2012 IEEE 4th International Conference on Cloud Computing Technology and Science

Large-Scale Multimedia Data Mining Using MapReduce Framework

Hanli Wang*†, Yun Shen*†, Lei Wang*†, Kuangtian Zhufeng*†, Wei Wang*† and Cheng Cheng*†
*Key Laboratory of Embedded System and Service Computing, Ministry of Education
Tongji University, Shanghai 200092, China
†Department of Computer Science and Technology, Tongji University, Shanghai 201804, China
Email: hanliwang@tongji.edu.cn

***Abstract*—In this paper, the framework of MapReduce is explored for large-scale multimedia data mining. Firstly, a brief overview of MapReduce and Hadoop is presented to speed up large-scale multimedia data mining. Then, the high-level theory and low-level implementation for several key computer vision technologies involved in this work are introduced, such as 2D/3D interest point detection, clustering, bag of features, and so on. Experimental results on image classification, video event detection and near-duplicate video retrieval are carried out on a five-node Hadoop cluster to demonstrate the efficiency of the proposed MapReduce framework for large-scale multimedia data mining applications.**

***Keywords*-MapReduce; Hadoop; Image classification; Video event detection; Near-duplicate video retrieval.**

I. Introduction

The past few years have witnessed a great success of social networks and mobile multimedia applications, leading to the generation of dramatically huge amount of Internet images/videos. These vast amounts of images and videos are shared and transmitted in Internet and it is becoming one of most major ways for humans to gather information by exploring these multimedia resources. It is desired to efficiently store, query, analyze, understand and utilize these immense multimedia data, which is quite a big challenge to the computing industry and research communities.

In order to process these large-scale multimedia data, not only advanced algorithms (such as that of computer vision and machine learning) are necessary but also powerful computing technologies or platforms are required. A number of recent studies and industry practices, such as [1], develop data-center scale computer systems (or cloud computing) to meet the high storage and processing demands for large-scale data applications. Such systems are usually composed of hundreds, thousands, or even millions of commodity computers connected through local area networks.

One of the most popular programming paradigms on data-center scale computer systems is the MapReduce programming model [1]. With this model, an application can be implemented as a series of MapReduce operations, each consisting of a Map phase and a Reduce phase to process a large number of independent data items. MapReduce supports automatic parallelization, distribution of computations, task management and fault tolerance. The MapReduce framework has been implemented and optimized in a number of researches. In [2], the MapReduce framework is further extended by introducing a Merge phase to form a Map-Reduce-Merge framework for performance improvement. He *et. al.* develop the computing system of Mars by implementing MapReduce on GPUs [3]. Later, MapReduce has been realized on FPGAs to form the framework of FPMR [4]. A MapReduce-based co-clustering approach is proposed in [5] to preprocess and co-cluster distributed data.

There is also a number of researches in the computer vision community employing MapReduce for efficient computing. Li *et. al.* in [6] apply MapReduce to a landmark classification algorithm to reduce the processing computations. In [7], the MapReduce framework is utilized to search nearest images to generate image tags. Yan *et. al.* in [8] develop a MapReduce algorithm to train a scalable concept detection model. In [9], the MapReduce framework is applied to multimedia data mining and the results of K-means and background substraction have been presented.

In this work, we further explore the MapReduce framework on a number of large-scale multimedia data mining applications, including image classification, video event detection and near-duplicate video retrieval. Extensive experiments are carried out to demonstrate the performance of the proposed MapReduce framework on the aforementioned three applications, which are the main contributions of this work. The rest of this paper is organized as follows. The framework of MapReduce and Hadoop is briefly introduced in Section II. Section III describes the proposed MapReduce-based implementation on the three multimedia data mining applications. Experimental results are presented in Section IV. Finally, Section V concludes this paper.

