TAGME Machine Learning Contest (IISc Bangalore)

Soumajyoti Sarkar

Sibashis Chatterjee

CST, 4th year

Bengal Engineering and Science University, Shibpur.

Team ID: **SSH**

Summary:

1) **Softwares Used** : The language used for writing the algorithms is **Matlab** and the toolboxes used are those available by default in **R2012 b** .

   The additional libraries used are:

   1. **Vlfeat:** It is an open source library that implements popular computer vision algorithms including feature descriptors, k-means clustering algorithms.

      Website: http://www.vlfeat.org/

   2. **VGG Computer Vision Practicals**: They are a collection of MATLAB-based self-contained hands-on lab experiences introducing fundamental concepts in visual recognition. They are open-sourced and are readily available on Github.

      Website: https://sites.google.com/site/vggpracticals/home

      Github: https://github.com/vedaldi/practical-object-category-recognition

   3. **Libsvm**: It is an open-source library for implementing Support Vector Machines for learning a classifier and then testing it on unknown images.

      Website: http://www.csie.ntu.edu.tw/~cjlin/libsvm/

   4. **Spatial Pyramid matching code**: We used source codes from the author of [2], that implemented the spatial pyramid matching kernels.

      Link: http://www.cs.illinois.edu/homes/slazebni/research/SpatialPyramid.zip

2) **Feature Extraction algorithm:** We have used **three** feature descriptors: **SIFT** (Scale Invariant Feature Transform) descriptors by David Lowe [4] , **HOG**(Histogram of Oriented Gradients) descriptors by Dalal and Triggs [3] and **oriented edge points** as has been used in [2] for spatial pyramid kernel matching.

   The **SIFT descriptor** has been shown to outperform a set of existing descriptors. The SIFT descriptor first computes a gradient orientation histogram within the support region. For each of 8 orientation planes, the gradient image is sampled over a 4×4 grid of locations, thus resulting in a $4 \times 4 \times 8 = 128$-dimensional feature vector for each region. A Gaussian window function is used to assign a weight to the magnitude of each sample point.

   In our algorithm we have used patch sizes of 8x8 and 16x16.

We also use dense SIFT in feature extraction while creating spatial pyramids for pyramid kernel matching as has been described in [2]. For that we use patches of size 16x16.

The **HOG** algorithm works in four main steps: gradient computation, orientation binning, descriptor blocks and block normalization. HOG has been found to perform very well compared to most other descriptors with an added advantage of fast computation over SIFT.

The **oriented edge points** also called the 'weak features' have been used in [2]. These are points whose gradient magnitude in a given direction exceeds a minimum threshold. We extract edge points at two scales and eight orientations, for a total of M = 16 channels.

3) **Similarity/Distance measures:** Two main distance measures have been used for computing the kernels used for learning a classifier using Support Vector Machines.

   They are:

   1. **Euclidean distance:** This is used in the computation of Radial Basis function (RBF) kernel.
   2. **Chi-square-distance:** This is used in the computation of chi-square kernel.

4) **Classifier:** For classification, we use Support Vector Machines (SVM) (Schlkopf and Smola, 2002).

   We use the concept of One vs All (OVA) method used for multi-category object classification used in SVM which is used in LibSVM. We have considered three kernels for classification – the **Radial basis function** (RBF), the **chi-square kernel** that incorporates the chi-square distance metrics and the **spatial pyramid matching kernel**.

   The RBF and the chi-square kernels have been used separately on both SIFT and the HOG descriptors leading to a total of 4 pre-computed kernel matrices.

   The spatial pyramid matching kernel has been used on the edge oriented features and the dense SIFT descriptor as has been described in the algorithm used in [2].

   So in total 5 kernels have been used for learning the classifier. We prepare 5 models on each of those pre-computed kernel matrices and calculate the probability estimates of the classes of each of those models on a test image. After that we take the average of all those probabilities and adopt a **winner-takes-it-all** strategy.

5) References:

1. The main algorithm has been used from the paper **Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study by Zhang et.a**l (algorithm used in PASCAL VOC 2006) where they presented a large-scale evaluation of an approach that represents images as distributions (histograms) of features extracted from a sparse set of keypoint locations and learns a Support Vector Machine classifier with kernels based on the χ 2 distance.
2. To consider the spatial orientation of the feature descriptors, we used the algorithm of spatial pyramid based matching in Beyond Bags of Features: **Spatial Pyramid Matching**

for Recognizing Natural Scene Categories by Lazebnik et.al.

