Incremental Multi-Classifier Learning Algorithm on Grid'5000 for Large Scale Image Annotation

Yubing Tong
Laboratoire d'Informatique de Grenoble – UJF, BP 53 – 38041 Grenoble Cedex 9 – France
Yubing.Tong@imag.fr

Bahjat Safadi
Laboratoire d'Informatique de Grenoble – UJF, BP 53 – 38041 Grenoble Cedex 9 – France
Bahjat.Safadi@imag.fr

Georges Quénot
Laboratoire d'Informatique de Grenoble – CNRS, BP 53 – 38041 Grenoble Cedex 9 – France
Georges.Quénot@imag.fr

ABSTRACT

With our previous research, active learning with multi-classifier showed considering performance in large scale data but much calculation was involved. In this paper, we proposed an incremental multi-classifier (SVM classifiers were used) learning algorithm for large scale imbalanced image annotation. For further accelerating the training and predicting process, Grid'5000, French National Grid, was adopted. The result show that the best performance was reached with only 15-30% of the corpus annotated and our new method could achieve almost the same precision while save nearly 50-60% or even more than 94% of the calculation time when parallel multi-threads were used. Our proposed method will be much potential on very large scale data for less processing time.

Categories and Subject Descriptors

H.2.0: Image

General Terms

Algorithm

Keywords

Incremental learning, multi-classifier, image annotation

1. INTRODUCTION

Active learning is a traditional method for labeling unknown samples [1]. A user subjectively labels the images as Positive or Negative, and these labeled images are used to train a classifier that performs a bi-class classification on the image database.

Images with the higher scores or probability values are retrieved as the most informative samples to be labeled by the user [2, 3, and 4]. Now For image annotation system, in practice, low predicting accuracy of informative class and huge calculation derived from large scale imbalanced data make high requirement on machine learning adopted in image annotation. In the TRECVID evaluation campaigns, the minor class is less than 1% [5]. This imbalance could make the predicting accuracy of informative class (normally minor class) very low. Some methods tried to generate the balanced data set by down sampling major class or over sampling minor class in the original training data set [6, 7, and 8]. Down sampling might lose information due to the fact that it ignored a lot of information from the non chosen negative samples and over sampling involved much artificial information. Ref [9] has shown the effectiveness of using the multi-learners approach to handle the problem of the imbalances between the major and minor classes. Our previous research also showed that combining of active learning with multi learners significantly increases the effectiveness of the active learning [2, 3]. But it still needs to train many classifiers at iterations and much slower comparing to a single learner approach. This makes a very big challenge in the task of automatically large scale image annotation. A good idea is to re-use the previous useful information and learn the incremental information derived from only a part of the new samples. Some incremental learning method mainly focused on the new added samples [10]. In this paper, we overcome the problem of time processing for the active learning with multiple-learners by proposing a robust incremental multi-SVM algorithm. The information from previous iteration could be reused to update part of the predicting results in the current iteration of active learning and only a small number of new classifiers need to be trained. Therefore much calculation on training could be saved.

Besides the above incremental learning algorithm, parallelism was another way to accelerate the annotating process for very large scale by dispatching the large amount experimental tasks onto distributed processors. Grid'5000, French National Grid, was adopted as a high performance and parallel computing platform in this paper. An effective method for multi-thread arrangement and resource reserve on Grid5000 was also proposed in this paper, by which only 40% resource was reserved compared with the common method. Experiment results from TRECVID 2007 and 2008 show that almost 50-60% calculation could be saved with the proposed method and furthermore almost 94% calculation could be saved with parallel mechanism on the proposed method.

