DETECTION OF NANO PARTICLES IN TEM IMAGES USING AN ENSEMBLE LEARNING ALGORITHM

by

Paranjith Singh Lohiya

Submitted in Partial Fulfillment of the Requirements

for the Degree of Master

in the

Computer Science and Information Systems Program

YOUNGSTOWN STATE UNIVERSITY

May, 2015

DETECTION OF NANO PARTICLES IN TEM IMAGES USING AN ENSEMBLE LEARNING ALGORITHM

Paranjith Singh Lohiya

I hereby release this thesis to the public. I understand that this thesis will be made available from the OhioLINK ETD Center and the Maag Library Circulation Desk for public access. I also authorize the University or other individuals to make copies of this thesis as needed for scholarly research.

Signature:

Paranjith Singh Lohiya, Student Date

Approvals:

Dr. Yong Zhang, Co-Major Advisor Date

Dr. Feng George Yu, Co-Major Advisor Date

Dr. John Sullins, Committee Member Date

Dr. Sal Sanders, Associate Dean of School of Graduate Studies Date

DEDICATION

I dedicate this thesis to my parents and friends who have encouraged me at all phases of my life and for their immense love on me.

ABSTRACT

Transmission electron microscopy (TEM) has an ability to depict material structures on nanoscales (~0.1 nm). High resolution TEM has found applications in a wide range of domains such as the studies of biological tissues, reactive chemical compounds and product defect inspection. For the past decade, Nano-research has generated a large number of TEM images, each containing immense amount of information that cannot be processed and interpreted manually. The combination of image processing and big data mining becomes the only viable solution. This thesis investigates the feasibility of using a Cascade AdaBoost algorithm to detect and count nanoparticles automatically. Experiments with cube-shaped objects have yielded very promising results with high detection rate (true positive rate) and low false alarm rate (false positive rate). The impacts of labeling variation, sample size and feature size on the detection accuracy were also discussed.

ACKNOWLEDGEMENTS

I would like to thank Dr. Yong Zhang for giving me this opportunity to work on the detection of nanoparticles project and helping me to complete this thesis successfully. He has provided me tremendous encouragement and knowledge of how to approach a challenging problem by thinking beyond the box.

I would like to thank Dr. John Sullins for being a great advisor during my entire graduate study at YSU and guiding me to choose courses of my interest. I also would like to sincerely thank Dr. Feng George Yu for taking time from his busy schedule to serve in the committee and supporting me on many issues.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

The transmission electron microscope (TEM) operates on the same basic principles as the optical microscope but using electrons instead of photons as its light source. Because electrons have much lower wavelengths, a TEM can achieve an image resolution that is thousand times higher than that of an optical microscope. A typical TEM system consists of three essential subsystems, namely an electron gun, image producing subsystem and image recording subsystem. In a TEM, electromagnetic lenses replace optical lenses and images are viewed on a screen rather than through an eyepiece. Transmission electron microscopy illuminates a sample with electrons within a high vacuum to produce images and receives the electrons that are transmitted through the sample. The image recording subsystem has a fluorescent screen for viewing and focusing the object and a digital camera for permanent recording. The advanced high-resolution TEM has an ability to capture atomic structure in nanomaterials with a resolution as low as 0.10 nm and with sensitivity at the single-atom level. In addition, an environmental TEM system is capable of characterizing a sample's morphologic, compositional, crystallographic, as well as its *in situ* reactive chemical properties [1, 2].

Because of the aforementioned features, TEM's has vast applications in many fields such as forensic analysis, biological tissue engineering, pharmaceutical quality control, 3D printing system, as well as the inspection of large scale integrated circuits and chips. This thesis studies the TEM images of nanoparticles used as chemical catalysts in air quality control, particularly the cube-shaped nanocrystals [3, 4].

1.1 MOTIVATIONS OF USING ADABOOST ALGORITHM

Nano research using TEMs has generated a large quantity of images and each image contains GB or TB digital information, leading to a typical situation of “big data” that cannot be processed and analyzed by human experts or specialists. Further analysis of the complex relationships among nanomaterial properties such as shape, size, effective surface contact area, crystal forming temperature, growth rate and chemical reactivity demands more sophisticated deep mining methods. Detecting and counting the number of particles in a TEM image is the first and the most critical step of performing a deep mining task. The presence of image noise and objects overlapping causes much challenges to the methods that rely upon geometric and photometric cues. An ensemble learning algorithm (Cascade AdaBoost) is chosen because its robustness and efficiency in handling difficult objects has been demonstrated in many applications, especially in the area of face detection.

1.2 TECHNICAL CHALLENGES

The main challenges encountered in this project are summarized as follows:

(1) Many particles are severely overlapped i.e., the well-known occlusion problem in computer vision and object detection research.

(2) Due to the large number of particles involved and the occlusion problem, class labeling errors in extracting positive samples (ground truth) become an issue. In other words, ambiguity in labeling accuracy must be taken into account.

(3) Computational cost is very high because a large number of features is needed to train a Cascade AdaBoost classifier that can perform well in the testing phase.

1.3 CONTRIBUTIONS

The contribution of this study is four-fold: (i) This is the first investigation of using an ensemble learning method to automatically detect cube-shaped nanoparticles in TEM images; (ii) A baseline performance of two-class classification is established using an extended set of Haar features; (iii) The potential impacts of multi-label-set, sample size and feature size on the classifier's performance are examined; (iv) The preliminary experiments show that the proposed method is very promising in detecting cube-shaped objects.

2. METHODS

2.1 CASCADE ADABOOST ALGORITHM

Adaboost is considered as an ensemble learning method (meta-algorithm) that is composed of many weak classifiers. Each classifier performs a simple task according to one dimensionality of the input vector [5]. Having many weak classifiers, the detection rate of an AdaBoost algorithm is improved but it also requires a long training time with a potentially high false alarm rate. To deal with these issues, Viola and Jones [6] proposed a Cascade-Adaboost approach as shown in figure 1. In a cascade architecture, *Neg* represents the number of negative sub-windows (objects) rejected and *Pos* indicates the number of positive sub-windows (objects) accepted. *X* is the input set, which includes both negative and positive samples. During a training, if a sub-window is determined negative, it is removed from the original training set, thus reducing the number of samples as the cascade stage gradually increases [7, 8].

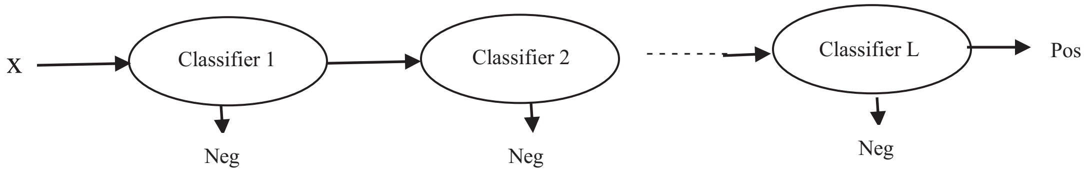

Figure 1: Cascade AdaBoost classifier

Before the training of a cascade classifier, a few parameters need to be set: d as a minimum detection rate, f as a maximum false alarm rate in each stage, and F_{target} as a target false alarm rate. Given two sets of samples (P for positive and N for negative), a training is done in two loops. Inside the inner loop, the target value will be checked each time a weak classifier is added. The training is completed at the stage where the overall target is reached. If the false alarm rate F_i is below F_{target}, the training is terminated, otherwise negative samples will be reset and false detections are put in set N. The external loop is repeated to train the cascade classifier for the next stage until the overall false alarm rate is below F_{target}. Figure 2 illustrates the entire training procedure [9, 10].

```
• $f$: maximum acceptable false alarm rate at a stage.
• $d$: minimum acceptable detection rate at a stage;  $F_{target}$: overall target false alarm rate.
• $P$: positive sample set;  $N$: negative sample set.
• $F_0$=1; $D_0$=1; i=0.
• while $F_i > F_{target}$
    o $i = i + 1$; $n_i = 0$; $F_i = F_{i-1}$
    o while $F_i > f$ x $F_{i-1}$
        ❖ $n_i = n_i + 1$.
        ❖ use $P$ and $N$ to train a classifier with $n_i$ features
        ❖ check current classifier on validation set to determine $F_i$ and $D_{i.}$
        ❖ determine threshold for the $i$th classifier until the current cascade classifier
          has a detection rate > ($d$ x $D_{i-1}$)
    o $N$ is NULL.
• if $F_i > F_{target}$ then evaluate the current cascaded classifier on the set of negative samples and put
  false detections into set $N$.
```

Figure 2: Cascade Adaboost training procedure.

