Feature Selection and Basic Pattern Recognition Algorithms

PLAKSHA UNIVERSITY

Statistics Revisit

$$Cov\,(x,y) = \frac{\sum_i^n (x_i - \overline{x}) \cdot (y_i - \overline{y})}{N-1}$$

Sample Covariance for two independent variables (features) x & y

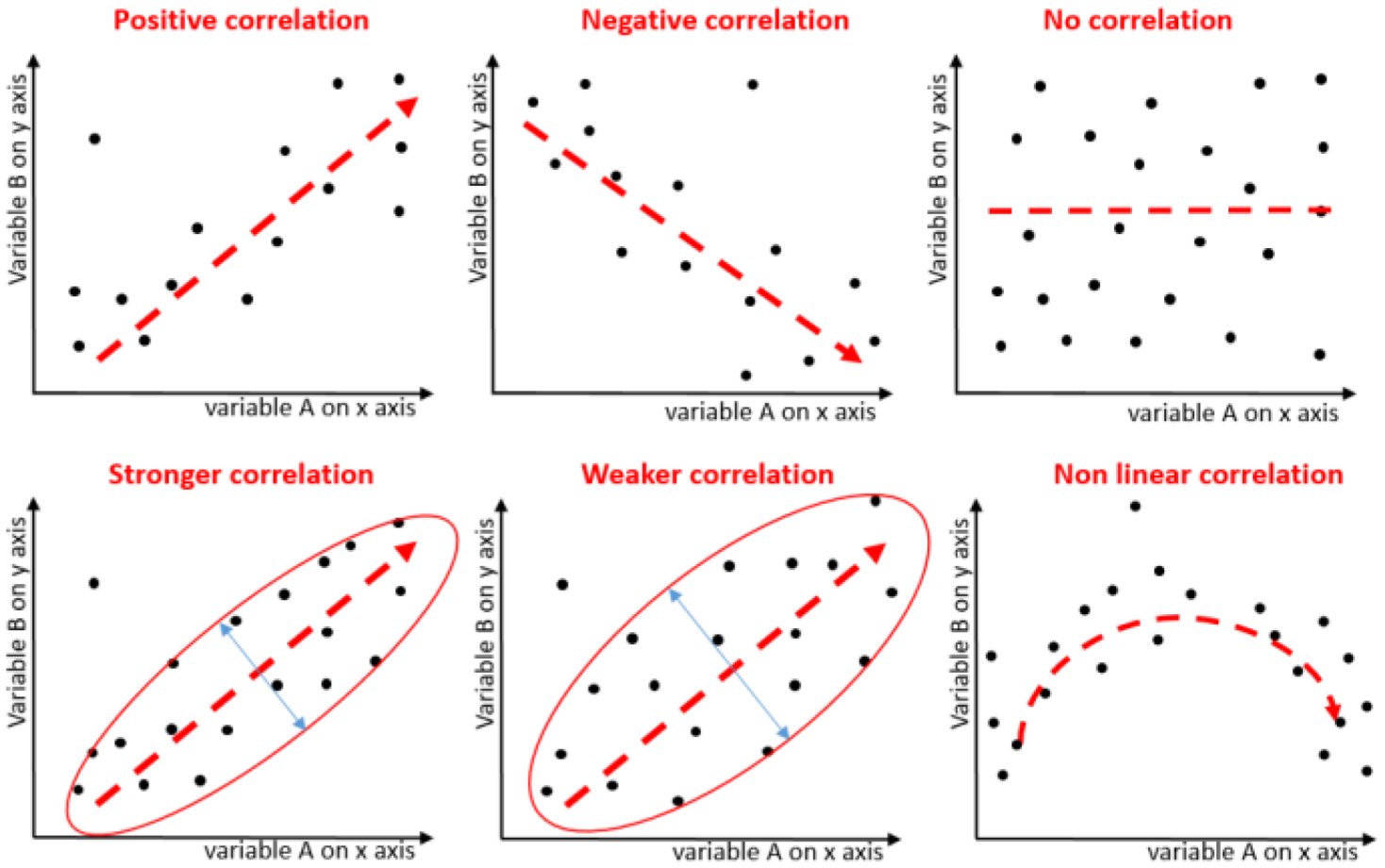
$$Correlation = \frac{Cov\,(x,y)}{\sigma x * \sigma y}$$

