SHARING THE TREES AMONG RANDOM FORESTS FOR EFFECTIVE AND EFFICIENT CONCEPT DETECTION

Tzu-Hsuan Chiu #1, Guan-Long Wu #2, Yu-Chuan Su *3,Winston H. Hsu #4

Graduate Institute of Networking and Multimedia, National Taiwan University
No. 1, Sec. 4, Roosevelt Road, Taipei, 10617 Taiwan(R.O.C)

* *Department of Computer Science & Information Engineering, National Taiwan University*
No. 1, Sec. 4, Roosevelt Road, Taipei, 10617 Taiwan(R.O.C)

***Abstract*—In this paper, we focus on the random forest based concept detection system, and we intend to improve the efficiency of the system in testing phase and to save memory and storage usages by reducing the total number of trees (classifiers). However, reducing the tree number often results in poor performance. In this article, we proposed a method called tree-sharing to cope with this issue. Unlike the traditional method that treats each concept independently, our work shares the trees among concepts, and leave the most important ones from the view of whole system. Experiments on different concept sets show tree-sharing can greatly reduce the number of total trees while the performance decreases slightly. Even in the worst case, we achieve 80% of original performance with only 5% of trees.**

I. INTRODUCTION

Concept detection, recognizing the concept (usually objects, locations, or people) of an image/video from visual content, has been the focus of considerable interest and research over the past years. Typically, concept detection is accomplished by machine learning techniques, and to train a binary classifier for each concept is a general scheme. For each concept, the training set (positive and negative samples) is given, and we apply the machine learning methods to learn a model from it. The machine learning-based concept detection has been extensively studied[1][2][3].

To further improve the concept detection performance, some studies suggest utilizing concepts relationship. Since concepts do not occur in isolation, the probability of the appearance of a specific concept in the test image should relate to other concepts[4][5], and this method is often referred as context-based concept fusion (CBCF).

Due to the prevalence of social media, the size of image/video collections to be indexed is increasing rapidly. Thus, take efficiency problem into account while designing the concept detection system is necessary. One of the promising approaches is to implement the concept detection system on distributed environment, such as Rong Yan's work[6], which is implemented on top of MapReduce.

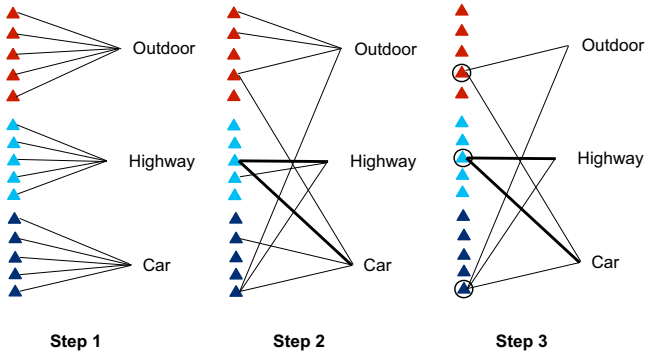

Fig. 1. Tree-sharing Scheme. A triangle denote a tree, and the thickness of line indicates the weight of each classifier for a specific concept. If there is no line, it means the weight is zero. Step 1. Constructing forests: The forests are constructed independently as the traditional method. Step 2. Deciding the weights: For each concept, we employ all constructed trees and decide the weight for each tree through optimizing the pre-defined loss function. Step 3. Reducing the total tree number: We leave several most significant trees (circled) according to the system budget. The system budget is assumed to be 20% of total trees (three) in this figure.

Ensemble learning is another emerging research topic[7][8][9]. It uses multiple weak models to form a stronger classifier. In this scheme, the models are often constructed with randomization, either in subsampling the training data or selecting the attributes. The test is applied on all constructed models, and the aggregation of responses from all models forms the classification result. Since each model can be treated independently in training and testing phase, ensemble learning is suitable for parallel computing in natural[7].

