Object Class Recognition Using Multiple Instance Learning with Image as a Bag of Subimages

-- Shadab Khan
COSC 174, Dartmouth College
Project Report

1. Introduction

Object class recognition in a given image is a difficult problem. To classify a test image as hit or match for a particular object class requires the abstraction of an object, developed using real world images. This is where the difficulty lies - since the images of an object class can have large-scale variations in terms of illumination, angle of view, scale, rotation, color, location, type etc (intra-class variations). For example, the different photographs of the object class 'car' from PASCAL VOC 2007 [1] dataset shown below differ from each other in aspects mentioned previously.

Figure 1 : Image showing object class 'car' from PASCAL VOC 2007 data-set. Note the variations in the illumination, scale, rotation, angle of view, occlusion, manufacturer and model etc.

This necessitates use of complex algorithms to develop as robust abstraction as possible within computational budget, to handle the variations while still producing useable classification performance. The complete process of object class recognition can be understood as a three step process involving : 1) Representation of the object, 2) Learning an abstraction or a model using set of training images which are represented by method developed in previous step, 3) Testing the object class recognition. A discussion of proposed approach for these steps follows.

For image representation, I used Bag of Visual Words approach. For learning an image model, I tried Diversity Density and Citation-K-Nearest-Neighbor (CKNN) approaches and found CKNN approach to be better suited for the task of object class recognition. For testing these algorithms, I collected free-usage images available on the internet.

To perform the experiments, I decided to use only 4 object classes from the PASCAL VOC 2007 data-set, namely : Aeroplanes, Buildings, Cars and Monitors. My objective was to have 4 classes that would range from relatively easier to more difficult object classes for the purpose of object-class recognition. Looking at the images in the data-set, I observed that images of the Aeroplanes had a similar background (mostly sky) so I expected Aeroplanes to be the easier class to classify. Buildings and Cars usually had varying backgrounds and I expected them to be somewhat difficult object-classes. I expected Monitors to be the most difficult object-class in the four to be classified. Since Monitors typically occupied a smaller region, and had high dissimilarities in the background and the image being displayed in the monitor, majority of the visual words describe the background and would lead to inaccuracies in the classification. This report is organized as follows : Section 2 describes the object-class abstraction. Section 3 Describes the Multiple Instance Learning (MIL) approach. Section 4 details the MIL algorithms that I tried. Section 5 presents the results and discusses their comparison against the baseline methods. Section 6 concludes the report and references are provided towards the end of the report.

2. Object Class Abstraction

To represent each image, we need image-features which are not only highly discriminative, but are also robust to the variations. To decide which features to use, I did a survey of features which are typically used for object class recognition and shortlisted the following features.

1. Bag of Visual Words (BOVW) : A commonly used technique to represent image as a histogram of visual words. Introduced by Csurka et al. in 2004 [2], this has grown up to be a very popular method of image representation. This works by first finding a number of interest points in an image - this step is often called ‘detection’. Next, the region around interest points is windowed and abstracted - this step is often called ‘description’. These descriptors are then quantized into visual words, often using k-means clustering with Mahalanobis or Euclidean distance [3]. The feature vector then contains the number of occurrence of each visual word in an image.
2. GIST : Developed by Torralba et al. [4], this algorithm computes the histograms of gradient orientations which are localized at test point. It roughly abstracts the spatial arrangement of image structures and has been proven to work well for describing the general appearance of scene.
3. Histogram of Oriented Gradients (HOG): Developed by Dalal and Triggs [6], this is a well-known feature that has shows its promise in state-of-the-art part-based object detector [5].

Several auxiliary features can be combined with the ones listed above to suit a specific application and requirement, [3, pp. 205-266] has a comprehensive list of them. The three features listed above make a good choice and have been shown to work well for object-class recognition problem by Torressani et al. [7]. For the Bag of Visual Words feature, I used SURF (Speeded Up Robust Features), which is a fast and scale-invariant, rotation-invariant detector and descriptor [8]. The MATLAB implementation of SURF used for this project is [10].

During experiments with Multiple Instance Learning algorithms, I realized that using all three features listed above will significantly increase the computational cost. This is a critical consideration for this project since I used a Bag-of-Image representation for all of the images used in the experiments, which meant each sub-image of the image, whether small or large, would have a large feature vector length. Thus, I decided to use only BOVW to represent images, since it is the most com-

monly used feature [7], [11-13]. The BOVW representation of an image is essentially a histogram of frequently observed visual words, which have been previously extracted from a diverse set of images and stored in the computer in form of a visual-word-book. This visual-word-book is usually called 'Vocabulary' in the literature. Process of obtaining a histogram representing images can be divided into three parts :

1. Compute interest points for the given image : In case of SURF, the interest points are computed using Hessian matrix [8], [14]. Figure 2 shows the interest points computed using OpenSurf for one of the images taken from PASCAL VOC 2007 data-set.