Pearson Correlation Coefficient for two independent variables (features) x & y

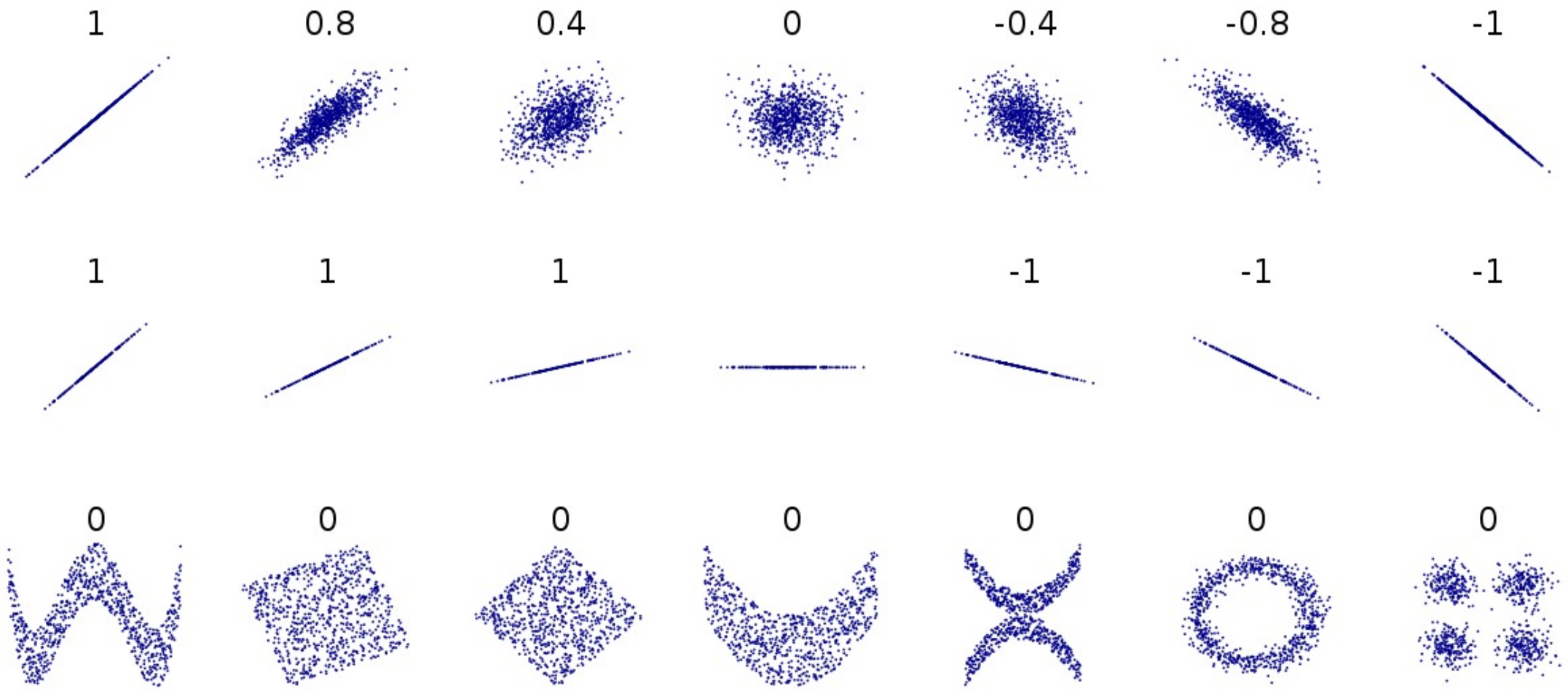
$$Correlation = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{\left[n\Sigma x^2 - (\Sigma x)^2\right]\left[n\Sigma y^2 - (\Sigma y)^2\right]}}$$

PLAKSHA UNIVERSITY

Correlation between features



Correlation Coefficient



Feature Selection

Let's say we have extracted many features from the data.

- Not all features may be equally good at having useful information about the problem at hand.

- Some features may have redundant information already present in other features.

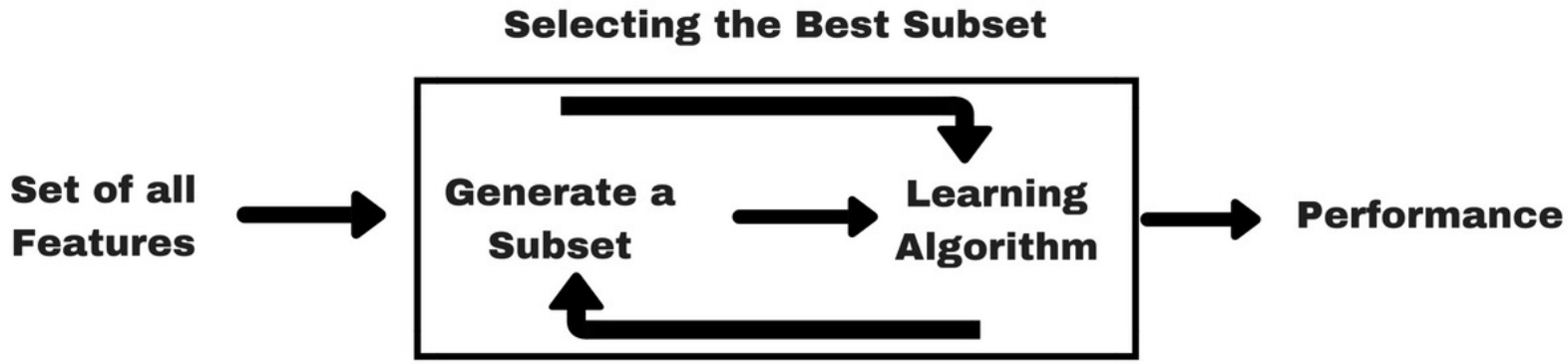
So, how do we select the best?

There are many methods proposed in the literature. They are all quite dependent on the dataset. Let's discuss three here.

Feature Selection

a) Wrapper method

Keep adding or keep dropping features from the features set till you get the best performance.

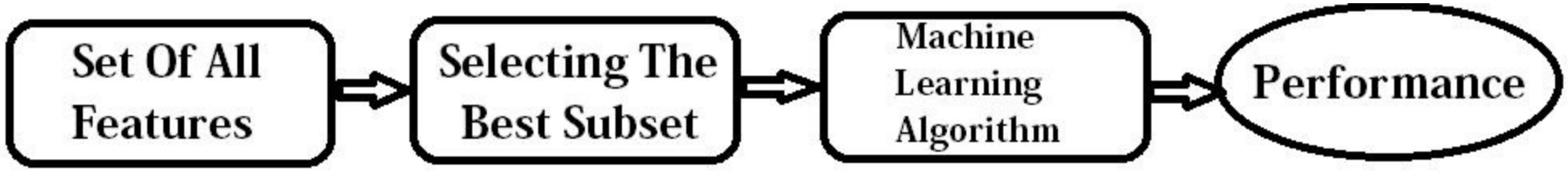


Obviously not a great method if you have hundreds or thousands of features! It will arrive at the best subset of features but could be computationally very expensive.

Feature Selection

b) Filter method

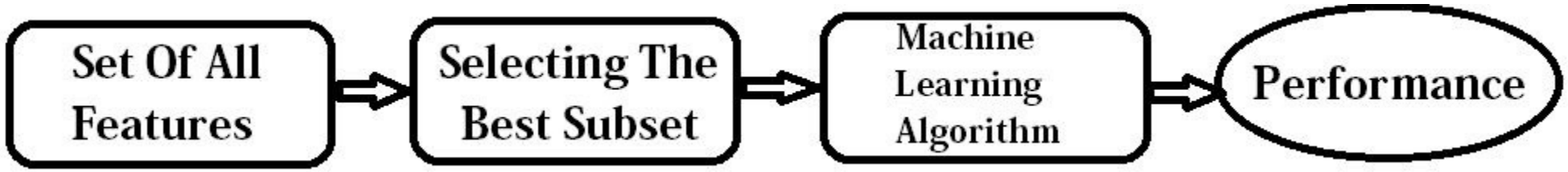
Uses statistical techniques to evaluate the relationship between each feature and the target (dependent) variable.