In this article, we employ the random forest for concept detection task. Random forest[7] is one of the most effective ensemble learning method and is widely used. We intend to improve the efficiency in testing phase and save the memory and storage usage by reducing the number of total trees with

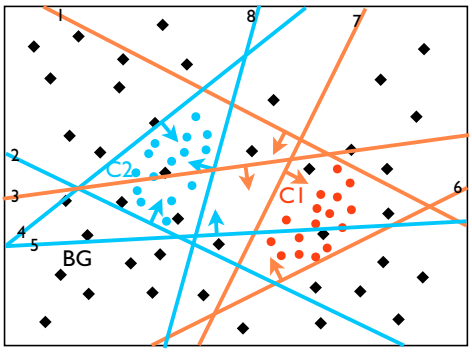

Fig. 2. Tree-sharing illustration. Each point represents a sample, and there are three concepts in the space, C1, C2 and background. The lines denote the classifiers constructed for distinguishing C1 and C2 from background and the arrow indicate the positive response part. It is obvious 1, 2, 4, 6, 7, 8 classifiers can be shared with either positive or negative weight, while 3, 5 are not suitable for sharing.

low performance degradation (ie. average precision). Unlike the traditional method that treats each concept independently, we share the trees among concepts; thus, although the total tree number is reduced, the serving tree number for each concept does not decrease as many. While this intuition sounds simple, how to share the trees is not trivial, which will be presented in the following sections. In this paper, we call the proposed method **tree-sharing**, and the words classifier, model and tree will be used interchangeably. Note that although we use random forest in this paper, this method could be applied to some other ensemble classifiers.

To the best of our knowledge, it is the first attempt on random forest to achieve this improvements by sharing trees among the concepts. Though there are other works discussing about sharing the classifiers, they are mostly focused on the boosting scheme[10][11]. The sharing processes are totally different and beyond the scope of this article.

The process of building concept detection system in our work contains three steps. First, the forests are created independently for each concept. Traditional concept detection system stops here, and each concept use the trees in its own forests to perform concept detection task with equal weight, and it is the baseline in this article. Second, we form a tree pool with all the constructed trees, and for each concept, the tree weights of all the pooled trees are decided. Third, we leave several most significant trees from the view of whole system according to the system budget. See the illustration in Figure 1, and we assume the system budget to be 20% of total trees (three) in this illustration. Traditionally, to obtain the scores of a test image on these three concepts, we need to test the image on fifteen classifiers; however, there are only three with the proposed tree-sharing method. Thus, we can improve the efficiency in testing phase and save the memory and storage usage at the same time. In the experiments on the different concept sets, we consistently find it decreases the total tree number greatly while the performance (average precision) drops slightly. Even in the worst case, we achieve 80% of original performance with only 5% trees left. Take Figure 1. for illustration, we are comparing the performance of Step 3. (using 20% of trees) to the Step 1. (using 100% of trees).

The weight deciding process is completed through optimizing a loss function. We all know that the loss function should be strongly related to the performance criterion (average precision in this article), so the best function should be AP. However, the challenge is AP cannot be optimized directly because it depends on the ranking results and cannot be expressed as a continuous function of the tree weight. In this article, we introduce a probabilistic framework to cope with this problem and finally we give a list-wise loss function which is strongly related to AP (section III-B2). The advantages of list-wise approach have been discussed in many works and would not be covered here.

The proposed method seems to be benefited from leveraging the correlation among concepts like CBCF, but the the leveraged information and the goal are quite different. They will be discussed in the next section.

This paper is organized as following. In section II, we introduce some related works and distinguish this work from them. In section III, we describe the proposed tree-sharing method in detail. The experiments are shown in section IV and the conclusions are given in section V.

II. Related Work

In II-A, we are going to introduce random forest briefly. In II-B and II-C, we introduce context-based concept fusion (CBCF) and distinguish our work from it.

A. Random forest

Random forest is an ensemble classifier consisted of a number of decision trees. Since the performance of the ensemble classifier is highly related to the correlation among each model in it, the trees are often constructed with some randomization. Generally speaking, randomization comes from two points: (1) Subsampling the training data and each tree are grown with different data. (2) For each internal node, selecting some attributes for split. Besides, each internal node contains a best split of training data. While testing, the test sample is applied on each tree, and the final result is the aggregation of the results from all trees. This method was first introduced in [9], and further developed in [7].