II. Background

A. MapReduce

MapReduce [1] is a parallel programming model for various data intensive applications by hiding the inter-machine communication, fault tolerance and load balancing, etc, and thus suitable for processing large-scale datasets. A standard MapReduce program can be performed mainly in two phases, including the Map phase and the Reduce phase. Each phase has (key,value) pairs as input and output. In the Map phase, each input in terms of the (key,value) pair is processed

978-1-4673-4510-1/12/$31.00 ©2012 IEEE

independently and a list of (key,value) pairs are produced. In other words, for each input it may emit a list containing zero, one or more (key,value) pairs. The logical view is $map(key1, value1) \rightarrow list < key2, value2 >$. Then, all the intermediate values associated with the same intermediate key are grouped and passed to the Reduce phase. In the Reduce phase, the (key,value) pairs with the same key are processed to generate a final output. The logical view is $reduce(key2, list < value2 >) \rightarrow list < value3 >$. The MapReduce structure is shown in Fig. 1.

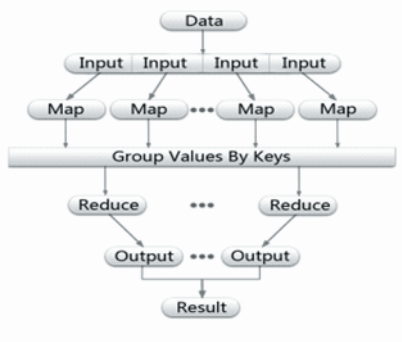

Figure 1. Illustration of MapReduce structure.

For optimization, the Combine phase [9] can be implemented between the Map and Reduce phases, which is similar to the Reduce phase except that it operates directly on the local output of mappers. The design of the Combine phase would easily cut down the amount of data sent to the reducer and help decrease the cost of network communications.

B. Hadoop

Hadoop [10], [11] is an open source software library which provides the Hadoop Distributed File System (HDFS) and MapReduce architecture. HDFS is designed to store very large data reliably, and to process high-speed data streams for user applications, which is the primary storage system designed to work with MapReduce. In Hadoop, a large file is generally divided into blocks that are replicated to multiple nodes for fault tolerance, in addition to support real time access of data to prevent network bandwidth bottlenecks. In this way, the Map and Reduce functions can be executed on smaller sub-datasets and thus accelerate big data processing.

The master-slave architecture is employed by Hadoop. The master serves in two key functional roles: one is to store data in terms of HDFS managed by NameNode, and the other is to run parallel programs arranged by JobTracker which implements MapReduce. The slaves consist of a number of computing nodes and each slave performs as both DataNode and TaskTracker. Specific efforts such as performing block creation, deletion and replication upon instruction from the NameNode are realized by DataNode. The TaskTracker is employed for running programs to achieve the task. The Hadoop flowchart is illustrated in Fig. 2.

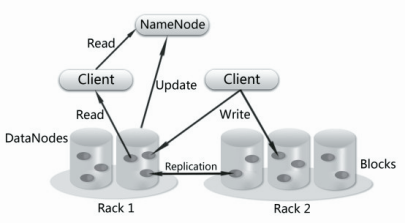

Figure 2. Illustration of Hadoop flowchart.

III. Multimedia Data Mining with MapReduce

In general, the algorithms of computer vision and machine learning are usually applied for multimedia data mining. It is not trivial to implement these algorithms from the viewpoint of computations, not even to mention the exploration on large-scale datasets where huge amounts of computations are generally required. Therefore, the MapReduce framework is very useful to be applied for speeding up such multimedia data mining applications. In this work, we focus our discussions on three hot multimedia data mining topics, including image classification, video event detection and near-duplicate video retrieval.

A. Data Representation

Hadoop provides support for reading input data in several different formats. In each type of implementations, the data can be meaningfully split and then passed in parallel to different mappers [1]. Traditional Hadoop implementations are good at processing text data and may face difficulties in representing image and video data in a reasonable format. In order to achieve data representation for image and video, we propose to allow one mapper to process an entire input file. To this aim, we customize the InputFormat and RecordReader, which are two classes in Hadoop implementation. After customization, the video and image files can be presented as original stream, and hence make the subsequent processing more effective.

B. Feature Extraction

Feature extraction is one of the most important steps for most multimedia data mining tasks. In this work, we focus on image classification, video event detection and near-duplicate video retrieval. Therefore, an efficient MapReduce-based implementation is necessary to detect interest points for both images such as the Harris-Laplace detector [12] and the dense sampling detector [13] and for videos such as the detector of Space-Time Interest Points (STIP) [14]. After 2D/3D interest points are detected, feature description is carried out to describe the local characteristics of these keypoints for images such as the Scale Invariant Feature Transform (SIFT) descriptor [15] and the Histograms of oriented Gradient (HoG) for videos [16].

