes and object category of all the objects that appear.

I choose to use the validation set of the DET dataset in this project for the following reasons. First, the bounding box information for the objects was available (as XML files). Second, this dataset does not include images in which the object occupies more than 50% of the image. This is important since we add occlusion only inside the bounding box of the object, and it is important that the background is still present in order to give the models a chance to identify the image after the occlusion is added. Moreover, the 200 categories in the DET dataset are more general, easier to handle and more relevant for a project like this one. Finally, the validation set has a more manageable size, which is important given the limited computational resources and time that I had to work on this project.

Unfortunately, more than one object appears, in general, in the DET images. This is a problem since the models used are only trained to distinguish one object, and adding occlusion to one while leaving the others untouched would be hard to track. For this reason I chose to filter the DET validation dataset and only use the images that have only one object in them. The resulting database contains 7706 images.

However, using the DET dataset introduces a problem. Since all deep models were trained with the CLS-LOC dataset, their predictions are the relative weights of each of the 1000 classes, while the ground truth labels of the DET dataset are one of the 200 categories in the dataset. In order to solve this problem, I found a mapping from the CLS-LOC classes to the DET ones. This is discussed in the next section.

3.1 CLS-LOC to DET conversion

The categories used in the ImageNet datasets are organized according to the WordNet structure[6]. The WordNet is a tree of terms that describe how words are associated with each other. For example, the tree describes that a *labrador retriever* is a *dog* that is an *animal*. In general, the categories in the CLS-LOC database are very specific (deep in the tree), such as dog breeds, whereas the DET categories are much more general (types of animals, objects, etc).

In order to find the conversion between CLS-LOC and DET classes, I used a file that I found online that describes all the *is-a* relations between nouns in the WordNet. Then, for each of the CLS-LOC categories I walked back its ancestor tree until finding one of the 200 categories in the DET dataset. Figure 2, for example, shows the family tree of the *watercraft*, which is one of the DET categories. The terms with the blue background are categories in the CLS-LOC database. In this case, the *watercraft* DET category corresponds

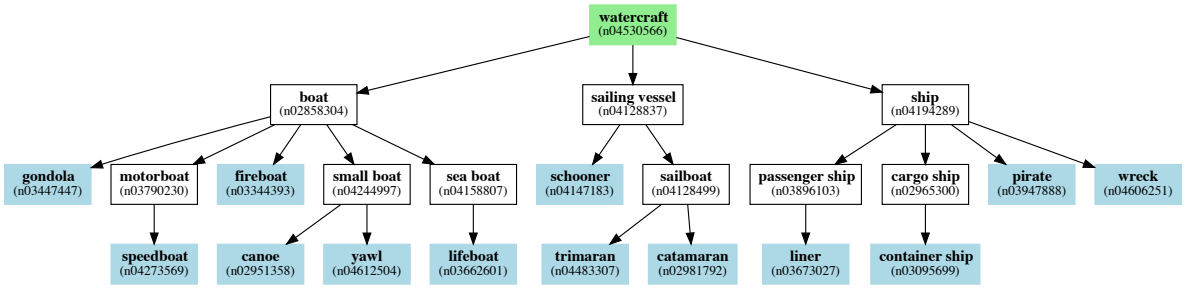


Figure 2: Watercraft family tree

to 13 CLS-LOC categories. For a watercraft image, the answer of the model is considered correct if it outputs any of these 13 CLS-LOC categories.

3.2 Random Chance calculation

If the output of the models used was one of the 200 categories directly, the random chance of guessing the correct category for an image would be $RC_1 = 1/200 = 0.005$. Similarly, the chance of having one of its top 5 predictions being the correct class would be $RC_5 = 1 - \frac{199 \cdot 198 \cdot 197 \cdot 196 \cdot 195}{200 \cdot 199 \cdot 198 \cdot 197 \cdot 196} = 0.025$ (computed as one minus the probability of guessing it wrongly).

However, the models output 1000 categories and these are mapped to the 200 categories, and the mapping is very irregular. Dogs, for example, have 116 classes out of the 1000 classes, birds have 52 and snakes have 17. Other DET classes only map to a single of the 1000 CLS-LOC categories. For this reason, the random chance of guessing the category of an image depends on the category and can be computed with the following formula

$$\psi_j(k) = \frac{\sum_{i=1}^{k} \binom{n_{cj}}{i} \binom{N-n_{cj}}{k-i}}{\binom{N}{k}} \tag{1}$$

, where $\psi_j(k)$ is the random chance of guessing DET category j among the top-k predictions, $N = 1000$ is the number of the CLS-LOC categories and n_{cj} is the number of CLS-LOC categories that the j DET category has assigned. Classes with $n_c = 1$ have top 1 and top 5 random chances of 0.001 and 0.005 respectively. However, the dog category (with $n_c = 116$) has top 1 and top 5 chances of 0.116 and 0.461. The random chances for most categories are shown in Appendix A.