B. Context-based concept fusion

Since concepts do not occur in isolation, the probability of the appearance of a specific concept in the test image should relate to other concepts. The basic idea of CBCF is to model the probability $P(Ci|S1, S2, ...)$, where $S1, S2, ...$ is the raw probabilities $P(Ci|\mathbf{x})$ generated from each concept detector, where $\mathbf{x}$ is the test image/video.

In [4], W. Jiang proposed a method called a Boosted Conditional Random Field (BCRF). In this method, they use CRF to model the probability $P(Ci|S1, S2, ...)$. Each node is the concept and the edges represent the pair-wise relationships between concepts. The BCRF method has two layers. It takes

the posterior probabilities of the test image/video produced from independent concept detectors as the inputs, and the detection results for each concept are refined with these inputs in the second layer.

In [5], Kennedy proposed a classification based reranking method. For each concept, it first gets the ranking list of a set of images using the baseline concept detector. Then it takes the higher-ranked and lower-ranked images as pseudo-positive and pseudo-negative samples, and takes the responses of images on baseline concept detectors of related concepts as the feature to train a SVM (support vector machine) classifier. Finally, they considered the normalized classification score of each image to be the refined ranking score.

C. Difference between CBCF and our work

Although our work seems to be benefited from leveraging the correlation among concepts, there are two main differences between CBCF and our work. First, CBCF utilize the concept level information (the output of concept detector) while we utilize the sub-concept level information (the output of individual trees). Second, our work aims to reduce the total number of trees, which is not the consideration of CBCF.

There are some conditions which can be dealt with sub-concept but not concept level information. For example, there are two concepts to be detected: Dog and outdoor. Suppose dogs appear in half of the outdoor images; thus, the probability of test image belonging to outdoor gives no information about whether dogs appear. However, in our tree-sharing scheme, there might be trees suitable for sharing between these two concepts. Suppose there is a tree constructed from outdoor concept that classifies the test image according to wether it contains tornado or not. When it gives positive response, the probability of appearance of dog in the test image should decrease (suppose dog rarely appear with tornado) while outdoor should increase. Therefore, this tree is useful for both concepts, and can be used by dog concept to improve the detection accuracy. Although the situation is much more complex in practice since the attribute are selected randomly for each tree, even each node, and may have no human comprehensive meaning, we can still find the trees suitable for sharing with carefully designed method.

III. Tree-Sharing

In this section, we are going to describe why sharing the classifiers among concepts is feasible and how we accomplish weight decision and classifier number reduction in detail. The classifiers we discuss here are all binary classifiers.

A. Concept

The scenario of tree sharing is shown in Figure 2. The classifiers 1, 2, 4, 6, 7, 8 are suitable for sharing while 3, 5 are not. More specifically, for concept 1, it can use classifiers 1, 2, 3, 4, 6, 7 with positive weight, and classifier 8 with negative weight. Since the classifiers (trees) can be shared among concepts, we call this method **tree-sharing** . Then, for each concept, the weight of each tree needs to be decided. After deciding the tree weights for all concepts, we can remove the less used trees according to the system budget. Finally, The testing result is the linear combination of the responses from all left trees with the decided weights.

B. Deciding the tree weights

1) Formulating the problem: Given m concepts and construct n trees for each of them (In figure 1., m is 3 and n is 5).We denote the j-th tree of the i-th concept as t_{ij}. For each concept, we would like to decide the weights of total mn trees. We denote the weights for the i-th concept by a $mn \times 1$ vector $\mathbf{w}^i$. We will obtain $\mathbf{w}^i$ through minimizing a loss function $Loss^i(\mathbf{w})$.

$$\mathbf{w}^i = arg\ min_{\mathbf{w}}\ Loss^i(\mathbf{w}) \tag{1}$$

2) Defining the loss function: For the i-th concept, we define the scoring function to be $F^i_{\mathbf{w}}(img) = Rsp^i(img) * \mathbf{w}$, where $Rsp^i(img)$ is a $1 \times mn$ vector formed by testing img on all mn trees, and img is the test image. Since the trees are binary classifiers here, the mn elements are either 0 or 1. Suppose we have p images $I = \{img^i_1, img^i_2, ..., img^i_p\}$ and their relevance scores $r = \{r^i_1, r^i_2, ..., r^i_p\}, r^i_1, r^i_2, ..., r^i_p \in \{0, 1\}$ for the i-th concept. Since the evaluation criterion in this paper is AP (average precision), given the training images, we want to find $\mathbf{w}^i$ which makes $F^i_{\mathbf{w}}$ optimize the AP. For convenience, we ignore the superscript i later.

