A SIMPLE AUTOMATED IMAGE-CLASSIFIER

Author: J. Horner, jhorner@cybermesa.com

Last modified: 11 November 2019/0800 US Central Time

Environment: *Mathematica* (Wolfram 2019)
Windows 10
Dell Inspiron 545
-- Intel 8200 quadprocessor, clocked at 2.33 GHz
-- 8 GB memory

This *Mathematica* notebook illustrates how to implement a simple automated image-classifier. Among other things, the example shows the importance of providing an automated classifier with a sufficiently large, and sufficiently diverse, universe of examples in the classifier's training phase. If we don't do this, we are likely to get questionable results. If, for example, we "tell" such a classifier that any given object in the universe is either green cheese or beef jerky, the classifier will subsequently identify anything we ask it to classify, as green cheese or beef jerky.

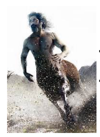
The images in this notebook were chosen from relevant randomly selected online examples, captured in Microsoft Word (tm) using the (keyboard-selectable) function PrtScr, then saved as a bitmap. This capture method eliminates any metadata in the original image that could be used by a classifier to bias classification.

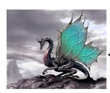
Method and results

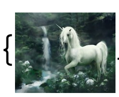
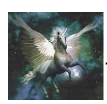
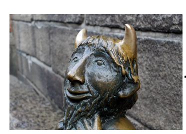
In Mathematica, create and train a classifier for five fantastical creatures, using eight training examples per creature type, letting the Mathematica function **Classify** choose the classification method:

In[10]:= **FantasticalCreatures = Classify[<|**

"Griffin" → {, , ,
, , , , **},**

"Centaur" → {, , , ,
, , , **},**

"Dragon" → {, , , ,
, , , **},**

"Unicorn" → {, , , , , ,
, **}, "Devil" → {**, , ,
, , , , **}|>]**

Out[10]= ClassifierFunction[Input type: Image
Classes: Centaur, Devil, Dragon, Griffin, Unicorn]

Data not in notebook; Store now »

Describe some properties of the classifier.

```
In[11]:= Information[FantasticalCreatures]
```

Out[11]=

Classifier information

Data type	Image
Classes	Centaur, Devil, Dragon, Griffin, Unicorn
Accuracy	(61. ± 13.)%
Method	LogisticRegression
Single evaluation time	373. ms/example
Batch evaluation speed	2.35 examples/s
Loss	0.976 ± 0.21
Model memory	1.14 MB
Training examples used	40 examples
Training time	21.9 s

Apply the classifier to a set of five test images (i.e., images whose classification has not yet been performed by the classifier). For each image and each class, report the probability that the image is in the class.

In[12]:= **FantasticalCreatures**[

{, , , , }, **"Probabilities"**]

Out[12]= {⟨| Centaur → 0.0000379637, Devil → 0.999957,
Dragon → 7.35805×10^{-7}, Griffin → 2.99336×10^{-6}, Unicorn → 1.36183×10^{-6} |⟩,
⟨| Centaur → 1., Devil → 1.73438×10^{-27}, Dragon → 1.75514×10^{-21}, Griffin → 3.25811×10^{-12},
Unicorn → 1.76055×10^{-9} |⟩, ⟨| Centaur → 0.000657658, Devil → 0.000605823,
Dragon → 0.978078, Griffin → 0.00820106, Unicorn → 0.0124571 |⟩,
⟨| Centaur → 2.91468×10^{-27}, Devil → 4.89079×10^{-13}, Dragon → 1.29537×10^{-40},
Griffin → 2.35579×10^{-42}, Unicorn → 1. |⟩, ⟨| Centaur → 1.47235×10^{-14}, Devil → 1.,
Dragon → 8.10348×10^{-30}, Griffin → 2.71738×10^{-22}, Unicorn → 8.4186×10^{-18} |⟩}

Here’s how to read the output of this report. The report is a list of lists. The entire report is delimited by “{...}”, where “...” is an ellipsis for what lies between the outermost report delimiters. A classification-descriptor for a given image is delimited by “<| ... |>”, where “...” stands for what lies within that classification-descriptor. The classification-descriptors are separated by commas. The order of occurrence of the classification-descriptors for each test image, first to last, is the same as the order of the test images, left to right, in the Mathematica instruction that appears immediately above this report. Within the classification-descriptor of a given image, the report states the probability (as determined by the classifier) that the given image is of each given image class. The probability that a given image has a given classification is of the form “*ClassificationCategory* → *probability*”.

For example, consider the the second test image from the left of the classification instruction above. That image is intended to be a centaur. Locate the second classification-descriptor in the output immediately below the Mathematica instruction. That classification-descriptor reads

⟨| "Centaur" → 1., "Devil" → 1.73438×10^{-27}, "Dragon" → 1.75514×10^{-21},
"Griffin" → 3.25811×10^{-12}, "Unicorn" → 1.76055×10^{-9} |⟩

This descriptor means:

The probability that the image is a centaur is 1.
The probability that the image is a devil is 1.73438×10^{-27}.
The probability that the image is a dragon is 1.75514×10^{-21}.

and so on.

Discussion

The classifier, trained on the set shown above, classifies the first image from the left as a devil (arguably incorrect). The classifier classifies the second image as a centaur (correct). It classifies the third image as a dragon (correct), the fourth image as a unicorn (correct), and the fifth image (of Trump) as a devil. The tool's classification of the Trump image shows that the tool will classify an image X as one of the kinds of entities on which the classifier has been trained, *regardless* of whether the training example set is adequate for classifying images of X. This problem is not peculiar to automated image-classifiers: it is likely one of the sources of sensory illusions and tribalisms.

The example above uses eight training examples per classification category. This is likely to be too small to produce reliable results for all classification categories and test images. It is not unusual in machine-learning regimes to use hundreds to thousands of training images per classification category.

References

Wolfram Research. (2019). *Mathematica* Home Edition v12.0.