Introduction

Image classification is a widely researched topic due to its complex nature and wide domain of applications. In recent years, a variety of supervised learning models have been proposed to tackle the many different types of image datasets. Some examples of the different image datasets that can be constructed are grayscale images, high resolution images, low resolution images, images with multiple objects and images with contextual information.

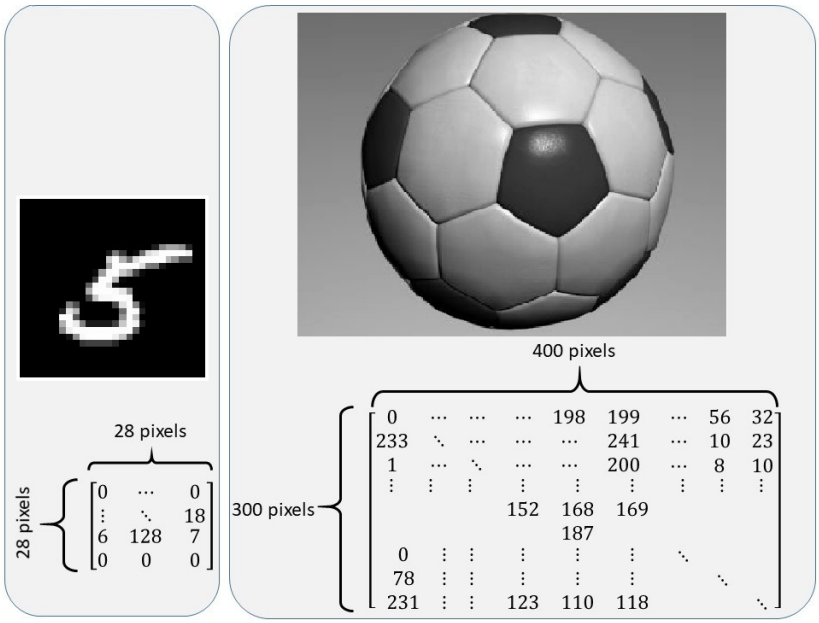
In this report, we explore a variety of supervised learning techniques for image classification. At the heart of our models are image feature descriptor algorithms from computer vision and image processing research. These algorithms are combined with dimensionality reduction and classification techniques such as PCA, SVM; and the strengths and weaknesses of each model are analysed separately for high and low-resolution image data sets. In the end, we propose a mixture of models technique that is able to classify both high and low-resolution datasets with greater than 80% accuracy.

Literature Review

We begin by summarizing some significant recent developments in image classification. In 2012, Noridayu et al. proposed a supervised object recognition methodology that used shape-boundary based features alongside shape-interior information. Using an SVM classifier they were able to report classification accuracy of 80% in feature rich image data. In 2007, Fei-Fei et al. proposed a technique to extract both objects and scenes from images. Scenes from images were constructed by extracting local features. By using the combination of categorized objects and scenes, this model was able to classify images with complex contexts or multiple objects accurately. For example, if the image comprised of the event "rowing", the objects identified with this approach would be boat, person, tree or water; and the scene identified would be lake or sporting competition. This model was able to identify the events in complex images at 73.4% accuracy. A variety of Convolutional Neural Networks and Deep-CNN architectures have also been explored in image classification. The famous CNN architectures: Yann LeCun's LeNet was able to report 99.1% accuracy on handwriting data; Alex Krizhevsky et al.'s AlexNet was able to report 84.7% accuracy on a subset of the ImageNet data; Szegedy et al.'s GoogLeNet was able to report 93.4% accuracy on ImageNet data with 12 times fewer parameters than AlexNet.

Dataset

In this report, we choose to focus on building models to classify two types of image datasets: High and Low-Resolution images. For consistency, both datasets contain 100 grayscale training images per class and 50 grayscale test images per class. The high-resolution image dataset is comprised of 5 classes of labeled images, where each image is 400 pixels wide and 300 pixels long. The hi-resolution image classes are ball, car, money, motorbike and person. The low-resolution image dataset is a subset of the MNIST handwriting digit database. These images are 28 pixels and 28 pixels long, and the class labels for the low-resolution dataset are 0, 1, 2, . . ., 8, 9.