2. PARALLEL INCREMENTAL MULTI-CLASSIFIER LEARNING ALGORITHM

2.1 Incremental Multi-classifier Learning Algorithm

For solving the large scale imbalanced data, active learning with multi-classifier was adopted in this paper. Active learning includes much iteration and new samples will be annotated and added to the training data set at each iteration. Classifiers could be trained on the training samples to get the new classifier model which was used for predicting testing samples. The outputs from prediction with the higher scores were retrieved as the most informative samples to be labeled by the users. In this paper, multiple SVMs were adopted for training sample in iterations. Classifier was learned from the training data set $\boldsymbol{S_T}$ by minimizing the following

$$\min \quad \Phi(w,\xi,b) = \frac{1}{2}(w^T w) + C\sum_{i\in S_T}\xi_i \tag{1}$$

subject to the constraint:

$$y_i(w\cdot x_i + b) \ge 1-\xi_i, \xi_i \ge 0$$

, where, ξ_i is slack variable and C is a regularization parameter.

If the global training data set $\boldsymbol{S_T}$ is balanced and $S_T = \bigcup_{j=1\ldots,nl(k)} S_j$, $\bigcap_{j=1,\ldots,nl(k)} S_j = 0$ where, *nl(k)* is the number of subsets, the minimum in equation (1) can be rewritten as follows,

$$\min \quad \Phi(w,\xi,b) = \frac{1}{2}(w^T w) + C\sum_{i\in(\bigcup_{j=1\ldots,nl(k)} S_j)}\xi_i \tag{2}$$

Then machine learning with multi-learners could be proposed on all the subsets $S_0,\ldots,S_{i,\ldots},S_{nl(k)}$ with the following format,

$$\begin{aligned}
&\min \quad \Phi_1(w,\xi,b) = \frac{1}{2}(w^T, w) + C'\sum_{j\in S_1}\xi_j \\
&\qquad subject \quad to \quad y_j(w^T x_j + b) \le 1-\xi_j; \xi_j \ge 0;\ j\in S_1 \\
&\ldots \\
&\min \quad \Phi_k(w,\xi,b) = \frac{1}{2}(w^T, w) + C'\sum_{j\in S_k}\xi_i \\
&\qquad subject \quad to \quad y_j(w^T x_j + b) \le 1-\xi_j; \xi_j \ge 0;\ j\in S_k
\end{aligned} \tag{3}$$

For the imbalanced problem, we proposed an active learning with multi-classifiers. First, we randomly down sample the negative class samples (here negative class was assumed to be the major class) to set up balanced subsets. Since the data was extremely imbalanced, all the positive samples were used in the current iteration. This was done as a way to compensate for the natural bias against that class embodied by the individual classifier trained on the class-imbalanced domain. The scale of the subsets was adjusted by a global parameter, *fpos*, the ratio of negative and positive samples. Then based on the modified balanced training subsets, multi-classifiers were adopted on multiple subsets. One learner is corresponding to one training subset; *nl(k)* in equation (2) could be calculated as follows,

$$nl(k) = \frac{nn(k)}{fpos * np(k)} \tag{4}$$

, where, at iteration *k*, *nn(k)* is the number of negative samples and *np(k)* is the number of positive samples. Then multiple learners can be trained according to equation (3) in which multiple learners are trained independently at iterations. But some information from previous training can still be useful for the current iteration. Then previous training can be relative to the current training. Some strategy can be designed to select suitable new classifiers for training and predicting, at the same time, the information from previous iteration could be reused to update part of the predicting results in the current iteration. Finally only some new classifiers need to be trained in the current iteration and therefore much calculation can be saved. Then equation (3) can be modified by introducing previous training subset with different weights such as C''_{k-1} in equation (5),

$$\begin{aligned}
&\min \quad \Phi_1(w,\xi,b) = \frac{1}{2}(w^T, w) + C'\sum_{j\in S_1}\xi_j \\
&\qquad subject \quad to \\
&\qquad\quad y_j(w^T x_j + b) \le 1-\xi_j; \xi_j \ge 0;\ j\in S_1 \\
&\ldots \\
&\min \quad \Phi_k(w,\xi,b) = \frac{1}{2}(w^T, w) + C'\sum_{i\in S_k}\xi_i + C''_{k-1}\sum_{j\in S'_{k-1}\subseteq S_{k-1}}\xi_j \\
&\qquad subject \quad to \\
&\qquad\quad y_{i\_or\_j}(w^T x_{i\_or\_j} + b) \le 1-\xi_{i\_or\_j}; \xi_{i\_or\_j} \ge 0; i\in S_k, j\in S'_{k-1}
\end{aligned} \tag{5}$$