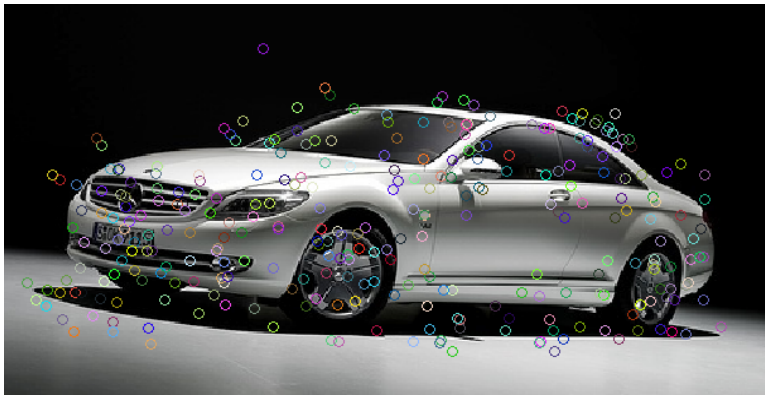


Figure 2 : Image showing the interest points that were detected for one of the image from PASCAL VOC 2007 data-set.

2. Compute descriptors for the interest points : SURF descriptors describe a distribution of Haar-Wavelet responses within the interest point neighborhood [8].

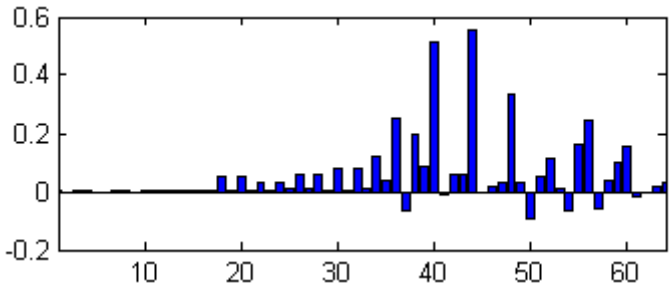


Figure 3 : Plot showing descriptor computed for one of the interest points of the image shown in Figure 2.

3. Quantize all feature vectors to obtain a histogram of visual words. This is done by assigning the computed descriptors to the closest matching visual word in the Vocabulary.

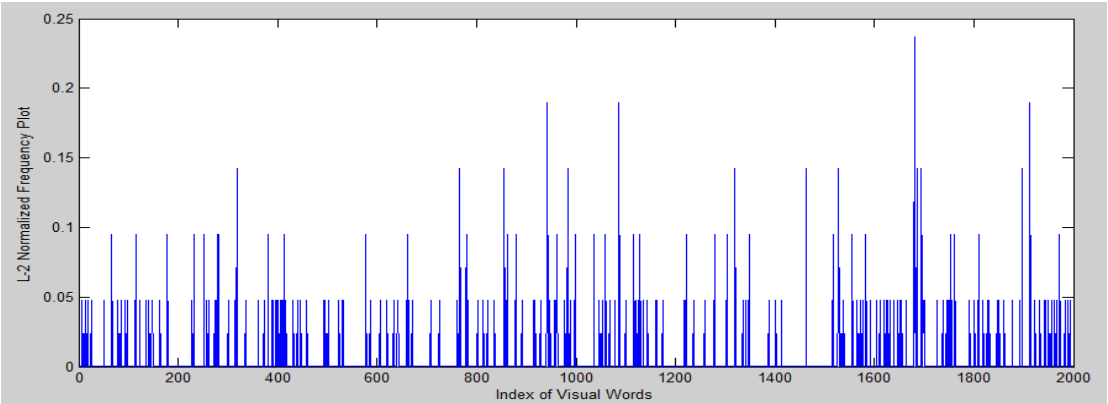


Figure 4 : Plot of the histogram of visual words obtained for the image shown in Figure 2.

Once the interest points have been detected, descriptors are calculated to describe the 'neighborhood' of the interest points. Descriptors are

Vocabulary Generation : Step 3 requires a code book of visual words which stores all visual words that define our algorithm's 'vocabulary'. I created a Vocabulary using 157 images which had images drawn from the four object classes chosen for this project, as well as other images of common scenarios. To create the vocabulary, I first extracted descriptors from all of the images, and then clustered them using K-Means clustering algorithm to have 2000 visual words. The histograms obtained are vectors of length 128 elements that represent an image.

3. The Multiple Instance Learning Approach

Consider the image of the car and its interest points computed using SURF shown below :

Figure 5 : Image of the car with its interest points superimposed. Notice the number of interest points that are on the background.