Naïve Classifier

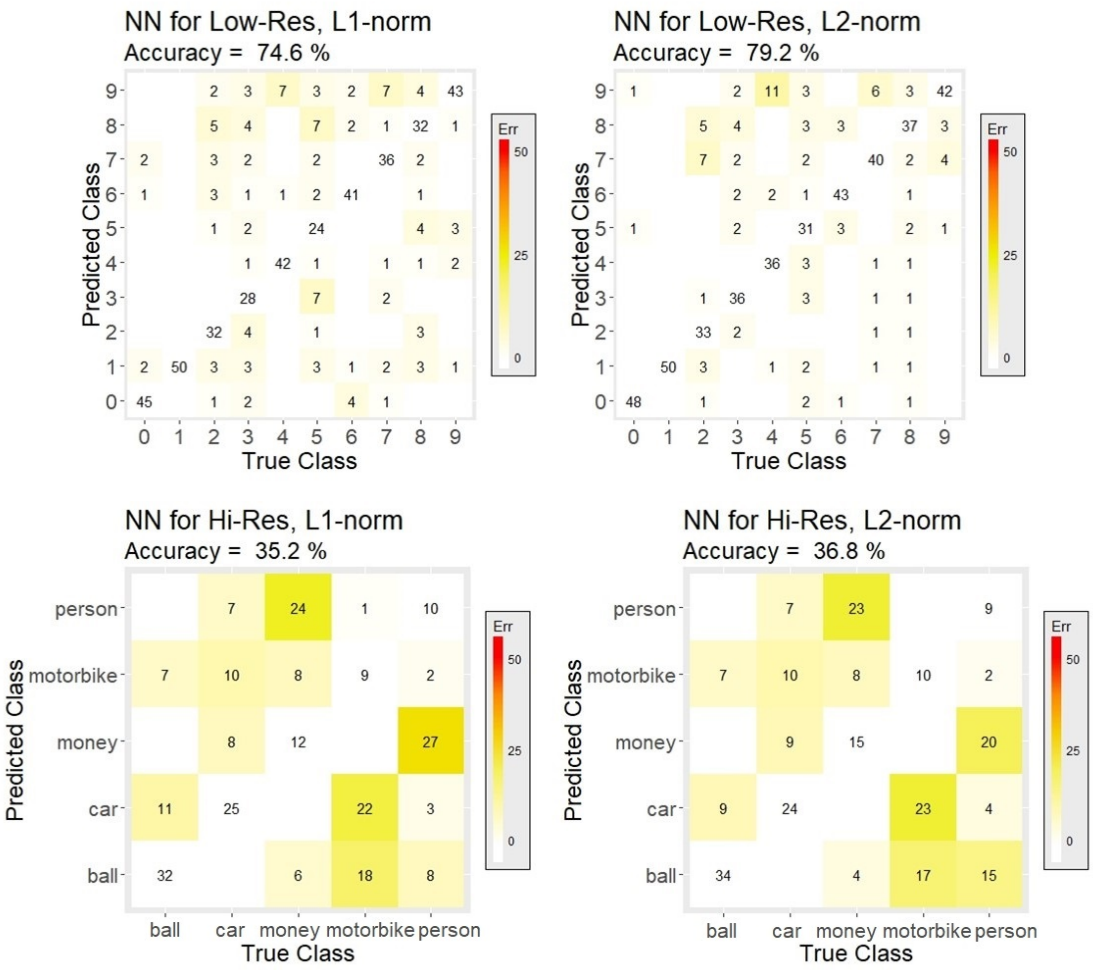
As a gentle introduction, we begin with a very simple Nearest Neighbor image classification idea. In practice, this idea is never used due to its poor classification performance and computational complexity.