That leads to the following incremental learning based on multiple classifiers as shown in equation (6),

$$\begin{aligned}
&\min \quad \Phi_1(w,\xi) = \frac{1}{2}(w^T, w) + C'\sum_{i\in S_1}\xi_i \\
&\qquad subject \quad to \quad y_i(w^T x_i + b) \le 1-\xi_i; \xi_i \ge 0; i\in S_1 \\
&\ldots \\
&\min \quad \Phi'_k(w,\xi) = \left\{\frac{1}{2}(w^T, w) + C'\sum_{i\in S_k}\xi_i\right\} + \left\{\frac{1}{2}(w^T, w) + C''_{k-1}\sum_{j\in S'_{k-1}}\xi_j\right\} \\
&\qquad\qquad => new\_classifier_k + pre\_classifier_{k-1} \\
&\qquad subject \quad to \\
&\qquad\quad y_{i\_or\_j}(w^T x_{i\_or\_j} + b) \le 1-\xi_{i\_or\_j}; \xi_{i\_or\_j} \ge 0; i\in S_k, j\in S'_{k-1}
\end{aligned} \tag{6}$$

The classifiers following the first one can be viewed as two parts: new classifiers and pre classifiers. In practice, equation (6) could be described in figure 1, where we use previous information to update the predicting result in the current iteration.

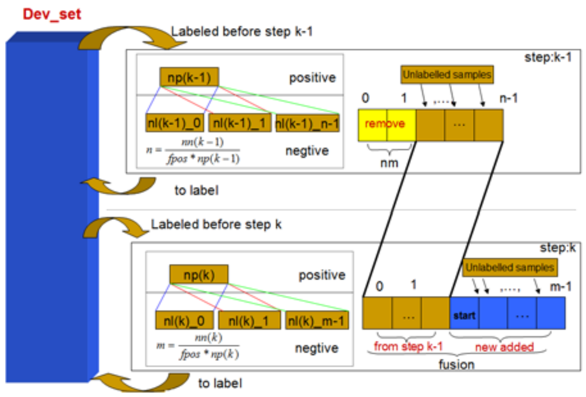


Figure 1. Incremental learning based on Multi-classifiers.

At each iteration the incremental algorithm would remove the learners with the minimum number of positive samples (normally the learners taken from the oldest iterations), and each new learner that should be added, consists of all the positive samples and of a subset randomly selected from the negative samples in the training set. Let *nl(k)* be the number of learners needed at *k-th* step and *nm[k]* be the minimum number of learners to be changed at *k-th* step, then the actual number of learners which should be trained at *k-th* step is equal to *nl(k)-(nl(k-1)-mn[k])* and the ‘start’ position shown in figure 1 can be calculated as

$$start = nl(k) - (nl(k-1) - nm[k]) \tag{7}$$

At the end, we merge/fuse these new learners with *nl(k-1)-mn[k]* learners from the previous steps, so we have *nl(k)* learners but we only train part of them at each iteration *k*. The *nm[k]* parameter is calculated by the conditions described as follows.