2.2 FEATURE COMPUTATION

Integral image is an intermediate representation of an input image. It can be used to compute rectangle features (Haar features) rapidly [6, 11]. The integral image at location (*x, y*) represents the sum of the pixels above and to the left of (*x, y*) as shown in figure 3.

The integral image at location (x, y) can be represented mathematically as:

$$ii(x, y) = \sum_{\mathrm{z}^1 \leq \mathrm{z},\, \mathrm{y}^1 \leq \mathrm{y}} i(x^1, y^1) \tag{1}$$

where *ii(x, y)* is the integral image and *i (x ,y)* is the original image. The integral image can be computed in one pass over the original image using the following equations:

$$s(x, y) = s(x, y - 1) + i(x, y) \tag{2}$$

$$ii(x, y) = ii(x - 1, y) + s(x, y) \tag{3}$$

where *s(x, y)* is the cumulative row sum with *s(x, -1)* = 0 and *ii(-1, y)* = 0.

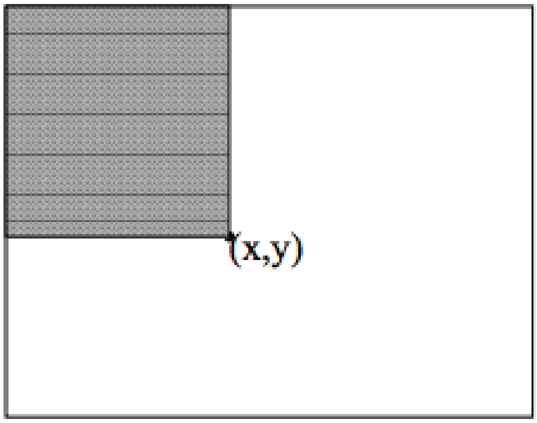

Figure 3: Integral image at point (x, y)

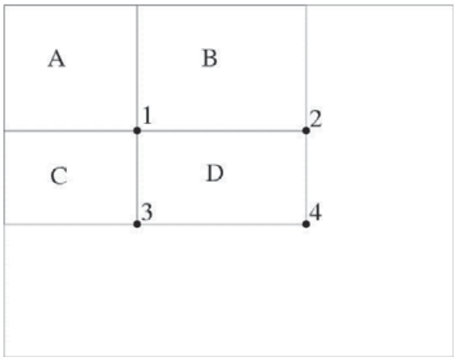

Figure 4: Computing rectangle features using integral image.

Given the integral image representation of an image, the values of rectangular features such as the regular and extended set of Haar features can be obtained in a constant time. As shown in Figure 4, if the integral image values at four points are known, the integral sum inside rectangle region of D can be computed using the following formula:

$$ii(4) + ii(1) - ii(2) - ii(3) \qquad (4)$$

2.3 OCCLUSION HANDLING

Various methods have been proposed to handle the occlusion problem, which is an instance when a part of the object or the whole object is blocked by other objects [12, 13, 14]. The commonly used cubes include the shape prior, trajectory of a moving object and color consistency or difference. However, the occlusion in TEM images is unique in the sense that the objects are semi-transparent. In other words, the intensity values reveal the degree of occlusion (layers of overlapping). Therefore, an implicit approach is adopted that counts overlapped objects as a single one without a separate labeling. In the future work, a more explicit counting method will be considered.

3. DATA PROCESSING

3.1 DATA SET

The TEM images used for the training and testing are of 20-50 nm scales (see Figures 5, 6, and 7). The image set and cube objects count information is summarized in Table 1. All of the objects are classified into three basic categories: (i) the internal objects: fully visible boundaries and not or slightly overlapped; (ii) the overlapped objects: fully visible boundaries and largely overlapped (>20% area); (iii) the boundary objects: across the image boundaries and fully or partially visible. Since the boundary objects were not cropped and included in the positive training sample sets, they were not counted in the detection rate calculations. In addition, since an implicit approach was used for the occlusion handling, the fully overlapped objects (> 70% overlapping area) were counted as one detection hit.

TABLE 1: IMAGE SET AND OBJECTS

Images	**Resolution**	**No. of objects**	**No. of internal objects**	**No. of overlapped objects**
cube_a	1002 X 668	79	66	23
cube_b	1002 X 668	72	53	9
cube_c	1002 X 668	235	191	67

3.2 SAMPLE PREPARATION

3.2.1 Positive Samples (Label Sets)

Multiple students have worked on a machine learning algorithm as a part of their class projects or independent studies. They used the cube images to run and learn the algorithm. They first cropped cubes from the TEM images and then used the cubes as input data for their own project experiments. Cropping was done using a GIMP software which is a free image editing tool available in both Linux and Windows environments. Before cropping, a cube was rotated so that its boundary lines were aligned vertically or horizontally. The cropped cubes were then saved in the standard jpg format. Cropping is a very simple task that does not require a special training. This thesis utilizes those already cropped cubes as positive samples (referred henceforth as “label sets” also) to train the Cascade AdaBoost classifier. Table 2 and Figure 8 show the label sets. It should be noted that (i) a student cropped cubes based on his or her own judgement and need and hence the object counts could be different from the actual number of objects as shown in Table 1; (ii) this thesis does not evaluate students’ cropping results and performances; (iii) this thesis only assesses the detection accuracy of the Cascade AdaBoost method given different cropped cubes as the positive samples (label sets).

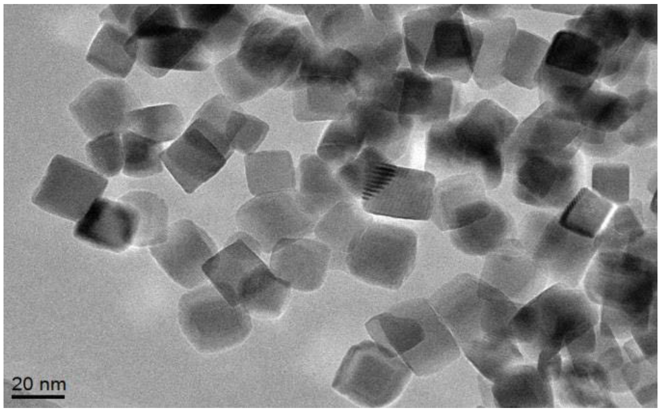

Figure 5: A sample image (cube_a).