The best subset is selected by computing a statistical measure (such as the Pearson Correlation Coefficient) between each feature and the target variable and choosing K (user-provided) best features with highest absolute correlation values.

Feature Selection

b) Filter method



Instead of Pearson Correlation Coefficient, chi-squared test may also be used.

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

where:

c = degrees of freedom

O = observed value(s)

E = expected value(s)

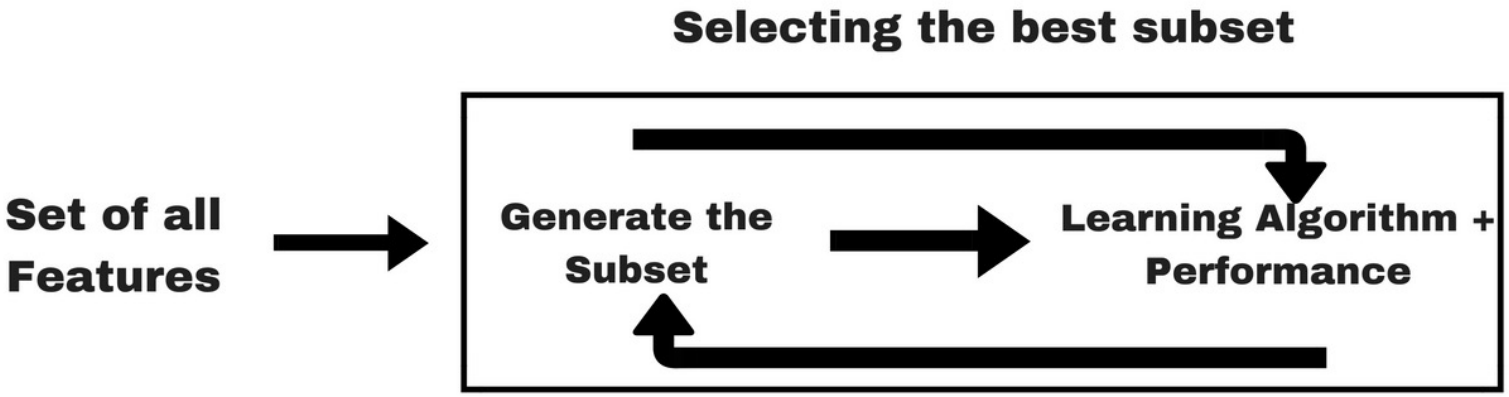
We select the feature if the null hypothesis is rejected ($p<0.05$) since the feature is independent. Else, we skip the feature.

Filter method may unfortunately provide many features conveying similar kind of information but it is faster than Wrapper method.

Feature Selection

c) Embedded method

Combination of Wrapper and Filter methods.

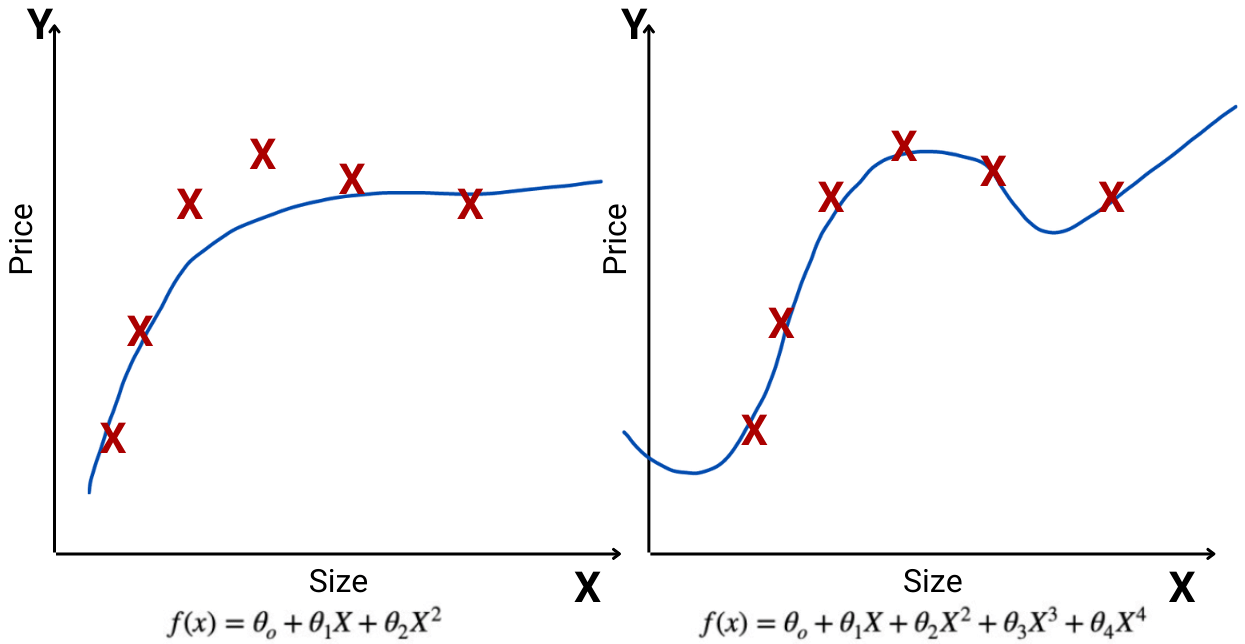


Fast like Filter method and finds good independent features without much information overlap between them like Wrapper method.

Feature Selection

c) Embedded method

L1 Regularization (LASSO Regression)



$$Cost = \sum_{i=0}^{N} (y_i - \sum_{j=0}^{M} x_{ij} W_j)^2 + \lambda \sum_{j=0}^{M} |W_j|$$

The method uses a cost function to penalize the irrelevant features by shrinking their coefficients (θ) to 0.