$$AP = \frac{1}{R} \cdot \sum_{k=1}^{p} r_k \cdot \frac{\pi_{rel}(k)}{\pi(k)} \tag{2}$$

Where R is the number of relevant images in I, π is the permutation according to scoring function $F_{\mathbf{w}}$, $\pi_{rel}(k)$ is the rank of the k-th image in relevant images in I , $\pi(k)$ is the rank of the k-th image in all images in I. Since r_k is either 0 or 1,we only need to consider the relevant images, and $\frac{\pi_{rel}(k)}{\pi(k)}$ is simply the precision at $\pi(k)$.

It is challenging to maximize the above equation for the reason that the dependence of image ranks on $F_{\mathbf{w}}$ is not expressed explicitly ($\mathbf{w}$ does not appear in this equation). To overcome this problem, we introduce a claim[12] and a probabilistic framework inspired by [13][14].

Claim 1: **R-precision is approximately AP**

In [12], the author states that under some reasonable assumptions, the R- precision approximates the area under the precision-recall curve. Since AP is also an approximation of the area under the precision- recall curve. R-precision is approximately AP. In [12], they have confirmed this claim on the TREC8 collection.

According to Claim 1, we maximize the R-precision instead of AP. The R-precision is computed as following:

$$RP = Rel(R)/R = \frac{1}{R} \cdot \sum_{k=1}^{p} r_k \cdot \mathbf{I}(\pi(k) \leq R) \tag{3}$$

Where $Rel(R)$ means the number of relevant images in the R first ranked images, and $\mathbf{I}$ is an indicator function. With this equation, we avoid dealing with $\frac{\pi_{rel}(k)}{\pi(k)}$ in equation (2).

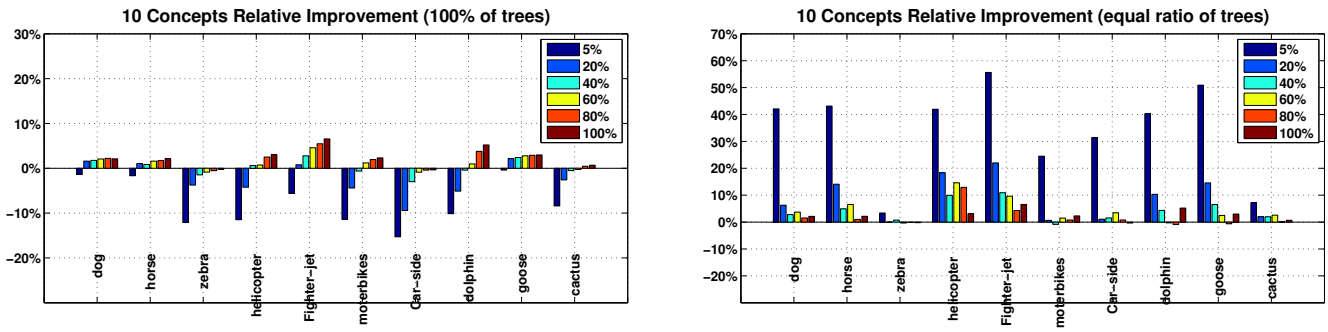

Fig. 3. Relative improvement of 10 concepts experiment. Different colors of bar indicate the ratio of employed trees in tree-sharing method. Left: The relative improvement with respect to baseline method employing 100% of constructed trees ($\frac{AP\ of\ Tree-sharing\ (using\ various\ ratio\ of\ trees)}{AP\ of\ Baseline\ (using\ 1000\ trees)}$). Right: The relative improvement with respect to using baseline method employing the same percentage of total trees as tree-sharing (take some classifiers off) ($\frac{AP\ of\ Tree-sharing\ (using\ various\ ratio\ of\ trees)}{AP\ of\ Baseline\ (using\ the\ same\ ratio\ of\ trees\ as\ tree-sharing)}$).