Images such as this one, often have a significant number of detected interest points on the background. Thus, the histogram representation of this entire image will often have visual words that describe the background. This leads to inaccuracies in the classification, since we do not have an accurate representation of the class.

To overcome this problem, I decided to try Multiple Instance Learning approach, where each image is represented as a bag of sub-images. MIL algorithms are used to solve the problems where instead of having a label corresponding to each instance, there is a label for a bag of instances. This is an example of weakly supervised learning. Popular MIL algorithms have been reviewed in [15]. The image shown below, (taken from [15] shows a graphical illustration of the idea behind MIL approach.

Figure 6 : Illustration of the idea behind "Bag of Instances" having a single label, as used in MIL approach. Image reference [15].

The intuition behind using MIL approach is that if subimages can be obtained which contain a good portion of the object-class, then there will be a stronger match between them as compared to the match between the corresponding original images, which typically contain a significant contribution of background and clutter in the histogram of the image. Thus using MIL, we can possibly get better classification results as compared to approach using each image as an instance. Please note that in MIL, bags of instances are labeled 1 if there is at least one instance belonging to the class being searched for. However, negative bags of instances are labeled 0 only if none of the instances belong to the object class under search.

To obtain the subimages from the window, I used Objectness measure proposed by Ferrari et al., [9] which returns a score of 'objectness' of many sampled subimages. This objectness score is generic over classes. The objectness score takes four visual cues into account to measure the relative chances of a sampled subimage containing a generic object. The four visual cues are :

1. Multi Scale Saliency : This is based on the spectral residual of the FFT, which favors regions with a unique appearance within the entire image.

2. Color Contrast : This cue is a measure of the localized dissimilarity of a window to it's neighborhood.

3. Edge Density : This cue measures the density of edges near the window borders.

4. Superpixel Straddling: This cue captures the closed boundary characteristic of the objects by using superpixels as features. Superpixels divide the image into small connected-components that have uniform texture. A key property of superpixels is that they preserve object boundaries, and all pixels within a superpixel constitute the same object [16].

Using the Objectness Measure 1.5 code available from [17], I generated a bag of 10 sub-images corresponding to each image in the training and test set of the four object-classes. In my trials, I found 10 subimages to be a good number to cover the most important objects as determined by the objectness measure. Figure below explains this idea.

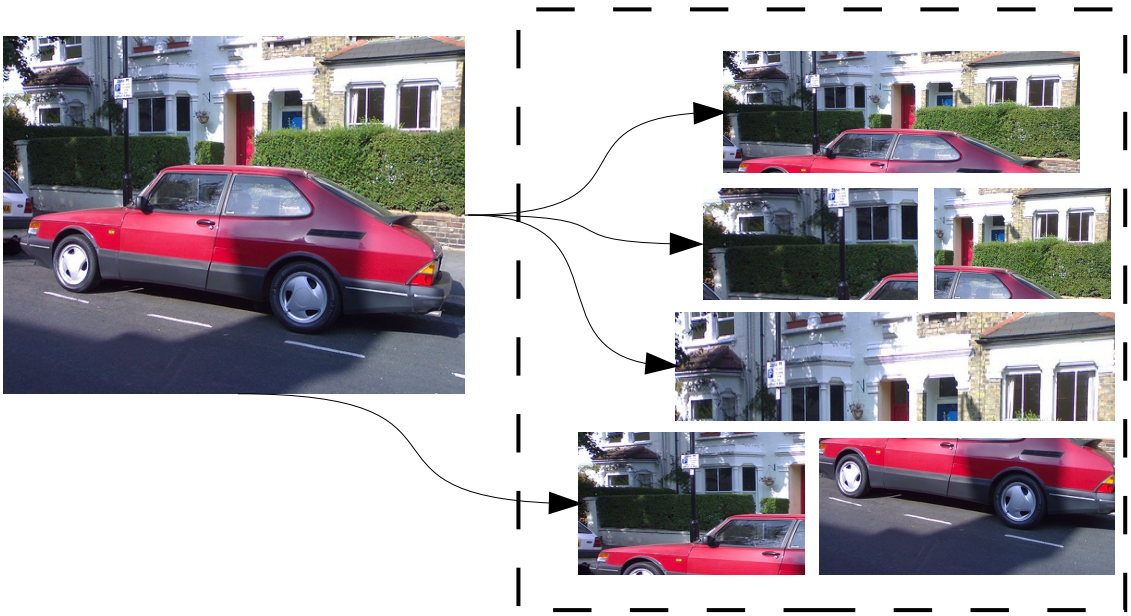


Figure 7 : An image represented as bag of sub-images.

Once the sub-images have been obtained, histograms are computed for each sub-image that serve as an instance. Thus we get a bag of instances corresponding to each image (instance).