The overall top-k random chance, Ψ_k, across the full database of the 7706 images is computed as

$$\Psi_k = \frac{\sum_j m_j \psi_j(k)}{\sum_j m_j} \tag{2}$$

, where m_j is the number of images of the DET category j in the dataset.

4 Occlusion Generation

In order to generate occlusions I used Gaussian Blur that was added with the OpenCV library. The Gaussian Blur was added inside the bounding box of the object and the occluded amounts varied among 0%, 10%, 20%, 50%, 75% and 100%. The aspect ratio of the blur was kept as the aspect ratio of the object bounding box. The blur, with the coverage amounts specified, was added in *center*, *corner* and *random* locations. For the corner locations, the corner was chosen randomly for each image, but all models were evaluated with the same choices. This is also true for the random location occlusion. Figure 3 shows some examples of images generated using this method.

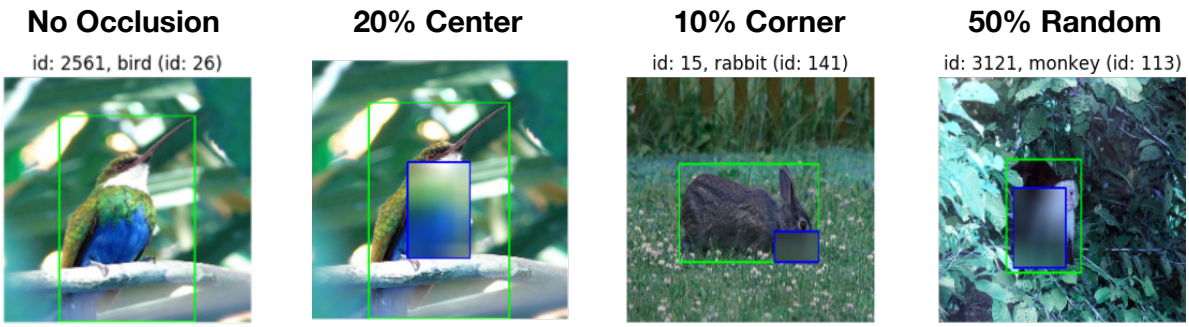
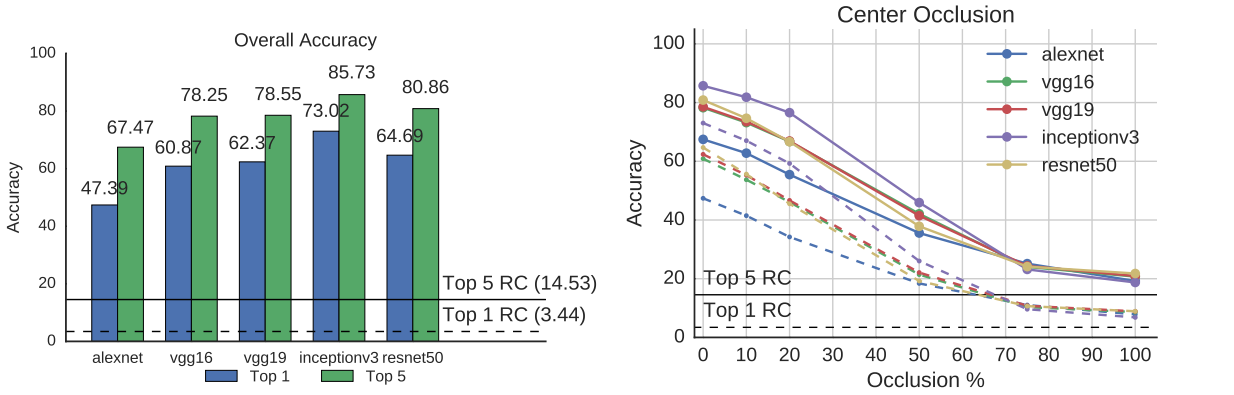


Figure 3: Occlusion examples



(a) Baseline performance with no occlusions

(b) Performance decrease with center occlusions. Top-5 in solid, top-1 in dashed lines.

