A Max-Margin Riffled Independence Model for Image Tag Ranking

Tian Lan and Greg Mori
School of Computing Science, Simon Fraser University, Canada
{tla58, mori}@cs.sfu.ca

Abstract

We propose Max-Margin Riffled Independence Model (MMRIM), a new method for image tag ranking modeling the structured preferences among tags. The goal is to predict a ranked tag list for a given image, where tags are ordered by their importance or relevance to the image content. Our model integrates the max-margin formalism with riffled independence factorizations proposed in [10], which naturally allows for structured learning and efficient ranking. Experimental results on the SUN Attribute and LabelMe datasets demonstrate the superior performance of the proposed model compared with baseline tag ranking methods. We also apply the predicted rank list of tags to several higher-level computer vision applications in image understanding and retrieval, and demonstrate that MMRIM significantly improves the accuracy of these applications.

1. Introduction

Image understanding is a central problem in computer vision and has been extensively studied over the past decade. A number of recognition algorithms have been proposed to simultaneously recognize multiple objects and attributes from an image [5, 8, 7]. Most of these systems treat all objects and attributes as equally important. However, there is evidence humans do not perceive them as such. Psychologists have shown that we tend to only memorize the overall scene and a few visually important objects and properties after looking at pictures [17]. Consider the image shown in Fig. 1. Although more than a dozen of tags are associated with the image, it is arguable only a few of them are perceptually important to humans. In this paper, our goal is to rank tags according to their importance or relevance to the image content. We demonstrate that *tag ranking* leads towards better image understanding.

Predicting the importance of image content is a challenging problem in computer vision and has been addressed by relatively small amount of work [21, 1]. Spain and Perona [21] take an object-centric stance, predicting the importance of objects in an image. We consider a wider range of image tags including both objects and attributes. Berg et al. [1] consider prediction of importance for objects, scenes and attributes, and study the influence of relations among objects in the same scene on people's perception of importance in images, which is similar to our goals. However, instead of modeling importance prediction as a binary decision problem, we predict image tags with multiple importance levels.

Figure 1: **Image tag ranking.** Not all tags associated with an image are equally important. The goal of this paper is to rank tags according to their importance or relevance to the image content.

Understanding tag importance in images can potentially facilitate a variety of web-based applications that involve image tags. With the increasing popularity of social photo sharing websites like Flickr, tons of images with user-specified tags are available. However, these massive quantities of tags are collected in extremely uncontrolled settings, thus lots of irrelevant tags can be associated with images. This limits the benefits of these tags in potential applications such as visual search and image organization. In computer vision, a variety of methods has been developed to explore the correspondences between tags and images, for automatic image annotation [2, 3, 9, 20, 22]. However, none of these methods considers the relevance of tags to the image content. Our work is inspired by the recent research that explores the ordering of tags associated with an image. Hwang and Grauman [12, 13] observe that human taggers usually name prominent tags first, and gradually expand to irrelevant tags. This implicit cue is used to improve object

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.399

localization and image retrieval. Liu et al. [15] develop an unsupervised tag ranking scheme on Flickr images with associated tag lists. Tag relevance is estimated based on probability density estimation followed by a random walk refinement. In contrast to all of the above mentioned work, our method directly learns the ordering of tags from the training data. So when given a new image, our model can predict an ordered list of tags for the image, where the tags are ranked according to the importance to the image content.

In this paper, we address the problem of tag ranking. Compared to the familiar problem of image ranking, tag ranking outputs a rank list for each image, and the tags are more closely correlated than images. For example, given four tags "bear", "furry", "stripe" and "zebra", knowing "stripe" is preferable to "furry" will indicate that "zebra" is preferable to "bear". These complex and structured relations can be represented by a densely connected graph, where nodes are tags and edges indicate the pairwise interactions between tags. This naturally forms a problem of learning with structured output. Structured output learning has been utilized for document ranking [4] and image ranking [19] for optimizing specific ranking measures. We demonstrate it for tag ranking, a natural structured output problem.

As with many challenging learning problems, learning to rank tags involves intractably large state spaces, e.g. ranking n tags results in $n!$ possible permutations. In order to achieve efficient inference, Huang and Guestrin [10] proposed a generalized notion of probabilistic independence on permutations (called *riffled independence*) for ranking applications. Riffled independence for ranking is similar to shuffling a deck of cards, one ranks two disjoint sets of items independently, then interleaves the ranked items together to form a full ranking. The interleaving stage characterizes riffled independence for rankings and distinguishes it from other probabilistic independence assumptions. In [11], a structure learning algorithm is proposed to automatically discover groups of items that are riffled independently.

