Fast Feature Selection and Training for AdaBoost-based Concept Detection with Large Scale DataSets

Shi Chen[1, 2], Jinqiao Wang[1,2], Yang Liu[1,2], Changsheng Xu[1, 2], Hanqing Lu[1,2]
[1]National Lab of Pattern Recognition, Institute of Automation, CAS, Beijing 100190, China
[2]China-Singapore Institute of Digital Media, Singapore, 119615, Singapore
Email: {schen, jqwang, liuyang, csxu, luhq}@nlpr.ia.ac.cn

ABSTRACT
AdaBoost has been proved a successful statistical learning method for concept detection with high performance of discrimination and generalization. However, it is computationally expensive to train a concept detector using boosting, especially on large scale datasets. The bottleneck of training phase is to select the best learner among massive learners. Traditional approaches for selecting a weak classifier usually run in $O(NT)$, with N examples and T learners. In this paper, we treat the best learner selection as a Nearest Neighbor Search problem in the function space instead of feature space. With the help of Locality Sensitive Hashing (LSH) algorithm, the best learner searching procedure can be speeded up in the time of $O(NL)$, where L is the number of buckets in LSH. Compared with the *T (~500,000)*, the *L (~600) is* much smaller in our experiments. In addition, through studying the distribution of weak learners and candidate query points, we present an efficient method to try to partition the weak learner points and the feasible region of query points uniformly as much as possible, which can achieve significant improvement in both recall and precision compared with the random projection in traditional LSH algorithm. Experimental results reveal our method can significantly reduce the training time. And still the performance of our method is comparable with the state-of-art methods.

Categories and Subject Descriptors
H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing

General Terms
Algorithms, Experimentation

Keywords
Concept Detection, AdaBoost, Local Sensitive Hashing.

1. INTRODUCTION
Semantic concept detection is an essential and general classification task that determines whether an image or a video segment is relevant to a given concept. To detect multiple high level concepts, various features were utilized and exhibited excellent performance. However, with the increasing number of features, the bottleneck of computation complexity is gradually prominent. In [1, 2], bag-of-visual-words (BOW) representation was utilized for both scene categorization and object classification. The size of visual vocabulary increased from 1,000 [1] to as high as 20,000 [2]. Meanwhile, more discriminative and complex features, such as geometric blur, ColorSIFT were applied in [3, 4]. In addition, the scale of datasets and number of concepts also increased rapidly in recent years.

Large scale of datasets and rich feature sets provide a good resource for the statistic learning model such as SVM, AdaBoost to develop concept detectors. Particularly, AdaBoost with cascade structure, which is faster in detection than other statistic learning methods due to only assembling few weak learners to detect, is regarded as a promising way for concept detection task [5, 6]. However, one of the greatest obstacles to extensive use of AdaBoost is its time-consuming training process. All these increments of datasets and feature sets further aggravate the time cost in concept detection task. The main time consuming of AdaBoost lies in training and selecting the best learner. Traditional training algorithms usually run in $O(NT)$, with N and T are the example and learner numbers respectively.

Several methods have been proposed to speed up this training process. Guyon *et al.* [7] filtered the feature set by using mutual-information and Pham *et al.* [8] proposed a method to project features set into several directions by some statistics information. However, both methods cannot guarantee the selected learner to be the best one. Wu *et al.* [9] introduced a method to decrease the training time by using *caching*. However, the memory storage requirement was much larger than other methods.

In this paper, we convert the best learner selection of AdaBoost to Nearest Neighbor Search (NNS) in the function space. Thus LSH can be used to speed up the search procedure with the time complexity of $O(NL)$ where L is the number of buckets in LSH. Furthermore, compared with traditional LSH, the distribution of weak learners and candidate query points in function is more certain and regular than the generic data. Hence, we can design a more effective and efficient way to divide the candidate query space and encode the features so as to achieve better detection performance in contrast to the random projection in traditional LSH algorithm.

