Computational Social Science with Images and Audio

Elliott Ash, **Philine Widmer**

29 September 2023

For classical ML, we (often) extract features explicitly.

- Recall the typical pipeline for classical machine learning:

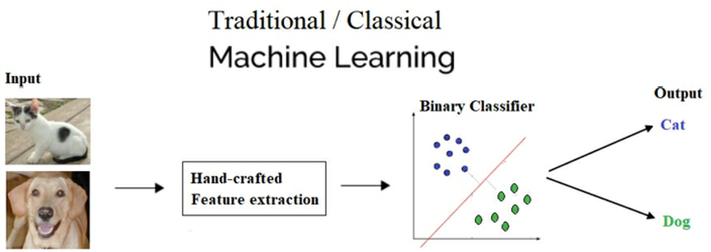


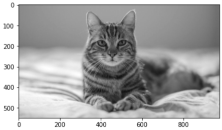
Figure: Dey (2018)[1]

[1] Dey, S. (2018). Hands-On Image Processing with Python: Expert Techniques for Advanced Image analysis and Effective Interpretation of Image Data. Packt Publishing Ltd.

Can we run a direct pixel-based analysis with classical ML?

```
array([[131, 131, 131, ..., 107, 106, 106],
       [131, 132, 132, ..., 107, 107, 106],
       [132, 132, 133, ..., 108, 107, 106],
       ...,
       [192, 192, 192, ..., 189, 187, 185],
       [193, 193, 193, ..., 189, 187, 185],
       [194, 194, 194, ..., 188, 186, 184]], dtype=uint8)
```

Figure: Can we directly provide our model with an array like this (above)?



Sometimes, we can use raw pixel values as features...

- Generally, images are typically flattened: $2D \rightarrow 1D$
- **Uniform image size** is required (no varying dimensions)
  - (Applies to many deep learning models, too, not just classical ML)
- Grayscale images may be preferable (each pixel represented by a single intensity value)
  - With multiple channels, high(er) dimensionality!
- Centered objects can help, as well as objects occupying most of the image (i.e., not much background irrelevant to the task)
- Depends on the task (simple vs. complex features)
- Can you think of an example?

There are MANY options for feature extraction, such as...

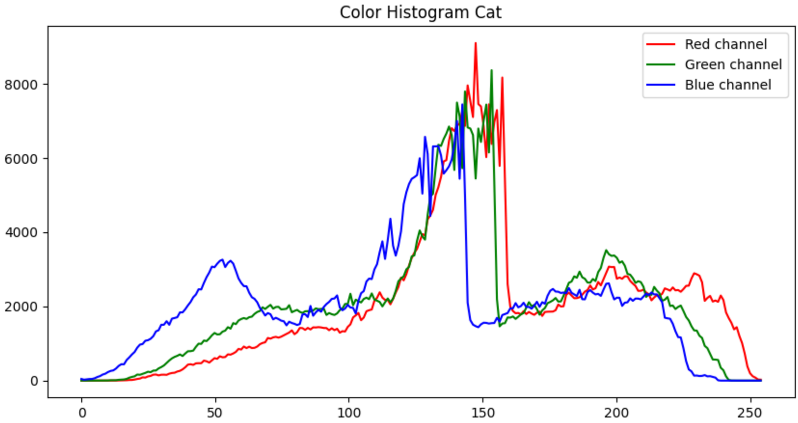
- **Color Histograms:** Represents the distribution of color values in an image.
- **Texture Features:** Calculate texture features like entropy, energy, contrast, and homogeneity from the grayscale co-occurrence matrix.
- **Histogram of Oriented Gradients (HOG):** Compute the histogram of gradient orientations in localized portions of the image.
- **Haar-like Features:** Use integral images to efficiently compute features that recognize edges, lines, and rectangles for object detection.

$\rightarrow$ Different feature types provide a different kind of information about the image content, and the best one to use depends on the specific task at hand!

Color histograms show the pixel intensity distribution

- Represents the distribution of pixel intensity levels
- Mathematically, $h(r_k) = n_k$ where h is the histogram, r_k is the k-th intensity value, and n_k is the number of pixels with intensity r_k
- Typical use cases: image retrieval, color-based segmentation
- What color histogram do you expect for the cat?

This cat's histogram shows the frequencies for each channel (RGB).



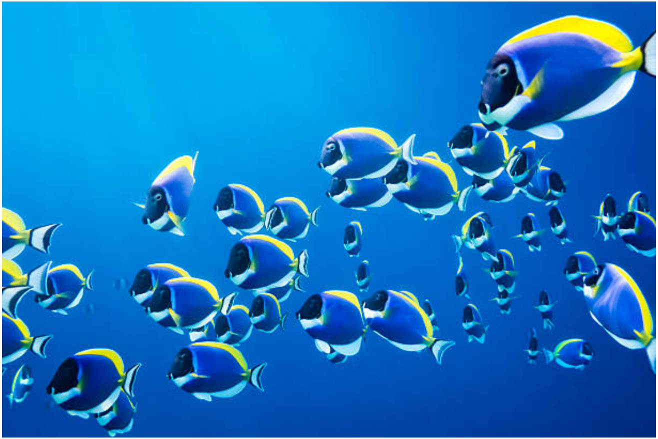
Texture features reveal image complexity and uniformity.

