This is the Pre-Published Version.

The following publication J. Hu, K. -M. Lam, P. Lou and Q. Liu, "Constructing a hierarchical tree for image annotation," 2017 IEEE International Conference on Multimedia and Expo (ICME), Hong Kong, China, 2017, pp. 265-270 is available at https://doi.org/10.1109/ICME.2017.8019512.

CONSTRUCTING A HIERARCHICAL TREE FOR IMAGE ANNOTATION

Anonymous ICME submission

ABSTRACT

Image annotation is always an easy task for humans but a tough task for machines. Inspired by human's thinking mode, there is an assumption that the computer has double systems. Each of the systems can handle the task individually and in parallel. In this paper, we introduce a new hierarchical model for image annotation, based on constructing a novel, hierarchical tree, which consists of exploring the relationships between the labels and the features used, and dividing labels into several hierarchies for efficient and accurate labeling.

Index Terms— Annotation, hierarchical, tree, label

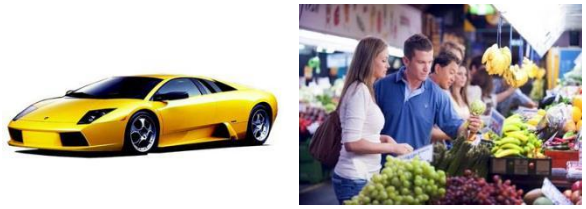
1. INTRODUCTION

In pursuit of the ultimate goal of building an intelligent system capable of managing images in the way humans do, it is necessary to explore the intrinsic essence of machine-based technologies for real-world applications and human knowledge. Annotation, a very hot topic these days, allows for an image to be searched by the use of text. On this point, automated annotation tends to be much more practical than using a manual process, when databases are large. If we can find a reliable mapping between low-level features and high-level concepts, keyword-based image retrieval will be more meaningful than content-based image retrieval [1].

Human beings see image-annotation tasks as an easy problem. The related tags that we assign to an image can be classified into 2 categories. As shown in Fig. 1, one category includes those basic or obvious tags that we do not need to think about, e.g. car, sky, dog, etc. The other is the more complex or abstract tags that we need to think over, e.g. market, Asia, indoor, etc. We wonder if a machine can have such two systems, like human beings? Therefore, in this paper, we propose a hierarchical framework to mimic the two systems for handling tags, i.e. with solid concepts and abstract concepts, respectively.

In order to exploit the correlations between the class labels, we introduce a method by constructing a tree structure, which classifiers image labels into different levels of a hierarchy according to their level of abstraction. In other words, the labels or graphs of labels are linked to each other through the tree structure.

The remainder of this paper is organized as follows. In Section 2, a brief introduction to related works will be given. We present our proposed method in detail in Section 3. The experiment set-up and results, and a conclusion, are given in Sections 4 and 5, respectively.



(a) Car (b) Market

Fig. 1. (a) An image with a simple, solid tag, and (b) an image with a confusing, abstract tag.

2. RELATED WORKS

In this section, we discuss image-annotation models that are relevant to our proposed algorithm. The literature can be grouped into three models: generative models, discriminative models [2,3,4,5], and nearest-neighbor-based models. Most generative models [6,7] construct a joint distribution over an image's contents and the keywords to find a mapping between the image features and the annotation keywords. These generative models aim to learn a single model for all the vocabulary terms, which yields better modeling in terms of dependencies. Some methods treat the task of image annotation as several binary classification problems. This means that the joint distribution of the unobserved variables and the observed

©2017 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

variables is not needed. In this situation, discriminative models [4] can generally yield a superior performance. The discriminative models learn a separate classifier for each single label, and use the classifier to judge whether the test image belongs to this class or not. Although the training process is complicated and time-consuming, this approach can, with a smart design, achieve more promising performances than the generative models. The third model, as one of the oldest, simplest, and most effective methods for pattern classification, is the KNN-based model [10], which is accurate, especially with an increasing number of training data.

