Zero-shot Learning - Animals with Attributes

Yannis Karakozis

Methodology

Animal Attribute Classification

For the attribute classification step, I used a ResNet18 architecture [1] pretrained on Imagenet [3]. I maintained the pretrained weights of the convolutional layers but replaced the last fully connected layer with a fully connected layer with an output dimensionality equal to the number of animal attributes. I initialized the fully connected layer parameters by drawing values from $[-s, s]$ uniformly at random. The spread s is given by $2/\sqrt{\text{fan-in} + \text{fan-out}}$, where fan-in is the number of input neurons and fan-out the number of output neurons of the fully connected layer. I apply dropout with a 0.5 factor to the inputs of the fully connected layer as a form of regularization to prevent overfitting.

I train on the full training split of the Animals with Attributes (AwA) dataset [4] using just the animal attribute labels. I preprocess the image data by normalizing them by the Imagenet per color channel mean and standard deviation values, as is required for data to be used in Imagenet-pretrained models [3]. I data augment my training set by performing random horizontal flips in the training data with a probability of 0.5. I train for 20 epochs with batches of 64 images using an Adam optimizer [2] with learning rate 0.00001 and weight decay 0.00001.

I apply the sigmoid function to the network outputs and then compute the aggregate Binary Cross Entropy (BCE) Loss over all attribute classes using the sigmoid values. Since there is no validation set provided, I also compute the BCE Loss on the test set using the animal attributes labels. I save the model corresponding to the epoch in which the BCE Loss on the test set is the lowest. Despite training for 20 epochs, I observe that the model typically needs less than 5 epochs to converge to the least aggregate BCE Loss on the test set.

Zero-shot Animal Class Recognition

For the zero-shot class recognition step, I used the trained ResNet18 model to predict animal attributes for each image in the AwA test set. To do so, I freeze gradient backpropagation, forward feed each image through my ResNet18 model, and apply the sigmoid function to the N-dimensional vector produced by the network ($N = 85$, i.e. the number of animal attributes). Each value in the final vector corresponds to an animal attribute.

To identify the animal classes, I use a simple top1 nearest neighbor classification hieuristic. I first initialize a KD-Tree nearest neighbor classifier using the ground truth face attribute label vectors of the 10 animal classes of the AwA test split [4]. I then identify the animal class of each test image by finding the top1 nearest neighbor of the image's N-dimensional sigmoid valued vector. To maximize my chance of attaining high performance, I find the top1 nearest neighbor using both L2 and the cosine similarity distance metrics.

Evaluation

My system achieves 45.14% accuracy with L2-distance and 42.62% accuracy with cosine similarity in the zero-shot class recognition task, vastly outperforming the given baseline. Interestingly, if I instead use the predicted label vectors (i.e. the sigmoid valued vectors with their entries thresholded at 0.5) as inputs to the nearest neighbor classifiers, accuracy drops to 42.65% and 41.09% respectively.

Error Analysis

Figure 1 presents the confusion matrix among the 10 classes predicted at test time with un-normalized confusion values. Figure 2 presents the normalized confusion matrix for the 10 test-time classes. The pairs (hippopotamus, seal), (humpback+whale, seal), and (persian cat, rat) are significantly confused for one another, due to the high values in their paired confusion matrix entries.

I also observe that giant pandas are heavily predicted as persian cats (0.82), hippopotamus as pigs (0.46), pigs as persian cats (0.23), and raccoons as leopards (0.24) or rats (0.38). The numbers inside the parentheses are the fraction of times an image of the first class is predicted as an image of the second class. These values are taken from Figure 2. There is no more than 20% confusion among the remaining animal class pairs according to Figure 2.