Figure 4: Overall results

5 Results

The combination of occlusion locations and coverage amounts produce 14 different datasets of images, each having 7706 images. All these datasets were classified using each of the 5 models (AlexNet, VGG16, VGG19, Inception v3 and ResNet50). This resulted in 539,420 images being analyzed, which took a few hours in the GPU I had access to. Figure 4a shows the baseline, that is, the performance results of each model in the absence of occlusions. As seen in the graph, the best model in this dataset was Inception v3.

Figure 4b shows the performance decrease in all models as center occlusion is added. As it can be seen in the figure, the performance drops significantly as occlusion is added. However, it is very interesting to notice that by the time the occlusion reaches 75%, the performance of all models is essentially the same, even when the initial performance was very different. This indicates that, as expected, the main object is much more important that the background in order to perform classification.

Figure 5 shows the performance decrease for the center, corner and random locations of the occlusion as a function of the occlusion amount. As seen in the figure, the performance decrease due to center occlusion is the worst of all, while the corner occlusion is the least severe and the random one lies between these two. This results meet expectations because, unlike their bounding boxes, most objects do not have rectangular shapes and, in many situations, corner occlusions only slightly block the actual object.

After discussing the performance decrease across all categories, I proceed to assess the performance of these models in the individual DET categories. Figure 6 left shows that the performance across categories has a wide spread, with some categories having a very low performance even in the absence of occlusions, while others show a 100 % top-1 accuracy even with 20 % random occlusion. Figure 6 right shows the relative performance decrease across categories with respect to the performance with no occlusions. As seen

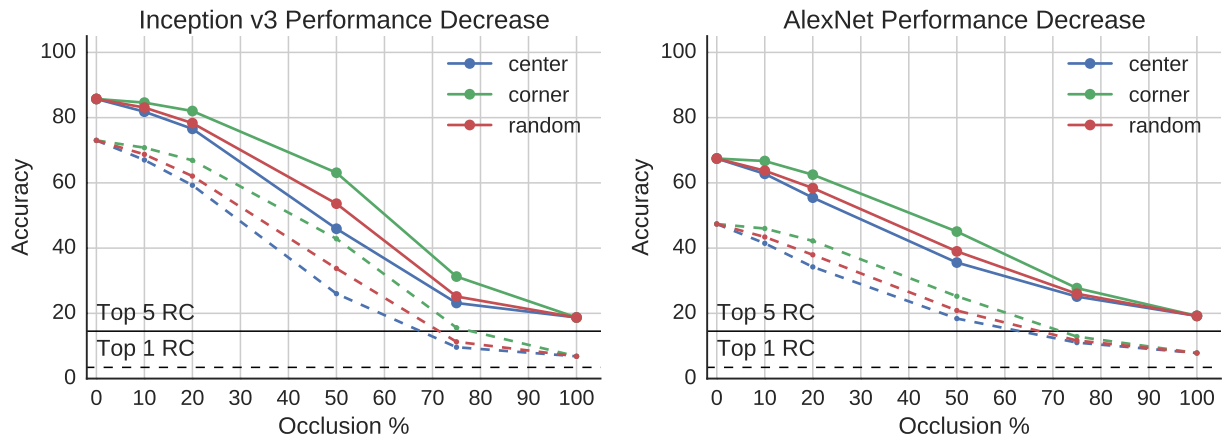


Figure 5: Performance decrease depending on the location of the occlusion for Inception (a) and AlexNet (b)