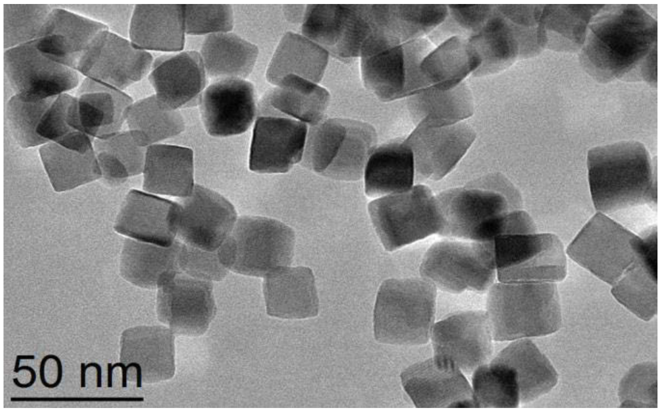

Figure 6: A sample image (cube_b).

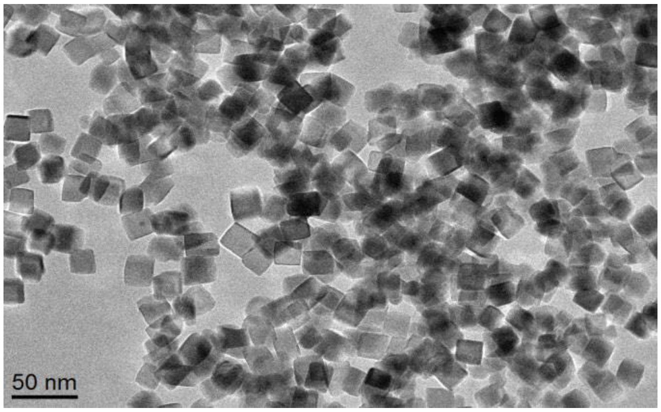

Figure 7: A sample image (cube_c).

TABLE 2: POSITIVE SAMPLES (LABEL SETS)

Label set	Image	No. of Cropped Objects
Label set 1	cube_a	79
Label set 2	cube_a	70
Label set 3	cube_b	72
Label set 4	cube_b	71
Label set 5	cube_b	58
Label set 6	cube_c	235
Label set 7	cube_a	60
Label set 8	cube_c	215
Label set 9	cube_c	71

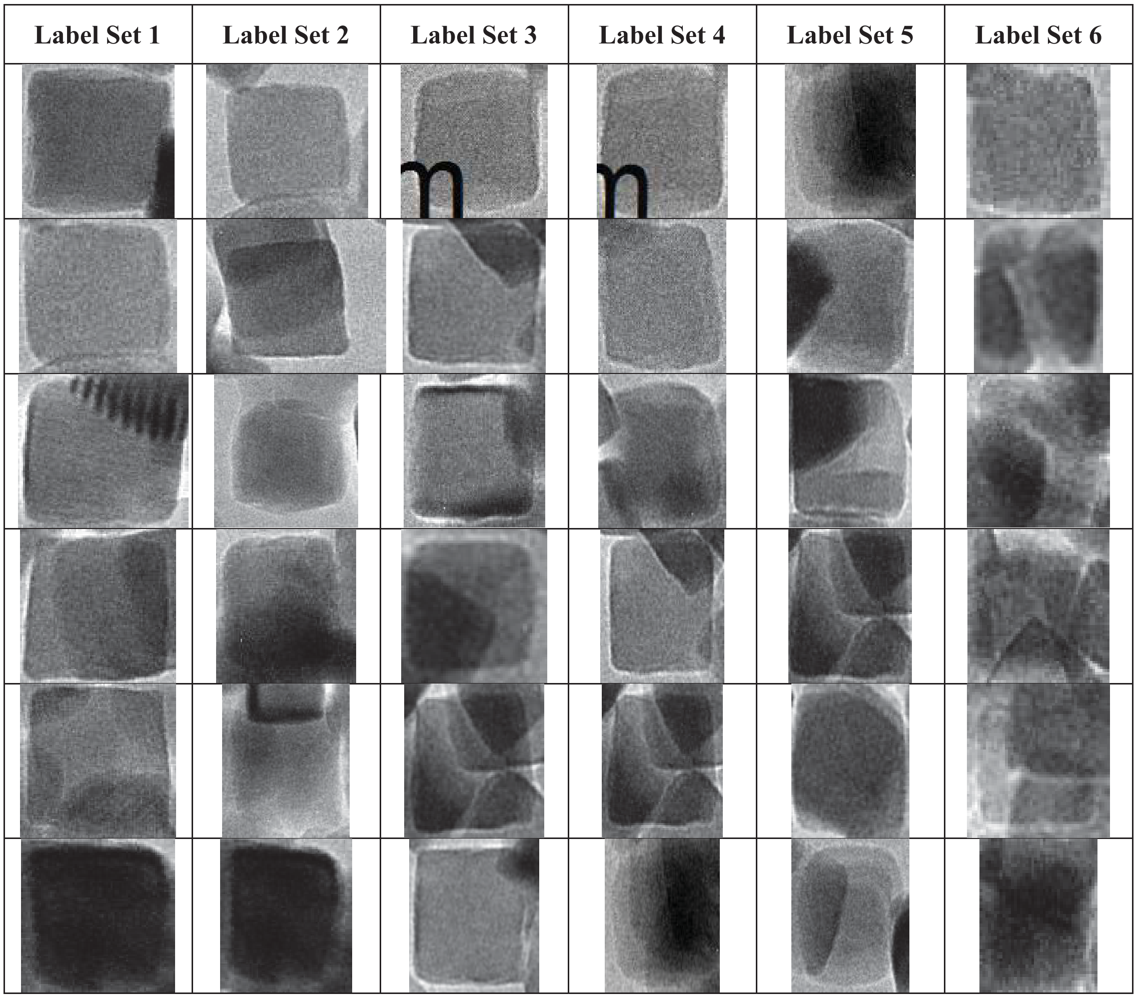

Figure 8: Cropped cube samples of different label sets.

3.2.2 Negative Samples

A total of 770 negative samples were generated by slicing a few random images of various background scenes (including those of similar intensity distributions to that of TEM images, see Figure 9). All negative samples are of the same size that is slightly larger than that of positive samples. The diversity of the negative samples will help reduce the false alarm rate and hence improve the robustness of the classifier when tested with unseen images.

