Depth of Field Prediction

Ζούρος Μιχάλης

Problem to Solve

- Image classification to Deep and Shallow Depth of Field images from RAW images
- Implementation in Python3 with the use of various libraries (OpenCV, Numpy, Matplotlib/Seaborn, Sklearn/Keras)

- Github: https://github.com/mzouros/ml_dof

Data Collection

- Toy Dataset from personal portofolio
- Main Dataset downloaded from https://github.com/sniafas/photography-style-analysis/tree/main/dataset (over 25k images)

Many thanks to @tygiannak and @sniafas for the data supply

Data Preparation & Pre-processing

- Handcrafted preparation of the Training/Test datasets
- 500 images labeled as Deep(0) or Shallow(1) DoF
- 80/20 ratio (Training: 400 / Test: 100)
- Input dataset (RAW images)

- Image Standardization - resize images to 200x200

Data Exploration

- Changing Colorspaces (GBR, RGB, Grayscale)
- Geometric Transformations (Scale, Rotate)
- Smoothing (Blur, GaussianBlur)
- Thresholding
- Edge Detection
- Morphological Transformations (Open, Gradient)
- Histograms (RGB, Grayscale, Edges)
- Gradients (Sobel, Laplacian)

Feature Extraction

- For kNN, SVC:
  - Load images as 16-bit
  - Remove noise by blurring with a Gaussian filter
  - Convert the image to grayscale
  - Apply Laplace function
  - Convert images back to 8-bit
  - The output of this process is a features vector

- For CNN there is no need for Feature Extraction

Data Revision & Augmentation

- Consider image size
  - from 100x100 back to original size (200x200)

- Consider more data
  - Augment dataset via Image Transformation (Rotation)
  - from 200 images to 500

Classification

- Fit Models:
  - kNN
  - SVC
  - CNN
- Train / Test Accuracy Difference
- Metrics:
  - Accuracy
  - Recall
  - Precision
  - F1 Score
  - Confusion Matrix
- Average & Standard Deviation of all Metrics (kNN)
- Precision / Recall and ROC Curves (SVC)

Results - kNN

- Algorithm = auto (same results for all)
- k = range(1, 15)

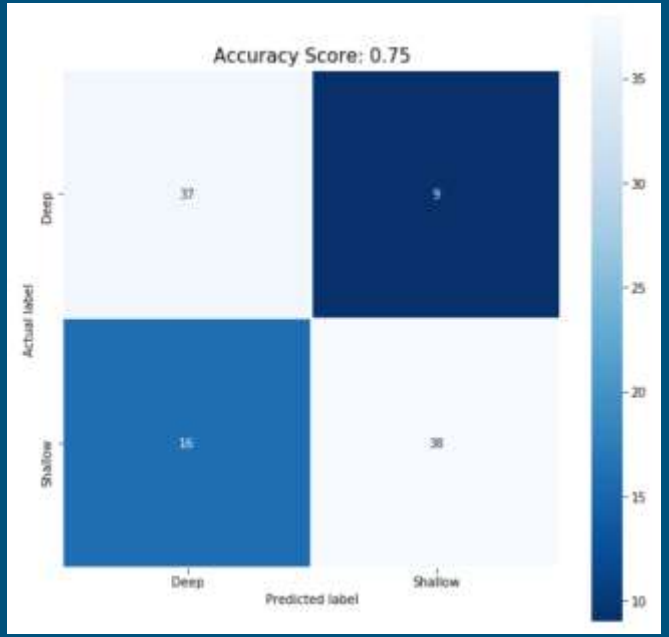
```
==========================================================
~~~~~~~~~~~~~~~~~~~~~~~~ MEAN ~~~~~~~~~~~~~~~~~~~~~~~~~~~
Average Accuracy: 0.64
Average Precision: 0.76
Average Recall: 0.52
Average F1 Score: 0.60
~~~~~~~~~~~~~~~~~ STANDARD DEVIATION ~~~~~~~~~~~~~~~~~~~~
Standard Deviation of Accuracy: 0.04
Standard Deviation of Precision: 0.12
Standard Deviation of Recall: 0.12
Standard Deviation of F1 Score: 0.04
==========================================================
```

