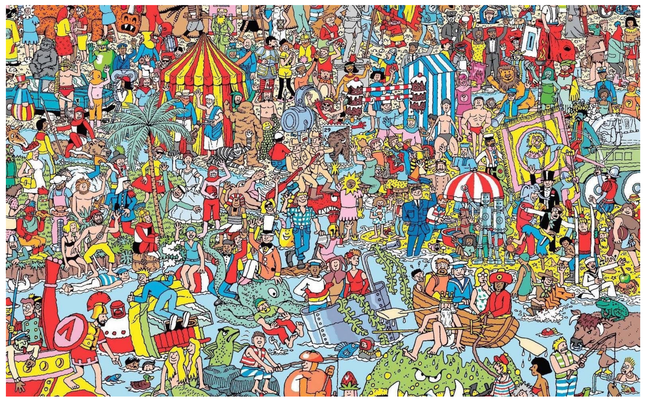
Where's Waldo

Kushagra Singh (2014056), Shrey Bagroy (2014099)
CSE 343: Machine Learning,Monsoon 2016, IIIT-Delhi

Motivation

Where's Waldo (Wally, outside the US) is a popular series of books in which readers are challenged to find Waldo hidden in a "group".

The "group" is usually a huge gathering of people who, both, resemble Waldo(a little) and those who don't.



A sample picture where we have to find Waldo

Clearly, it's not that easy a task, which is why we aim to automate this process and detect Waldo.

The Learning Task & Reason

The learning task is Identifying Waldo given an input image. Where's Waldo has been an enjoyable experience in our childhood(s). It was challenging (frustrating, at times) but worthwhile, which is why we feel it'd be a great idea to automate the process and train a (binary) classifier to find Waldo in a given picture.

Dataset

We came across a set of human annotated images for some of these pictures [1], which we plan to use to train our classifier.

Description of the dataset:
19 Original Images, each of which have Waldo hidden somewhere in them. These 19 images have subdivided into patches to generate the following

- 256 x 256 pixels (317 patches)
- 128 x 128 pixels (1244 patches)
- 64 x 64 pixels (5376 patches)

One concern which we have regarding the dataset, is that it is imbalanced. The number of positives (images having Waldo) is quite less in comparison to the negatives (images not having Waldo).

We plan to resolve this issue by some common balancing techniques like sampling or synthetically generating images.

Preprocessing techniques to be explored

- Feature extraction; we will explore Image Analysis techniques like extracting the histogram, looking contrast/brightness etc.
- Since our dataset is imbalanced (a lot of negative class instances), we will explore techniques to balance this dataset like creating synthetic data points, sampling and penalising the model.

Learning Techniques

- Naive Bayes Classifier (Bernoulli and Gaussian)
- Multilayer Perceptron
- Convolutional Neural Networks

We will use Stochastic Gradient Descent as the training approach for the Neural Networks. We plan to use the python library Scikit-Learn[2] and TensorFlow[3] for our experiments.

Evaluation Metrics

- Accuracy (TP + TN)/Total
- Precision TP/(TP + FP)
- Recall TP/(TP + FN)
- F1-Score 2TP/(2TP + FP + FN)
- AUC Value/ROC curve

TP : True Positives TN : True Negatives
FP : False Positives FN : False Negatives

References

1. Dataset: https://github.com/vc1492a/Hey-Waldo
2. Scikit-Learn: http://scikit-learn.org
3. TensorFlow: https://www.tensorflow.org