To express the dependence of image ranks on $F_{\mathbf{w}}$ explicitly, we maximize its expectation instead.

$$\varepsilon(RP) = \frac{1}{R} \cdot \varepsilon(Rel(R)) = \frac{1}{R} \cdot \sum_{k=1}^{p} r_k \cdot Pr(\pi(k) \leq R) \quad (4)$$

In equation (4), the probability $Pr(\pi(k) \leq R)$ needs to be defined. Besides, we model $Pr(\pi|F_{\mathbf{w}})$ to explicitly relate image ranks to $F_{\mathbf{w}}$. We conform to [13], and model $Pr(\pi|F_{\mathbf{w}})$ using equation (5).

$$Pr(\pi|F_{\mathbf{w}}) = \frac{1}{Z(F_{\mathbf{w}})} \cdot \exp[\sum_{k=1}^{p} \sum_{l:\pi(l)>\pi(k)} (F_{\mathbf{w}}(img_l) - F_{\mathbf{w}}(img_k))] \quad (5)$$

Then, we can derive lemma 1.

Lemma 1: **The expectation of the rank of the l-th image can be approximate with the equation**

$$\varepsilon(\pi(l)) \approx 1 + \sum_{k=1}^{p} \frac{1}{1 + \exp[2(F_{\mathbf{w}}(img_l) - F_{\mathbf{w}}(img_k))]} \quad (6)$$

The proof is provided in [13].
Finally, we model $Pr(\pi(k) \leq R)$ with a logistic function $L(\alpha(R - \varepsilon(\pi(k))))$, and the equation is as following:

$$\varepsilon(RP) \approx \frac{1}{R} \cdot \sum_{k=1}^{p} r_k \cdot L(\alpha(R - \varepsilon(\pi(k)))) \quad (7)$$

Replacing $\varepsilon(\pi(k))$ in equation (7) with equation (6), we can see it clearly that the $\varepsilon(RP)$ is now a function of $\mathbf{w}$. We define the loss function $Loss^i(\mathbf{w}) = -\varepsilon(RP)$. Therefore, minimizing the loss function is strongly related to maximizing the average precision. We simply set the α in equation (7) to 1.0 here, and solve this optimization problem with the active set algorithm.

C. Reducing the tree number

To reduce the tree number, we need to measure the significance of each tree from the view of whole system. After deciding the weights, we get $\mathbf{w}^1, \mathbf{w}^2, ..., \mathbf{w}^m$. We gather these weight vectors to form a $mn \times m$ matrix $\mathbf{W} = [\mathbf{w}^1, \mathbf{w}^2, ..., \mathbf{w}^m]$. Each row of this matrix is consisted of the weights of each tree on all concepts. Since the absolute value of the weight can be considered as how important the tree is for a specific concept, we take the L1-norm of each row vector as the significance of each tree. Then, we sort the trees by their significances and leave the most important ones according to the system budget. In testing phase, the final result of a test image on each concept is the linear combination of the response of the left trees with the computed weights.

IV. EXPERIMENTS

We present two sets of experiments in this section, (a) The relative performance of tree-sharing with respect to baseline method. (Section IV-B and IV-C) (b) The ontology constructed using the computed weight vector $\mathbf{w}^1, \mathbf{w}^2, ..., \mathbf{w}^m$. (Section IV-D).

A. Experimental settings

All experiments are conducted on the subset of Caltech-256 database[15]. To examine the consistent improvement, we experimented on two different concept sets. One has ten and another has twenty categories (concepts). The visual feature we used is SIFT[16] with Hessian-affine region detector and one hundred trees were constructed for each concept. The training set for each concept are formed by sampling thirty images from its category and five hundred from background. However, since we don't want the training set to be too unbalanced, we actually use only two hundred and forty of the five hundred background images to construct the trees. The test set of each concept consists of twenty-five images from its concept, and twenty-five images from background. AP is the performance criterion in these experiments. Besides, for each concept, we use its whole training set for deciding the tree weights in tree-sharing method. Since part of the training sets are used for constructing the trees of its concept, using the training set for weight decision would prefer the trees constructed with it. To alleviate this problem, we can multiply the responses of the trees of its concept by a constant (between 0 and 1). However, in these experiments, we simply