4. Multiple Instance Learning Algorithms

Diversity Density : Input histograms of 2000 visual words can be considered as a combination of 2000 basis vectors in feature space. The idea behind Diversity Density is to find a combination of these basis vectors, which lies at the intersection of the positive bags minus the union of the negative bags [18]. Such combinations are called 'Concepts'.

Let us represent the i-th positive bag, that is the bag-of-subimages containing the object-class as B_i^+ and the i-th negative bags as B_i^-. The j-th instance of positive i-th bag is given by B_{ij}^+. Then, finding the Concepts is equivalent to maximizing the objective function shown below :

$$\underset{x}{argmax} \prod_i Pr(x = t|B_i^+)Pr(x = t|B_i^-)$$

To calculate the *Concepts* the objective function mentioned above is maximized with respect to the data. This is the general definition of the Diversity Density. To instantiate this, we need to define the terms in the product. Maron et al. [18], suggest using a noisy-or model where the probability that not all instances belong to the object-class is :

$$Pr(x = t \mid B_i^+) = Pr(x = t \mid B_{i1}^+, B_{i2}^+,...) = 1 - \prod_j (1 – Pr(x = t \mid B_{ij}^+))$$

and similarly,

$$Pr(x = t \mid B_i^+) = Pr(x = t \mid B_{i1}^+, B_{i2}^-,...) = \prod_j (1 – Pr(x = t \mid B_{ij}^-))$$

The causal probability of an individual instance on a potential concept is given by a Gaussian kernel which uses Euclidean metric to measure the 'closeness' of the two. To tone down the contribution of irrelevant features in measuring this probability, weights corresponding to each feature is calculated as well. Note that the assumption that there exist a common single point for all of the positive bags is not necessarily correct. Thus, multiple concepts can be calculated and the maximum probability of an instance for any of these concepts can be chosen to obtain the probability of concept given the bag.

I implemented Diversity Density algorithm using the code provided by [21] and obtained the concepts for object-class Aeroplanes. As the starting point to begin line search of concepts and weights, I used all of the instances from all of the positive bags of training set. For 370 instances in total, the algorithm took over 24 hours to compute the concepts. Results that I obtained for test bags of the Aeroplanes object-class were surprisingly bad. I observed that The Euclidean distances between the concepts and the positive & negative bags from the training set varied from 0+eps to 26. Researching on this problem, I learned that there are several problems with the Diversity Density method. Some of the problems are : 1. The algorithm requires multiple starts with different starting points. 2. The algorithm takes very long time to compute the concepts. 3. Performance varies depending on metric used to define the "closeness".

Euclidean metric, Earth Mover's Distance, χ^2-metric, etc. have been tried before with Gaussian Kernel for object classification. Previous studies report that Euclidean distance does not carry topological information of the histograms, and relative scaling of the distances should be performed to have an acceptable threshold for classification. Rubner *et al.* [20] have pointed out that EMD & χ^2 are better measures of “closeness” for object classification tasks. Due to the very long computational time and no guarantee of obtaining globally optimal concepts, I decided to not to continue experiments with the Diversity Density algorithm.

Citation-K-Nearest-Neighbors (CKNN) : CKNN [19] is a kNN-like approach to multiple instance learning. In CKNN, distance between the bags is given by modified Hausdorff distance

Hausdorff distance : for a set $A = \{a_1, a_2, \ldots\ldots, a_m\}$ and set $B = \{b_1, b_2, \ldots\ldots, b_n\}$, the Hausdorff distance between sets A and B is given by H(A,B) = max{h(A,B),h(B,A)} where h(A,B) is given by:

$$h(A, B) = \underset{a \in A}{max} \underset{b \in B}{min} \|a - b\|$$

An illustration of minimum Hausdorff distance taken from [21] is shown below.

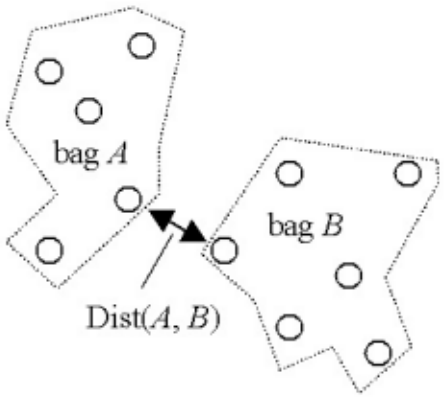


Figure 8 : Illustration of minimum Hausdorff distance taken from [21].