Recently, a NN-based keyword-transfer approach was proposed in [11]. In this method, the labels are transferred from neighbors to a given image after a simple distance calculation. The nearest neighbors are determined using Joint Equal Contribution (JEC) only, which finds the average distance obtained from the differences in image features. The method was extended in [12] to filter out most of the irrelevant labels, with a promising result obtained.

To address the problem of a large number of labels, contextual modeling [8,9] has become a recent focus. In [13], structured prediction models are proposed for image labeling, which explicitly takes the dependencies among image labels into account. In the tree-structured models, the nodes represent image labels, and the edges between the nodes encode the dependency relations. To allow for more complex dependencies, labels are combined in a single node, and mixtures of trees are used.

This paper aims to devise a learning algorithm to handle labels in a human way. Although it is difficult for machines to handle labels in different ways according to each label's nature, we can construct a hierarchical tree structure among the labels to facilitate this capability.

For image representation, we follow our previous work [14] using pools of features. We will learn the mapping between the tag space and the image-representation space. We aim to boost the performance of feature-label selection, based on the constructed hierarchical structure. Our proposed approach can achieve a good balance between efficiency and accuracy.

3. CONSTRUCTION OF A TREE HIERARCHY

Suppose that the set $\Gamma = \{l_1, l_2, \ldots, l_n\}$ represents the dictionary of the labels for a whole image dataset. The training images are denoted as $\{i_1, i_2, \ldots, i_N\}$, while l_{Si} ($Si \in \{l_1, l_2, \ldots, l_n\}$) represents the labels of the image l_i in the training set. Then, assuming that each image contains no more than 4 labels, a tree structure can be constructed using the following steps:

Step 1. Only images containing label pairs (i.e. two labels) are considered in this step. We count the frequency of each of these label pairs (e.g. Fruit-Apple, assigned to *Layer* 4), which appears in the same training images. Sort these label pairs in descending order according to their frequencies.

Step 2. Only images containing label triplets (i.e. three labels) are considered in this step. We count the frequency of each of these label triplets (e.g. Fruit-Apple-Market, assigned to *Layer* 3), which appears in the same training images. Sort these label triplets in descending order according to their frequencies.

Step 3. We count the frequency of each single label which appears with a label pair (e.g. Garden-(Fruit, Apple), where the label "Garden" appears with the label pair "Fruit, Apple", assigned to *Layer* 2). Sort these labels in descending order according to their frequencies.

Step 4. We count the frequency of each single label which appears with a label triplet (e.g. Crowd-(Fruit-Apple-Market), assigned to *Layer* 1). Sort these labels in descending order according to their frequencies.

Step 5. We can now construct a 6-layer tree-hierarchical structure. The bottom layer, namely "*Layer* 0", consists of all the individual class labels (*n* classes), and each node in this layer contains a class label. Then, *Layer* 1 to *Layer* 4 are constructed according to Steps 1 to 4, such that each node in each of the layers contains different combinations of the labels. Finally, the top layer, i.e. *Layer* 5, contains all the labels in a single cluster.

The above steps illustrate the construction of a hierarchy for a number of labels. Specifically, the single labels, label pairs, and label triplets will form individual small clusters. Each of these clusters is considered a node in the tree structure. The constructed tree structure represents the relationships between the labels and the corresponding image features. The tree structure can be extended to include images having even more labels. However, the number of these types of images is usually small.

Using our previous work [12,14], we can learn image exemplars for each of the label classes (i.e. single label, label pairs, single label+label pairs, label triplets, single label+label triplets, and all). These image exemplars can be obtained by incorporating image patches into a hypergraph. The exemplars are good representations of each class label, which are used to represent the nodes in the tree structure in our proposed framework. Then, we can extract the corresponding image features relating to each node in the constructed hierarchical graph.

In this paper, we propose a novel algorithm which incorporates different feature pools for a hierarchical training of node classifiers. Unlike our previous work, classifiers are not trained for each label class. We learn a classifier for each node in the tree structure instead of learning a classifier for each single label. As was done in [15], we train a regression model with the use of the tree structure. But unlike [15], we extend the nodes which combine labels in different ways in our proposed framework.