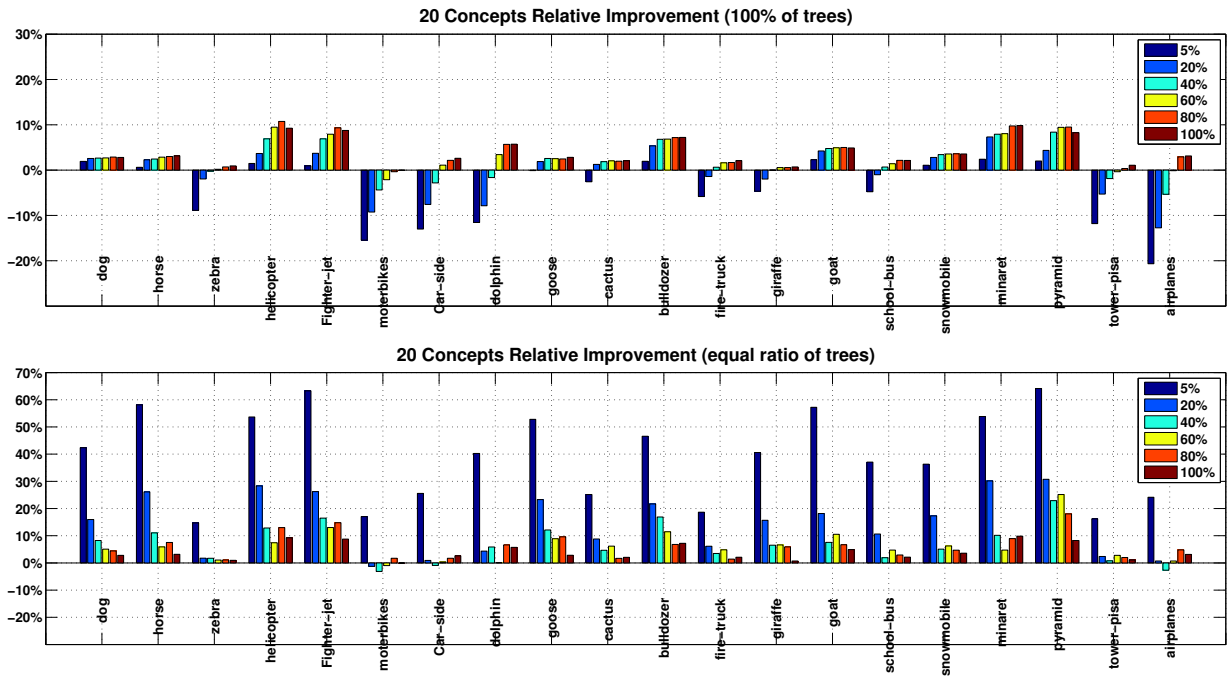

Fig. 4. Relative improvement of 20 concepts experiment. Different colors of bar indicate the ratio of employed trees in tree-sharing method. Up: The relative improvement with respect to baseline method, which uses 100% of constructed trees. Down: The relative improvement with respect to baseline method, which employs the same percentage of total trees as tree-sharing (take some classifiers off).

set the constant to 1 because we think it is also reasonable to prefer the trees constructed with its own training set. Matlab optimization tool is employed to minimize the loss function. In addition, all the numbers reported here are the average of ten runs.

B. Relative performance on ten concepts

The ten concepts we choose here are dog, horse, zebra, helicopter, fighter-jet, motorbikes, car-side, dolphin, goose and cactus. The experiment of ten concepts is shown in Figure 3. Figure 3. Left. shows tree sharing is capable of reducing the tree number greatly without serious performance loss. Even in the worst case (car-side), the performance drops about 15% while there are only 5% trees being used.

From Figure 3. Right, we see tree-sharing benefit all concepts more or less. Note we get more improvement while the used tree ratio is decreasing. It is because the baseline method is much more sensitive to the decreasing tree number. Imagine while there are only 5% of trees left, each concept has only 5 trees without tree-sharing, but there are still 50 trees for each concept with tree-sharing method.