Results - SVC

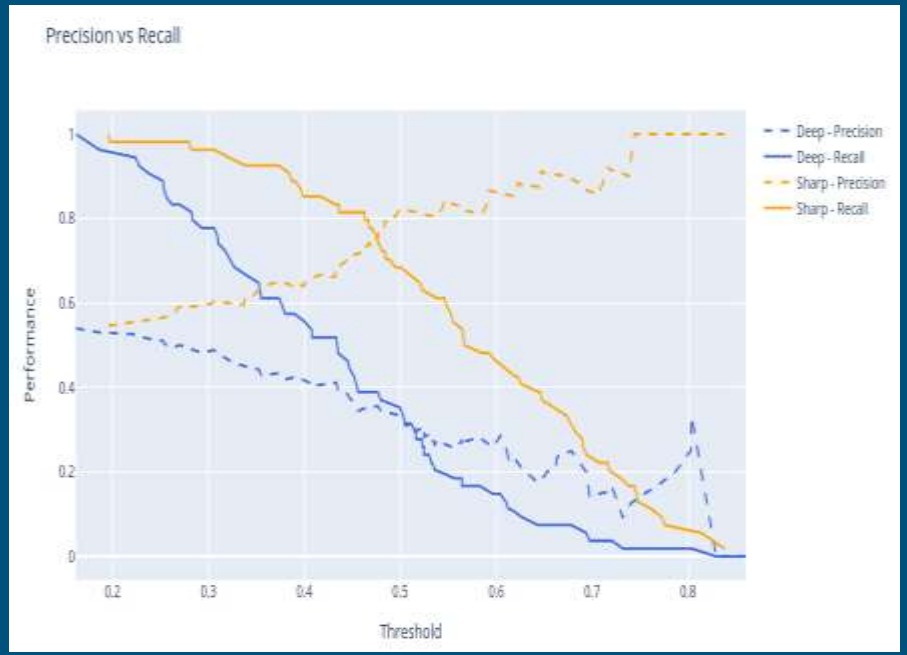
- kernel = ‘rbf’
- C = 2, gamma = ‘scale’

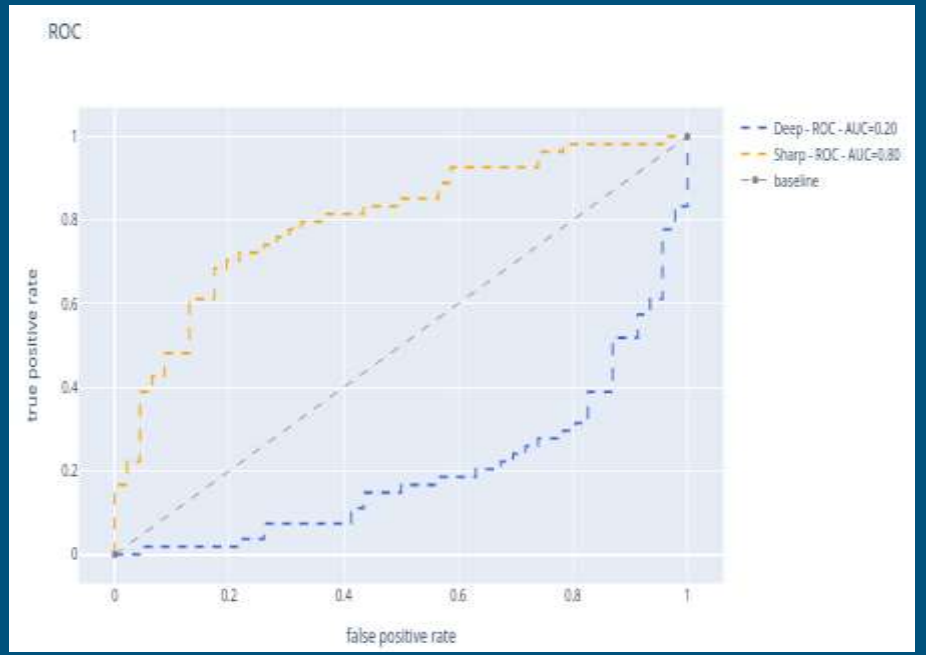
Accuracy: 0.75
Precision: 0.8085106382978723
Recall: 0.7037037037037037

	precision	recall	f1-score	support
0	0.70	0.80	0.75	46
1	0.81	0.70	0.75	54
micro avg	0.75	0.75	0.75	100
macro avg	0.75	0.75	0.75	100
weighted avg	0.76	0.75	0.75	100



Precision/Recall & ROC Curves





Discussion

- Problems during implementation:
  - DoF is a concept closely related to the 3D world and trying to identify it inside a 2D image with traditional ML methods proved challenging
  - DoF estimation (for labeling) in 200x200 size images was eye-hurting
  - OpenCV library's default colorspace is BGR. Extra transformations needed to plot
  - Bias -> shallows are humans/animals/plants, deeps are landscapes and many are from top down angle
  - White background TOP DOWNs and deep DoF images which include sky and/or sea reflection were becoming a noise for our Deep dataset after the Laplacian Filter

Future Work

- Different libraries like Mahotas, which provides more advanced features such as haralick, local binary patterns, etc
- Different algorithms, like Decision Trees (..but, CNN FTW)
- RGB, HSV instead of Grayscale
- Bigger Dataset (1000+ images)
- Bigger image size (maybe 600x600)
- SMOTE for Dataset’s balancing

References

- https://towardsdatascience.com/from-raw-images-to-real-time-predictions-with-deep-learning-ddbbda1be0e4
- https://medium.com/swlh/image-classification-using-machine-learning-and-deep-learning-2b18bfe4693f
- https://github.com/hasabo/Machine-Learning/blob/master/Python/Image_Classification%20/Image_Classification.ipynb
- https://www.analyticsvidhya.com/blog/2019/08/3-techniques-extract-features-from-image-data-machine-learning-python/
- https://freecontent.manning.com/the-computer-vision-pipeline-part-4-feature-extraction/

Thank you!


IMAGE
CLASSIFICATION TASK
TRADITIONAL
ML
ALGORITHMS
DEEP LEARNING
imgflip.com