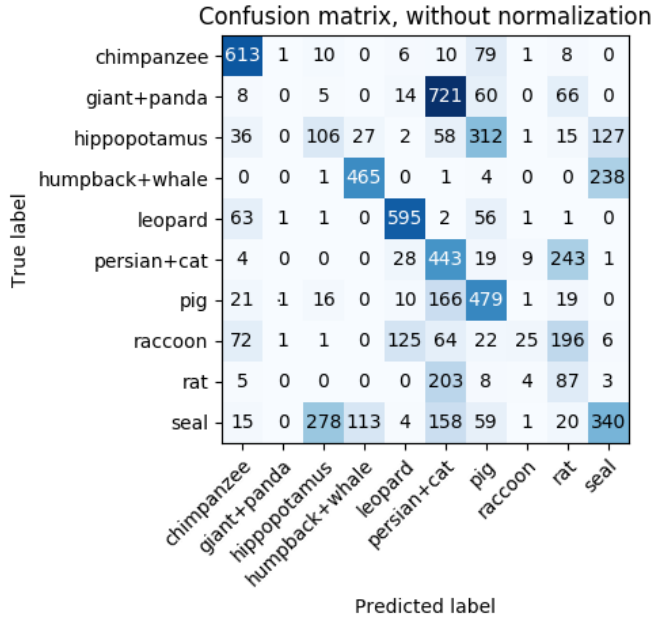

Figure 1: Confusion Matrix

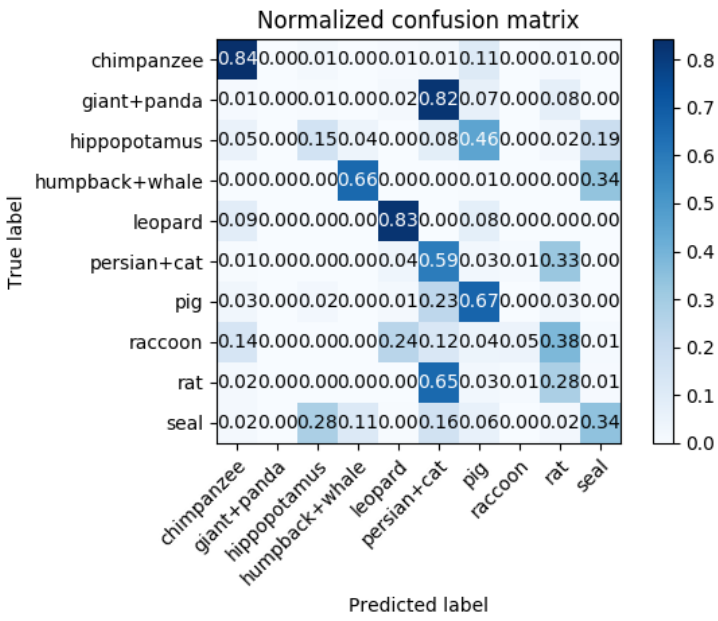

Figure 2: Normalized Confusion Matrix

In Figure 3, I present the per attribute percentage classification accuracy achieved on the test set, i.e. on the images of the 10 animal classes that are not presented to the model during training. The attributes 'red', 'yellow', 'orange', 'hands', 'longneck', 'horns', 'tusks', 'flys', 'hops', 'tunnels', 'plankton', 'oldworld', 'coastal', 'desert', 'ocean', 'cave' are predicted correctly more than 90% of the time. On the contrary, the attributes 'brown', 'spots', 'lean', 'smelly', 'active', 'fish', 'fierce', 'domestic' have less than 55% accuracy. This is bad if one considers that predicting uniformly at

random would yield a 50% accuracy due to the binary nature of the labels of the attributes.

Due to the fact there are no specific types of attributes (e.g. color, location, limb types, environmental elements) that the model performs consistently well or consistently bad in, there is no good justification for why these attributes have such low accuracy while others do not. A surprising observation though is that the attributes with low accuracy have typically 15,000+ positive examples in the training set while the ones with high accuracy have typically less than 5000. This might indicate that the network is better at identifying attributes that make animals distinct from one another, i.e attributes that are not typically observed across species, such as bright colors, tusks and horns. This is unlike elements that can be found across animals, such as spots, dark colors such as brown and black, or the abstract properties of being an 'active' or a 'fierce' animal. Such attribute sharing probably makes it harder for the network to pinpoint the visual manifestation of the attribute across images of different species.