The rest of this paper is organized as follows: Section 2 presents a method to transfer the best learner selection of AdaBoost to 1-Nearest Neighbor in function space. Section 3 describes a new projection scheme which is more effective than random projection in conventional LSH. The experimental results on concept detection are reported in Section 4. In Section 5, we conclude the paper with future work.

MM'10, 10 October 25-29, 2010, Firenze, Italy.

2. FROM THEN BEST LEARNER SELECTION TO 1-NN

The basic idea of AdaBoost is to train an ensemble of T decision stumps or weak classifiers $h_t(x)$ in the form:

$$f_T(x) = \sum_{t=1}^{T} a_t\, h_t(x)\,. \quad (1)$$

where a_t are weights of weak learners. This is done by greedily selecting $h_t(x)$ and computing a_t for t^{th}-iteration from 1 to T. It is proved [10] that selection of the best learner at t^{th}-iteration in Eq.2 can minimize the upper bound of training error.

$$h_t(x) = \; argmax_{h(x)} \sum_{i=1}^{N} w_i^t h(x_i) y_i \quad (2)$$

where y_i and w_i^t are the label of sample x_i and the weight of x_i at the t^{th}-iteration. Actually, before selecting the best one in all learners, we must choose a best threshold for each learner on current distribution of sample weights. Hence, there is a two-level search for the best learner. The analysis of feature selection is shown in Figure 1. We can find out that the best learner selection $O(NT)$ in the iteration is the bottleneck compared with $O(N)$ to update weights.

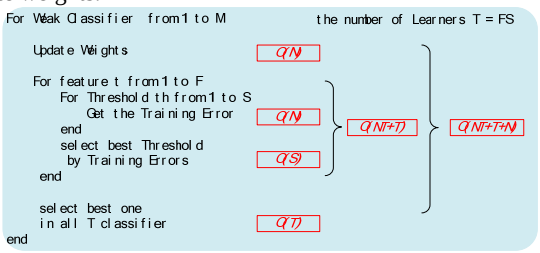

Figure 1.Tradition Weak Learner Selection in AdaBoost.

It is noted that both N and T are not only dominant, but also important in constructing the weak classifiers. In this paper, we present a method that takes a different view to address this problem without trying to reduce either N or T. We convert the linear search of the best learner in the feature space into the Nearest Neighbor Search in the function space. Each learner $h_i(x)$ in the feature space can be projected as an *N-dimension* point P_i in the function space, and we construct a query point Q_t for each iteration *t*. We will prove that the nearest neighbor of Q_t of all the P, donated as P^t, is the best learner in Eq.2.

THEOREM 1 Assuming each learner $h_i(x)$ is projected in function space by $P_i(x)$ $\{h_i(x_1), h_i(x_2),..., h_i(x_j),..., h_i(x_N)\}$ and query point Q_t is built by $\{w_1^t y_1, w_2^t y_2, \dots, w_j^t y_j, \dots, w_N^t y_N\}$, the 1-NN of Q_t in Euclid distance is the best learners in Eq.2.

Proof: The Euclid distance of $P_i(x)$ and Q_t is

$$\begin{aligned} \text{Dist}(P_i(x), Q_t) &= \|P_i(x) - Q_t\|_2 \\ &= \sum_{j=1}^{N} (h_i(x_j) - w_j^t y_j)^2 \\ &= N + \sum_{j=1}^{N} (w_j^t)^2 - \sum_{i=1}^{N} w_i^t h(x_i) y_i \end{aligned}$$

For $h_i(x_j)$, y_j can be either in -1 or +1 in a two-class classification. In each iteration, $\sum_{j=1}^{N} (w_j^t)^2$ is constant for every weak learner. Hence the nearest neighbor $P^t(x)$ is the best learner:

$$\begin{aligned} P^t(x) &= \min_{P_i(x) \in P} \text{Dist}(P_i(x), Q_t) \\ &= \min_{P_i(x) \in P} N + \sum_{j=1}^{N} (w_j^t)^2 - \sum_{i=1}^{N} w_i^t h(x_i) y_i \\ &\propto \max_{P_i(x) \in P} \sum_{i=1}^{N} w_i^t h(x_i) y_i \end{aligned}$$