Nearest Neighbor Classifier

We treat each image $I_{m \times n}$ as a matrix $M_{m \times n}$, where each pixel value from the image represents one element in the matrix. First, we compute and store the distances between each test and train image. Next, the class prediction for each test image is determined by its closest neighbor from the training data.

To compute the distance between two image matrices, we calculate the $L_1 - norm$ and $L_2 - norm$ of their difference. Reminder, the $L_1 - norm$ and $L_2 - norm$ are defined as:

- L_1-norm of M $= ||M||_1 = max_{1 \leq j \leq m} \sum_{i=1}^{m} |m_{ij}|$, which is the maximum absolute column sum of the matrix.
- L_2-norm of M $= ||M||_2 = \sigma_{max}(M) \leq \left(\sum_{i=1}^{n} \sum_{j=1}^{m} |m_{ij}|^2 \right)^{1/2}$, where $\sigma_{max}(M)$ is the maximum singular value of matrix M.



Surprisingly, the classification matrices above show that the naive classifier was able to achieve 79% accuracy using the $L_2 - norm$ distance. Predictably, this simple approach does not fare well in high-resolution image classification (accuracy is below 40% with both distance calculations).

It is worth noting that the L_2 distance got better result in both image sets than L_1. This is because pixel values come from a continous bounded set. As seen in its formulation, the L_2 distances are less sparse than the L_1 distance, which is better for our continous values.

In the following sections we will construct more informative models that outperform this naive approach for both datasets.

Feature Descriptors

Feature descriptors are algorithms motivated by recent works in computer vision and image processing that take an image and output a set of numerical feature vectors that encode *interesting* information from the image. This set of numerical feature vectors can be used to define a unique "fingerprint" for any given image. Recent works have uncovered ways of extracting information from images that is invariant under image transforamtions; this means that the same set of features are identified from an image if it is tranformed in ways such as scale, rotation, shift etc.

Histogram of Oriented Gradient (HOG)

The histogram of oriented gradient was proposed as a feature descriptor for object detection by Dalal and Triggs in 2005. For simplicity of presentation, HOG is considered on gray-scale images throughout our report.

Let D be an image, we would like to describe using HOG. We divide D into non-overlapping sub-images called cells, denoted C. Next, we form image blocks, denoted B, by combining cell sub-images. An important difference between cells and blocks is that cells are non-overlapping, while blocks can overlap. Starting with the cells definded in the upper-left corner of the image, blocks are formed in a slidding window format per row, with the last block consisting of the cells in the bottom-right corner.

The gradient ∇D of image D is vector field over the image's domain (x, y) , where (x, y) is a pixel location in the image. This can be approximated by partial derivative along the x and y directions as,

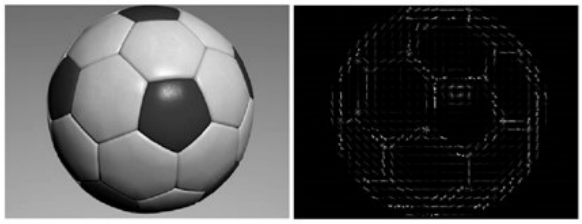
$$\nabla D \approx \begin{bmatrix} \frac{D(x+\delta,y)-D(x-\delta,y)}{2\delta} \\ \frac{D(x,y+\delta)-D(x,y-\delta)}{2\delta} \end{bmatrix}$$

The gradient provides two important pieces of information about the image's local stucture. $||\nabla D||$ tells us the magnitude of the local gradient and $\Theta = tan^{-1}\left(\frac{\nabla_y D}{\nabla_x D}\right)$ gives us the angle of the gradient.