CKNN approach is different from regular kNN. Instead of using minimum Hausdorff distance, CKNN uses k-th Hausdorff distance for finding out k-nearest citations and references of a given instance. However, the major difference arises from the idea of using both citations and references of a given instance, as opposed to only references of an instance as in the kNN algorithm. That is, for a given instance, we not only look at the references of an instance (also defined as it's nearest neighbors in the kNN algorithm), we also consider the instances from the training set that *refer* to the given instance. Including the citations improves the classification results [19] since the negative instance in the positive bags, along with the negative instances from the negative bags might overwhelm the simple KNN algorithm. This idea is illustrated below.

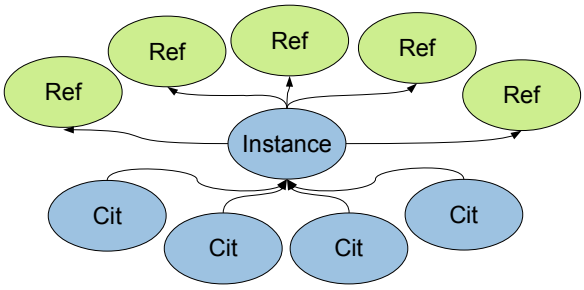


Figure 9 : Illustration showing the idea of citations and references of a given instance.

The algorithm for Citation-KNN requires number of citations and number of references to be used for classification as the inputs. In case of a tie in the positive versus negative poll using the instances from the training set, I set the classification as negative. I obtained the CKNN code from [22].

5. Results and Discussion

Results obtained using CKNN method for all of the four classes are summarized below. I ran 100 iterations of CKNN code to compute the classification accuracy for number of citations and references each varying from 1 to 10. Please note that for all of the ribbon plots, left axis is the number of citations, right axis is the number of references and the z-axis is the classification accuracy.

Aeroplanes

I expected the results for this class to be generally better than the results for the other class since many of the test images had a relatively cleaner background. Results of all 100 iterations is shown.

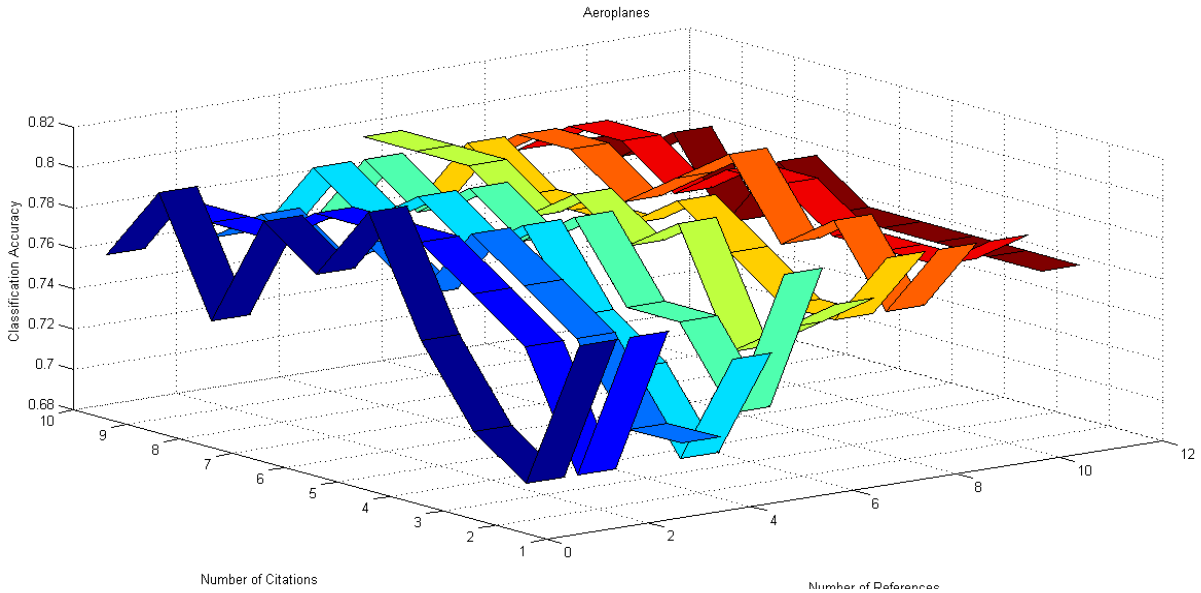


Figure 10 : Results for Aeroplanes object-class.

Buildings

I expected Buildings to be a more challenging class as compared to the Aeroplanes, since Buildings typically made background in the PASCAL VOC 2007 data-set.