Assume that the correlations among the labels can be represented using a tree structure consisting of a set of vertices or nodes V. In the tree structure, each bottom-leaf node corresponds to a single label(i.e. A, B, C, D, E, F, G and H in Fig. 2) in the dictionary, while the middle nodes represent the label pairs(e.g. AB), label triplets(e.g. ABC), and two types of extended nodes, which are formed by combining a single label with either a label pair or a label triplet, as illustrated in Fig.2.

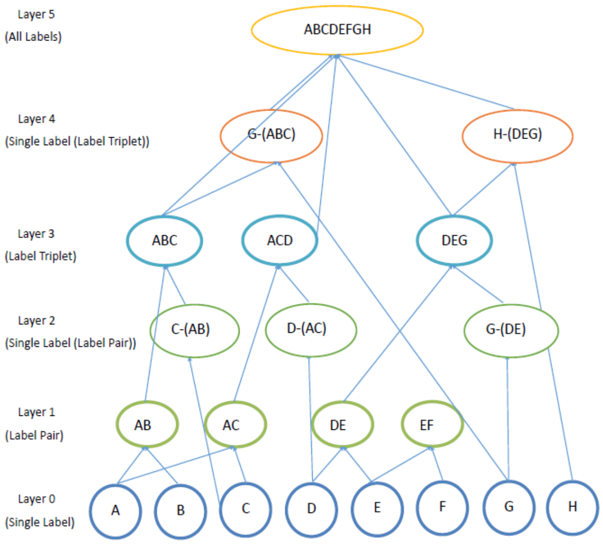


Fig. 2. The tree structure constructed using our proposed model, where *Layer* 0 contains eight bottom-leaf nodes corresponding to eight labels (*A*, *B*, *C*, *D*, *E*, *F*, *G*, and *H*) in the dictionary; *Layer* 1 and *Layer* 3 contain label pairs and label triplets, respectively; *Layer* 2 and *Layer* 4 are the extended nodes, which are those nodes composed of a single label with either a label pair or triplet, respectively; and *Layer* 5 is a single node which contains all the labels.

To train a node classifier, a feature pool will first be selected from a set of feature pools, and then a specific classifier is trained using the selected feature for the node. In the first step of our algorithm, the transferable knowledge between nodes and the common features among the different nodes are not considered. Only the most suitable feature pool for each node is identified. It should be noted that the same feature pool may be selected and used for a number of nodes. In the second step of our training, we incorporate the multi-task learning algorithm in [16] in our framework to train the node classifiers. Fig. 3 shows the details of our tree-structure-based feature-label selection algorithm, which consists of a training stage and a testing stage. In testing a query input, we aim to find the likelihood of a test image belonging to each node. Then, the final score for each label class is computed according to hierarchical factors (i.e. the weights learned for the nodes). The Problem Statement and Formulation will be given in the following.

Training Step:

Step 1: Feature Selection (AdaBoost) – A classifier is trained for each node used to find the features from the pool that result in the best performance for the class labels, with the help of the hierarchical factors.

Step 2: Use the multi-task learning algorithm in [16] to learn a classifier for each node based on the selected feature pool.

Step 3: Train a biased classifier for each feature pool for labeling novel images.

Testing Step:

Step 1: Use the biased classifier trained for each feature pool to judge which of the feature pools is the most suitable for classifying a test image.

Step 2: Use the node classifiers to output scores (in the form of probabilities) which represent how likely it is that the test image belongs to the corresponding node.

Step 3: Consider the hierarchical factor for each node in computing the final score for each label class, and choose the labels with the highest scores.

Fig. 3. The algorithm for training the nodes and for label selection.