C. Relative performance on twenty concepts

The twenty concepts are the previous ten and bulldozer, fire-truck, giraffe, goat, school-bus, snowmobile, minaret, pyramid, tower- pisa and airplane. In twenty concepts experiment, we get results consistent with the previous one (Figure 4). Note that we seem to get more improvement on the first ten concepts than the previous experiment. It is because the concept number is twice now; thus, the number of total tree and the number of the trees serving for each concept become twice. More specifically, the number of trees serving each concept is independent of concept number with traditional method while it is in proportion to concept number with the tree-sharing method. This implies tree-sharing prefer large number of concepts in nature, and it should be an advantage for large-scale problem.

D. Constructing the ontology

In addition to reducing total tree numbers in the system, we show that the computed weights can be used to construct the concept ontology automatically, which is useful for representing the relationships among concepts.

Similar concepts should have similar weight distributions on the trees. Therefore, we can take the computed weight vectors $\mathbf{w}^1, \mathbf{w}^2, ...,$ and $\mathbf{w}^m$ as the representation of each concept, and perform the agglomerative clustering on them to construct the ontology. We adopt the cosine distance to be the distance metric because the behavior of a concept detector is invariant to the scale of weights. (Multiplying the weight vector of a concept by a positive constant doesn't affect the behavior of the concept detector). Figure 5. shows the clustering result on twenty concepts. For convenience, we set a threshold 0.78, and the concepts in the same cluster are marked with the same color.

It is reasonable to see that dog, goat, goose, and horse in the same cluster, fire-truck and school bus be together, helicopter be clustered with fighter-jet, and motorbikes and car-side in the same cluster. Nevertheless, it seems wiered to cluster cactus, bulldozer and giraffe, or zebra and tower-pisa together.

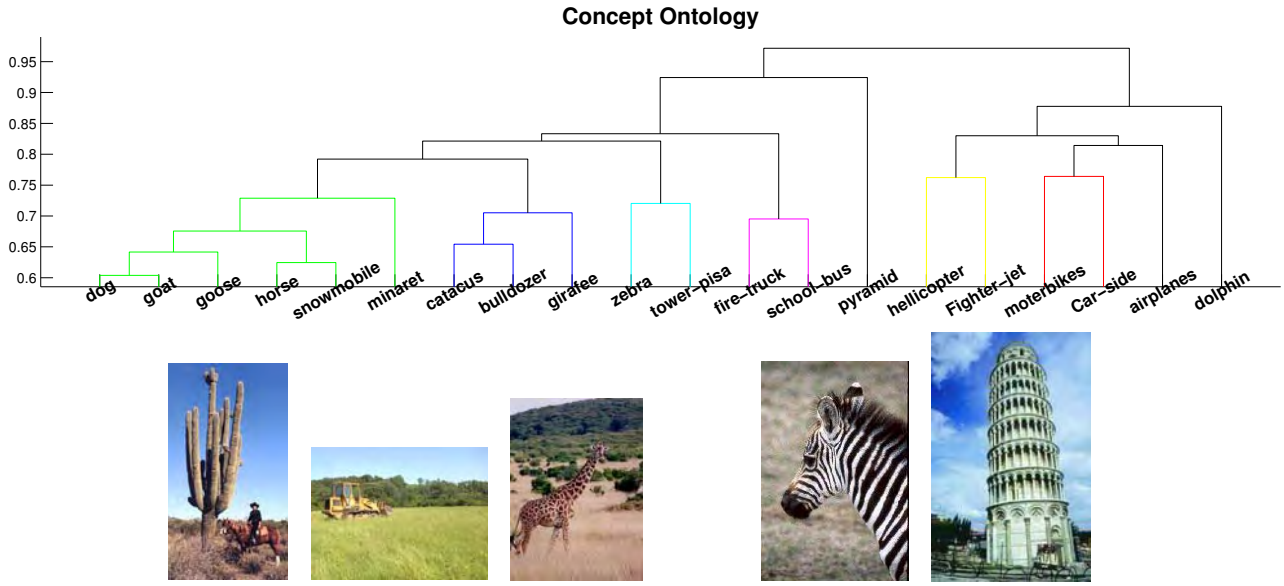

Fig. 5. Up: The constructed ontology of 20 concepts. Down: The images sampled form cactus, bulldozer, giraffe, zebra and tower-pisa. Note the similar background in the first three images and the interleaved black and white patterns in the last two. These make the first three and the last two visually related.