The number of learners to be removed from the previous step is calculated by the following:

if ($nl(k) >= nl(k-1)$) : nm

if ($nl(k) < nl(k-1)$) : $nl(k-1) - nl(k) + nm$

the number of learners to be added by:

if ($nl(k) <= nl(k-1)$) : nm

if ($nl(k) > nl(k-1)$) : $nl(k) - nl(k-1) + nm$

With previous subset adjustments biased minor class, normally the learners in *k-th* step will generate more or at least the same number minor labeled samples as that in last step. The incremental learning algorithm can be finally described as follows,

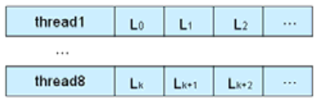
```
S: all data samples.
L_i;U_i: labeled and unlabeled subsets of S.
A=(Train, Predict): the elementary learning algorithm.
Q: the selection (or querying) function.
Initialize L_i (e.g. 10 positives & 20 negatives).
while $S = S \setminus L_i \neq \phi$ ; do
    Calculate $nl(k)$, nm[k] and start position
    for all $j \in [start, ..., nl(k)]$ do
        Select subset $T_j$ from $L_i$ for training
        $C_j \longleftarrow Train(T_j)$
        $P_{un}^j \longleftarrow Predict(U_i, C_j)$
    end for
    $P_{un} \longleftarrow Fuse(P_{un}^j)$
    Apply $Q$ on $P_{un}$.
    Select $\tilde{x} \in U_i$ samples
    $\tilde{y}$ = Label $\tilde{x}$
    $L_{i+1} \longleftarrow L_i \bigcup (\tilde{x}, \tilde{y})$
    $U_{i+1} \longleftarrow U_i \setminus \tilde{x}$
end while
```

In our experiments, we fixed the minimum and maximum values of *nm* as {1, 10} and initial nm[k] =20% *nl*(*k*).

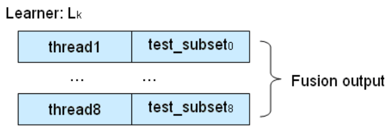
2.2 Parallel Incremental Multi-classifier Learning Algorithm on Grid'5000

Grid'5000 initial goal was to provide 5000 CPU cores distributed over 9 sites in France and now it is leaning towards an international computer science grid with more and more sites involved. Users can submit resource reservations and experiment jobs using the OAR resource and job management system [11]. With middle ware, Taktuk, tasks can be effectively dispatched to more computing nodes with flat chain structure or complicate cascade structure [12]. In practice, no only SVM train but also SVM predict cost much calculation in image annotation. Both the training and predicting of a single SVM were parallelized with Pthreads in this paper.

As described in the above incremental active learning algorithm based on multi-SVMs, only new added classifiers need to be trained at each iteration. For some concepts used for image annotation, such as "Two_People" in TRECVID 2007, only 1 or 2 classifiers are trained at each iteration. But SVM prediction will always be done on all the testing data. Then parallelization of SVM training will be not high. While SVM predict can be arranged with high parallelization by dividing the testing sets. So the different multi-thread strategies for SVM training and predicting are proposed in this paper. For SVM training, multi-threads are created and each with its own task sequences as shown in figure 2 (a),



(a) Threads arrangement for SVM training



(b) Threads arrangement for SVM predicting

Figure 2. multi-threads arrangement for SVM training and predicting.

Here 8 threads were used as an example. The threads arrangement for SVM prediction was different from SVM training. For every learner, testing set is divided into 8 subsets and each is filled into the task sequence of the corresponding thread. So every thread will always be full filled.

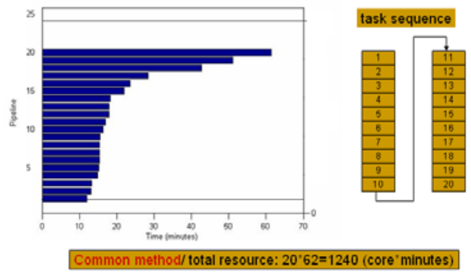
Before multi-threads run on Grid'5000, resource should be reserved in advance. There were several ways to apply for resource on Grid'5000. Grid'5000 arrange multi-thread task on pipelines with work stealing methods that means if there is no task for the current thread then it became a 'thief' to steal a task which might be firstly arranged for another thread [13]. Work stealing strategy made the pipeline arrangement on Grid5000 much different from that for common DSP for which the pipeline will run independently once the tasks are arranged on pipeline and it will never happen for the current pipeline schedule to 'steal' the task prepared for another pipeline. But this may happen on Grid5000 because of the work stealing strategy described in the above.