In the image-annotation problem, upon receiving a query or test image I_q, the annotation algorithm will output its corresponding labels $l(I_q, t)$, where $l(I_q, t)$ refers to the set of tags t related to the query image I_q. Assume that the label set S contains n classes, and that $l(I_q, t)$ is a subset of S. An exemplar image for the i^{th} class label is denoted as I_e^i. The aim of annotation is to find the tags t^* that maximize the conditional distribution $p(t|I_q)$. The feature pool used in learning is denoted as $F = \{f_1, ..., f_j, ..., f_M\}$, where f_j represents one type of features (e.g. color histogram, local shape descriptors, etc.). Denote d_j as the dimension of the feature f_j. Then, the total dimensionality of all the image features in the feature pool F is d, where $d = \sum_{j=1}^{M} d_j$. We can form a feature matrix $X_j \in \Re^{n' \times d_j}$ to represent the features of n' training images using the j^{th} feature pool. Then, we can learn a corresponding regression coefficient vector $\beta_{kj} \in \Re^{d_j}$ for the k^{th} node. Since we have to output the final scores for each label class in the last step, we have $\beta_k = (\beta_{k1}^T, ..., \beta_{kM}^T)^T$, which can be solved.

Following our previous work, the weak classifier for the k^{th} node T_k can be defined as follows:

$$\Phi_{T_k}(X) = W_k^T X + b, \tag{1}$$

where $W_k = W_c + V_k$, W_c is a common regularization term shared by those classifiers using the same feature pool, and V_k is the specific regularization term for the individual node class. X is the feature vector of the training samples (which will be described in Section 4), and b is an offset. For the second and third steps of our training algorithm, we aim to learn a classifier for each node using the same feature pool, and then we train a biased classifier for each feature pool. Following are the specific details of our training algorithm.

The training samples which result in the best performance with the same feature pool j are denoted as f_j: $W = \{X_{jk}, Y_{jk} \mid j = 1,...N; k = 1,...,L\}$, where L is the number of training samples using the j^{th} feature pool, X is the feature vector, Y is the label, and k is the node index. Training a multiple number of classifiers for each node class using the same feature pool f_m is then transformed into a joint optimization problem as follows:

$$\min\{C \sum_{k=1}^{L} \sum_{j=1}^{N} \xi_{ij} + \beta_1 \sum_{k=1}^{L} \|V_k\|^2 + \beta_2 \|W_c\|^2\} \tag{2}$$

subject to:

$$\forall_{j=1}^{N} \forall_{k=1}^{L} : Y_{jk}(W_c + V_k) \cdot X_{jk} + b \geq 1 - \xi_{jk}, \xi_{jk} \geq 0,$$

where $\xi_{jk} \geq 0$ represents the training error rate, β_1 and β_2 are positive regularization parameters, and C is the penalty term. The dual optimization problem for the above equation is to determine the optimal α_{jk}^* by:

$$\max\left\{\sum_{k=1}^{L}\sum_{j=1}^{N}\alpha_{ij} - \frac{1}{2}\sum_{k=1}^{L}\sum_{j=1}^{N}\sum_{h=1}^{L}\sum_{l=1}^{N}\alpha_{ih}Y_{ih}\alpha_{kl}Y_{kl}K_{kh}(X_{jh,}X_{kl})\right\} \tag{3}$$

subject to:

$$\forall_{j=1}^{N} \forall_{k=1}^{L} : 0 \leq \alpha_{jk} \leq C, \sum_{k=1}^{L}\sum_{j=1}^{N}\alpha_{jk}Y_{jk} = 0,$$

where $K_{kh}(.,.)$ is the underlying kernel function. Our multi-task learning algorithm is able to handle the visual similarity among the node classes using the same feature pool.

4. EXPERIMENTS AND RESULTS

Four benchmark image databases are used in our framework. The first database used in the experiments is Corel 5K, which contains 5,000 images, comprising 4,500 training samples and 500 testing samples. Each image in the dataset is annotated with about 3.5 keywords on average, and the dictionary has a total of 374 words or labels. Another dataset used is Corel 30K, which is similar to Corel 5K except that it is substantially larger, containing 31,695 images and 5,587 words or labels. The third dataset used is the ESP Game dataset, which contains 18,689 training images and 2,081 testing images.

To make a fair comparison with other state-of-the-art methods, we choose precision and recall as our evaluation criteria. The precision rate and recall rate for each test image are measured by comparing the annotated results to the ground-truth, and then the average precision and recall of all the test images are computed to form the final results.