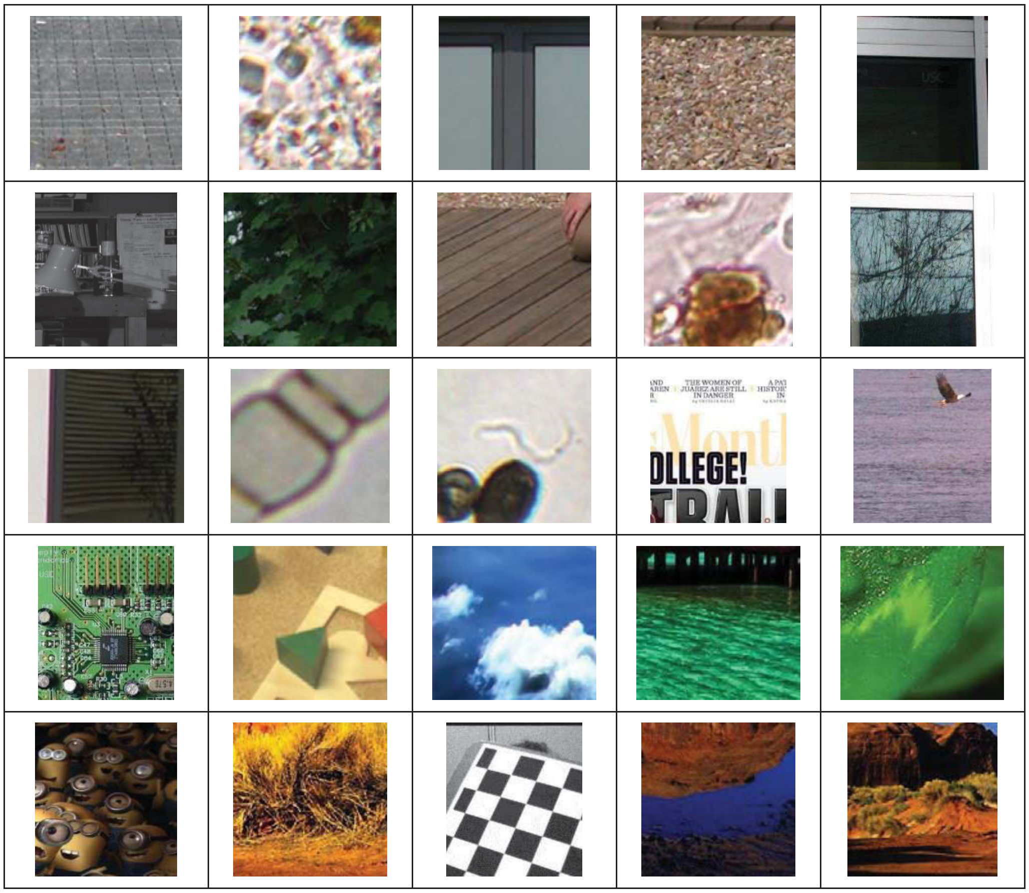

Figure 9: A few negative samples used in the training.

4. EXPERIMENTS

Experiments were carried out using the OpenCV packages installed in a Linux system. The training of a cascade classifier was performed in four steps: (1) Select all images used in training; (2) Create positive training samples; (3) Merging individual training files into a single one; (4) Train the cascade classifier.

(1) Select all images:

Two files are generated that lists positive and negative images, respectively. A subset of negative samples can be selected depending on the test design.

```
find ./posImgDir -name "*.jpg" | sort -V -f > posImgList.txt
find ./negImgDir -name "*.jpg" | sort -V -f > allNegImgList.txt

#=======Select a subset from allNegImgList.txt=========
numNeg=222
k=0
while read varLine
do
     echo "$varLine" >> negImgList.txt
     ((k++))
     if [ $k -ge $numNeg ]
        then
        break
     fi
done < allNegImgList.txt
```

(2) Create positive training samples:

Positive samples were positioned vertically and horizontally during the cropping process. However, the actual objects in the test image could be of any rotation angle. Therefore, multiple positive samples of various degrees of rotation must be generated. To do this, a rotation angle of 360 degrees around the *z*-axis was specified. In addition, the total number of positive samples (actual positive samples multiplied by a user selected number) and the window size (*h, w*) also need to be specified. For example, if a dataset has 120 positive images and each image can generate 8 training samples of various rotation degrees, then a total of 120 * 8 = 960 training samples will be created.

```
perl  ./bin/step2.pl\
      posImgList.txt\
      negImgList.txt\
      vecSampleDir\
      960\
      "opencv_createsamples\
      -bgcolor       0\
      -bgthresh      0\
      -maxxangle 0.005\
      -maxyangle 0.005\
      -maxzangle 3.141\
      -maxidev       3\
      -w            20\
      -h            20"
```

(3) Merging individual training files into a single one:

All individual positive samples can be merged into a single file that has the vector format defined in the OpenCV packages.

```
find ./vecSampleDir/posImgDir -name '*.vec' | sort -V -f >
./vecSampleDir/vecList.txt
./bin/mergevec ./vecSampleDir/vecList.txt
./vecSampleDir/allPositiveSamples.vec
```

(4) Train the cascade classifier:

During the training, several key parameters must be specified: number of positive and negative samples used, number of stages, window size (*h, w*) which should be the same as those in step (2), minimum hit rate, maximum false alarm rate, and weight trim rate. It should be noted that the number of positive samples should be 80-90% of number specified in step (2). The overall false alarm rate is calculated as a product of maximum false alarm rate and number of cascade stages.

```
opencv_traincascade -data trainedClassifier\
                    -vec  ./vecSampleDir/allPositiveSamples.vec\
                    -bg   negImgList.txt\
                    -numPos              770\
                    -numNeg              222\
                    -numStages            20\
                    -precalcValBufSize   512\
                    -precalcIdxBufSize   512\
                    -stageType         BOOST\
                    -featureType        HAAR\
                    -w                    20\
                    -h                    20\
                    -bt                  GAB\
                    -minHitRate        0.996\
                    -maxFalseAlarmRate 0.500\
                    -weightTrimRate    0.950\
                    -maxDepth              1\
                    -maxWeakCount        100\
                    -mode                ALL
```

5. RESULTS AND DISCUSSIONS

5.1 BASELINE PERFORMANCE ANALYSIS

5.1.1 Best Detection Rates

To assess the optimal capability of Cascade AdaBoost method, a comparative study of two tests was conducted: (i) the first test used positive samples of cube_a image (label set 1) as the training set and cube_b image as the test set; (ii) the second test used positive sample of cube_b (label set 4) as the training set and cube_a image as the test set. The same parameter values were used in two tests to ensure a fair compassion. The results are given in Table 3 and shown in Figures 10 and 11. The first test gave a better detection rate. There are two possible explanations: (a) cube_a had more diverse objects in terms of their shape, size, and overlapping degree, which provided a richer training set; (b) The positive samples in label set 1 have tight background margins, which might help the classifier select more discriminative features. Note that only the internal objects are counted and overlapped objects are counted as one hit.

TABLE 3: BEST DETECTION RATE

Training sets	**Test sets**	**Detection rate**	**False alarm rate**
cube_a (label set 1)	cube_b	94.34%	0%
cube_b (label set 4)	cube_a	87.88%	0%

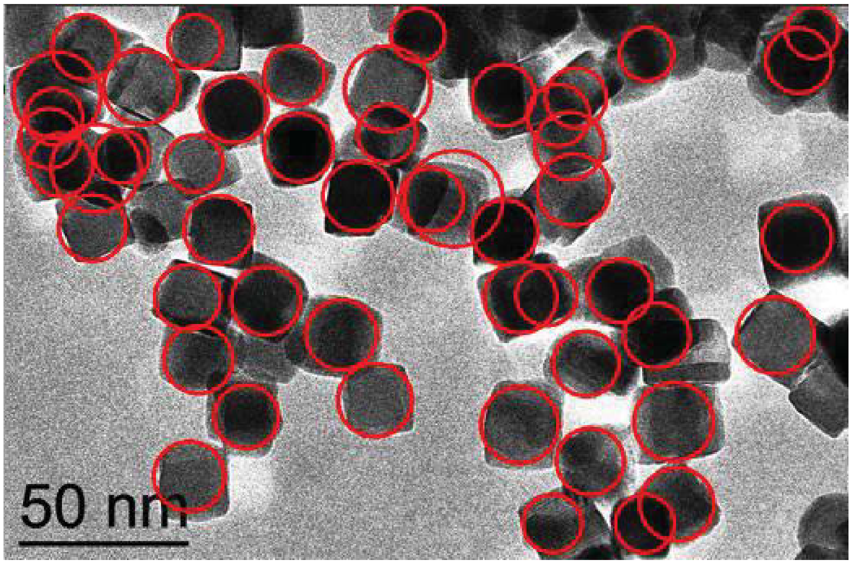

Figure 10: Best detection result in cube_b using label set 1.