Concretely, the proposed learner selection is shown in Figure 2. Firstly, before performing AdaBoost loop, the learners are pre-computed and indexed as follows. For each feature and possible threshold, we construct a weak learner $h_{\{i,th\}}(x)$ where $\{i, th\}$ denotes the index and threshold of the feature. Then we get binary classification result $h_{\{i,th\}}(x_j)$ (+1 or -1) by using the weak learner $h_{\{i,th\}}(x)$ to classify the sample *j*. Finally, we project each learner $h_{\{i,th\}}(x)$ to a point of the function space $\Re^N$, represented by $P_{\{i,th\}}$ with the form of ($h_{\{i,th\}}(x_1)$, $h_{\{i,th\}}(x_2)$, ..., $h_{\{i,th\}}(x_N)$), where N is the number of the samples. This projection will consume $O(NT)$ where T is the number of learners, but we only have to compute it once before the loop. And then we index and store all the points $P_{\{i,th\}}$ for query. The store of our method is also very low for we only need store the binary classification result $h_{\{i,th\}}(x_j)$ instead of the all feature values. In the t^{th} iteration, we first compute a virtual query point Q_t by $\{w_1^t y_1, w_2^t y_2, \dots, w_j^t y_j, \dots, w_N^t y_N\}$ where w_j^t is the weights of sample j at the current iteration. Then we will adopt fast K-NN algorithms to speed up the process of the best learner selection.

A lot of promising relevant work [11, 12] has been done to speed up the Nearest Neighbor Search process. One of the most popular approximate algorithms is Locality Sensitive Hashing (LSH) [12]. LSH can reduce the search time to $O(NL)$, where L is the number of LSH buckets. For example with T (~500,000) learners and N(~2,000) examples, we only need L(~600) buckets to get the best learner at probability of 90%.

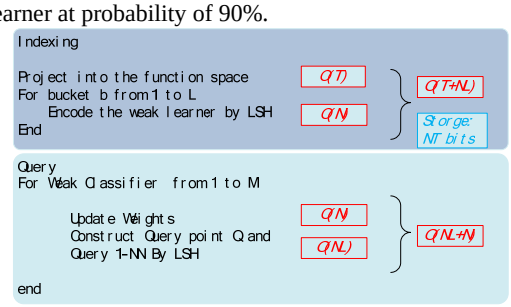

Figure 2.Our Weak Learner Selection By LSH for AdaBoost

3. EFFICIENT LSH BY UNIFORM PARTITION

LSH is a probabilistic technique to solve the approximate NNS problems. Given a corresponding hash family, LSH maintains a number of hash tables containing the points in the dataset. A solution to the K-NN query is found by hashing the query point and scanning the buckets to which the query point is hashed.

Although LSH achieves remarkable performances in high-dimensional similarity search, it suffers from one drawback: its search quality is sensitive to several parameters that are quite data dependent. The sensitivity of data comes from the hash family which is always defined as random projection. As shown in Fig.3, if the data is encoded by a random linear projection followed by a random threshold, it will generate very inefficient hash codes, which contain both lots of collisions and empty buckets. Random projection is the only choice if we don't know the distribution of data. However, we find out two statistical facts of our data which can be used to guide the projection instead of random scheme.

Fact 1. All weak learner points $P_i(x)$ locate on the grid of *{-1, +1}*^{N}, because the decision stump $h_i(x_1)$ is either -1 or +1. Hence, the distribution of these points are much regular than other generic data. It's very easily getting the data divided evenly.

Fact 2. Assume y_i is the data label of the sample *i*, the query points Q_t { $q_1, q_2, \dots, q_N$ } constructed in THEOREM 1 is

constrained to the feasible region: $y_1q_1 + y_2q_2 + \cdots + y_Nq_N$ =1, for $y_i^2 = 1$ and the sum of weights equals to 1. The point C $\{ {}^{y_1}/_N, {}^{y_2}/_N, \ldots, {}^{y_j}/_N, \ldots, {}^{y_N}/_N \}$ will be the center of this feasible region.