3. Histogram of Oriented Gradients – Navneet Dalal and Triggs.

   Site: http://lear.inrialpes.fr/pubs/2005/DT05/

4. Scale Invariant Feature transform – David Lowe

   Link: http://www.cs.ubc.ca/~lowe/keypoints/

5. Recognizing and learning object categories – Rob Fergus, Lei-fei

   Link: http://people.csail.mit.edu/torralba/shortCourseRLOC/

6. Apart from that the concept of **winner-takes-it-all** strategy has been used for incorporating multiple kernels in multi-class classification via Support Vector Machines.

   Competitive learning networks - http://www.gc.ssr.upm.es/inves/neural/ann1/unsupmod/CompetLe/CompetLe.htm

Algorithm Details:

1.Prepocessing Stage:

Since some of the images were blurred in the second validation set, so we decided to go for two pre-pocessing stages:

i) **Contrast enhancement**: This ensured that some of the images that had low intensities compared to other images were scaled up to make the images sharper as well as reduce the contrast differences between the training and test images.

ii) **De-blurring:** We used a de-blurring algorithm based on convolution regularization using a point spread function.

These images are then given as input to the feature extraction algorithms for learning a classifier.

2.Feature Extraction Algorithm:

Bag-of-words Model:

The main idea behind the image classification algorithm used in this competition is based on the bag-of-visual-words model demonstrated in [5]. We obtain a global texton vocabulary (or visual vocabulary) by clustering descriptors from a training set, and then represent each image in the database as a histogram of texton labels. Given a global texton vocabulary of size **m**, the **ith** entry of a histogram is the proportion of all descriptors in the image having label **i.**

The steps for the bag-of-words image representation is as follows:

i) Take a set of training images of each class. For our purpose we have taken a set of 30 images per class from the training set. We then detect interest points in each of those training images. For our purpose, we have used the HOG and SIFT feature descriptors for the bag-of-words model in the first stage.

ii) We then apply k-means algorithm to the descriptors to cluster them into a fixed number of regions. For this competition, we kept the number of clusters equal to the number of

images per class in the training set which is equal to 30. This stage creates the visual vocabulary (also called dictionary) for each of the five classes. So we have five visual vocabularies in all., one for each class.

iii) We then concatenate all these visual vocabularies to form a global visual vocabulary of all the training images. Now for each of those training images of each of the classes, we form histograms for each of those classes using the global visual vocabulary. For that we first vector quantize the global vocabulary that we created, into visual words by using the SIFT or the HOG descriptors (separately for each case).

iv) Then we compute the normalized histogram of word frequencies using the keypoints extracted from the feature descriptors on each training image.

So our final feature vector representation for each of those 5 classes is a normalized histogram of feature descriptors. We obtain 5 histograms from each of the classes and then we concatenated them into a global histogram that would be fed as input to the kernel matrix computation stage. We apply the same procedure for each test image separately to ultimately obtain histogram of that test image. We use these histograms for computing the kernel matrices between the training set histograms and the test histogram.

Feature descriptors used:

i) **SIFT** – We use sift since it is one of the most widely used feature descriptors for obtaining the visual words model. The drawbacks of SIFT is that it incurs huge computation time, and hence for our purpose we use only patches of size 8x8 and 16x16 while obtaining the bag-of-words model and patches of size 16x16 while obtaining the feature descriptors for spatial pyramid matching kernel.
For using the SIFT descriptors we used Fisher encoding to encode the SIFT descriptors.

The **Fisher Vector** (FV) is an example of second order encoding. It records both the residuals and also the covariance of the SIFTs assigned to each Voronoi cell. Its implementation uses a Gaussian Mixture Model (GMM) (instead of k-means) and consequently SIFTs are softly assigned to mixture components.

ii) **HOG** – The advantage of using HOG over SIFT is that HOG is computed for an entire image by dividing the image into smaller cells and summing up the gradients over every pixel within each cell in an image while SIFT computes the gradient histogram only for patches (usually 16*16 divided into 16 cells) around specific interest points obtained by taking the DoG's (as an approximation to LoG's) in the scale space.