An effective method for multi-thread arrangement and resource reserve on Grid'5000 is proposed here. Resource on Grid'5000 is measured by the number of cpu-cores and the keeping time, so core*minutes can be used as the measurement unit of resource. The time of 20 runs has been sorted as shown in table 1 which is used as an example to illuminate our proposed resource reserving method.

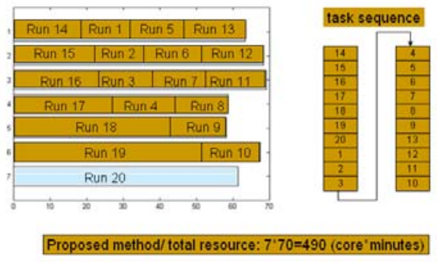
Table 1. Running time for 20 experiments.

run	1	2	3	…	18	19	20
time(minutes)	12.03	13.19	13.23	…	42.69	51.16	61.46

One common method to reserve resource is shown in figure 3 (a), task sequence is arranged from 1 to 20. 20 pipelines are used and run at the same time. Since the maximum value for the pipeline is the value used by the 20th pipeline, then totally we need 20 cores*62 minutes = 1240 core*minutes. In figure (b), task sequence is arranged from small to large according to the costing time. Normally we can run a part of the final experiment to estimate their costing time and sort them. The largest 7 tasks (task 20,…,14) were arranged first, then the smallest ones are used to fill their following position in each pipeline. After that small adjustment was used to make total time on each pipeline almost same then each pipeline is balanced. Finally, only 7 pipelines are generated and totally we need 7 cores* maximum ($time_{16}$+ $time_3$+ $time_7$+ $time_{11}$) = 7 *70 =490 core*minutes. Compared with the common reserving method, our proposed reserving resource method almost save (1240-490)/1240=60.5% resource. Working schedule in this case is a NP problem essentially. Our proposed method is effective in practice and has shown much better than common method although it might not be the optimal solution.



(a) Common resource reserving method



(b) Proposed resource reserving method

Figure 3. Resource reserve methods.

3. EXPERIMENTS

Four types of image descriptors using the SVM with RBF kernel as classifier and the relevance sampling strategy for active learning have been used to evaluate our proposed algorithm. The cold start problem was not really explored; a random set of 10 positive and 20 negative samples was used. *fpos* was taken from our previous work [2]. For the number of samples to be added at each iteration, a variable step size was used since we observed in previous experiments that having small steps in the beginning of the active learning process is better for the speed of performance improvement. In practice, we used a logarithmic scale with 40 steps. The evaluations were conducted using the TRECVID 2007 and TRECVID 2008 test collection and protocol [3].

3.1 TRECVID 2007 and 2008 collections

20 concepts in TRECVID 2007 and 2008 collections were used to evaluate our algorithm. TRECVID 2007 collection contains 21532 video shots as a training set and 22084 shots as test set, while TREC2008 contains 43616 video shots as a training set and 35766 shots as a test set [3]. In our experiments active learning methods are executed as if very few annotations are available in the training set. Each time a human annotation is needed, the corresponding subset of the full annotation is made available to the active learner.

3.2 Image representation

Concepts and Images can be represented by their vector descriptors or features. We evaluated the different methods with descriptors of different types and sizes. These descriptors have been produced by various partners of the IRIM project of the GDR ISIS [14].

LIG_hg104: early fusion with normalization of an RGB histogram 4×4×4 and a Gabor transformation (8 orientations and 5 scales), 64+40 = 104 dimensions.

CEALIST_global_tlep: early fusion of local descriptors of texture and of an RGB color histogram, 512+64 = 576 dimensions.

ETIS_ global_qwm1x3x256: 3 histograms of 3 vertical bands of visual descriptors, standard Quaternion wavelet coefficients at three scales, 3×256 = 768 dimensions.