Table 1 shows the performances of our proposed method versus some state-of-the-art methods. Three performance indices are measured: the mean precision rate (P %), the mean recall rate (R %), and the number of total keywords recalled (N+), respectively. Our method outperforms all the other methods in terms of the mean precision rate, which is the most important measurement. Our method also achieves a better performance than most of the other methods in terms of the mean recall rate. Although the recall of HPM is slightly better than ours, it is unfair to compare ours directly to those methods using already-known labels. Nevertheless, our method is still better in terms of other performance indices.

Table 1. Performances based on the Corel 5K dataset for some existing methods and our proposed method.

Methods	*P%*	*R%*	N^+
MBRM[17]	24	25	122
SML [18]	23	29	137
JEC [11]	27	32	139
LASSO[11]	24	29	127
TagProp[9]	33	42	160
HPM(with prior knowledge)[19]	33	47	162
HPM(without prior knowledge)[19]	25	28	136
LFA[12]	31	40	151
Proposed Method	**36**	**44**	**165**

The tree structure has hierarchical relationships among the nodes. Our system treats the different labels (simple or complex) in their own ways. With the help of the tree structure, the normal layers and extra branches are weighted differently. Figure 4 shows the performance with and without using the tree, as a trade-off between precision and recall for this Corel 5K dataset.

Fig 5 shows the precision and recall rates when the annotation length changes from 1 to 10 on the Corel 30K dataset. Table 2 compares the performances of the different methods in terms of precision, recall, and number of recalled keywords. The results show that our method always achieves better results on the ESP Game dataset.

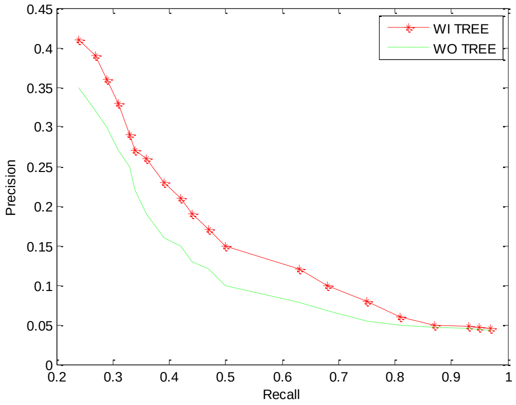


Fig. 4. Precision-recall plots generated by varying the number of keywords assigned to an image with and without using the tree structure.

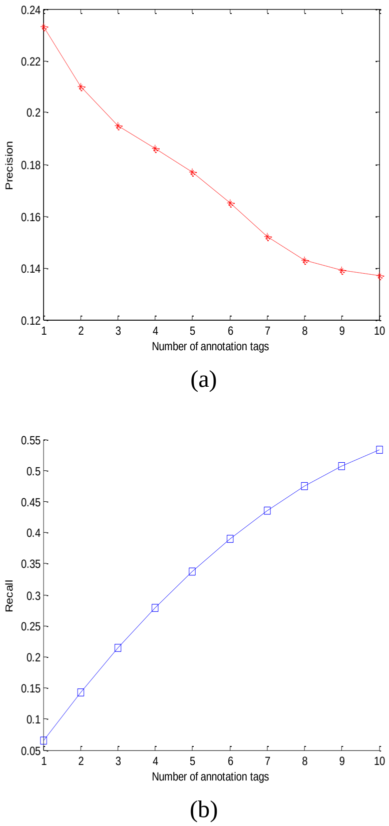


Fig. 5. Performance of our proposed method based on a test set of 1500 tags: (a) Precision, and (b) Recall.

Table 2. Performance comparison on the ESP Game dataset.

Method	JEC [11]	TagProp [9]	AICDM [19]	LFA	Prop. Work
Avg. Prec.	0.22	0.39	0.24	0.35	0.44
Avg. Recall	0.25	0.27	0.26	0.25	0.30
N^+	224	239	231	228	261

5. CONCLUSIONS

This paper presents a new hierarchical model for efficient image annotation, which employs an adaptive learning algorithm to select an optimal feature subset for each label. A tree structure is constructed and a regression model is trained using the tree structure based on our proposed feature-label-selection algorithm. Making use of the tree, the relationships among the labels are considered, which can highly improve the performance of our multi-task learning algorithm. The experiment results have shown that our proposed framework achieves a promising performance, and that it can achieve both efficiency and accuracy in image annotation.