black: 56.54%	lean: 54.43%	flys: 100.00%	fish: 52.70%	plains: 84.35%
white: 60.69%	flippers: 87.76%	hops: 96.75%	meat: 64.70%	forest: 68.26%
blue: 86.04%	hands: 93.54%	swims: 88.59%	plankton: 91.51%	fields: 79.79%
brown: 50.29%	hooves: 88.22%	tunnels: 96.84%	vegetation: 68.78%	jungle: 66.49%
gray: 71.54%	pads: 76.41%	walks: 87.24%	insects: 85.14%	mountains: 72.86%
orange: 90.29%	paws: 82.56%	fast: 70.82%	forager: 68.02%	ocean: 91.84%
red: 99.99%	longleg: 84.65%	slow: 58.53%	grazer: 69.18%	ground: 89.59%
yellow: 96.54%	longneck: 96.81%	strong: 80.79%	hunter: 57.59%	water: 88.00%
patches: 57.81%	tail: 71.58%	weak: 87.39%	scavenger: 87.14%	tree: 65.31%
spots: 49.69%	chewteeth: 61.92%	muscle: 73.43%	skimmer: 89.86%	cave: 98.25%
stripes: 88.76%	meatteeth: 63.21%	bipedal: 83.34%	stalker: 84.32%	fierce: 53.40%
furry: 71.94%	buckteeth: 78.17%	quadrapedal: 87.19%	newworld: 75.75%	timid: 65.60%
hairless: 77.25%	strainteeth: 82.89%	active: 50.34%	oldworld: 95.50%	smart: 71.73%
toughskin: 86.36%	horns: 96.23%	inactive: 59.96%	arctic: 83.08%	group: 57.31%
big: 82.76%	claws: 76.26%	nocturnal: 74.93%	coastal: 91.60%	solitary: 70.82%
small: 67.10%	tusks: 98.17%	hibernate: 79.87%	desert: 99.94%	nestspot: 61.96%
bulbous: 60.40%	smelly: 53.89%	agility: 65.44%	bush: 74.60%	domestic: 54.46%

Figure 3: Per Class Attribute Classification

According to the diagonal entries of Figure 2 which present the prediction accuracy of each animal class, giant pandas (0.00) are predicted entirely wrong, while raccoons (0.05) and hippopotamuses (0.15) are also major sources of error. One possible explanation is that these are the only three test classes that satisfy the following property: The mean number of attributes they share with the train animal classes is lower than the mean number of attributes their most similar test animal class shares with the train animal classes.

For example, the most similar test class to the raccoon is the rat, with 67 shared attributes. This means that the network should output highly similar N-dimensional vector representations for these two animal classes. However, the mean number of attributes the raccoon shares with the train classes is 54, while for the rat it is 56. This means that the train time attribute distribution is more similar to the rat attribute distribution than the raccoon attribute distribution. Thus, the raccoon vectors are very likely to be similar to the groundtruth rat attribute vectors, leading to the high misclassification of raccoons as rats as observed in Figure 2.

Reflection

Because there was no validation set, I was initially training for 60 epochs and saving the network state produced at the end of the 60 epoch iterations, without checking the loss on the test set. This network was heavily overfitted to the training set leading to extremely low performance. I easily worked around this by keeping track of the BCE Loss on the test set and picking the model achieving the minimum test loss instead of the one produced at the last epoch.

Future Work

As observed in the 'Error Analysis' section, there is a good number of animal attributes that are predicted accurately at a rate close to 50%. This implies that the sigmoid values for these attributes convey erroneous information about those attributes of the depicted animal in a large number of cases. I hypothesize that if I use only the animal attributes that the model can predict decently well for, the zero-shot class recognition performance would improve as the feature representation of the images will more accurately reflect the attributes present in the image.

Since this is a test time inference improvement, it was very easy to implement it. I performed the zero-shot inference using the attributes for which the model achieves at least 0, 0.5, 0.55, 0.60, 0.65, 0.70, 0.75, 0.80, 0.85, and 0.90 classification accuracy.

The largest improvement is observed when evaluating using attributes that are classified correctly at least 65% of the time. The L2-distance nearest neighbor classifier achieves 49.78% accuracy. The cosine similarity nearest neighbor classifier achieves 55.02% accuracy, a whopping 10% improvement over the best performance of the previous system. Both classifiers use only 65 out of the 85 attributes to do the classification. This indicates that removing the attributes that are harder to predict correctly provides a representation of the images that is more conducive to zero-shot animal class recognition.

Python Libraries

- **pytorch** for Imagenet [3] pretrained ResNet models [1].
- **scikit-learn** for Nearest Neighbors classification algorithm.
- **numpy** for data preprocessing and loading.
- **matplotlib** for the confusion matrices visualization

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[2] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[3] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li, "Imagenet large scale visual recognition challenge," *CoRR*, vol. abs/1409.0575, 2014. [Online]. Available: http://arxiv.org/abs/1409.0575

[4] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata, "Zero-shot learning - A comprehensive evaluation of the good, the bad and the ugly," *CoRR*, vol. abs/1707.00600, 2017. [Online]. Available: http://arxiv.org/abs/1707.00600