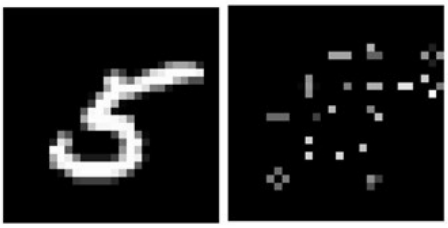
The vector h_i, called the HOG of block B_i, is calculated by computing the $\left(\Theta, ||\nabla D||\right)$ for each pixel in the block. Downsampling by binning the values of Θ is used to reduce the gradients per block. Finally, the HOG of image D is defined by stacking all h_i in order. Therefore, the HOG feature descriptor for an image can be represented as a 2-dimensional vector.

The images below display image features identified by HOG for high and low resolution images.

HOG Results for Hi-Res Image



HOG Results for Lo-Res Image



PCA of Histogram of Oriented Gradient (HOG)

We explore a multi-class classifier based on principle component analysis (PCA) of histogram of oriented gradient (HOG) for accurate and fast image classification.

Supposed we are given K classes of labeled images. For each image $x_i^{(j)}$ in class j where $i = \{1, \cdots, n_j\}$ and $j = \{0, \cdots, K-1\}$, we compute the HOG vector $h_i^{(j)}$. For each class j, the HOG vectors are arranged in a feature matrix $H^{(j)} = \left[h_1^{(j)}\, h_2^{(j)}\, \cdots h_{n_j}^{(j)}\right]$.

For each class j, we compute the mean vector and covariance matrix

$$\bar{h_j} = \frac{1}{n_j} \sum_{i=1}^{n_j} h_i^{(j)}$$

$$C_j = \frac{1}{n_j - 1} \sum_{i=1}^{n_j} \left(h_i^{(j)} - \bar{h_j}\right)\left(h_i^{(j)} - \bar{h_j}\right)^T$$

and then perform eigen-decomposition of C_j as

$$C_j = T_j \Lambda_j T_j^T$$

and construct matrix $T_P^{(j)} = \left[t_1^{(j)}\, t_2^{(j)}\, \cdots t_P^{(j)}\right]$ which contains P eigenvectors of C_j that are associated with the P largest eigenvalues. As a dimension reduction technique, the K image classes can be though of as being well represented by the reduced image set $\{\bar{h_j}, T_P^{(j)}\}$. $T_P^{(j)}$ defines a P dimensional subspace representing the variation of the individual images in class j from $\bar{h_j}$.

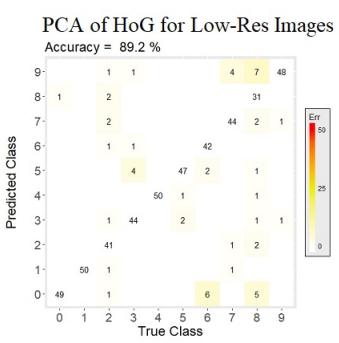
To classify a test image x, we first compute HOG features h of x. Then for each class j, we project the image to the P-dimensional subspace by computing $z_j = T_P^{(j)T}\left(h - \bar{h_j}\right)$. Next, we reconstruct the image by mapping it back to its original HOG dimensions by inverse transforming as $\hat{h_j} = T_P^{(j)} z_j + \bar{h_j}$. Finally, we compute the euclidean distance $E_j = ||h - \hat{h_j}||_2$ for $j = \{0, \cdots, K-1\}$ and classify x to the class j^* that minimizes $\left\{E_j, j = 0, \cdots, K-1\right\}$.

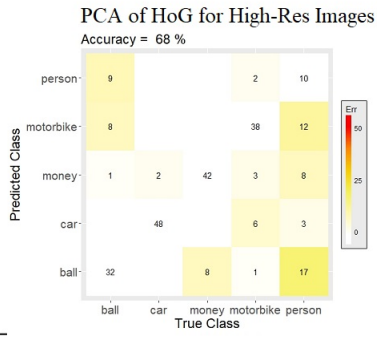
Algorithm 2: Multi-category Classification using HOG + PCA