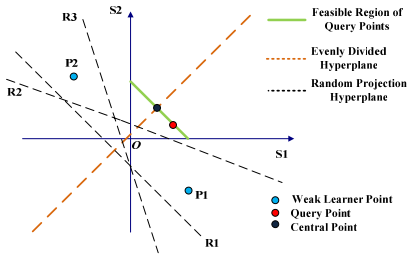

Figure 3.Comparison of two projection methods in LSH

In [14], Weiss *et al.* proposed several rules for perfect encoding. One of most importance rules is to require each bit to enjoy equal chances of being -1 or +1, which means the encoding or projection should be divided evenly as much as possible. Inspired by this rule and the two facts, we design the projection as follows.

Firstly, we randomly choose two points of the learner and a perpendicular hyper plane of the line connecting these two points is chosen to divide the weak learner data efficiently. Then we make this hyper plane through the central point *C* to make feasible region is evenly divided.

A toy example in 2D shown in Figure 3 can illuminate the efficiency of our methods. Assume query point is *Q*, and weak learner points are P_1, P_2. The encoding of random projection R1 is inefficient because P_1, P_2 are collision. Though R2, R3 make an efficient divided of whole function space, these projections are still not good enough for they do not make divided for all possible query points in feasible region, which means that for all possible query points, they will recommend the same points (P_2, P_1 respectively). Compared with the random projection, our evenly divided projection makes a better and efficient encoding for trying to divide both the weak learner points and feasible region evenly. The algorithm for the best learner selection is sketched in Algorithm1.

Algorithm 1. Efficient LSH by Uniform Partition

Given the training data $(x_1, y_1), \ldots, (x_N, y_N)$, where $x_i \in \chi$ and $y_i \in \{+1, -1\}$.

Indexing Process:

Construct weak learner $h_{\{i,th\}}(x)$ and use it to classify all the samples to get $P_{\{i,th\}}$ in function space $\Re^N$ by $\{ h_{\{i,th\}}(x_1), h_{\{i,th\}}(x_2), \ldots h_{\{i,th\}}(x_N) \}$.

For $l = 1, \ldots, L$:

Randomly choose two weak points P_1, P_2 in dataset. Calculate the direction of perpendicular hyper plane by $(h_1(x_1) - h_2(x_1), h_1(x_2) - h_2(x_2), \ldots, h_1(x_N) - h_2(x_N),)$ and calculate the shift to make this hyper plane through $\{ {}^{y_1}/_N, {}^{y_2}/_N, \ldots, {}^{y_N}/_N \}$.

Encode all points by this perpendicular hyper plane.

End

Query Process:

Encode the query points Q_t $\{w_1^t y_1, w_2^t y_2, \ldots, w_N^t y_N\}$.

Get all similar points *P* in the same buckets with Q_t.

Compute all dot product of *P* with Q_t and select the minimum one as the best learner.

4. EXPERIMENTS

To evaluate the performance, we conduct thorough experiments on TRECVID 2008 dataset.The dataset contains about 200 hour videos and is divided into two parts: the training(development) dataset and the test (evaluation) dataset and both contains 219 video files.

Basically, we conduct two experiments to evaluate our proposed approach. In Experiment 1, with large scale extracted features, our LSH-based algorithm is compared with classic AdaBoost algorithm, the training time and performance of selection of weak learner is reported on the selected concept in TRECVID 2008 due to unbearably long time training for original AdaBoost. In Experiment 2, to compared with the state-of-the-art concept detectors [2, 6], our fast AdaBoost algorithm is utilized to train all 20 concept detectors.