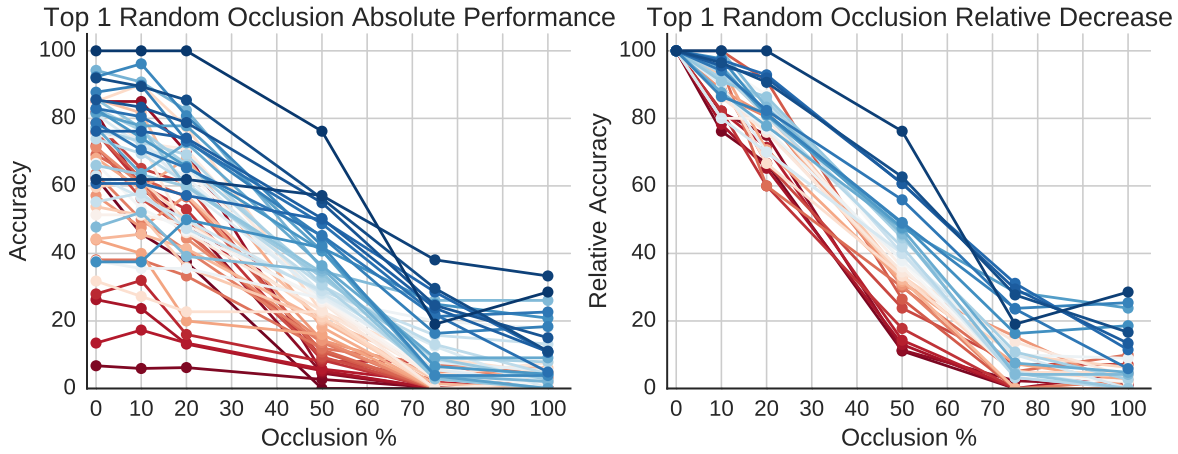


Figure 6: Absolute and relative performance of Inception v3 across DET categories

in the graph, the performance is maintained well up to 50 % random occlusion for some categories, while other categories see their performance decrease drastically as soon as some occlusion is introduced. These graphs show the results for the Inception v3 model, but the plots for the other models look fairly similar.

Finally, Table 1 shows the Inception v3 absolute performance across categories in the absence of occlusion and the relative performance decrease in the presence of 50 % random occlusions. We can observe in these tables that simpler objects with round, rectangular or elongated shapes (such as a golf ball) are easier to classify in the presence of obstacles. Moreover, we can hypothesize that objects that appear in distinctive backgrounds such as butterflies, birds or bees are easier to distinguish even in the presence of high amounts of occlusion compared to other objects that often appear in very common non-distinctive scenarios (such as domestic cats).

6 Conclusions and Future Work

As expected, we have seen that the performance of state of the art Deep CNN models decreases sharply in the presence of occlusions. Perhaps more unexpectedly, I have shown that in the presence of high rates of occlusion all models perform essentially the same. This is surprising since the top-5 performance advantage between Inception v3 and AlexNet is almost 20 points. This indicates that these models get most of the

	name	ψ^5	#	Top 1	Top 5
1	golf ball	0.005	21	1	1
2	dog	0.461	1365	0.94	0.98
3	skunk	0.005	26	0.92	0.96
4	butterfly	0.03	199	0.92	0.96
5	whale	0.01	49	0.88	0.94
6	airplane	0.005	35	0.86	0.97
7	swine	0.015	49	0.86	0.94
8	bird	0.235	1629	0.86	0.95
9	giant panda	0.005	20	0.85	0.95
10	koala bear	0.005	40	0.85	0.93
..					
49	tape player	0.005	42	0.38	0.95
50	train	0.005	45	0.38	0.47
51	mushroom	0.005	24	0.38	1
52	wine bottle	0.005	22	0.32	0.86
53	purse	0.005	25	0.28	0.76
54	bowl	0.01	38	0.26	0.5
55	cup or mug	0.005	52	0.13	0.52
56	lamp	0.005	34	0.12	0.29
57	person	0.015	371	0.07	0.11
58	table	0.005	64	0	0.05

(a) Baseline performance without occlusions

	name	ψ^5	#	Top 1	Top 5
1	golf ball	0.005	21	0.76	0.95
2	bird	0.235	1393	0.63	0.8
3	watercraft	0.063	112	0.61	0.87
4	butterfly	0.03	183	0.61	0.81
5	ray	0.01	34	0.56	0.74
6	bear	0.02	59	0.49	0.62
7	whale	0.01	43	0.49	0.74
8	bee	0.005	21	0.48	0.91
9	snake	0.082	292	0.47	0.59
10	skunk	0.005	24	0.46	0.6
..					
26	porcupine	0.005	26	0.31	0.53
27	antelope	0.015	46	0.3	0.55
28	chime	0.005	20	0.3	0.65
29	koala bear	0.005	34	0.26	0.49
30	swine	0.015	42	0.24	0.43
31	domestic cat	0.025	45	0.18	0.28
32	snail	0.005	28	0.14	0.31
33	scorpion	0.005	23	0.13	0.1
34	turtle	0.025	88	0.11	0.3
35	frog	0.015	63	0.11	0.35

(b) Performance decrease with 50% random occlusion

Table 1: Performance across categories using the Inception v3 model

information from the actual object as opposed to the background. We have also seen that, as expected, center occlusion degrades performance significantly more than random and corner occlusion.