Image Pyramid model:

The algorithm for obtaining the feature descriptor representation in brief is as follows:

1. **Extract interest point descriptors (dense SIFT)** – As used in the algorithm described in the original paper, we have used oriented edge points, i.e., points whose gradient magnitude in a given direction exceeds a minimum threshold. We extract edge points at two scales and eight orientations, for a total of $M = 16$ channels. For better discriminative power, we also use SIFT descriptors of 16×16 pixel patches computed over a grid with spacing of 8 pixels.

2. **Construct visual word dictionary** – This is similar to the bag-of-words model where we use vector quantization and k-means clustering to form a visual vocabulary.
3. **Build spatial histograms** – Then we form spatial histograms on pyramids of level 0, 1 and 2.

   For step 3, we have used source codes available in [2] written by the author herself.

So like in the previous step, the output of this step would a concatenation of spatial histograms of different levels which would be used as input arguments for computing the kernel matrices.

3.Training Algorithm:

For the training algorithm, we have used Support Vector Machines for learning the classifier models.

Input Arguments: Precomputed kernel matrices, training set labels.
Tunable Parameters: The kernel matrices and the value C used in SVM.
Output arguments: The SVM classifier model.

In a two-class case, the decision function for a test sample x has the following form:

$$g(x) = -\sum_i \alpha_i y_i K(x_i, x) - b$$

where $K(x_i, x)$ is the value of a **kernel function** for the training sample x_i and the test sample x, y_i the class label of x_i $(+1 \text{ or } -1)$, a_i the learned weight of the training sample x_i, and b is a learned threshold parameter.

We have used two different kernels on the Bag-of-words model described above and the histogram intersection kernel for spatial pyramid kernel matching:

1) **Chi-square-kernel:** To compare two histograms $S1 = (u1, \ldots, um)$ and $S2 = (w1, \ldots, wm)$, we use the $\chi 2$ distance defined as

$$D(S_1, S_2) = \frac{1}{2}\sum_{i=1}^{m} \frac{(u_i - w_i)^2}{u_i + w_i}$$

   To incorporate $\chi 2$ distance into the SVM framework, we use extended Gaussian kernels

$$K(S_i, S_j) = \exp(-\frac{1}{A} D(S_i, S_j))$$

   Where A is the mean of all the chi-square distances between the histograms of the classes.

2) **Radial basis Function Kernel:** The RBF kernel on two histograms S_i and $\mathbf{S_j}$, represented is defined as

$$K(S_i, S_j) = \exp(-\frac{|| S_i - S_j ||_2^2}{2\sigma^2})$$

   Where sigma is a free parameter that can be tuned on the validation set.

3) **Histogram Intersection kernel:** This kernel is used for spatial pyramid matching.

So, in brief we precompute the kernel matrices using the first two statistics on feature histograms obtained from SIFT and HOG. So we obtain a total of 4 kernel matrices from them and one kernel from 3.

Now using these precomputed kernel matrices we create an SVM model using LibSVM library.

4.Prediction Algorithm:

We use LibSVM's **svmpredict** function to predict the labels on the test data. For that we followed the following steps:

i) First we estimate the probabilities of a test image belonging to each particular class based on all the models from each of the 4 bag-of-words kernels and one spatial pyramid kernel (total of 5 kernels).
ii) Then we take the mean of the probabilities of each class summing up the probabilities found among all the kernels.
iii) The class that gives the highest probability is the predicted class of the test image.

5.Interpretation of results on validation data:

The validation data fetched us an accuracy of **90%.**

We did not use the spatial pyramid kernel matching algorithm on the validation set, but upon testing it on images we found the results to be better than by just using SIFT/HOG descriptors since the bag of words model on SIFT/HOG do not keep spatial information. So to incorporate that we used spatial pyramid matching algorithm.

Why do you think your algorithm got the accuracy that it did on the validation data? Are there scope for improvements?

The main reason as to why the algorithm didn't work on several car images was that the car images in the training set and the validation set differed a lot in the context of interest point detectors. There were many car images that were cropped from either the top or bottom in the validation and test images provided, but there were no such images in the training set. So we had to keep spatial information as well for which we used spatial pyramid matching on the test image. Had we applied that on the validation set, would have had a better score on the car images in particular.