In this work, we utilize the toolkits of LIP-VIREO [17] and STIP [18] to achieve interest points detection and local

978-1-4673-4510-1/12/$31.00 ©2012 IEEE

feature description. During the MapReduce-based implementation, each image/video file is passed to a corresponding computer node in the form of byte stream as an entire load, each mapper receives the data and generates a new copy of the file in the local disk. Then, the feature extraction proceeds in parallel without interfering with other nodes and mappers. The executable programs are invoked directly and natively to process the local temporary file and produce the desirable features. The name of the image/video is used as the *key*, and the information of the features is designed to be the *value*. Once the Map phase is finished, the temporary files mentioned-above will be cleaned up. In the Reduce phase, all the (key,value) pairs are aggregated together to output a final image/video feature dataset.

C. Clustering

In this work, K-means clustering is utilized for bag of features generation. In the proposed K-means implementation with MapReduce, each iteration takes 3 stages, including Map, Combine and Reduce. The Map function is designed to assign each point to the closest center while the Reduce function is figured out to update the new centers. For optimization, a Combine function is developed to aggregate the partial intermediate values with the same key within the same Map task. The detailed implementation is described as follows. The input data for clustering is stored on HDFS as a text file with each line representing a point. The dataset is split to a number of sub-datasets and passed to mappers. For each Map task, the K-means algorithm constructs an array containing centers of the clusters. Then, a mapper can compute the closest center for each point. The intermediate output of mappers are (key,value) pairs where *key* indicates the index of the closest center and *value* consists of two components, including the location of the point and its count which is equal to 1 because only one point exits for each (key,value) pair. After the Map phase, the outputs are stored locally in the host computer and then processed in the Combine phase. The combiner groups the (key, <value1,1>) pairs with the same key and then generates a new (key,<value, k>) pair where it is assumed that the amount of the corresponding points is k. Finally in the Reduce phase, the total number of points and their locations are used to produce the mean location of the points which is taken as a new center.

D. Bag of Features

The Bag of Features (BoF) technique, which can represent an image or a video as an orderless collection of keypoint features, has been shown to produce the state-of-the-art performance in a number of multimedia data mining/computer vision researches. To generate BoF, a visual dictionary is usually created by clustering the detected keypoints in their feature space and treating each cluster as a visual word of the dictionary. Then, the BoF vector is generated by assigning each keypoint in an image or a video to visual words. Moreover, the generation of BoF is further refined by integrating the spatial pyramid framework [19] and soft-weighting scheme [20], which have achieved significantly improved performance and robustness.

The proposed MapReduce-based BoF algorithm with soft-weighting scheme is composed of two main stages: Map and Reduce. In the Map stage, the nearest neighbor search is performed by calculating similarities between one keypoint and all the visual words. The keypoint is mapped to the top-k nearest visual words instead of only a single one. Therefore, the number of output pairs from each mapper is k. Regarding the output of (key,value) pair, *key* is the index of i-th ranking visual word in the top-k results, and *value* is the similarity score with partial weight according to the rule that the closer to a visual word, the higher the corresponding similarity score. In the Reduce stage, the (key,value) pairs are arranged to compute histograms to form a new image/video representation. In particular, the *value* (i.e., partial weight) of each (key,value) pair is accumulated together according to its *key* index. If the spatial pyramid framework [19] is applied to improve BOF representation, the dimension of a final BoF vector is multiple times larger than the original size of dictionary. The MapReduce-based BoF algorithm with spatial pyramid is a little different from the former approach. The difference is mainly in the Map stage: when the nearest neighbor search is performed, the location of the keypoint should be considered to determine in which bin the keypoint situates. At the end of the Map task, two indexes should be outputted, one for the location of image/video and the other for bin location.