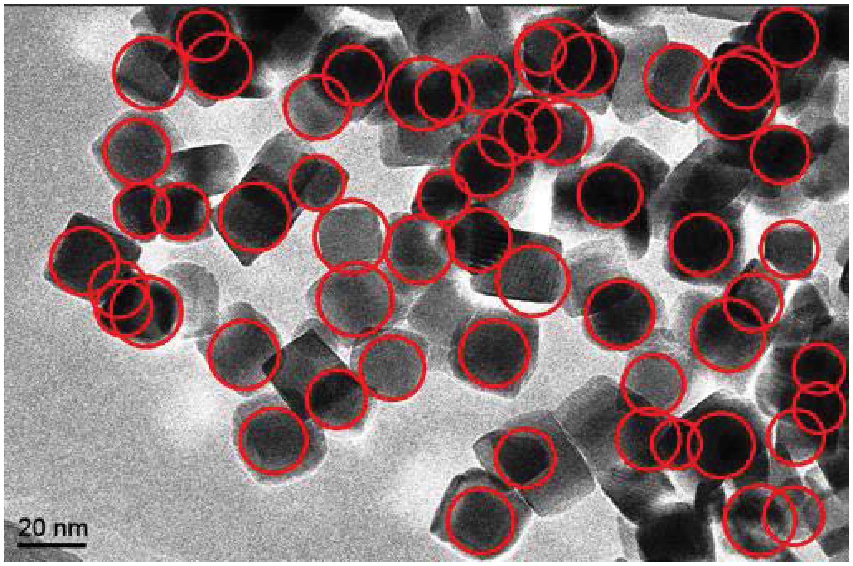

Figure 11: Best detection result in cube_a using label set 4.

5.1.2 Impact of sample size

One of the common factors that could greatly affect the training quality of a cascade classifier is the number of positive samples used. It is generally believed that a large sample size will likely improve the classification performance, but at the cost of a much longer training time and hence the requirement of more computing resources. Table 4 lists the results of a series of tests that used the positive samples (label set 6) from cube_c image as the training set and cube_b image as the test set. The detection rate, false alarm rate and training time were plotted against the number of samples in Figures 12 and 13. As expected, the training time grew rapidly (almost linearly) with the sample size. It should be noted that the results were obtained using a much simplified test design such as a smaller window size, a small number of stages and a low minimum hit rate. If normal parameter values were used, the training time would be several orders of magnitude longer (in the range of a few days to weeks). Another important observation is that the detection rate only improved mildly with an increasing false alarm rate as the sample size grew, suggesting that the performance gain from a larger sample size would likely reach a plateau and the benefit would be negated by a higher false alarm rate and a longer training time.

TABLE 4: IMPACT OF SAMPLE SIZE

No of objects in the training set (cube_c)	**Test sets**	**Detection rate**	**False alarm rate**	**Training time**
100	cube_b	81.13%	0%	52 seconds
250	cube_b	84.90%	0%	60 seconds
500	cube_b	92.45%	0%	101 seconds
750	cube_b	87.13%	0%	143 seconds
1000	cube_b	90.57%	3.77%	178 seconds

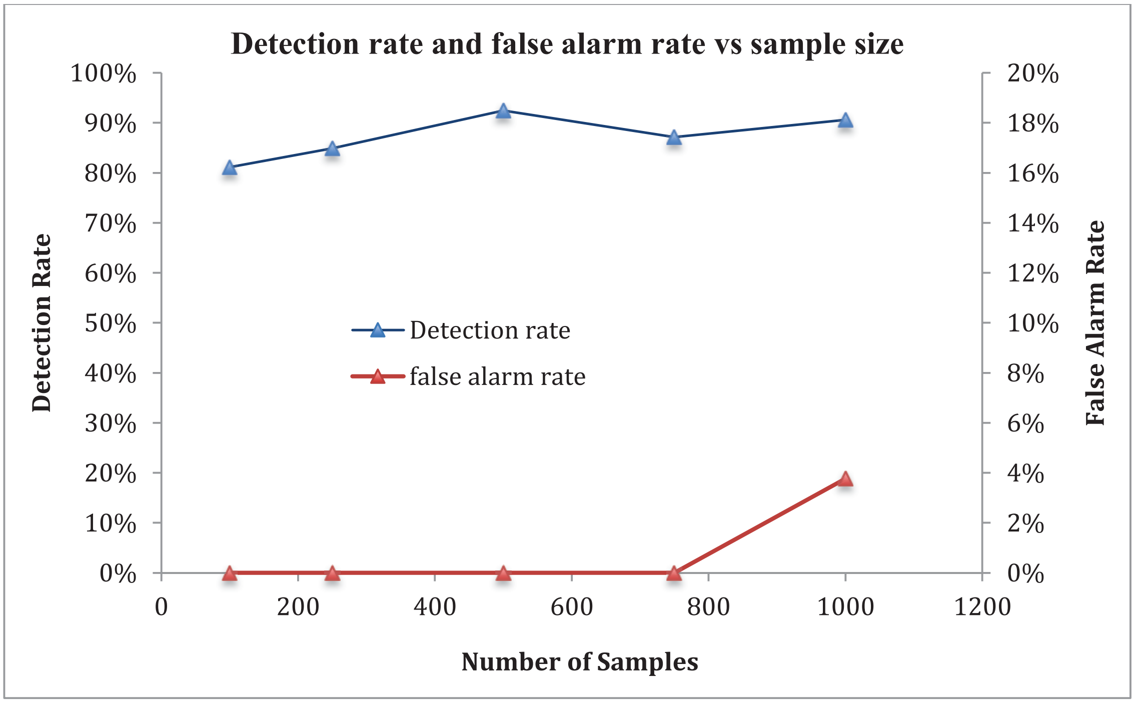

Figure 12: Impact of sample size on detection rate and false alarm rate.

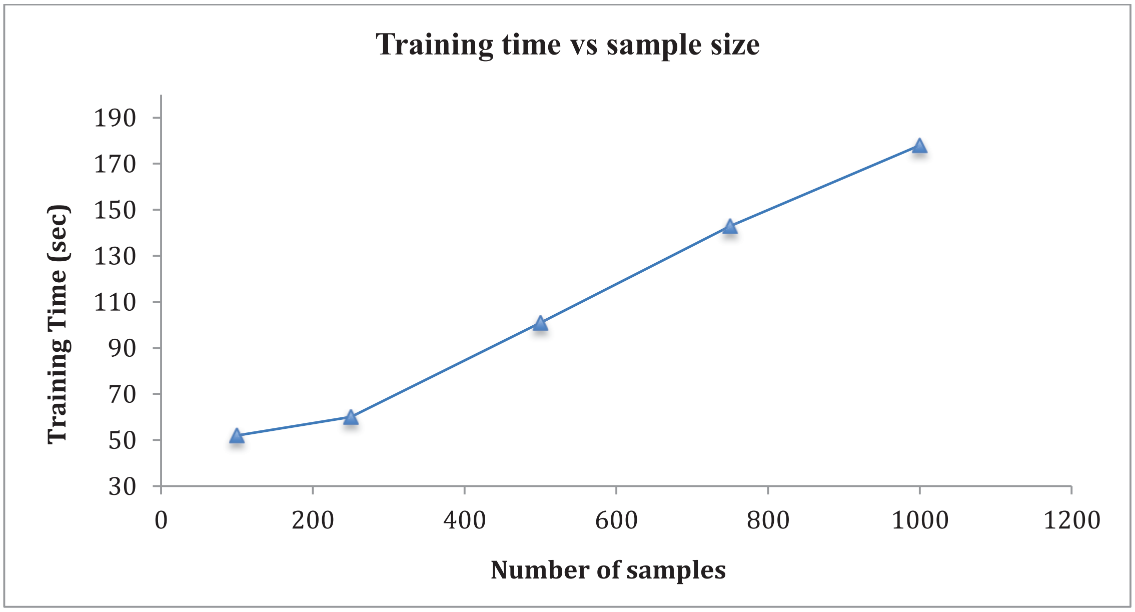

Figure 13: Impact of sample size on the training time.