Finally, the data also shows that there is a big spread in the performance under occlusion across different classes, and that classes that appear often in very distinctive scenarios are more resilient to occlusion, compared to other classes that can appear in all kind of unrelated scenarios.

As future work it would be interesting to redo this analysis with circular occlusion, as opposed to the rectangular occlusion that I used here, since the artifacts added by the circular occlusion should be smoother than the sharp rectangular occlusion that I artificially introduced. In the same way it would also be interesting to understand the effect of adding occlusions with salt and pepper noise as opposed to the Gaussian Blur that was used in this work. Finally, it would be interesting to fine tune these models with additional images in the presence of occlusions to see if the neural networks would be forced to learn smaller parts of the objects in order to be able to differentiate them in the presence of occlusions.

For someone with very limited knowledge (and zero experience) with neural networks, with only elementary biology knowledge and with an absolute lack of familiarity with neuroscience, this has been a fascinating course. I have enjoyed tremendously the contagious enthusiasm and hunger of knowledge that the speakers showed while telling us about their quest deciphering the mysteries of the human brain.

A DET Categories

	name	#	nc	ψ^1	ψ^5	inceptionv3	resnet50	vgg19	vgg16	alexnet
1	bird (id: 26)	1629	52	0.052	0.235	1393 (0.86)	1275 (0.78)	1283 (0.79)	1257 (0.77)	1159 (0.71)
2	dog (id: 58)	1365	116	0.116	0.461	1286 (0.94)	1230 (0.90)	1198 (0.88)	1198 (0.88)	906 (0.66)
3	person (id: 124)	371	3	0.003	0.015	25 (0.07)	26 (0.07)	15 (0.04)	12 (0.03)	21 (0.06)
4	snake (id: 159)	353	17	0.017	0.082	292 (0.83)	237 (0.67)	249 (0.71)	228 (0.65)	184 (0.52)
5	lizard (id: 105)	246	11	0.011	0.054	192 (0.78)	169 (0.69)	179 (0.73)	167 (0.68)	79 (0.32)
6	monkey (id: 113)	229	13	0.013	0.063	190 (0.83)	147 (0.64)	144 (0.63)	145 (0.63)	110 (0.48)
7	butterfly (id: 34)	199	6	0.006	0.03	183 (0.92)	174 (0.87)	165 (0.83)	159 (0.80)	150 (0.75)
8	watercraft (id: 197)	147	13	0.013	0.063	112 (0.76)	105 (0.71)	99 (0.67)	105 (0.71)	68 (0.46)
9	car (id: 37)	112	10	0.01	0.049	74 (0.66)	71 (0.63)	66 (0.59)	64 (0.57)	26 (0.23)
10	turtle (id: 188)	112	5	0.005	0.025	88 (0.79)	68 (0.61)	62 (0.55)	59 (0.53)	44 (0.39)
11	frog (id: 72)	83	3	0.003	0.015	63 (0.76)	55 (0.66)	49 (0.59)	39 (0.47)	36 (0.43)
12	fox (id: 70)	76	4	0.004	0.02	57 (0.75)	49 (0.64)	47 (0.62)	46 (0.61)	23 (0.30)
13	bear (id: 20)	75	4	0.004	0.02	59 (0.79)	49 (0.65)	38 (0.51)	42 (0.56)	15 (0.20)
14	rabbit (id: 141)	70	2	0.002	0.01	31 (0.44)	19 (0.27)	30 (0.43)	29 (0.41)	16 (0.23)
15	table (id: 177)	64	1	0.001	0.005	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)	0 (0.00)
16	domestic cat (id: 59)	63	5	0.005	0.025	45 (0.71)	39 (0.62)	36 (0.57)	35 (0.56)	12 (0.19)
17	antelope (id: 4)	55	3	0.003	0.015	46 (0.84)	40 (0.73)	41 (0.75)	40 (0.73)	34 (0.62)
18	dragonfly (id: 60)	54	1	0.001	0.005	41 (0.76)	45 (0.83)	33 (0.61)	34 (0.63)	29 (0.54)
19	cup or mug (id: 54)	52	1	0.001	0.005	7 (0.13)	10 (0.19)	12 (0.23)	11 (0.21)	1 (0.02)