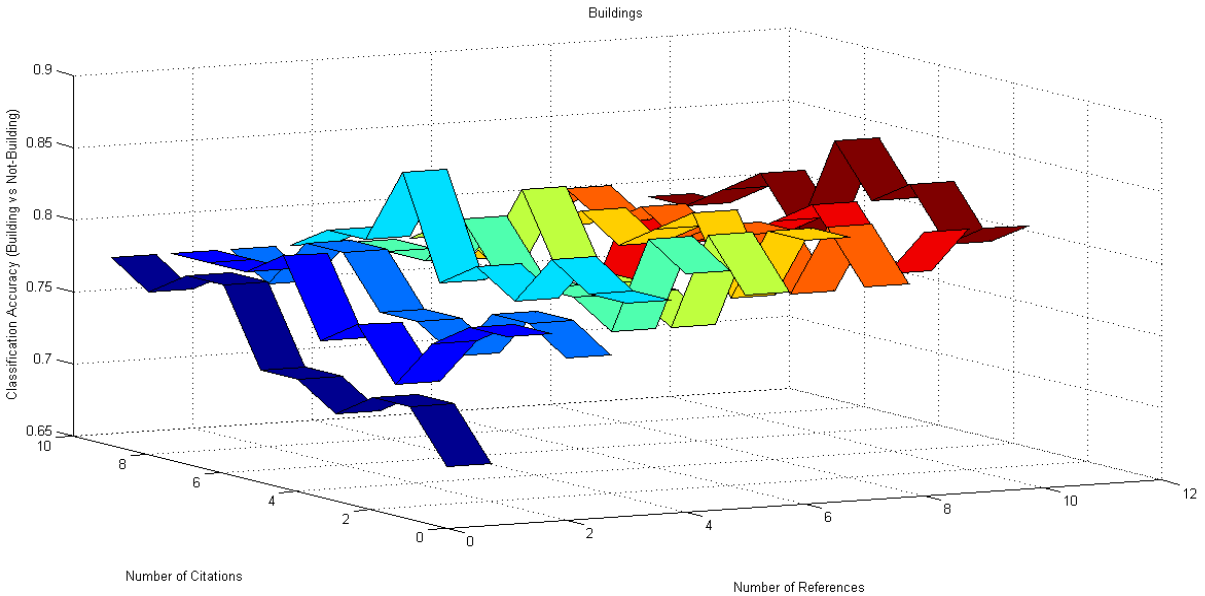


Figure 11 : Results for the Buildings object-class.

Cars

Cars made for a more difficult class as compared to the buildings and aeroplanes. Primarily because images of cars in the PASCAL VOC 2007 data set have large-scale intra-class variations. The images are not only angled differently, but have many differences in the illumination, angle of view etc. See figure 1 for more details.

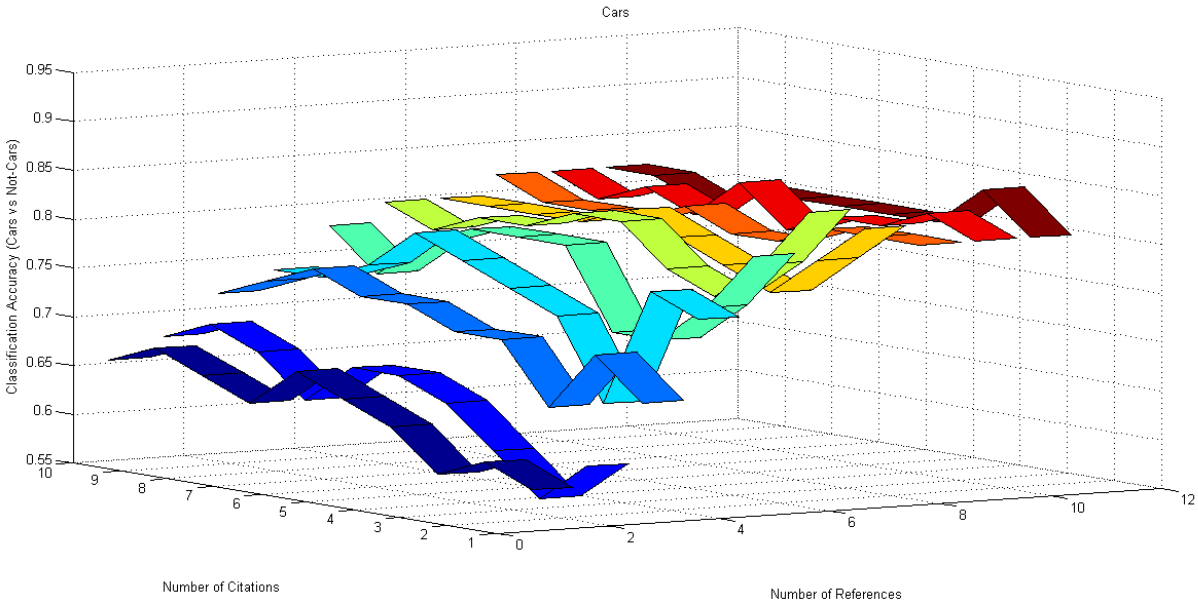


Figure 12 : Results for Cars object-class.