5.1.3 Impact of window size:

In many data mining projects, a common dilemma is that a small or moderate set of positive samples is often accompanied with a high dimensional feature space, which leads to the phenomenon of the curse of dimensionality. For example, a 20 by 20 sliding window in the Cascade AdaBoost classifier can generate hundreds of thousands of Haar features computed for all of the derived sub-windows. As a consequence, the feature size poses a much more serious challenge to the design of a classification task than the sample size. In fact, the Cascade AdaBoost algorithm can be viewed as a semi-feature selection method that finds the most discriminative feature subset (weak classifiers) through a sequence of weighted decision tree tests. To evaluate the impact of window size, i.e., feature size, on the classifier's performance, four tests were conducted that used the same training/test sets as in Section 5.1.2. The window sizes and test results are given in Table 5 and plotted in Figures 14 and 15. It is clear that the training time increases almost exponentially as the window size grows, much faster than the rate in the tests with an increasing sample size. At the same time, a 20% increase in detection rate was observed, indicating that the feature size is a much more influential parameter.

TABLE 5: IMPACT OF WINDOW SIZE

Training set (cube_a)	**Test sets**	**Detection rate**	**Training time**
$h = 10, w = 10$	cube_b	56.60%	6 seconds
$h = 15, w = 15$	cube_b	67.92%	27 seconds
$h = 20, w = 20$	cube_b	69.81%	79 seconds
$h = 25, w = 25$	cube_b	81.13%	161 seconds

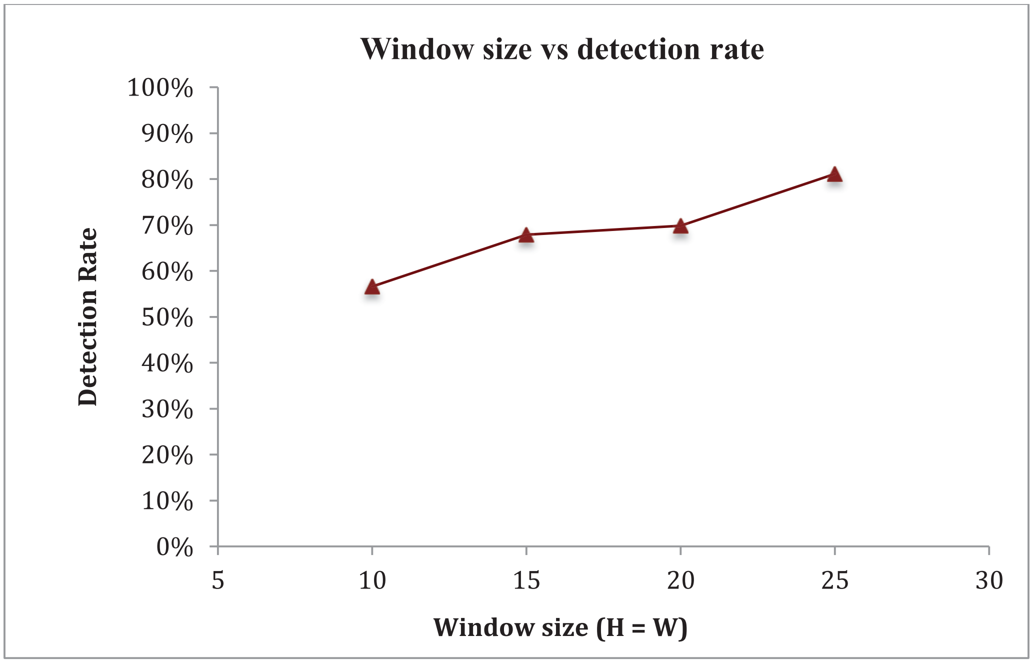

Figure 14: Impact of window size on detection rate.

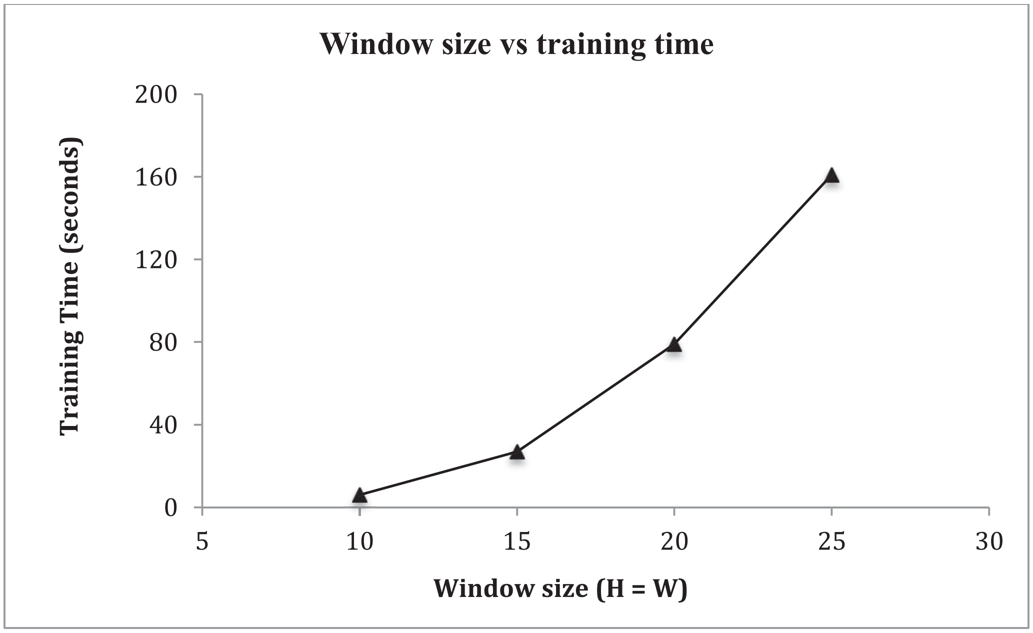

Figure 15: Impact of window size on training time.