4.1 Experimental Setup

For each keyframe, we use BOW [1] to represent and use k-means clustering to generate visual vocabularies. The size of vocabulary is 10,000, which generates the best result through a comprehensive study [2]. In addition, we get 50 thresholds by dividing the range of each feature into 50 sub-range evenly. Hence, a set of 500,000 weak learners is built to generate an optimal classifier in each iteration. In addition, for all LSH-based algorithm, we use the method in [13] to selection the number of the buckets. The Experiments were done on a 2.4G Intel Xeon PC Server with 16G memory.

4.2 Comparison of Training Time

To verify the training speed of proposed approaches, we compare our approach with original AdaBoost algorithm using the five concepts that occur most frequently. The training time is reported in Table.1. We can observe that our method achieves about 12-23 faster in the running time than traditional AdaBoost by converting the best learner selection to the Nearest Neighbor search by LSH.

Table 1. The report training time of two methods

Concept	#Learners	#Samples	Traditional AdaBoost	Our method	Speedup Ratio
Two People	500,000	7,000	156.35s	7.63s	22.28
Hand	500,000	4,500	110.42s	5.43s	20.32
Street	500,000	2,000	46.34s	2.48s	18.65
Boat_ship	500,000	860	17.54s	1.42s	12.87
Nighttime	500,000	800	14.34s	1.25s	11.91

4.3 Comparison of Detection Performance

In Fig.4, we further compare the detection performance of three learning methods with the same number of weak learners on the concept of 'two people' for it contains the most examples in TRECVID 2008. We set the maximal number of weak learners to 500. In addition, due to the training dataset is very imbalanced, we use the method in [6] to under-sample of negative examples to get the size of the negative examples equal to twice size of the positive samples.

From the testing error curves, we can see that the detection performance of the proposed method is almost the same as the original AdaBoost. However, the test error of Random LSH is higher than ours by 2%~3% because the discriminative ability of the learner selected in random LSH sometimes is much lower than the best ones (verified in Fig.5). In Fig.5 the training error of selected learner in three methods at each iteration are compared. For the sake of fairness, the number of buckets is set to 200 for

two LSH-based methods, and in each iteration, all three methods all add the best learner of whole learners to make the same weight distribution for the next iteration. We can observe that the training error of our method is almost as the same as the original AdaBoost. However, the Random LSH is much lower than the best learner for its inefficient encoding scheme.

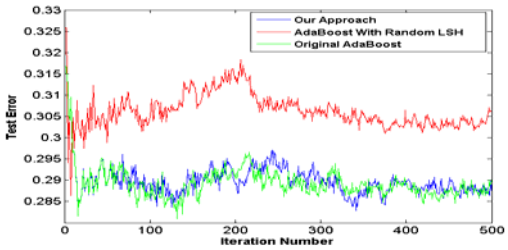

Figure 4. Comparisons of Detection rates of three AdaBoost Based methods on the Concept "Two-people"

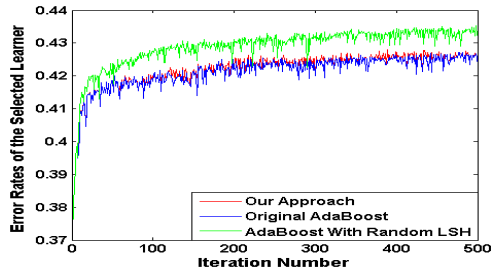

Figure 5. Comparisons of Detection Performance of Weak Learner on the Concept "Two-people"

4.4 Comparison with Other Approaches

We use our fast LSH-based AdaBoost method to train detectors for all 20 concepts in HLFE task of TRECVID 2008. We compare our method with some state-of-the-art methods in TRECVID 2008 as shown in Figure 6. In [2], Jiang use the same BOW features and kernel SVM to classifier the concepts. [6] is one of best results using AdaBoost in TRECVID 2008. We can conclude that our method can achieve comparable results with the state-of-the-art methods with extremely fast training speed. In addition, our method achieves better performance for concepts with more samples, such as *Two People*, *Street* and *Boat*, which shows that our approach is very useful for large scale training dataset.