Input: K,Number of principle components to retained
Data: $X_{Train}^{(j)}, X_{Test}, Y_{Train}$, where $j = 0, 1, ..., \#Class$
Result: Classification prediction forX_{Test}, $\hat{Y_{Test}}$
begin
  foreach j **do**
    Step 1 : HOG feature
    Transform the raw data $X_{Train}^{(j)}$ into HOG feature matrix $H^{(j)} = [h_1^{(j)}\, h_2^{(j)}\, \cdots\, h_{n_j}^{(j)}]$
    Step 2 : Apply PCA
    Compute following: (Standardization procedure)
    - $\bar{h_j} = \frac{1}{n_j} \sum_{i=1}^{n_j} h_i^{(j)}$
    - $C_j = \frac{1}{n_j - 1} \sum_{i=1}^{n_j} \left(h_i^{(j)} - \bar{h_j}\right)\left(h_i^{(j)} - \bar{h_j}\right)^T$

    Compute first K eigenvectors from C_j
    $\Longrightarrow T_K^{(j)} = [t_1^{(j)}\, t_2^{(j)}\, \cdots\, t_K^{(j)}]$
  Step 3 : Test data classification
  foreach $x \in X_{Test}$ **do**
    Compute HOG feature h of x
    foreach j **do**
      Compute
      - projection $z_j = T_K^{(j)}(h - \bar{h_j})$
      - approximation $\hat{h_j} = T_K^{(j)} z_j + \bar{h_j}$
      - Euclidean distance $E_j = ||h - \hat{h_j}||_2$

    Find $c \in j$ s.t. $E_c = \text{argmin} E_j$
    Append c to $\hat{Y_{Test}}$
  return $\hat{Y_{Test}}$





As seen in the classification matrices above, the PCA of HOG method is able to achieve an impressive 89.2% classification accuracy for low-resolution images. However this method does not fare as well in classifying high-resolution images; it is only able to achieve 68% classification accuracy for high-resolution images.

HOG+PCA	Low-resolution image		High-resolution image	
#PCs	Accuracy	Classify Time (ms)	Accuracy	Classify Time (ms)
10	0.876	9.9	0.68	175.2
20	0.868	10.5	0.68	179.2
50	0.892	11.5	0.68	179.6
100	0.848	11.9	0.68	262.8

A notable feature of the PCA of HOG model is its fast classification performance. The table above shows that on a 2-core Intel-i7 processor, this model is able to classify low-resolution images in approximately 10 milliseconds. This benchmark is important because the broadcasting standard for videos is 24 frames per second. This means that each frame (or image) in a video is on screen for 42 milliseconds. With PCA of HOG, we should be able to perform real-time streaming low-resolution video classification.

Another important interpretation of the results above is that the performance of classification in high-resolution images does not improve as we reduce the effect of dimensionality reduction (or increase the number of principal components). This result informs us of the limitation of HOG as a feature descriptor for high-resolution images. This is why, in the following models proposed in this report, we will try to improve HOG with spstial locality information.

Bag of Features

Bag-of-features model (BoF) for image processing is an adaptation of the bag-of-words model from text mining. In this approach, image features are treated analogously to words, and images to documents. The complete space of image features defines the visual vocabulary, and each image is represented as a set of words (features) from this vocabulary. The flowchart below displays the steps in the BoF model.