Monitors

Monitors were the most challenging class, with most of the iterations having accuracy in 60's or low 70's with only a few iterations closing towards 80. The plot below shows the results for all of the iterations.

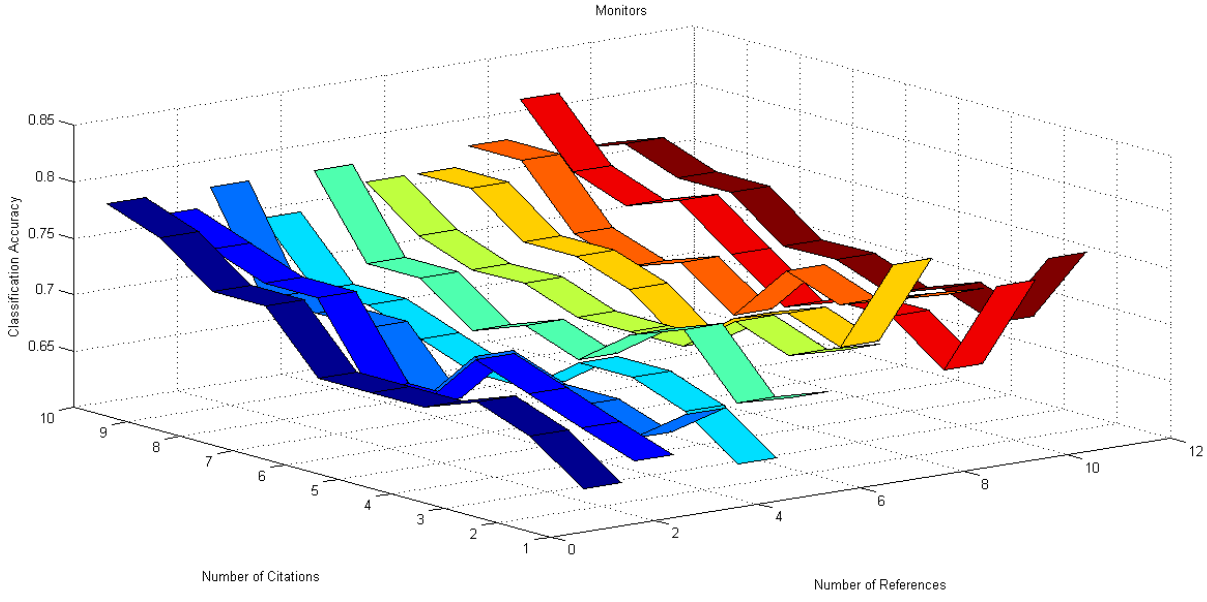


Figure 13 : Results for the Monitors object-class.

Statistics and The Comparison with the Baseline Algorithms

To evaluate the effectiveness and the efforts involved in this procedure, I computed the classification accuracy using Naive-Bayes and KNN algorithms, where each instance represented a complete image from the training or test set. For KNN algorithm, K varied from 1 to 13. The results are summarized below :

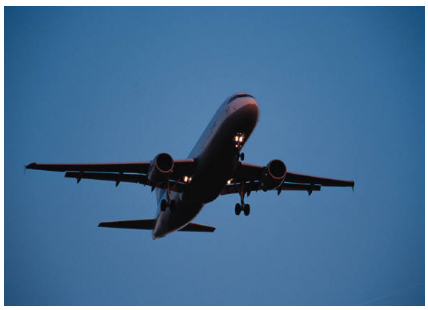
Object Class	Classification Accuracy Range – CKNN (MIL)	(Ref,Cit) - Min	(Ref,Cit) - Max	Classification Accuracy Naive-Bayes	Classification Accuracy - KNN
Aeroplanes	69.81 – 81.13	(1,2) (2,2)(4,2)	(1,5)(8,5)	41.79	71.64 – 79.10
Buildings	68.42 – 85.96	(1,1)	(10,5)	49.12	49.25 – 64.17
Cars	57.14 – 85.71	(2,2)	(6,1)(10,2)	50	50.74 – 61.19
Monitors	63.79 – 81.03	(1,1)(3,6)	(9,10)	51.72	61.19 – 74.62

The table above supports the initial hypotheses that using Multiple Instance Learning approach to the problem of object-class recognition Improves the classification accuracy ! The results clearly reveal an improvement in the classification accuracy, most prominently in the case of Buildings, Cars and Monitors. However, a more exhaustive testing, as well as the comparison against more baseline algorithms may further the usefulness of the MIL approach.

Trends Observed : The ribbon plots shows some interesting trends. In the case of Buildings, Cars and Monitors, it can be observed that classification accuracy improves not only with increasing number of references, but also with increasing number of citations. This is in agreement with literature where the context of application was different [19]. Further, improvements in case of Aeroplanes is visible with first few increments in the number of references and citations, however, the improvement beyond isn't readily visible.