20	ladybug (id: 99)	52	1	0.001	0.005	31 (0.60)	20 (0.38)	22 (0.42)	18 (0.35)	25 (0.48)
21	swine (id: 175)	49	3	0.003	0.015	42 (0.86)	36 (0.73)	34 (0.69)	32 (0.65)	6 (0.12)
22	whale (id: 198)	49	2	0.002	0.01	43 (0.88)	32 (0.65)	30 (0.61)	30 (0.61)	40 (0.82)
23	chair (id: 43)	46	4	0.004	0.02	29 (0.63)	23 (0.50)	18 (0.39)	18 (0.39)	15 (0.33)
24	train (id: 185)	45	1	0.001	0.005	17 (0.38)	15 (0.33)	16 (0.36)	13 (0.29)	13 (0.29)
25	tape player (id: 178)	42	1	0.001	0.005	16 (0.38)	8 (0.19)	10 (0.24)	10 (0.24)	10 (0.24)
26	ray (id: 143)	41	2	0.002	0.01	34 (0.83)	23 (0.56)	19 (0.46)	20 (0.49)	16 (0.39)
27	koala bear (id: 97)	40	1	0.001	0.005	34 (0.85)	34 (0.85)	31 (0.78)	29 (0.72)	22 (0.55)
28	red panda (id: 144)	39	1	0.001	0.005	21 (0.54)	6 (0.15)	8 (0.21)	10 (0.26)	13 (0.33)
29	bee (id: 21)	38	1	0.001	0.005	21 (0.55)	18 (0.47)	19 (0.50)	17 (0.45)	23 (0.61)
30	bowl (id: 30)	38	2	0.002	0.01	10 (0.26)	9 (0.24)	9 (0.24)	5 (0.13)	2 (0.05)
31	snail (id: 158)	37	1	0.001	0.005	28 (0.76)	22 (0.59)	18 (0.49)	17 (0.46)	14 (0.38)
32	airplane (id: 2)	35	1	0.001	0.005	30 (0.86)	27 (0.77)	26 (0.74)	24 (0.69)	14 (0.40)
33	squirrel (id: 166)	35	1	0.001	0.005	18 (0.51)	15 (0.43)	13 (0.37)	9 (0.26)	12 (0.34)
34	lamp (id: 100)	34	1	0.001	0.005	4 (0.12)	6 (0.18)	4 (0.12)	4 (0.12)	5 (0.15)
35	porcupine (id: 134)	32	1	0.001	0.005	26 (0.81)	23 (0.72)	21 (0.66)	23 (0.72)	14 (0.44)
36	scorpion (id: 152)	32	1	0.001	0.005	23 (0.72)	21 (0.66)	18 (0.56)	18 (0.56)	15 (0.47)
37	otter (id: 120)	30	1	0.001	0.005	20 (0.67)	15 (0.50)	17 (0.57)	17 (0.57)	5 (0.17)
38	soap dispenser (id: 162)	28	1	0.001	0.005	17 (0.61)	15 (0.54)	12 (0.43)	13 (0.46)	6 (0.21)
39	armadillo (id: 6)	27	1	0.001	0.005	22 (0.81)	20 (0.74)	19 (0.70)	18 (0.67)	14 (0.52)
40	bus (id: 33)	26	3	0.003	0.015	18 (0.69)	17 (0.65)	13 (0.50)	15 (0.58)	7 (0.27)
41	chime (id: 44)	26	1	0.001	0.005	20 (0.77)	21 (0.81)	16 (0.62)	15 (0.58)	1 (0.04)
42	skunk (id: 157)	26	1	0.001	0.005	24 (0.92)	22 (0.85)	22 (0.85)	23 (0.88)	13 (0.50)
43	tennis ball (id: 179)	26	1	0.001	0.005	20 (0.77)	16 (0.62)	16 (0.62)	17 (0.65)	11 (0.42)
44	purse (id: 140)	25	1	0.001	0.005	7 (0.28)	7 (0.28)	6 (0.24)	6 (0.24)	2 (0.08)
45	sheep (id: 155)	25	1	0.001	0.005	11 (0.44)	6 (0.24)	6 (0.24)	7 (0.28)	1 (0.04)
46	centipede (id: 41)	24	1	0.001	0.005	15 (0.62)	17 (0.71)	12 (0.50)	9 (0.38)	12 (0.50)
47	mushroom (id: 115)	24	1	0.001	0.005	9 (0.38)	12 (0.50)	11 (0.46)	8 (0.33)	2 (0.08)
48	digital clock (id: 56)	23	1	0.001	0.005	17 (0.74)	15 (0.65)	13 (0.57)	14 (0.61)	9 (0.39)
49	flower pot (id: 68)	23	1	0.001	0.005	11 (0.48)	11 (0.48)	8 (0.35)	10 (0.43)	2 (0.09)
50	ant (id: 3)	22	1	0.001	0.005	17 (0.77)	15 (0.68)	15 (0.68)	15 (0.68)	10 (0.45)
51	baby bed (id: 9)	22	3	0.003	0.015	14 (0.64)	12 (0.55)	8 (0.36)	13 (0.59)	5 (0.23)
52	hippopotamus (id: 90)	22	1	0.001	0.005	15 (0.68)	18 (0.82)	18 (0.82)	16 (0.73)	12 (0.55)
53	isopod (id: 95)	22	1	0.001	0.005	18 (0.82)	14 (0.64)	11 (0.50)	12 (0.55)	2 (0.09)
54	wine bottle (id: 199)	22	1	0.001	0.005	7 (0.32)	10 (0.45)	6 (0.27)	7 (0.32)	6 (0.27)
55	golf ball (id: 76)	21	1	0.001	0.005	21 (1.00)	18 (0.86)	17 (0.81)	18 (0.86)	12 (0.57)
56	nail (id: 116)	21	1	0.001	0.005	13 (0.62)	16 (0.76)	14 (0.67)	15 (0.71)	10 (0.48)
57	pencil sharpener (id: 122)	21	1	0.001	0.005	12 (0.57)	12 (0.57)	9 (0.43)	9 (0.43)	2 (0.10)