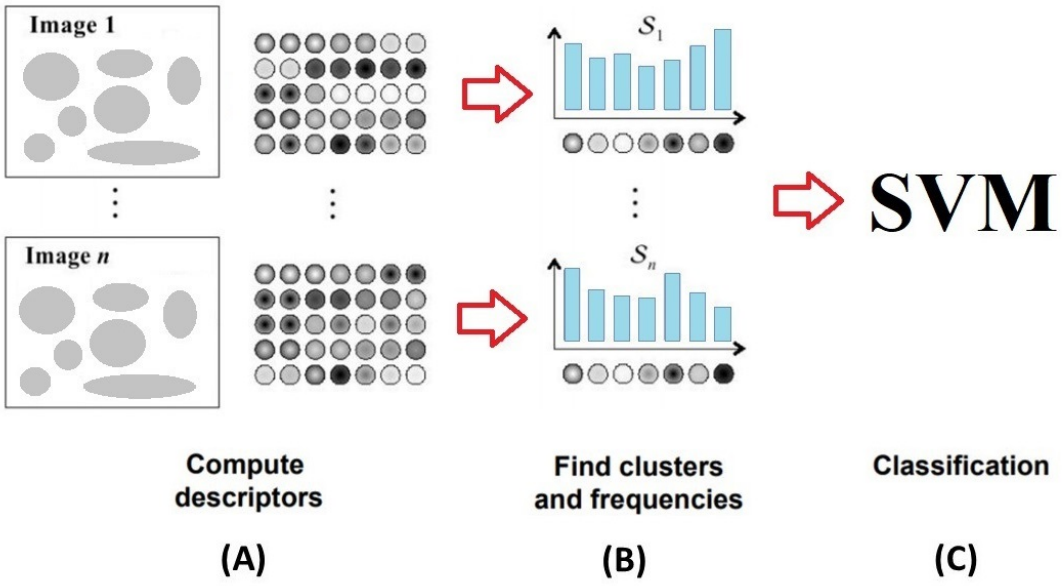


Image credit: Adapted from Arénaire project, Laboratoire de l'Informatique du Parallélisme(LIP)

Step (A): Compute descriptors

Feature detection and description algorithms must be applied to each image to extract local image features. These features are represented as numerical vectors.

Step (B): Find cluster and frequencies

The numerical vector representation of a local image features are converted to "codewords" (analogous to words in text documents). These "codewords" are collected to produce a visual vocabulary (analogous to a word dictionary). One simple method of converting feature vectors to the visual vocabulary is by performing K-means clustering over all local image features. The learned cluster centers define the "codewords" and the number of clusters defines the vocabulary size.

At this stage, each image can be represented as a combination of "codewords" from our visual vocabulary. It is worth noting that an image may contain multiple instances of the same "codeword", which is why each image is typically described as a histogram of "codewords".

Step (C): Classification

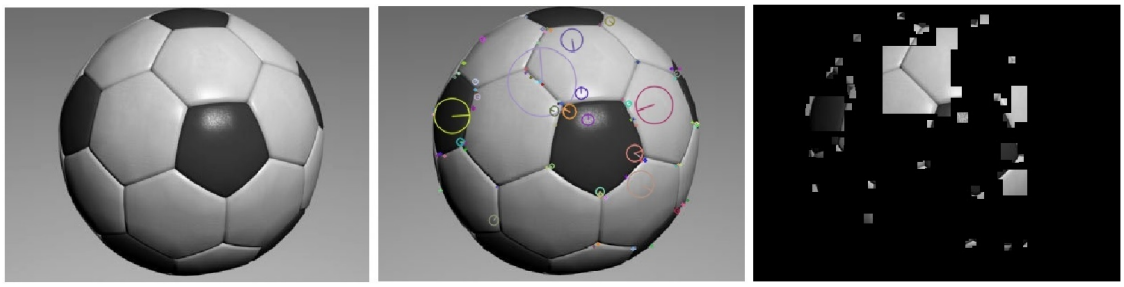
A classification decision boundry can be computed using the set-of-codewords representation of training images for each class. A variety of high-dimensional supervised learning models can be used for this step (such as SVM and AdaBoost).

To classify a test image, feature extraction is conducted as in step A. These test image features are mapped to the nearest "codewords" in our visual vocabulary. In the case where K-means is used to originally define the visual dictionary, this nearest mapping is simply minimizing the Euclidean Distance between each test feature vector and "codewords". Now, the test image can be represented as a collection of "codewords" and the classifier from step C can be used to predict the label of the test image.