Examples of Correctly and Incorrectly Classified Test Images (Taken from the max accuracy case)

Aeroplanes Correctly Classified


Aeroplanes Incorrectly Classified



Buildings Correctly Classified



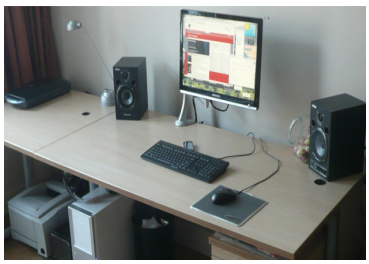
Buildings Incorrectly Classified



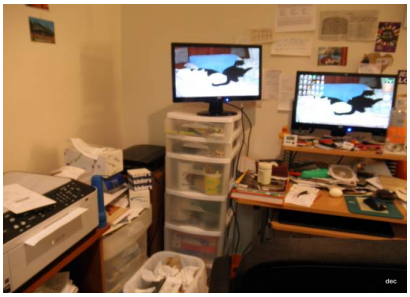
Cars Correctly Classified


Cars Incorrectly Classified


Monitors Correctly Classified



Monitors Incorrectly Classified


6. Conclusion

For this project, I investigated the effectiveness of Multiple Instance Learning algorithms in improving the classification accuracy as compared to the baseline algorithms (Naive-Bayes and KNN). I found that although the number of processing and computations involved in the case of MIL are more than the fully supervised approach, the results obtained using MIL approach are significantly better than the fully supervised approach.

References

[1] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2007 Results",

Web: http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html

[2] G. Csurka, C R Dance, L Fan, J Williamowski, C Bray, “Visual Categorization with Bags of Keypoints”, in *Proceeding of ECCV*, 2004.

[3] **BOOK:** Computer Vision: Algorithms and Applications, Richard Szeliski, Springer Publications, 2010.

[4] Aude Oliva, Antonio Torralba, “Modeling the shape of the scene: a holistic representation of the spatial envelope”, *International Journal of Computer Vision*, Vol. 42(3): 145-175, 2001.

[5] P Felzenszwalb, R B Girshick, D. McAllester and D Ramanan, “Object detection with discriminatively trained part based models” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009.

[6] N Dalal and B Triggs “Histogram of Oriented Gradients for Human Detection”, in *Proceeding of CVPR*,2005 .

[7] Alessandro Bergamo, Lorenzo Torresani, Andrew Fitzgibbon “PiCoDes: Learning a Compact Code for Novel-Category Recognition” in *Proceeding of Neural Information Processing Systems (NIPS)*, 2011.

[8] H Bay, A Ess, T Tuytelaars, L V Gool, “SURF: Speeded up robust features”, *Computer Vision and Image Understanding*, pp. 346-350, 2008.

[9] V Ferrari et al. “What is an Object?”, CVPR, 2010.

[10] Web: http://www.mathworks.com/matlabcentral/fileexchange/28300

[11] I Khan, P Roth and H Bischof , “Learning Object Detectors from Weakly-Labeled Internet Images”, Proceedings of AAPR Workshop, Austria, 2011.

[12] T Deselaers, Bogdan Alexe and V Ferrari, “Localizing objects while learning their appearance”, Proceedings of the 11th ECCV, 2010.

[13] J Wang *et al.* “Evaluating bag-of-visual-words representations in scene classification”, Proceedings of the Int. Workshop on Multimedia Information Retrieval, USA, 2007.

[14] K. Mikolajczyk and C. Schmid, “Indexing Based on Scale-Invariant Interest Points”, ICCV, 2001.

[15] B. Babenko, “Multiple Instance Learning: Algorithms and Applications”, UCSD Research Exam, October 2008. Web : http://cms.brookes.ac.uk/research/visiongroup/talks/rg_dec_11_09/bbabenko_re.pdf

[16] B C Russell et al., “Using multiple segmentations to discover objects and their extent in image collections”, CVPR, 2006.

[17] Objectness Measure 1.5, Web : http://www.vision.ee.ethz.ch/~calvin/objectness/

[18] O. Maron , T L Pérez, “ A Framework for Multiple-Instance Learning”, Advances in NIPS 1998.

[19] J Wang, J D Zucker, “Solving the multiple-instance problem: A lazy learning approach”, ICML 2000.

[20] Y Rubner *et al.*, “The Earth Mover's Distance as a Metric for Image Retrieval”, IJCV, vol. 40, 2000.

[21] Z Zhou, M Zhang, “ Ensembles of Multi-Instance Learners”, Proceedings of the 14th ECML, 2003.

[22] Citation-KNN Code, Web : http://lamda.nju.edu.cn/code_MIL-Ensemble.ashx