LEAR_bow_sift_1000: histogram of local visual descriptors, "classical" SIFT [15], 1000 dimensions.

3.3 Active learning steps

In our evaluation, we used a total of 40 steps for the active learning algorithm, considering the geometric scale function with the following formula:

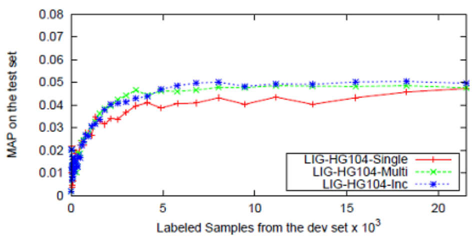
$$S_k = S_0 \times (\frac{N}{S_0})^{k/K} \qquad (8)$$

,where, N is the total size of the development set, S_0 is the size of the training set at the cold-start(30 samples), K is the total number of steps and k is the current step. At each step (or iteration) the algorithm calculates the S_k to be the size of the new training set and it chooses new samples to be labeled with size equal to S_k $-S_{k-1}$.

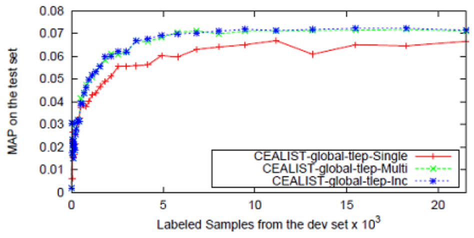
3.4 Performance of the incremental active learning based on Multi-SVMs

With figure 4, the effectiveness of the three methods (the single and Multi-learner and the Incremental) can be compared by using the four descriptors and the relevance sampling strategy. The performance of the Single-learner is also shown as baseline. For the multiple-learner and the incremental experiments, the fusion by harmonic mean has been used. These plots show the evaluation of the indexing performance of the test sets measured by the Mean Average Precision (MAP). For the plots, we consider that the faster it grows and the higher performance it achieves, especially in the beginning, the better. At the same time, the larger the area under the curve is, the better the performance will be.

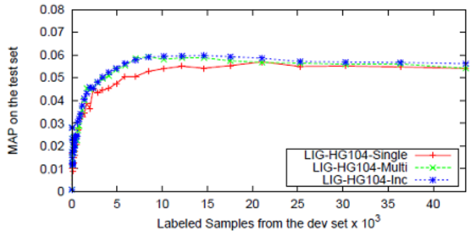
The proposed incremental algorithm has almost achieved the same performance as that of multi-learner. Both of them perform significantly better and faster to reach the highest value than single learner. With our incremental learning method, the highest performance can be reached with training by only 15-30 % samples that also means labeling 15-30 % samples instead of all the samples can still get the same result (in MAP).



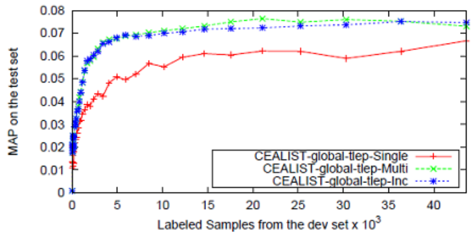
(a) HG104 descriptor on TRECVID 2007



(b) CEALIST descriptor on TRECVID 2007



(c)HG104 descriptor on TRECVID 2008



(d) CEALIST descriptor on TRECVID 2008

Figure4. The Map results on the TRECVID 2007 and 2008.

The performance of incremental active learning algorithm and multi-learners are almost the same and the corresponding MAP value is not high (about 0.075 for the best descriptor in figure (d)) as shown in figure 4. But it is quite good for individual descriptors considering that a classifier with a significantly higher performance can be built by fusing the outputs of several such classifiers. And we can also fuse the outputs from different descriptors to improve the performance. Although fusion of different descriptors is not the focus for this paper, we still show some results with fusion on different descriptors on TRECVID 2007. Figure 5 shows the MAP value by fusing the outputs of four descriptors which are much higher than any one of the four descriptors.