5.2 LABEL VARIATION ANALYSIS

5.2.1 Two-label-set comparison:

Since multiple label sets (cubes cropped by different labelers) are used, it is inevitable that labeling errors will be introduced in the training set. Therefore, it is essential to have a quantitative evaluation of the impact of label variation on the classifier's performance. Two evaluation tests were conducted, a two-label-set test and a three-label-set test. To ensure a fair comparison, all tests used the same parameter values (see box below). The two-label-set test used positive samples of cube_a (label set 1 and label set 2) as the training sets, and cube_b image as the test set. The test results are given in Table 6 and shown in Figures 16 and 17.

```
posImgList.txt\
negImgList.txt\
vecSampleDir\
 210\
"opencv_createsamples\
-bgcolor        0\
-bgthresh       0\
-maxxangle  0.005\
-maxyangle  0.005\
-maxzangle  3.141\
-maxidev        3\
-w             20\
-h             20"
```

```
-numPos              160\
-numNeg              150\
-numStages            20\
-precalcValBufSize   512\
-precalcIdxBufSize   512\
-stageType         BOOST\
-featureType        HAAR\
-w                    20\
-h                    20\
-bt                  GAB\
-minHitRate        0.996\
-maxFalseAlarmRate 0.500\
-weightTrimRate    0.950\
-maxDepth              1\
-maxWeakCount        100\
-mode                ALL
```

TABLE 6: TWO LABEL SET COMPARISON TEST

Training Set	Test Set	Detection Rate	False Alarm Rate
cube_a (label set 1)	cube_b	92.45%	1.88%
cube_a (label set 2)	cube_b	84.90%	5.66%

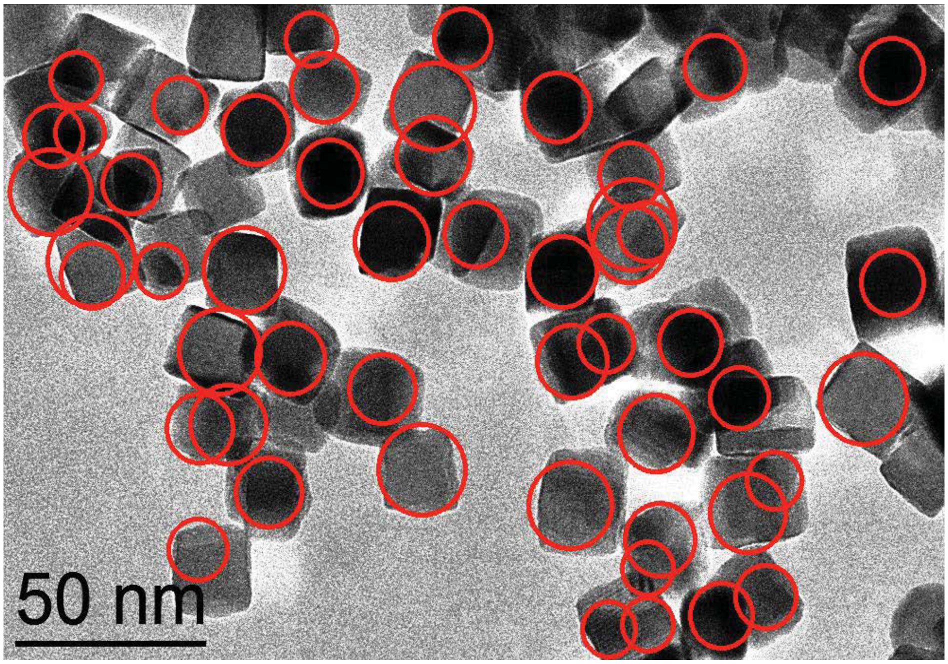

Figure 16: Detection result of cube_b using the label set 1.

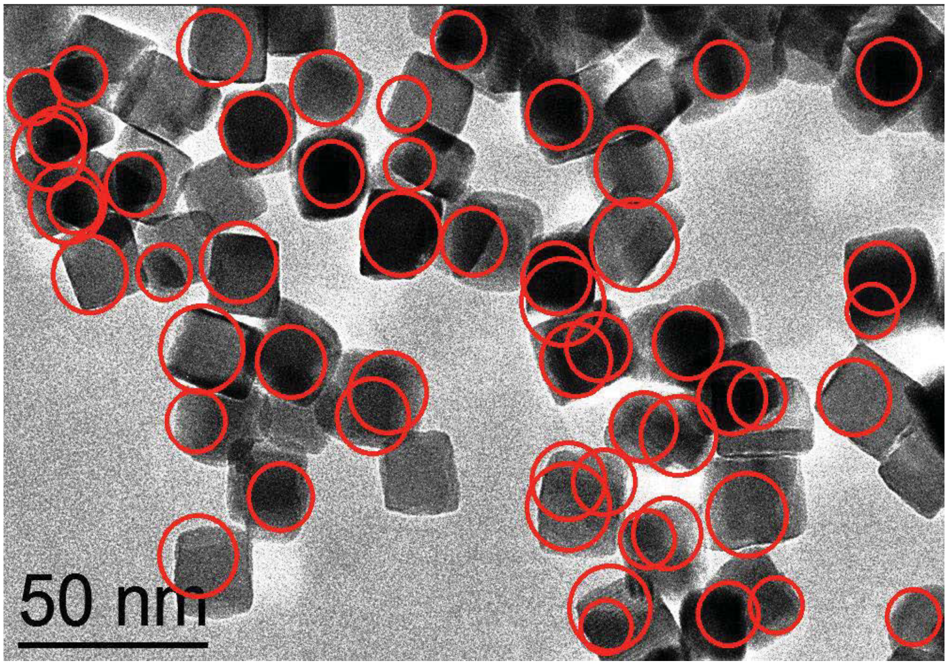

Figure 17: Detection result of cube_b using the label set 2.

The test of using positive samples of label set 1 showed a higher detection rate and a lower false alarm rate than the test of using the positive samples of label set 2. A close examination of the detection results (Figure 16 and Figure 17) indicates that the test of label set 1 was able to detect objects of larger sizes while the test of using label set 2 had a tendency of showing multiple hits on overlapped objects. Since two tests used exactly the same parameter values in both training and testing phases, the performance discrepancy is likely attributed to the ways of selecting and cropping positive samples. A few representative positive samples from the two label sets are shown in Figure 18. It is clear that the samples of label set 2 contain much larger margins (background areas) surrounding a cube. The presence of background intensities could generate “noisy” and “confusing” Haar features that weaken the discriminative power of the true cube features. However, the exact mechanism by which the detection rate and false alarm rate were affected by the margins is still not clear and more work is needed.

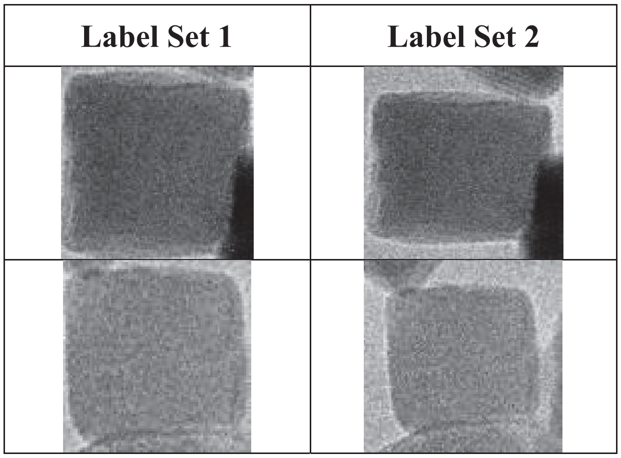

Figure 18: A few selected positive samples from two label sets.
The samples of label set 2 have much wider margins than that of label set 1.

5.2.2 Three-label-set comparison:

To further assess the impact of multiple labeling sources on detection accuracy, a three-label-set comparison test was conducted. In this test, positive samples of three label sets (different from label set 1 and label set 2 in the two-label-set test) from cube_b image were used as the training set and cube_a image was used as the test set. The same parameter settings as in the two-label-set test were used except the numbers of positive samples (see Table 7). The test results are given in Table 7 and the detected objects are shown in Figures 19, 20, and 21.

The detection rate of using the data of label set 3 is comparable to that of using the data of label set 1 and label set 2. But the results of using the data of label set 4 and label set 5 are much less accurate. The low detection rate of label set 5 is likely caused by the smaller set of positive samples used, as being observed in the impact test (Figure 12). The samples of label set 4 had slightly larger cube margins which may explains the low detection rate. Based on the two-label-set and three-label-set comparison tests, it seems that the inclusion of background scene in the positive samples is a major factor affecting the classifier's performance. A more comprehensive evaluation study with more data samples is needed to provide a statistically meaningful conclusion.