E. Image Classification and Video Event Detection

Identifying concepts and/or events in images and videos is becoming more prominent due to the ever increasing amount of images and videos and the rising need to ease the understanding of multimedia contents. The implementation of image classification and video event detection is in general achieved by learning a prediction model with the following steps. Firstly, a set of training samples are defined for model learning. Secondly, the feature extraction procedure in Subsection III-B is carried out on the training images or videos. Thirdly, clustering as mentioned in Subsection III-C is performed on the resultant keypoints to create the visual dictionary, and then the new feature vectors for each image and video are generated with the BoF technique as described in Subsection III-D. Finally, the BoF-based feature vectors and labels of the training samples are input to classifiers such as the Support Vector Machine (SVM) to generate the prediction model for image classification and video event detection. When a new image or video with label unknown is to be classified or event-detected, it is firstly feature extracted and then the new feature vector is derived based on the available visual dictionary. Then, the resultant

978-1-4673-4510-1/12/$31.00 ©2012 IEEE

prediction model is applied to the input feature vector and in consequence the class/event label is predicted.

F. Near-Duplicate Video Retrieval

Near-Duplicate Videos (NDVs) can be considered as approximately identical videos that might differ in coding parameters, photometric variations (color, lighting changes), editing operations (captions, logo insertion), or audio overlays [21]. It is an emerging trend to retrieve NDVs for a number of applications such as copyright detection, video monitoring, Internet video ranking, video recommendation, etc. In general, global features such as color or ordinal signature or local features such BoF can be employed for NDV retrieval. In this work, we focus on the local feature method [22] as briefly described below due to its superior performance to detect NDVs. Firstly, a training video dataset and a reference video dataset are given so that the training dataset is employed to generate the visual dictionary and the reference video dataset is utilized as the pool in which the query video searches for its NDVs. In order to generate the visual dictionary, the training videos are preprocessed by shot boundary detection and key frame selection approaches. The feature extraction procedure in Subsection III-B is implemented on these selected key frames, and then clustering in Subsection III-C is applied to generate the visual dictionary. With the visual dictionary available, the BoF features as described in Subsection III-D are computed for the key frames of each reference video and the query video. The similarity between a reference video and the query video can be counted by aggregating the number of key frame matches, which is realized with 2D Hough histogram [23].

In the proposed MapReduce-based implementation, the datasets including the training and reference datasets are stored in HDFS and index files are created to record each video's location. For generating the final BoF features, all the index files are sent to mappers in parallel. Each mapper firstly loads the related videos from HDFS to local disk based on the corresponding index file. Then the processes of key frame selection, feature extraction and BoF generation as mentioned above are performed. Finally, several subdivisions of the entire reference model (i.e., the feature representation of reference videos) are generated and stored in HDFS. Simultaneously, the addresses of these sub-reference models are recorded as index files in HDFS for query on demand. During the query process (where a single query and multiple queries in parallel can be made[1]), the query videos are uploaded to HDFS and the corresponding feature representations are derived by functional mappers. For each of the sub-reference models, the similarities between the query videos and the reference videos associated with the sub-reference model are calculated in the Map phase. Then, the top-n (where n indicates that n NDVs are desirable) highest similarity scores with the corresponding reference video indexes for each of the query videos are emitted to the reducer. In the Reduce phase, the data for the same query video are grouped together, and then n NDVs are selected with the top-n highest similarity scores.

[1]For the convenience of following discussion, it is assumed to make multiple queries in parallel. And this can be easily extended to the scenario of a single query.

IV. Experimental Results

In order to evaluate the proposed MapReduce-based framework for multimedia data mining, the experimental results on three kinds of applications are shown in this work, including image classification, video event detection and near-duplicate video retrieval. The platform dedicated for the experiments is a five-node Hadoop cluster with five computers denoted as Node0, Node1, Node2, Node3 and Node4, respectively. Each physical node runs on the GNU/Linux Ubuntu 10.04 operating system. The detailed information about these five nodes is provided in Table I.

Table I
Description of five computer nodes.