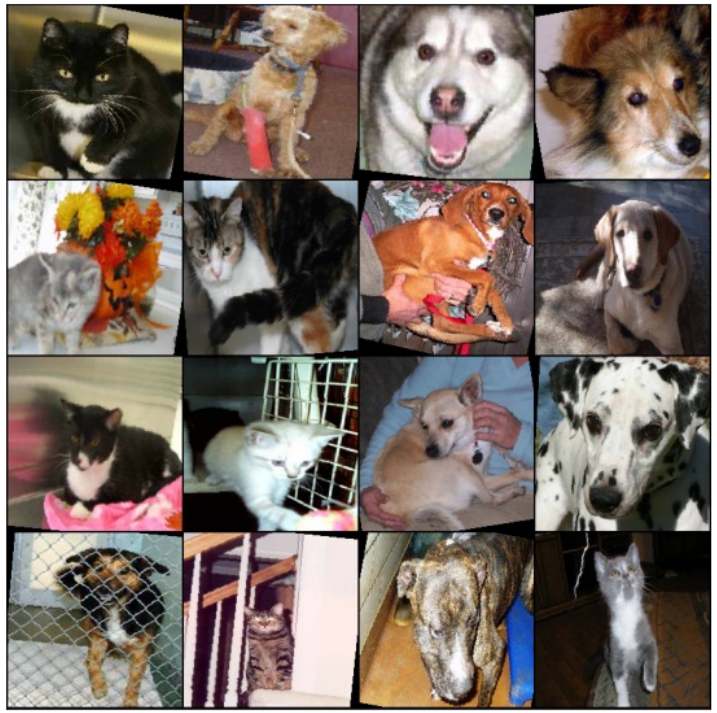
In this way, it removes irrelevant features from the feature set and creates a subset of relevant features.

Graphic courtesy: https://www.enjoyalgorithms.com/blog/regularization-in-machine-learning

Do we always need to extract features?

Short answer: **No**

Many deep learning algorithms do not require any feature extraction (and thus no feature preprocessing or selection is needed). They are quite helpful for complex problems where hand-crafted features do not work. For example:



Distinguishing between cats and dogs in these pictures would be quite hard based on features like color, size, fluffiness, etc.

Deep Learning algorithms automate the process of feature extraction. We will encounter them later in the semester.

Template Matching for Object Detection

Your first Pattern Recognition algorithm using Pearson Correlation Coefficient!

Template Image

$T(x_t, y_t)$
Value of a Template Image pixel

where, (x_t, y_t) are the coordinates of the Template Image

Search Image

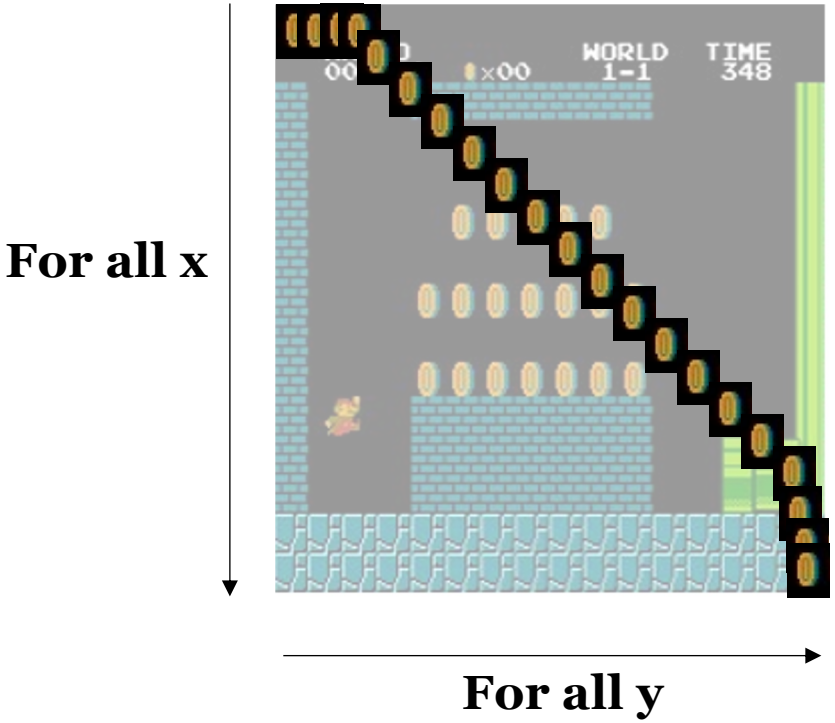
S(x, y)
Value of a Search Image pixel

where, (x, y) are the coordinates of the Search Image

Template Matching for Object Detection

To find the *Template Image* in the *Search Image*, we just need three steps:

1) Simply move the *Template Image* over the *Search Image* for all (x, y).

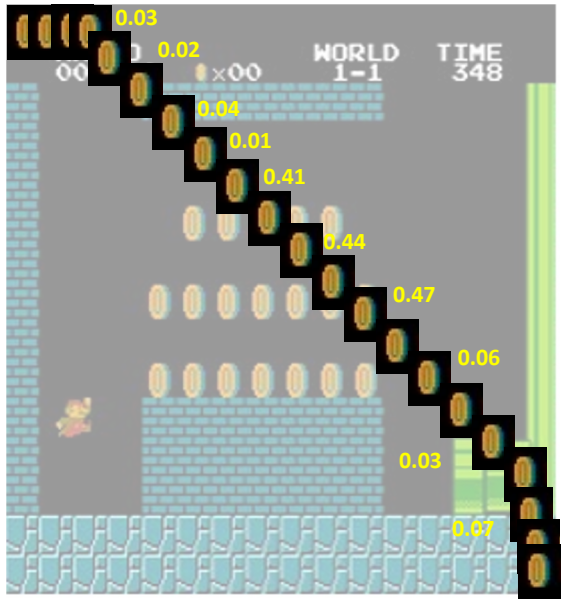


Template Matching for Object Detection

To find the *Template Image* in the *Search Image*, we just need three steps:

2) For each step in 1), find the Cross-correlation between the *Template Image* and the corresponding part of the *Search Image*.

$$(T \star S)(x, y) = \sum_{(x_t, y_t) \in T} T(x_t, y_t) \cdot S(x_t + x, y_t + y)$$



Template Matching for Object Detection

To find the *Template Image* in the *Search Image*, we just need three steps:

3) If the Cross=correlation > Threshold, label (x_i, y_i) in the *Search Image* as a location of the *Template Image*.