- Texture features are typically based on the grayscale co-occurrence matrix P
  - This matrix contains the occurrences of pairs of grayscale values *at a given offset*
- One example is entropy
  - It is defined as: $E = -\sum_{i,j} P(i,j) \log(P(i,j))$
  - $P(i,j)$ is the probability of co-occurrence of grayscale values i and j
- Typical use cases: material classification, defect detection

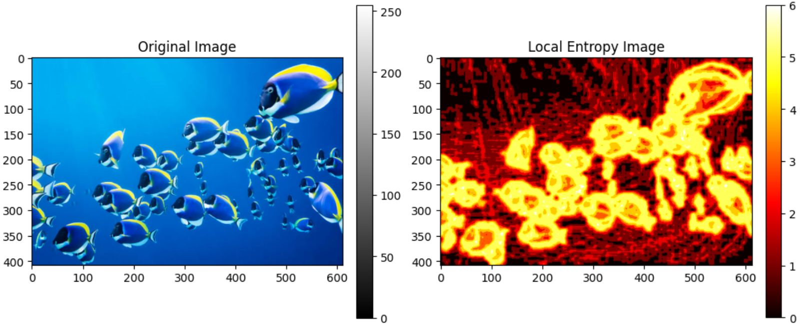
Entropy is an example of a texture feature.

- Entropy measures the amount of information or randomness in the image region
  - *Higher* entropy indicates a *more complex* texture with more details
  - *Lower* entropy indicates a *more uniform* texture with fewer details
- Example for local entropy:
  - Consider a 3×3 window in an image
  - Calculate the grayscale co-occurrence matrix for that window
  - Subsequently, calculate the entropy

What heatmap of local entropy is expected for this image?



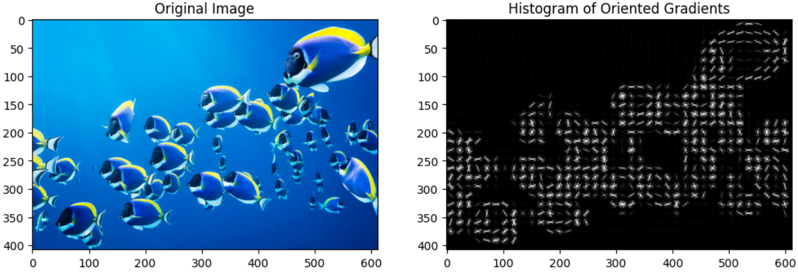
Local entropy is higher for the fish than the water.



Histogram of oriented gradients (HOG) features describe gradient distributions.

- HOG features capture the distribution and intensity of gradients in an image
- Computed by dividing the image into cells and creating a histogram of gradient directions for the pixels within each cell
  - Each pixel in the cell "votes"
- The gradient magnitude is used as a weight, emphasizing pixels with large gradients
- For intuition: for a given cell (of, say, 16 pixels), in which direction does the intensity change the most? And how significant is this change?
- Typical use cases: Human detection, object recognition

The HOG features hint at the fish.



Haar-like features can help to identify faces.

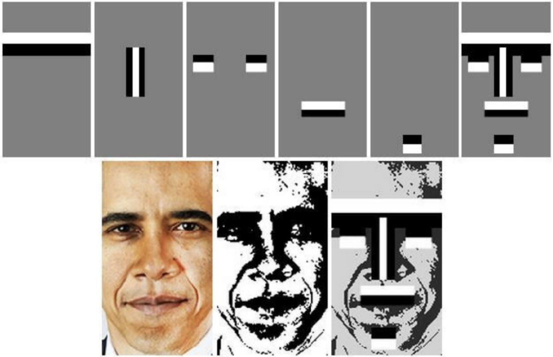


Figure: Kadir et al. (2014): an application of Haar-like features[2]

[2] Kadir, K., Kamaruddin, M. K., Nasir, H., Safie, S. I., and Bakti, Z. A. K. (2014). A Comparative Study between LBP and Haar-like Features for Face Detection using OpenCV. 4th International Conference on Engineering Technology and Technopreneuship (ICE2T) (pp. 335-339). IEEE.

An image-based analysis pipeline looks like other classical ML-based pipelines.

(This example assumes a supervised problem.)

1. **Data collection and pre-processing**
   - Gather a labeled dataset and pre-process the images (e.g., resize, convert to grayscale, normalize)
2. **Feature extraction and selection**
   - Extract relevant features (e.g., texture, color, edges, key points) and choose the relevant ones (or reduce dimensionality)
3. **Model training**
   - Train a model with the selected features (e.g., SVM, random forest)
4. **Model evaluation**
   - Use metrics and cross-validation to assess model performance
5. **Model deployment and prediction**
   - Deploy the trained model for making predictions on (new) images