	Node0	Node1	Node2	Node3	Node4
Processor	AMD Fx^{TM} -6100	AMD Fx^{TM} -6100	Intel Core^{TM} i5	Intel Core^{TM} i3	Intel Core^{TM} 2
CPU cores	6	6	4	4	2
CPU speed	3.9GHz	3.9GHz	3.7GHz	3.2GHz	3GHz
Memory	8G	8G	8G	2G	2G

Node0 acts as both the master node and slave node, and all the others are slave nodes. Based on the hardware configurations, it is designed that 9 map tasks are run on Node0 and Node1, 6 map tasks on Node2, and 4 map tasks on Node3 as well as Node4. The Java 1.7 version and Hadoop 1.0.1 version are used in the experiments. For comparative demonstrations, three test scenarios are used so that different combinations of computer nodes are employed to run the multimedia data mining applications, including (1) Scenario-0: only Node0 is used, (2) Scenario-1: Node0, Node2 and Node4 are used to form a three-node cluster, and (3) Scenario-2: all the five computers are used as a five-node cluster.

A. Image Classification

Due to the space limit, only the results on the fifteen scene categories [19] are shown in this work. Each of the fifteen categories has 200 to 400 images and the image resolution is 300×250 in pixels. In the proposed MapReduce-based implementation, 100 images per class are used for training and the rest for testing. The keypoint detector and descriptor are the dense sampling detector [13] and SIFT descriptor [15], respectively. The size of visual dictionary is 200 and two-level spatial pyramid are used for BoF generation. The LIBSVM tool [24] is employed as the learning classifier.

978-1-4673-4510-1/12/$31.00 ©2012 IEEE

The experimental results are given in Table II where the processing time (in unit of seconds) of the major procedures including feature extraction, clustering and BOF generation are listed for the three test scenarios. In addition, the classification accuracy results of the three test scenarios are given in Table II which are a little worse than the result of [19] (i.e., 79.0%). The difference of the accuracy results may reside in the fact that the detailed implementation of the proposed framework is different from that of [19], such as selection of training/testing set, parameters for keypoint detection, SVM implementation, etc. In addition, the comparison of the required computations of Scenario-1 and Scenario-2 against Scenario-0 are also listed behind the corresponding processing time in Table II in terms of $C_{\%} = \frac{C_i}{C_0} \times 100\%$, where C_i is the processing time for Scenario-i.

Table II
EXPERIMENTAL RESULTS OF IMAGE CLASSIFICATION.

	Scenario-0	Scenario-1	Scenario-2
Training			
Feature extraction (s)	2054	920 (45%)	620 (30%)
Clustering (s)	4426	3654 (83%)	2753 (62%)
BoF (s)	403	198 (49%)	161 (40%)
Testing			
Feature extraction (s)	4029	1794 (45%)	1199 (30%)
BoF (s)	718	362 (50%)	250 (35%)
Average accuracy (%)	74.4	75.0	74.0

B. Video Event Detection

Two benchmark video datasets, including KTH [25] and Hollywood2 Actions [26] are utilized to demonstrate the performance of the proposed MapRedce framework on video event detection. The KTH dataset contains 6 types of human action videos. There are 32 videos per class for training and 36 videos per class for testing. As far as the Hollywood2 Actions dataset is concerned, twelve frequent action classes are provided including 895 videos for training and 971 videos for testing. In the proposed MapReduce-based implementation, the keypoint detector and descriptor are the STIP detector [14] and HoG descriptor [16], respectively. The size of visual dictionary is set to 4000 and the LIBSVM tool [24] is employed as the learning classifier.

Firstly, the experimental results on the KTH video dataset are given in Table III where the processing time of the major procedures and average classification accuracy (which is consistent with the benchmark result of 91.8% in [16]) are listed. Moreover, the comparison of the required computations for the test three scenarios is also shown in Table III.

Then, the experimental results on the Hollywood2 Actions video dataset are given in Table IV where the processing time of the major procedures and the mean of average precision [27] (which is consistent with the benchmark result of 0.324 as given in [28]) are shown. Moreover, the comparison of the required computations for the test three scenarios is shown in Table IV.

Table III
EXPERIMENTAL RESULTS OF VIDEO EVENT DETECTION ON THE KTH VIDEO DATASET.