Template Image

Search Image

All (x,y) where threshold > 0.9

Template Matching for Object Detection

To handle the case with different sizes of template image and object in search image, one can iterate over the search image by taking progressively larger/shorter search images. Label the object as detected if you exceed a threshold in any iteration.

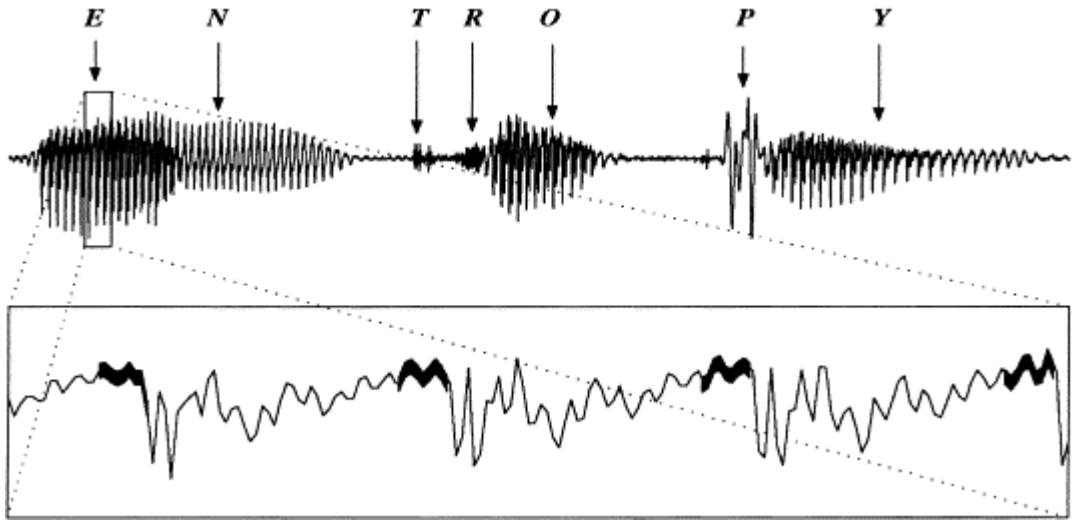
Template Image

Search Image

Template Matching for Audio

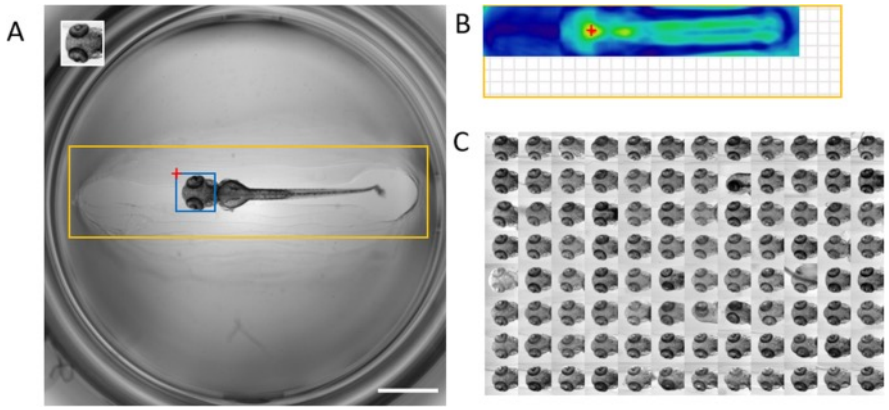
- Just like images, template matching can also be used on audio data to identify characters in the speech.

Speech

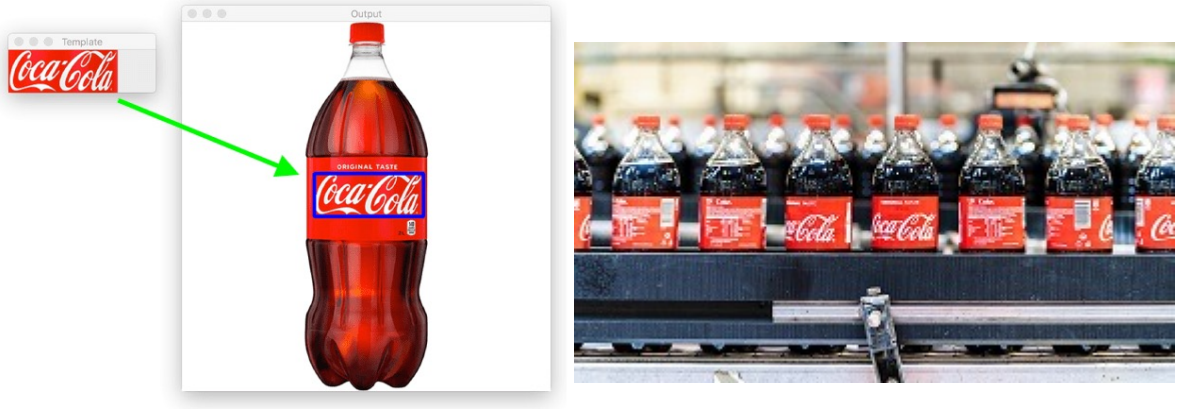


Template of the "E" sound

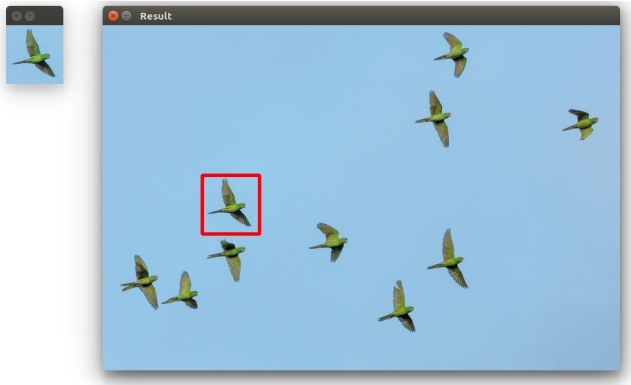
Template Matching Applications



Object Localization in Microscopy Images



Counting and Sorting in Industry



Video Surveillance Systems

Optical Character Recognition (OCR)

Template Matching Limitations

Template Matching works well when the *Template Image* is present "exactly" in the *Search Image*.