Scale-Invariant Feature Transform (SIFT)

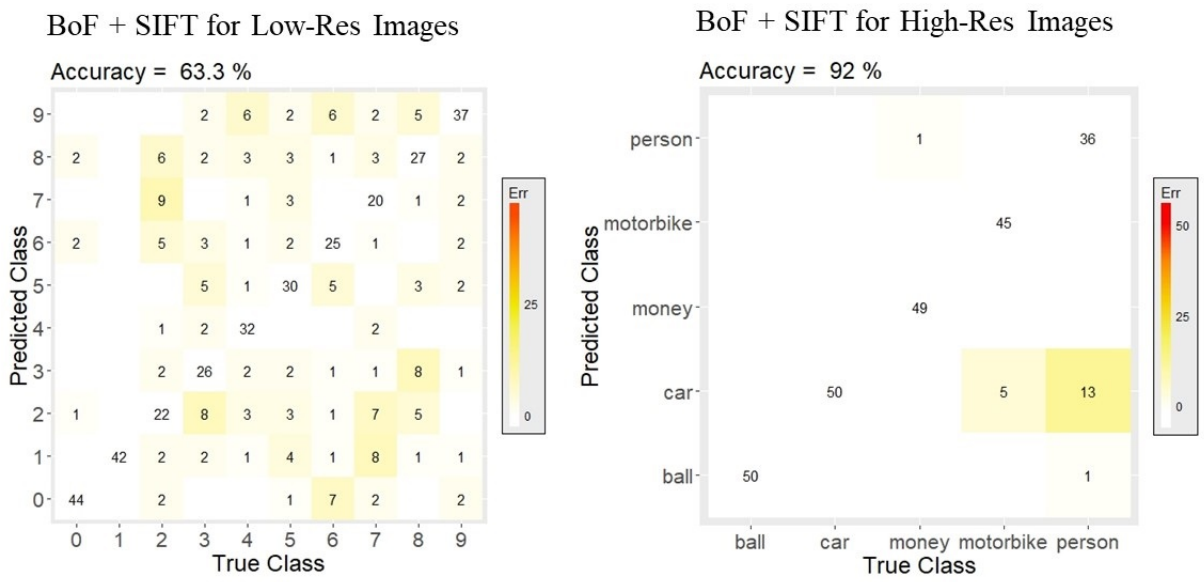
Scale-Invariant Feature Transform (SIFT) as a computer vision algorithm that was published by Lowe in 1999. This descriptor detects and describes a collection of local features in images that are invariant to image transformations of scaling, rotation, illumination changes, and geometric distortion.

The features highlighted by SIFT are computed by finding the local minimums and maximums of the *difference of gaussians* at various scales. Difference of gaussians is a feature enhancement technique that involves subtracting a series of blurred versions of an image from the original. In practice, difference of gaussians has been shown to increase the visibility of edges and object shape details in images. Similar to HOG, gradients can be computed on the image features highlighted by difference of gaussians. The direction of the local gradients, computed using the difference of gaussians processed image, is used to assign a location and one or more dominant orientations to each SIFT identified feature.



Bag-of-Features with SIFT

SIFT combines feature detection and description into one algorithm. We propose using it in the first step of the bag-of-features model. Next, for the clustering step to construct the visual vocabulary using the SIFT identified features, we suggest K-means for its simplicity. Finally, for the multi-class classification of image feature vectors, we employ SVM with the radial basis function kernel.



The classification matrices above show that the combination of a BoF and SIFT is able to achieve an impressive 92% classification accuracy for high-resolution images. However, this model does not fare well in classifying low-resolution images; it is only able to achive 63% classification accuracy for low-resolution images.