	Scenario-0	Scenario-1	Scenario-2
Training			
Feature extraction (s)	2505	1163 (46%)	596 (24%)
Clustering (s)	2873	2169 (76%)	2054 (72%)
BoF (s)	84	68 (81%)	45 (54%)
Testing			
Feature extraction (s)	2639	1258 (48%)	616 (23%)
BoF (s)	115	73 (63%)	55 (48%)
Average accuracy (%)	91.2	90.7	90.3

Table IV
EXPERIMENTAL RESULTS OF VIDEO EVENT DETECTION ON THE HOLLYWOOD2 ACTIONS VIDEO DATASET.

	Scenario-0	Scenario-1	Scenario-2
Training			
Feature extraction (s)	19629	8746 (45%)	5055 (26%)
Clustering (s)	2400	1685 (70%)	461 (19%)
BoF (s)	500	240 (48%)	151 (30%)
Testing			
Feature extraction (s)	20274	9150 (45%)	5422 (27%)
BoF (s)	623	346 (56%)	179 (29%)
Mean average precision	0.349	0.342	0.359

C. Near-Duplicate Video Detection

The benchmark video dataset CC_WEB_VIDEO [29] is used to evaluate the performance of the proposed MapReduce-based NDV retrieval implementation. The CC_WEB_VIDEO video dataset consists of 12790 videos and 398015 key frames. The available 398015 key frames are used to generate the visual dictionary with the size equal to 10000, and the Harris-Laplace detector [12] and SIFT descriptor [15] are employed.

The experimental results on the reference datasets generation and 1000 video queries in parallel are shown in Table V where the entire processing time is given. In addition, the speed-up of required computations of Scenario-1 and Scenario-2 as compared with Scenario-0 is also shown in Table V.

Table V
EXPERIMENTAL RESULTS OF NEAR-DUPLICATE VIDEO RETRIEVAL ON THE CC_WEB_VIDEO VIDEO DATASET.

	Scenario-0	Scenario-1	Scenario-2
Reference generation (s)	50405	21837 (43%)	13958 (28%)
1000 parallel queries (s)	3700	1844 (50%)	1193 (32%)

V. CONCLUSION

In this work, a MapRedce framework is proposed for speeding up large-scale multimedia data mining applications, including image classification, video event detection and near-duplicate video retrieval. The detailed MapReduce implementation for a number of commonly used computer

978-1-4673-4510-1/12/$31.00 ©2012 IEEE

vision approaches such as feature extraction, clustering and BoF generation are presented. Extensive experimental results on several benchmark image/video datasets demonstrate that the proposed MapReduce framework is able to efficiently reduce the processing time for image classification, video event detection and near-duplicate video retrieval.

ACKNOWLEDGMENT

This work was supported in part by the Program for Professor of Special Appointment (Eastern Scholar) at Shanghai Institutions of Higher Learning, the Program for New Century Excellent Talents in University of China under Grant NCET-10-0634, the Shanghai Pujiang Program under Grant 11PJ1409400, the National Natural Science Foundation of China under Grants 61102059 and 61103068, and the Fundamental Research Funds for the Central Universities under Grants 0800219158, 1700219104 and 0800219208, the Ph.D. Programs Foundation of Ministry of Education under Grant 20110072120017, and the Open Project Program of the National Laboratory of Pattern Recognition under Grant 201103187. The authors would like to thank Dr. Yugang Jiang and Dr. Wanlei Zhao for the helpful discussions.

REFERENCES

[1] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters", *Communications of the ACM - 50th anniversary issue: 1958-2008*, Vol. 51, No. 1, pp. 107-113, Jan. 2008.

[2] H. C. Yang, A. Dasdan, R. L. Hsiao, and D. S. Parker, "Map-reduce-merge: simplified relational data processing on large clusters", in *SCMD'07*, 2007, pp. 1029-1040.

[3] B. He and N. K. Govindaraju, "Mars: A MapReduce framework on graphics processors", in *PACT'08*, 2008, pp. 260-269.

[4] Y. Shan, B. Wang, J. Yan, Y. Wang, N. Xu, and H. Yang, "FPMR: MapReduce framework on FPGA", in *FPGA'10*, 2010, pp. 93-102.