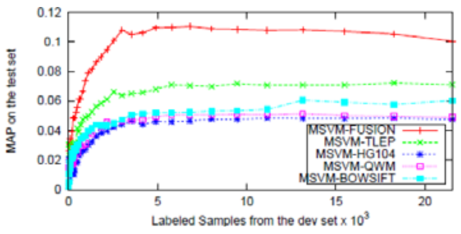


Figure 5. Fusion on descriptors.

We also consider the index of $G_{a\text{-}b}$ to measure the performance difference between two active learning curves as shown in table 2. $G_{a\text{-}b} = (A_a - A_b)/A_b$ was used to measure the performance difference. A_i, is the area under the curve i and $a, b \in \{S, M, I\}$,where S, M and I means single learner, multi-learner and incremental multi-learners.

Table 2. The gain of the system performance on TRECVID 2007 and 2008.

Descriptor	TRECVID 2007		TRECVID 2008	
	$G_{I\text{-}S}$ (%)	*$G_{I\text{-}M}$ (%)*	*$G_{I\text{-}S}$ (%)*	*$G_{I\text{-}M}$ (%)*
LIG_hg104	14.77	2.34	6.50	1.83
CEALIST_global tlep	12.84	0.62	22.42	-1.70
ETIS_global_qwm	4.76	0.73	1.20	0.02
LEAR_bow_sift	8.04	-3.16	5.22	0.75
Average value	10.1	0.13	8.83	0.23

3.5 Execution time

Table 3 gives the total execution times for the whole active learning process (40 iterations) on all 20 concepts on each experiment collection, per method and per descriptor, using the relevance strategy. *G* indicates the gain of time using our incremental method compared to the multi-learners. G (%) = ($time_{Multi}$-$time_{Inc}$) / $time_{Multi}$..

Table 3. Processing time on TRECVID 2007 and 2008

Descriptor	TRECVID 2007				TRECVID 2008			
	Single	Multi	Inc	*G (%)*	Single	Multi	Inc	*G(%)*
LIG_hg104	1.40	20.63	6.8	67	4.80	59.54	23.23	60
CEALIST_global_tlep	23.90	115.02	56	51	96.56	395.45	204.9	48
ETIS_global_qwm	13.40	142.97	64.10	55	45.67	460.60	212.3	54
LEAR_bow_sift	43.42	162.18	79.16	52	181.00	592.10	300.6	49
Average value	20.53	110.2	51.5	56.2	82	376.9	185.3	52.75

We can see that Single learner is faster than multiple learner and incremental methods. But considering the performance of single learner described in the above section, the performance of single learner is much lower than that of multi-learner. Compared with multiple learners, the new proposed incremental method has saved nearly 50-60% time without losing any performance.

Table 4 shows the costing time of 20 concepts with LIG_hg104 descriptor on TRECVID 2007. The costing time of the program with 8 threads can reduce to nearly 1/7th of that of the program without multithreads. With multi-threads on incremental multi-SVMs algorithm, (20.63-1.15)/20.63=94.43% calculation time was saved.

Table 4. The costing time on 20 concepts.

Concepts	MultiSVMs no_threads	8 threads (T&P)	Inc_ MultiSVMs	Inc_MultiSVMs _multithread
Airplane_flying	75.03	10.89	15.22	2.67
…	…	…	…	…
Two_people	63.51	18.50	32.31	9.68
Total Time in Minutes	1237.87	185.90	15.22	68.85
Total Time in Hours	20.63	3.10	6.8	1.15
Time saved (%)		20.63/3.1=6.65	20.63/6.8=3.03	20.63/1.15=17.94

4. CONCLUSION

In the paper, we proposed a new incremental active learning algorithm based on multiple classifiers for large scale image annotation. Experimental results show that the best performance is reached when 15-30% of the corpus is annotated. And our new method can achieve almost the same precision while nearly saved 50-60% or 94% with multi-threads of the calculation time compared with the multi-classifiers. SVM are adopted in this paper, but our method is not specific to SVM and other machine learning techniques can also adopt it. With the above considering performance on TRECVID2007 and 2008, our method will be much potential on very large scale data for less processing time.