1) However, Template Matching is unable to account for any design alterations such as change in font, artistic patterns, etc.

Template Images

Search Image

Template Matching

Template Images

Search Image

Template Matching

Template Matching Limitations

2) Template Matching is unable to account for any alterations in perspective, rotation, etc.

Template Image

Search Image

Template Matching

Template Image

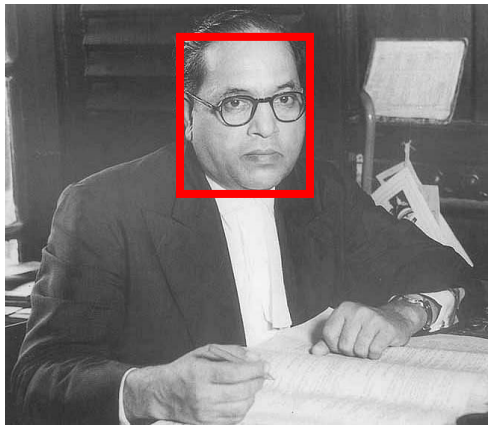
Search Image

Template Matching

Template Matching Limitations

3) Template Matching cannot translate from one object (such as face) to another

Template Image

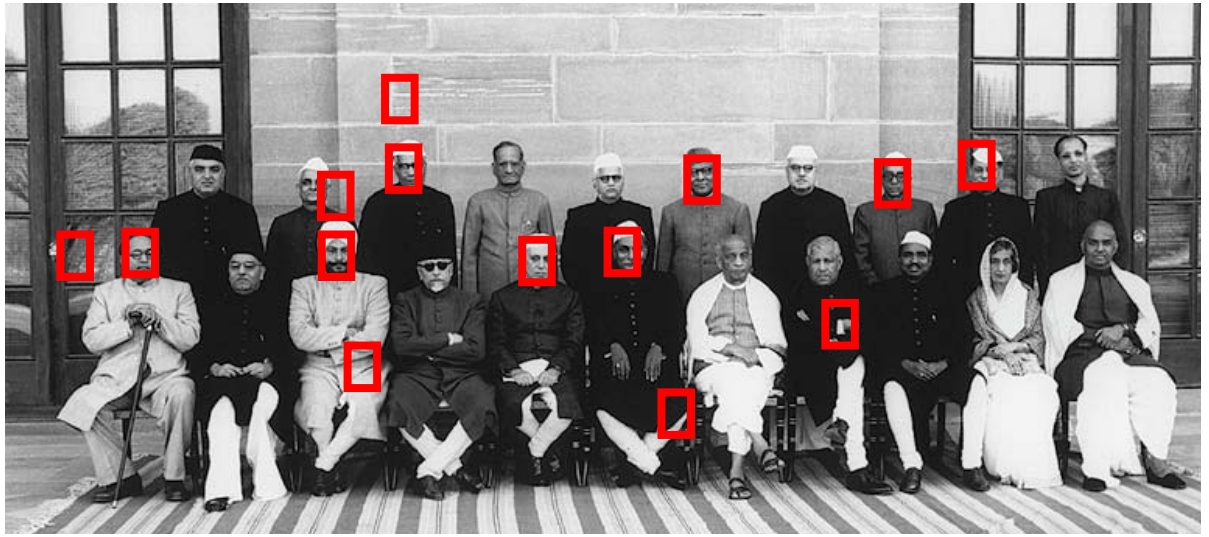


Search Image

✓

Template Matching

Template Image



✗

Template Matching

How to make Object Detection robust?

That's where Features come in from our previous lecture.

Extracting useful features from the *Template Image* allow us to represent it in a more robust form. These features represent information from the *Template Image* such that variations due to size, rotation, angle, etc. could be accounted for.

We cannot always "handcraft" features like we did in the last lecture during the class demos. Today, let us look at a more advanced feature representation.

Template Image

HOG Feature Representation

HOG Features

Histogram of Oriented Gradients (HOG) features. They can be computed in four steps:

1) Take a block of N x N pixels in the image. Iterate over the length and breadth of the image computing gradients in x and y directions.

$$G_x(r,c) = I(r,c+1) - I(r,c-1) \quad G_y(r,c) = I(r-1,c) - I(r+1,c)$$

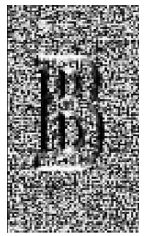
where r, c refer to rows and columns respectively

2) Calculate magnitude and angle of each pixel in the image using the above gradients.

$$Magnitude(\mu) = \sqrt{G_x^2 + G_y^2} \quad Angle(\theta) = |\tan^{-1}(G_y/G_x)$$