DET categories with more than 20 images in the single-object database ordered by the number of images. m is the number of images of that category in the database, ψ^k is the top-k random chance of the category as calculated with equation (1). The models columns show the number of images correctly classified by the model and the accuracy in terms of top-1 performance.

B Performance across categories with the Inception v3 model

	name	RC_5	#	Top 1	Top 5
1	golf ball	0.005	21	1	1
2	dog	0.461	1365	0.94	0.98
3	skunk	0.005	26	0.92	0.96
4	butterfly	0.03	199	0.92	0.96
5	whale	0.01	49	0.88	0.94
6	airplane	0.005	35	0.86	0.97
7	swine	0.015	49	0.86	0.94
8	bird	0.235	1629	0.86	0.95
9	giant panda	0.005	20	0.85	0.95
10	koala bear	0.005	40	0.85	0.93
..					
49	tape player	0.005	42	0.38	0.95
50	train	0.005	45	0.38	0.47
51	mushroom	0.005	24	0.38	1
52	wine bottle	0.005	22	0.32	0.86
53	purse	0.005	25	0.28	0.76
54	bowl	0.01	38	0.26	0.5
55	cup or mug	0.005	52	0.13	0.52
56	lamp	0.005	34	0.12	0.29
57	person	0.015	371	0.07	0.11
58	table	0.005	64	0	0.05

(a) No occlusion

	name	RC_5	#	Top 1	Top 5
1	golf ball	0.005	21	0.76	0.95
2	nail	0.005	21	0.57	0.71
3	butterfly	0.03	199	0.56	0.78
4	bird	0.235	1629	0.55	0.77
5	watercraft	0.063	147	0.5	0.85
6	soap dispenser	0.005	28	0.5	0.75
7	ray	0.01	41	0.49	0.71
8	bear	0.02	75	0.45	0.61
9	whale	0.01	49	0.45	0.71
10	skunk	0.005	26	0.42	0.58
..					
49	turtle	0.025	112	0.09	0.27
50	frog	0.015	83	0.08	0.31
51	purse	0.005	25	0.08	0.32
52	cup or mug	0.005	52	0.06	0.29
53	bowl	0.01	38	0.05	0.24
54	giant panda	0.005	20	0.05	0.3
55	isopod	0.005	22	0.05	0.14
56	person	0.015	371	0.03	0.07
57	centipede	0.005	24	0	0.12
58	table	0.005	64	0	0.08