TABLE 7: THREE-LABEL-SET COMPARISON TEST

Training Sets	**Positive Samples**	**Test Sets**	**Detection Rate**	**False Alarm Rate**
cube-b (label set 3)	70	cube_a	83.33%	0%
cube-b (label set 4)	70	cube_a	71.21%	0%
cube-b (label set 5)	58	cube_a	60.60%	0%

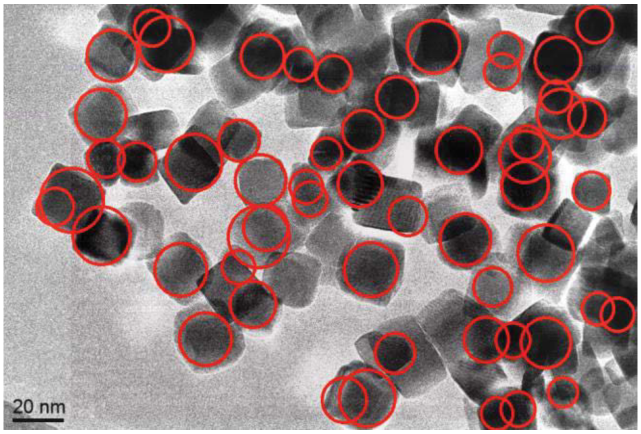

Figure 19: Detection result of cube_a using the positive samples of label set 3.

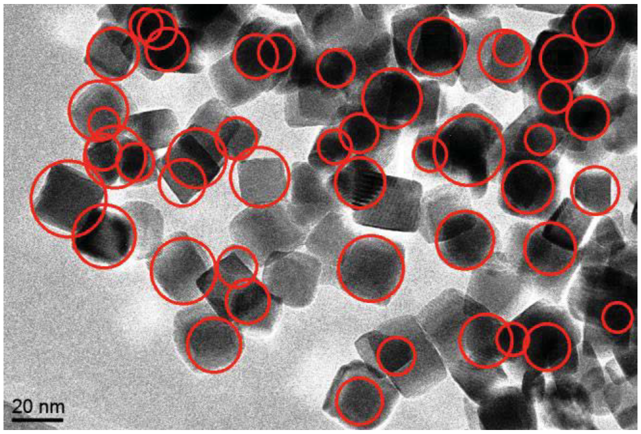

Figure 20: Detection result of cube_a using the positive samples of label set 4.

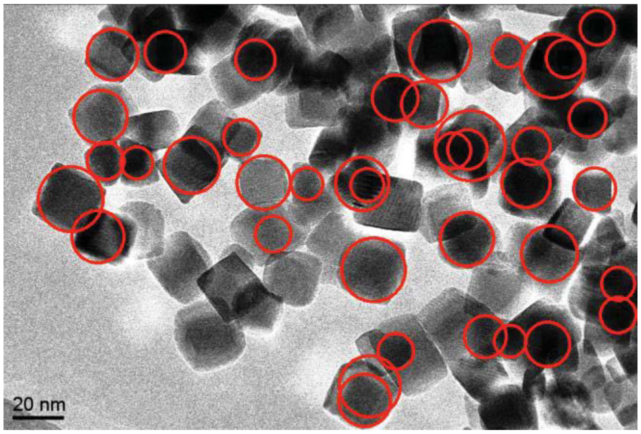

Figure 21: Detection result of cube_a using the positive samples of label set 5.

6. CONCLUSIONS

This thesis investigates the feasibility of using an ensemble learning algorithm to automatically detect cube-shaped nanoparticles in TEM images, which has a significant implication to scientific research in chemistry and material engineering. Preliminary experiments using limited positive samples delivered promising results, with the best hit rate of 94.34% being achieved. The primary findings are summarized below:

- The Cascade AdaBoost algorithm is effective in handling a large number of objects at the presence of image noisy and severe occlusions.
- The classifier's performance was affected by sample size and feature size. It is expected that a large sample set will improve the detection rate while the effect of feature size remains a complicated issue. How to select an optimal feature subset outside of the cascade training process is an interesting topic.
- It has been observed that a large detection rate variation exists with multiple label sets. A more comprehensive evaluation study is needed to quantify the impact of different label sources.
- Last but not least, the Cascade AdaBoost training is very expensive. Certain tests were not carried out due to the lack of computing resources. There is a strong need of a powerful computing infrastructure. One alternative solution is to utilize a GPU based method or a cloud computing framework.

7. REFERENCES

[1] J. R. Jinschek and S. Helveg, “Image resolution and sensitivity in an environmental transmission electron microscope,” *Micron*, vol. 43, no. 11, pp. 1156–1168, Nov. 2012.

[2] Z. L. Wang, “Transmission Electron Microscopy of Shape-Controlled Nanocrystals and Their Assemblies,” *J. Phys. Chem. B*, vol. 104, no. 6, pp. 1153–1175, Feb. 2000.

[3] R. Wang and M. Fang, “Improved low-temperature reducibility in ceria zirconiananoparticles by redox treatment,” *J Mater Chem*, vol. 22, no. 5, pp. 1770–1773, 2012.

[4] R. Wang, P. A. Crozier, R. Sharma, and J. B. Adams, “Measuring the Redox Activity of Individual Catalytic Nanoparticles in Cerium-Based Oxides,” *Nano Lett.*, vol. 8, no. 3, pp. 962–967, Mar. 2008.

[5] Y. Freund and R. E. Schapire, “Experiments with a new boosting algorithm,” *Int. Conf. on Machine Learning*, pp. 148–156, 1996.

[6] P. Viola and M. J. Jones, “Robust Real-Time Face Detection,” *Int. J Computer Vis.*, vol. 57, no. 2, pp. 137–154, May 2004.

[7] Z. Ou, X. Tang, T. Su, and P. Zhao, “Cascade AdaBoost Classifiers with Stage Optimization for Face Detection,” in *Advances in Biometrics*, vol. 3832, D. Zhang and A. K. Jain, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 121–128.

[8] R. Lienhart, A. Kuranov, and V. Pisarevsky, “Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection,” in *Pattern Recognition*, vol. 2781, B. Michaelis and G. Krell, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 297–304.

[9] H. Allende-Cid, R. Salas, H. Allende, and R. Ñanculef, “Robust Alternating AdaBoost,” in *Progress in Pattern Recognition, Image Analysis and Applications*, vol. 4756, L. Rueda, D. Mery, and J. Kittler, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 427–436.

[10] R. Lienhart and J. Maydt, “An extended set of haar-like features for rapid object detection,” *Int. Conf. on Image Processing*, pp. 900–903, 2002.

[11] A. Cordiner, P. Ogunbona, and W. Li, “Face detection using generalised integral image features,” *Image Processing (ICIP), 2009 16th IEEE International Conference on* , vol., no., pp.1229,1232, 7-10 Nov. 2009.

[12] Chen, J., Shan, S., Yang, S., Chen, X., & Gao, W. (2006, August). Modification of the adaboost-based detector for partially occluded faces. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on* (Vol. 2, pp. 516-519). IEEE.

[13] Yang, Tao, Quan Pan, Jing Li, and S. Z. Li. "Real-time multiple objects tracking with occlusion handling in dynamic scenes." In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, pp. 970-975. IEEE, 2005.

[14] Y.-Y. Lin, T.-L. Liu, and C.-S. Fuh, “Fast Object Detection with Occlusions,” in *Computer Vision - ECCV 2004*, vol. 3021, T. Pajdla and J. Matas, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 402–413.