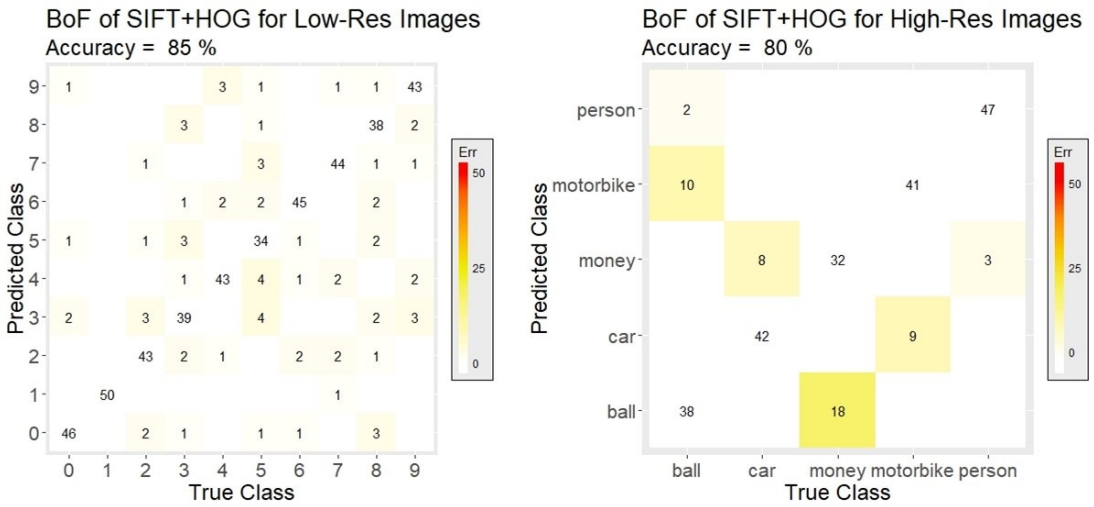
SIFT	Low-resolution image		High-resolution image	
Vocab. Size	Accuracy	Classify Time (ms)	Accuracy	Classify Time (ms)
30	0.474	0.70	0.80	47.2
50	0.552	1.26	0.84	48.0
100	0.602	1.48	0.92	49.6
200	0.633	1.78	0.92	51.2

The table above summarizes the performace of the BoF and SIFT models over a range of visual vocabulary sizes. The classification accuracy of this model increases as the vocabulary sizes increases. However, these gains only contiue up to a certain point, after which the model's performance stabilizes. Also, previously we mentioned the 42 millisecond classification-time threshold for real-time streaming video classification. As seen above, with a few technical refinements, the Bof and SIFT model should be able to classify high-resolution video in real-time.

Bag-of-Features with SIFT and HOG

In the PCA of HOG model previously, we saw that image features extracted by HOG were used to classify low-resolution images with high accuracy (89.2% accurate). In contrast, in the bag-of-features with SIFT model, we observed that the image features highlighted by SIFT could be used to classify the high-resolution images with sufficient accuracy (92% accurate). In this section, we propose a combination of these two models.

In this model, we will construct a bag-of-features using both SIFT and HOG algorithms. First, SIFT is used to identify key regions in the original images. Only these regions are extracted and HOG is applied to these subsets of the image. These HOG descriptors are then used to contruct the "codewords" in our visual vocabulary (as described before).



As intended, the conjunction of the two feature descriptors allows us to achieve satisfactory classification performance in both high and low-resolution datasets with a single model. As seen in the classification matrices above, this newly proposed BoF with SIFT and HOG is able to classify the high and low-resolution images with 84% and 92% accuracy, respectively.

There are some tradeoffs however. First, this new model does not perform as well as BoF with SIFT for classifying high-resolution images (BoF with SIFT achieved 92% classification accuracy earlier). We also observe a tradeoff in the performance of low-resolution image classification in the table below. The classification time for low-resolution images in the BoF with SIFT and HOG model is approximately 2 times slower than the PCA of HOG model.

BoF of SIFT+HOG	Low-resolution image		High-resolution image	
Vocabulary Size	Accuracy	Classify Time (ms)	Accuracy	Classify Time (ms)
10	0.646	21.0	0.60	327.6
30	0.788	21.8	0.72	419.2
50	0.850	22.2	0.78	442.0
100	0.850	24.8	0.80	468.0