5. CONCLUSION

We have presented a fast method to train and select the best learner for AdaBoost in concept detection on large scale datasets. It speeds up the weak classifier training time from *O(NT)* to O(NL). By substantially reducing the training time, this method empowers researchers to conduct more quickly experiments and explore solutions to other important research issues in this area.

6. ACKNOWLEDGEMENT

The research is supported by National Natural Science Foundation of China (Grant No.: 60903146, 60905008, 60833006,90920303), and National Basic Research Program (973) of China under contract No.2010CB327905.

7. REFERENCES

[1] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," IJCV 2007.

[2] Y-G. Jiang, J. Yang, C.-W. Ngo and A. G. Hauptmann, "Representations of Keypoint-Based Semantic Concept Detection: A Comprehensive Study," IEEE Transactions on Multimedia, 2010.

[3] S. Petrov, A. Faria, P. Michaillat, A. Berg, D. Klein, and et al., "Detecting categories in news video using acoustic, speech, and image features," in TRECVID workshop, 2006.

[4] C. G. M. Snoek, K. E. A. van de Sande, O. de Rooijand et al., "The MediaMill TRECVID 2009 semantic video search engine," in TRECVID workshop, 2009

[5] W. Jiang, S.-F. Chang and A. C. Loui, "Kernel sharing with joint boosting for multi-class concept detection", CVPR Workshop, 2007.

[6] Y. Peng and J. Yao "AdaOUBoost: adaptive over-sampling and under-sampling to boost the concept learning in large scale imbalanced data sets". Multimedia Information Retrieval 2010

[7] I. Guyon and A. Elisseeff. "An introduction to variable and feature selection". J. Mach. Learn. Res., 2003.

[8] M.T. Pham and T.J. Cham. "Fast training and selection of Haar features using statistics in boosting-based face detection". In ICCV 2007.

[9] J. Wu, S.C. Brubaker, M.D. Mullin and J. M. Rehg. "Fast asymmetric learning for cascade face detection." PAMI,2007.

[10] R.E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," in ACCLT, 1998.

[11] J. L. Bentley. "K-d trees for semi dynamic point sets." In SCG , 1990.

[12] P. Indyk and R. Motwani. "Approximate nearest neighbors: towards removing the curse of dimensionality." In STOC '98:

[13] A. Andoni and Piotr Indyk "E2LSH 0.1User Manual" http://www.mit.edu/~andoni/LSH/manual.pdf

[14] Y. Weiss, A. Torralba, R. Fergus."Spectral Hashing." In NIPS, 2008.

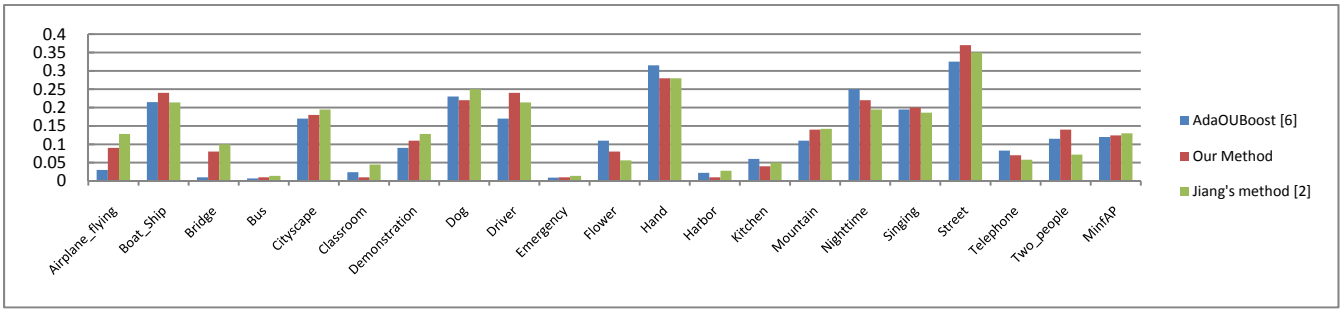

Figure 6. Concept Detection Performance onTRECVID 2008