Original Image

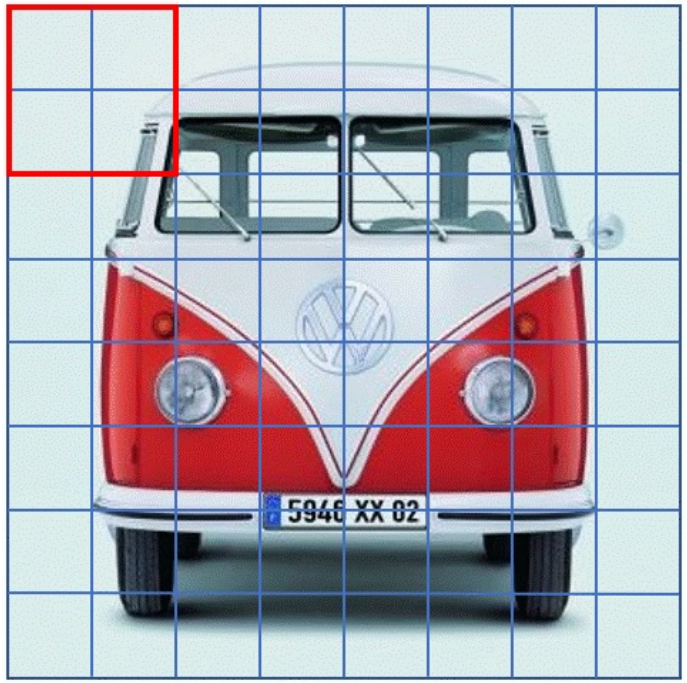
Magnitude Image



Angle Image

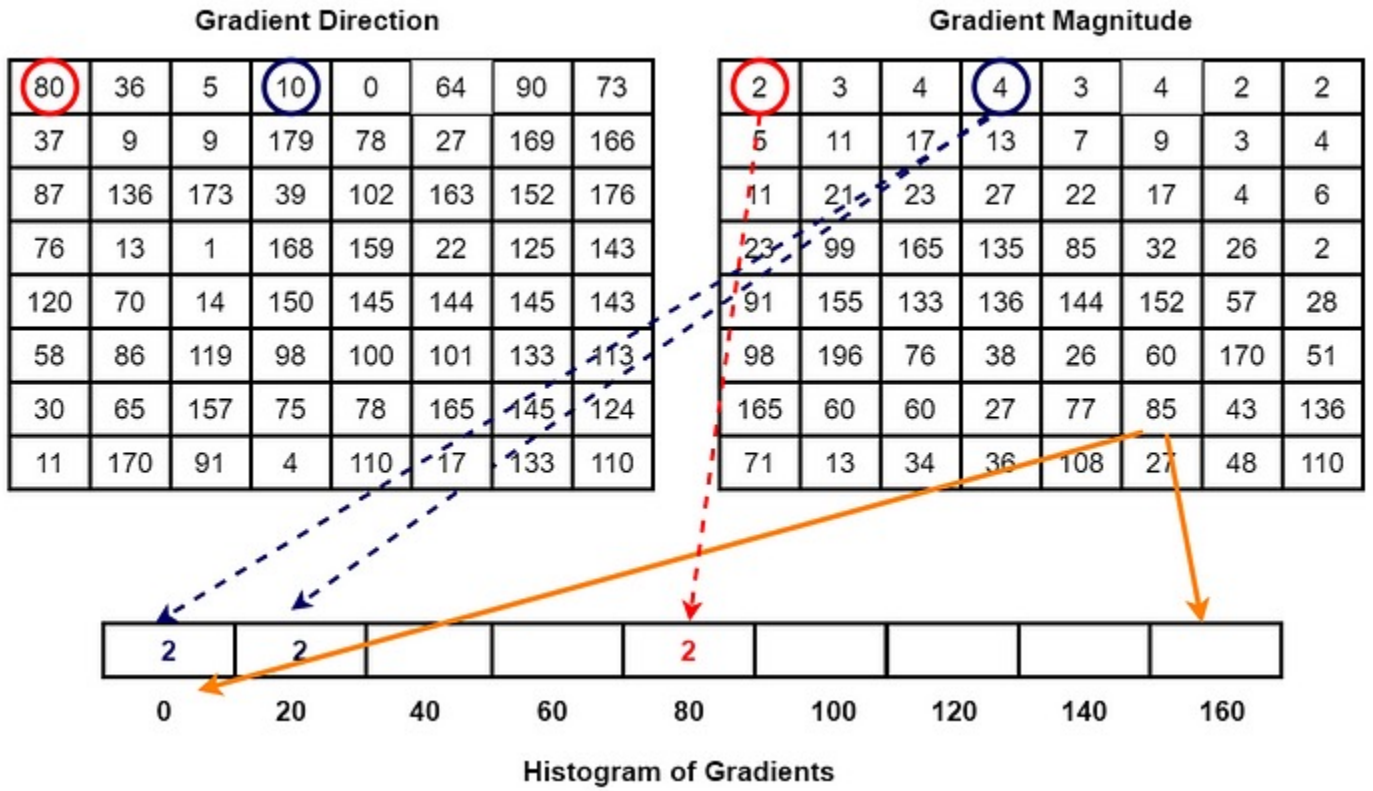
HOG Features

3) Use the magnitude and angles to generate a histogram for each bin in the image.