6. REFERENCES

[1] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z Wang. Image retrieval: Ideas, influences, and trends of the new age. ACM Computing Surveys (CSUR), 40(2):5, 2008

[2] Jing Zhang, Yongwei Gao, Shengwei Feng, Yubo Yuan, Chin-Hui Lee. Automatic image region annotationthrough segmentation based visual semantic analysis and discriminative classification. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), (2016) 1956-1960

[3] Xiao-Yuan Jing, Fei Wu, Zhiqiang Li, Ruimin Hu, David Zhang. Multi-Label Dictionary Learning for Image Annotation. IEEE Transactions onImageProcessing, Volume: 25, Issue :6 (2016) 2712-2725

[4] M. Grubinger, T. Mensink, J. Verbeek, C. Schmid. Tagprop: Discriminative Metric Learning In Nearest Neighbor Models for Image Auto-Annotations, in: Proceedings of the International Conference on Computer Vision, (2009) 309 - 316.

[5] Jiajun Wu, Yinan Yu, Chang Huang, Kai Yu. Deep multiple instance learning forimage classification and auto-annotation. in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2015) 3460–3469

[6] K.Barnard, P. Duygulu, D Forsyth, N. de Freitas, D. M. Blei, M. I. Jordanetc, Matching Words and Pictures, Journal of Machine Learning Research 3 (2003) 1107-1135.

[7] F.Monay, D. Gatica-Perez. PLSA-Based Image Auto-Annotation: Constraining the Latent Space, ACM Multi-media (2004) 348-351.

[8] Xinmiao Ding, Bing Li, Weihua Xiong, Wen Guo, Weiming Hu, and Bo Wang. Multi-Instance Multi-Label Learning Combining Hierarchical Context and its Application to Image Annotation.IEEE Transactions on Multimedia, Vol. 18, No. 8, (2016) 1616 - 1627

[9] Fei Wu, Zhuhao Wang, Zhongfei Zhang, Yi Yang, Jiebo Luo, Wenwu Zhu, Yueting Zhuang. Weakly Semi-Supervised Deep Learning for Multi-LabelImage Annotation. IEEE Transactions on Big Data, Volume: 1, Issue : 3 (2015) 109-122

[10] H. Zhang, A. Berg, M. Maire, J. Mailik. SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2006) 2126–2136.

[11] Ameesh Makadia, Vladimir Pavlovic, Sanjiv Kumar. A New Baseline for Image Annotation, in: Proceedings of the European Conference on Computer Vision, (2008) 316-329.

[12] J. Hu, K.M. Lam, G. Qiu. A Hierarchical Algorithm for Image Multi-labeling, in: Proceedings of the IEEE International Conference on Image Processing (ICIP'2010), (2010) 2349 – 2352.

[13] M.Choi, J.Lim, A. Torralba, and A. Willsky, "Exploiting hierarchical context on a large database of object categories," in CVPR,2010

[14] J.Hu, K.M.Lam, An Efficient Two-stage Framework for Image Annotation. Pattern Recognition 46(3): (2013) 936-947.

[15] Seyoung Kim and Eric P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In Proceedings of the 27th International Conference on Machine Learning, (2010) 543-550.

[16] Le, Q. V., Smola, A., Chapelle, O., Teo, C. H., Optimization of Ranking Measures, Journal of Machine Learning Research (2010).

[17] S.L. Feng, R. Manmatha, V. Lavrenko. Multiple Bernoulli Relevance Models for Image and Video Annotation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2004) 1002-1009.

[18] G. Carneiro, A.B. Chan, P.J. Moreno, N. Vasconcelos. Supervised Learning of Semantic Classes For Image Annotation and Retrieval, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (3) (2007) 394-410.

[19] N. Zhou, W. Cheung, G. Qiu, X. Xue. A Hybrid Probabilistic Model for Unified Collaborative and Content based Image Tagging, IEEE Transactions on Pattern Analysis and Machine Intelligence 33 (7) (2011) 1281-1294.