[5] S. Papadimitriou and J. Sun, "DisCo: distributed co-clustering with Map-Reduce", in *IEEE ICDM'08*, 2008, pp. 512-521.

[6] Y. Li, D. J. Crandall, and D. P. Huttenlocher, "Landmark classification in large-scale image collections", in *IEEE ICCV'09*, 2009, pp. 1957-1964.

[7] L. Kennedy, M. Slaney, and K. Weinberger, "Reliable tags using image similarity: mining specificity and expertise from large-scale multimedia databases", in *WSMC'09*, 2009, pp. 17-24.

[8] R. Yan, M. O. Fleury, M. Merier, A. Natsev, and J. R. Smith, "Large-scale multimedia semantic concept modeling using robust subspace bagging and MapReduce", in *LS-MMRM'09*, 2009, pp. 35-42.

[9] B. White, T. Yeh, J. Lin, and L. Davis, "Web-scale computer vision using MapReduce for multimedia data mining", in *MDMKDD'10*, 2010, article No. 9.

[10] Apache Hadoop, http://hadoop.apache.org/core/

[11] T. White, Hadoop: the definitive guide (2^{nd} Edition), *O'Reilly Media, Inc.*, 2010.

[12] K. Mikolajczyk and C. Schmid, "Scale & affine invariant interest point detectors", *Int. J. Comput. Vision*, Vol. 60, No. 1, pp. 63-86, 2004.

[13] E. Nowak, F. Jurie, and B. Triggs, "Sampling strategies for bag-of-features image classification", in *ECCV'06*, 2006, pp. 490-503.

[14] I. Laptev, "On space-time interest points", *Int. J. Comput. Vision*, Vol. 64, No. 2/3, pp. 107-123, 2005.

[15] D. G. Lowe, "Distinctive image features from scale-invariant keypoints", *Int. J. Comput. Vision*, Vol. 60, No. 2, pp. 91-110, 2004.

[16] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies", in *IEEE CVPR'08*, 2008, pp. 1-8.

[17] LIP-VIREO, http://www.cs.cityu.edu.hk/∼wzhao2/lip-vireo.htm

[18] STIP, http://www.di.ens.fr/ laptev/download.html#stip

[19] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bag of features: spatial pyramid matching for recognizing natural scene categories", in *IEEE CVPR'06*, 2006, pp. 2169-2178.

[20] Y. G. Jiang, C.-W. Ngo, and J. Yang, "Towards optimal bag-of-features for object categorization and semantic video retrieval", in *CIVR'07*, 2007, pp. 494-501.

[21] M. Cherubini, R. de Oliveira, and N. Oliver, "Understanding near-duplicate videos: a user-centric approach", in *ACM Multimedia'09*, 2009, pp. 35-44.

[22] W. L. Zhao, X. Wu, and C.-W. Ngo, "On the annotation of web videos by efficient near-duplicate search", *IEEE Trans. Multimedia*, Vol. 12, No. 5, pp. 448-461, 2010.

[23] M. Douze, A. Gaidon, H. Jégou, M. Marszatke, and C. Schmid, "INRIALEAR's video copy detection system", in *TREVCID'08*, 2008.

[24] LIBSVM, http://www.csie.ntu.edu.tw/∼cjlin/libsvm

[25] C. Schüldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local SVM approach", in *IEEE ICPR'04*, 2004, pp. 32-36.

[26] Hollywood2 Actions video dataset, http://www.irisa.fr/vista/actions/hollywood2

[27] M. Everingham, L. van Gool, C. Williams, J. Winn, and A. Zisserman, "Overview and results of the classification challenge", in *PASCAL VOC'08 Challenge Workshop*, 2008.

[28] M. Marszalek, I. Laptev, and C. Schmid, "Actions in Context", in *IEEE CVPR'09*, 2009, pp. 2929-2936.

[29] X. Wu, C.-W. Ngo, A. Hauptmann, and H.-K. Tan, "Real-time near-duplicate elimination for web video search with content and context", *IEEE Trans. Multimedia*, Vol. 11, No. 3, pp. 196-207, Feb. 2009.

978-1-4673-4510-1/12/$31.00 ©2012 IEEE