5. ACKNOWLEDGMENTS

This work was partly realized as part of the Quaero Program funded by OSEO, French State agency for innovation. It also got the support from Pole MSTIC APIMS project funded by University Joseph Fourier, Grenoble. Experiments presented in this paper were carried out using the Grid'5000 experimental testbed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr).

6. REFERENCES:

[1] Haibo He. Learning from Imbalanced Data. IEEE T. on Knowledge and Data Engineering, 21(9), 1263-1284. 2009.

[2] S. Ayache and G. Quénot. Evaluation of active learning strategies for video indexing. Image Commun., 22(7-8):692–704, 2007.

[3] S. Ayache, G. Quénot, and J. Gensel. Image and video indexing using networks of operators. EURASIP Journal on Image and Video Processing, 2007(4):1–13, 2007.

[4] B. Safadi and G. Quénot. Active learning with multiple classifiers for multimedia indexing. In CBMI, Grenoble, France, June 2010.

[5] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and TRECVID. In MIR'06: Proceedings of the 8th ACM International Worksh op on Multimedia Information Retrieval, 321–330, USA, 2006.

[6] Andrew Estabrooks, Nathalie Japkowicz. A multiple resampling method for learning from imbalanced data sets. Computing intelligence, 20(1), 2004.

[7] H. He, Y. Bai, E. A. Garcia, and S. Li, ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning, in Proc. Int. Joint Conf. Neural Networks (IJCNN'08), 1323-1329, 2008.

[8] Suzan K¨oknar-Tezel. Improving SVM Classification on Imbalanced Data Sets in Distance Spaces. ICDM, 259-267, 2009.

[9] Steven C.H. Hoi, Rong Jin, Jianke Zhu, and M.R. Lyu, Semi-supervised SVM batch mode active learning with application to image retrieval. ACM Transactions on Information Systems (TOIS), 27(3): 1-29, 2006.

[10] C. Wu, X. Wang, D. Bai, and H. Zhang. Fast incremental learning algorithm of svm on kkt conditions. In FSKD '09: Proceedings of the 2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery, 551–554, USA, 2009.

[11] Cappello, F. Caron, E. Dayde, M. Desprez, et al. Grid'5000: a large scale and highly reconfigurable experimental Grid testbed. International Journal of High Performance Computing Applications, 20(4): 481-494, 2006.

[12] Benoit Claudel, Guillaume Huard. Olivier Richard. Taktuk, adaptive deployment of remote executions, In Proceedings of the International Symposium on High Performance Distributed Computing (HPDC), 2009.

http://www.lrz.de/hpdc2009/PDF/huard.pdf.

[13] Stephane Letz and Yann Orlarey and Dominique Fober. Work stealing scheduler for Automatic Parallelization in Faust. LAC2010 Program: The Linux Audio Conference, 2010. http://lac.linuxaudio.org/2010/download/faust-LAC-2010.pdf.

[14] G. Quénot, B. Delezoide, H. le Borgne, P.-A. Moellic, D. Gorisse, F. Precioso, F. Wang, B. Merialdo, P. Gosselin, L. Granjon, D. Pellerin, M. Rombaut, H. Bredin, L. Koenig, H. Lachambre, E. E. Khoury, B. Mansencal, J. Benois- Pineau, H. Jégou, S. Ayache, B. Safadi, J. Fabrizio, M. Cord, H. Glotin, Z. Zhao, E. Dumont, and B. Augereau. Irim at TRECVID 2009: High level feature extraction. In TRECVID 2009 notebook, 16-17 Nov 2009.

[15] D. Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2):91–110, 2004.