(b) 50 % random occlusion

	name	RC_5	#	Top 1	Top 5
1	nail	0.005	21	0.33	0.52
2	golf ball	0.005	21	0.29	0.52
3	flower pot	0.005	23	0.26	0.52
4	bear	0.02	75	0.23	0.37
5	mushroom	0.005	24	0.21	0.33
6	whale	0.01	49	0.18	0.39
7	train	0.005	45	0.18	0.33
8	bird	0.235	1629	0.15	0.37
9	bee	0.005	38	0.13	0.66
10	butterfly	0.03	199	0.11	0.37
..					
49	lamp	0.005	34	0	0.09
50	otter	0.005	30	0	0.1
51	porcupine	0.005	32	0	0.03
52	purse	0.005	25	0	0
53	rabbit	0.01	70	0	0.11
54	sheep	0.005	25	0	0
55	snail	0.005	37	0	0
56	squirrel	0.005	35	0	0.11
57	table	0.005	64	0	0.02
58	tennis ball	0.005	26	0	0.04

(c) 100 % random occlusion

Table 2: Absolute performance in Inception v3

	name	RC_5	#	Top 1	Top 5
1	golf ball	0.005	21	1	1
2	butterfly	0.03	183	0.93	0.97
3	watercraft	0.063	112	0.92	0.97
4	koala bear	0.005	34	0.91	0.89
5	bird	0.235	1393	0.91	0.95
6	dog	0.461	1286	0.87	0.94
7	armadillo	0.005	22	0.86	0.96
8	car	0.049	74	0.85	0.94
9	tennis ball	0.005	20	0.85	0.92
10	skunk	0.005	24	0.83	0.96
..					
26	snail	0.005	28	0.71	0.75
27	dragonfly	0.005	41	0.71	0.87
28	airplane	0.005	30	0.7	0.97
29	otter	0.005	20	0.7	0.71
30	frog	0.015	63	0.67	0.72
31	red panda	0.005	21	0.67	0.7
32	swine	0.015	42	0.67	0.91
33	scorpion	0.005	23	0.65	0.72
34	chime	0.005	20	0.6	0.83
35	domestic cat	0.025	45	0.6	0.79

(a) 20 % random occlusion

	name	RC_5	#	Top 1	Top 5
1	golf ball	0.005	21	0.76	0.95
2	bird	0.235	1393	0.63	0.8
3	watercraft	0.063	112	0.61	0.87
4	butterfly	0.03	183	0.61	0.81
5	ray	0.01	34	0.56	0.74
6	bear	0.02	59	0.49	0.62
7	whale	0.01	43	0.49	0.74
8	bee	0.005	21	0.48	0.91
9	snake	0.082	292	0.47	0.59
10	skunk	0.005	24	0.46	0.6
..					
26	porcupine	0.005	26	0.31	0.53
27	antelope	0.015	46	0.3	0.55
28	chime	0.005	20	0.3	0.65
29	koala bear	0.005	34	0.26	0.49
30	swine	0.015	42	0.24	0.43
31	domestic cat	0.025	45	0.18	0.28
32	snail	0.005	28	0.14	0.31
33	scorpion	0.005	23	0.13	0.1
34	turtle	0.025	88	0.11	0.3
35	frog	0.015	63	0.11	0.35

(b) 50 % random occlusion

	name	RC_5	#	Top 1	Top 5
1	golf ball	0.005	21	0.29	0.52
2	bear	0.02	59	0.25	0.38
3	bee	0.005	21	0.24	0.71
4	whale	0.01	43	0.19	0.41
5	bird	0.235	1393	0.17	0.39
6	watercraft	0.063	112	0.13	0.36
7	butterfly	0.03	183	0.11	0.38
8	chime	0.005	20	0.1	0.22
9	monkey	0.063	190	0.09	0.32
10	swine	0.015	42	0.07	0.07
..					
26	armadillo	0.005	22	0	0
27	domestic cat	0.025	45	0	0
28	frog	0.015	63	0	0.01
29	koala bear	0.005	34	0	0.03
30	ladybug	0.005	31	0	0.12
31	otter	0.005	20	0	0.11
32	porcupine	0.005	26	0	0.03
33	rabbit	0.01	31	0	0.12
34	snail	0.005	28	0	0
35	tennis ball	0.005	20	0	0.04

(c) 100 % random occlusion

Table 3: Relative performance decrease in Inception v3

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv.org*, March 2016.

[2] Francois Chollet. Keras. Technical report, 2015.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv.org*, December 2015.

[4] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv.org*, February 2015.

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in neural ...*, pages 1097–1105, 2012.

[6] G A Miller. WordNet: a lexical database for English. *Communications of the ACM*, 1995.

[7] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. *arXiv.org*, October 2016.

[8] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[9] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv.org*, September 2014.

[10] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. *arXiv.org*, December 2015.