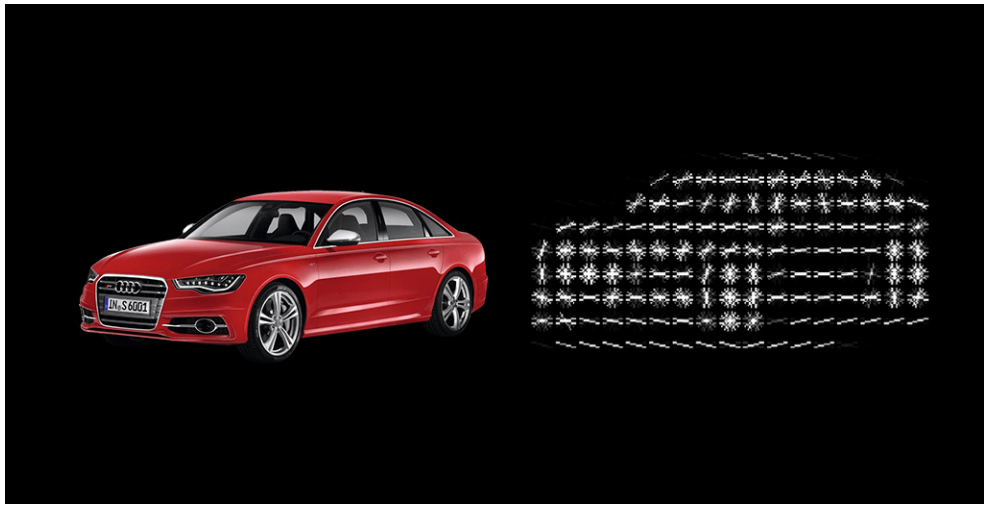
HOG Features

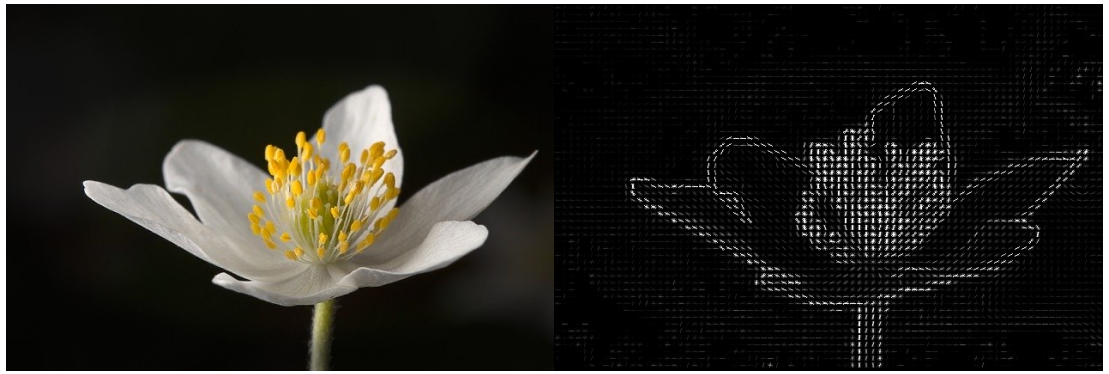
4) Use the magnitude and angles in four neighboring histogram bins to generate a Histogram of Gradients matrix.



HOG Features

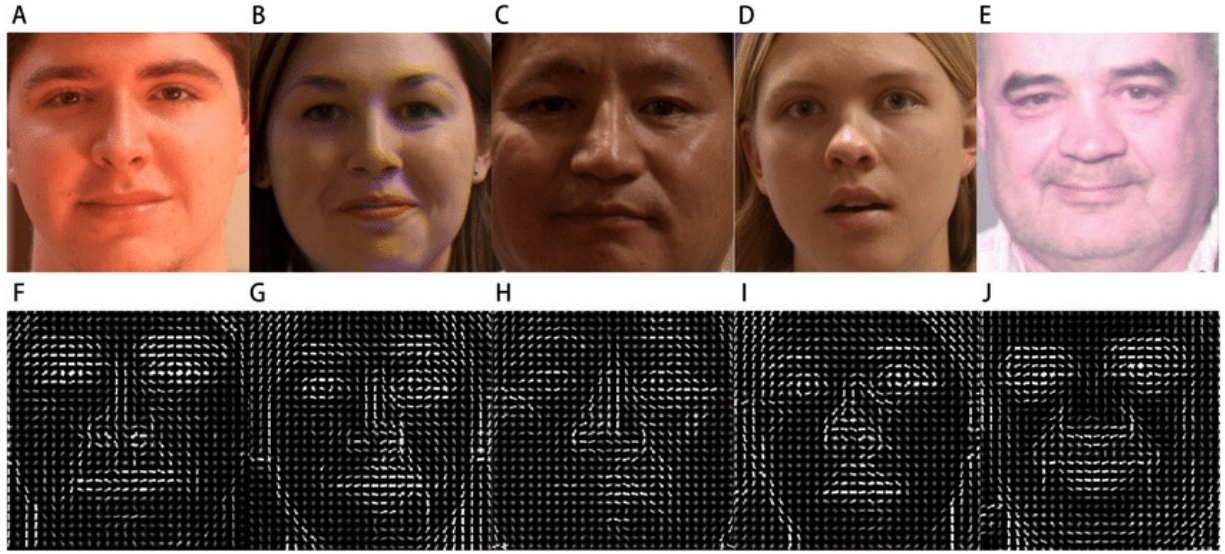
HOG features have the capability to represent the shapes, edges, and angles of various objects in images.



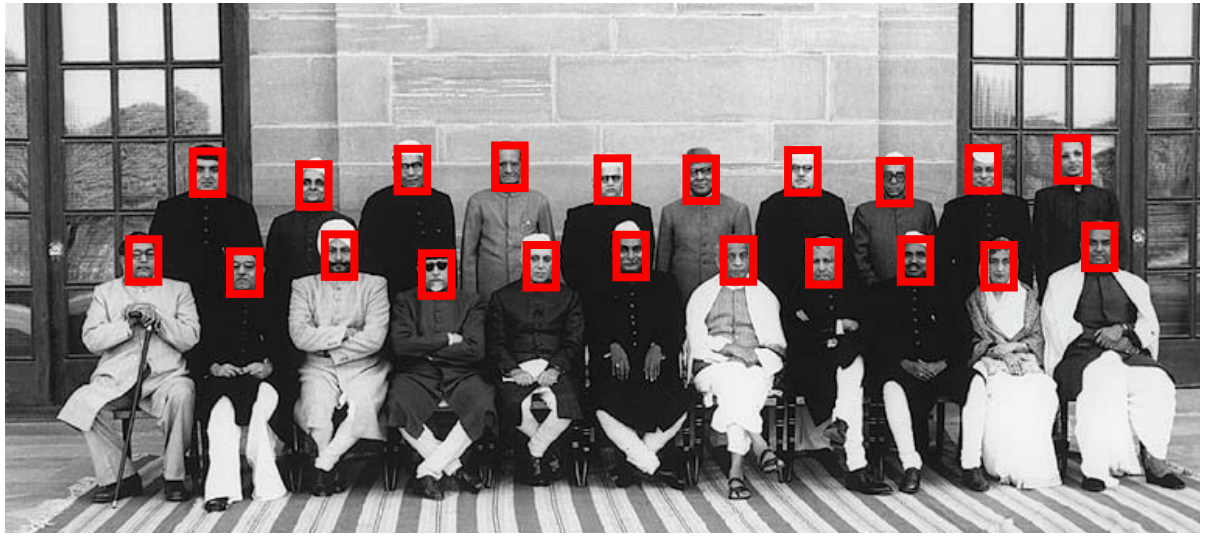


HOG Features-based Object Detection

HOG features can be used for more robust object detection than the Template Matching method because HOG features encompass more useful information to identify objects even with some variations.



Create a database of HOG features for many human face images



Using HOG features-based approach for face detection

Summary

This lecture

- Template Matching – advantages and limitations.

- Advanced Feature Extraction example – HOG Features

- HOG Features-based Object Detection

Next lecture

- Now that we know what are ML features and have seen how can we use them, how can we cluster unlabeled data using features?

- How do we form optimal clusters of unlabeled data using features?

Questions?