However, after sampling some images from these concepts, we found they are actually visually related. See Figure 5.

Compared with ontology built by human expert, this method may discover the concepts that are visually related.

V. CONCLUSIONS

This paper focus on random forest based concept detection system, and we enhance the efficiency in testing phase and save the memory and storage usage by the proposed tree-sharing method. To the best of our knowledge, it is the first attempt on random forest to achieve this improvements by sharing the trees among concepts. The proposed method greatly alleviate the impact of reducing the tree number. We experiment on different subsets of Caltech-256 to examine this method, and it shows we can achieve 80% of original performance with only 5% trees. Besides, we find it performs better in twenty concepts than ten with the same ratio of trees used. The reason is the serving trees for twenty concepts are twice than ten concepts. It implies this proposed method is suitable for large number of concepts detection task in nature.

REFERENCES

[1] A. Hauptmann, M. y. Chen, M. Christel, C. Huang, W. h. Lin, T. Ng, A. Velivelli, J. Yang, R. Yan, H. Yang, and H. D. Wactlar, "Confounded expectations: Informedia at trecvid 2004," in *In Proc. of TRECVID*, 2004.

[2] G.-J. Qi, X.-S. Hua, Y. Rui, J. Tang, T. Mei, and H.-J. Zhang, "Correlative multi-label video annotation," in *Proceedings of the 15th international conference on Multimedia*, ser. MULTIMEDIA '07. New York, NY, USA: ACM, 2007, pp. 17–26.

[3] J. Smith, M. Naphade, and A. Natsev, "Multimedia semantic indexing using model vectors," in *Multimedia and Expo, 2003. ICME '03. Proceedings. 2003 International Conference on*, vol. 2, july 2003, pp. II – 445–8 vol.2.

[4] W. Jiang, S.-F. Chang, and A. Loui, "Context-based concept fusion with boosted conditional random fields," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 1, april 2007, pp. I–949 –I–952.

[5] L. S. Kennedy and S.-F. Chang, "A reranking approach for context-based concept fusion in video indexing and retrieval," in *Proceedings of the 6th ACM international conference on Image and video retrieval*, ser. CIVR '07. New York, NY, USA: ACM, 2007, pp. 333–340.

[6] R. Yan, M.-O. Fleury, M. Merler, A. Natsev, and J. R. Smith, "Large-scale multimedia semantic concept modeling using robust subspace bagging and mapreduce," in *Proceedings of the First ACM workshop on Large-scale multimedia retrieval and mining*, ser. LS-MMRM '09. New York, NY, USA: ACM, 2009, pp. 35–42.

[7] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[8] T. K. Ho, "The random subspace method for constructing decision forests," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 8, pp. 832 –844, Aug. 1998.

[9] Y. Amit and D. Geman, "Shape quantization and recognition with randomized trees," *Neural Comput.*, vol. 9, no. 7, pp. 1545–1588, Oct. 1997.

[10] A. Torralba, K. Murphy, and W. Freeman, "Sharing features: efficient boosting procedures for multiclass object detection," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 2, june-2 july 2004, pp. II–762 – II–769 Vol.2.

[11] A. Torralba, K. Murphy, and Freeman, "Sharing visual features for multiclass and multiview object detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 5, pp. 854 –869, may 2007.

[12] J. A. Aslam, E. Yilmaz, and V. Pavlu, "A geometric interpretation of r-precision and its correlation with average precision," in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, ser. SIGIR '05. New York, NY, USA: ACM, 2005, pp. 573–574.

[13] H. Valizadegan, R. Jin, R. Zhang, and J. Mao, "Learning to rank by optimizing ndcg measure," in *NIPS*. Curran Associates, Inc., 2009, pp. 1883–1891.

[14] M. Volkovs and R. S. Zemel, "Boltzrank: learning to maximize expected ranking gain." in *ICML*, ser. ACM International Conference Proceeding Series, A. P. Danyluk, L. Bottou, and M. L. Littman, Eds., vol. 382. ACM, 2009, p. 137.

[15] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," *Annals of Physics*, no. 7694, pp. 1–20, 2007.

[16] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004.