We propose a novel max-margin learning framework for training tag ranking models with riffled independence assumptions. We introduce an efficient inference algorithm for predicting the rank list of tags. We call our approach the *Max-Margin Riffled Independence Model (MMRIM)*. To the best of our knowledge, this is the first max-margin learning framework designed for the problem of image tag ranking modeling structured preferences among tags.

2. Riffled Independence for Rankings

We first cover background on the *riffled independence* proposed in [10]. We use a simple example to illustrate this idea. Suppose we have six tags associated with an image: tree (T), furry (F), bear (B), grass (G), stripe (S) and zebra (Z). Our goal is to sort the tags into an ordered list, which

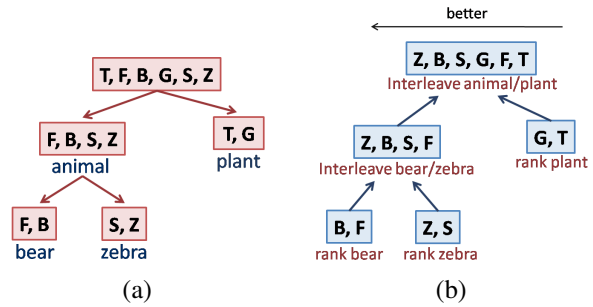


Figure 2: Illustration of the hierarchical riffle independent decomposition algorithm [11]. The algorithm consists of two steps: (a) top-down decomposition and (b) bottom-up ranking and interleaving. "T, F, B, G, S, Z" represents six image tags "tree, furry, bear, grass, stripe and zebra" respectively.

ends up in an intractably large state space of $n!$ (6! in this example) possible rankings for n items. The idea of riffled independence is to decompose the original set of tags into two subsets and rank them independently: {Z, B, S, F} (which represents "animal") and {T, G} (which represents "plant"). The two ranked subsets are then interleaved to form a full ranking: {Z, B, S, G, F, T}. The intuition of riffle independence is that the relative ranks of items in one subset is independent from items in the other subset, e.g. knowing "bear" ranks first gives no information on whether "grass" is preferable to "tree". This endeavor reduces the state space to $k! + (n-k)! + \binom{n}{k}$ if we consider two subsets with k and $n-k$ items respectively.

In order to automatically discover riffled independent groupings of items from the data, Huang and Guestrin [11] proposed the hierarchical riffle independent decomposition algorithm. The algorithm first performs a *top-down decomposition* that recursively partitions the full set of tags into riffled independent groups (see Fig. 2 (a)). We continue to use the same example to illustrate it. One can imagine that "animal" is further decomposed into: "furry bear" ({F, B}) and "zebra with stripes" ({S, Z}). To generate a full ranking, the algorithm then performs a *bottom-up ranking and interleaving* (see Fig. 2 (b)). The algorithm first ranks "furry bear" {B, F} and "zebra with stripes" {Z, S} independently, then interleaves them to generate the ranking for "animal" ($\{Z, B, S, F\}$). Finally, "plant" is ranked as {G, T} and interleaved with the rankings of the first subset to form a full ranking: {Z, B, S, G, F, T}.

We can summarize this example with three important *riffle independence properties*: 1) tags within each subset have strong correlations. For example, knowing "stripe" is preferable to "furry" indicates that "zebra" ranks higher than "bear". 2) The absolute rank of a tag in one subset gives no information about the relative ranks of tags in another subset. For example, knowing bear (B) ranks first among all the six tags does not tell that grass (G) is preferred to tree (T). Based on this intuition, one can rank tags in each subset independently. 3) Tags from different subsets are re-

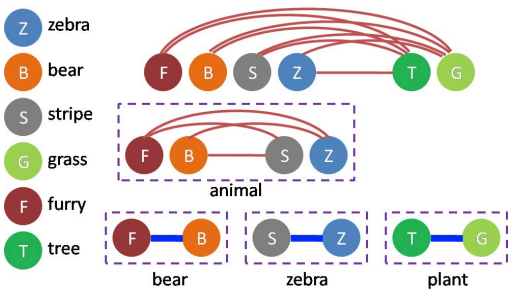


Figure 3: An example of the *riffled independence graph structure*. The graph structure is obtained from the riffled independence tree in Fig. 2 (a). The structure consists of three levels, where tags form into larger groups from bottom up. The bottom level captures the preferences among tags within each leaf set (blue lines), such as whether "tree" is preferable to "grass". The middle and top levels capture the preferences between tags in different groups (red lines), such as whether "furry" in "animal" is preferred over "grass" in "plant", where the preference is not only determined by their tag labels ("furry" versus "grass"), but also the group preferences ("animal" versus "plant"). The full rank list predicted from the tags is shown on the left side.

lated through group-level interactions. This interaction tells whether one group is preferable to another.

3. Max-Margin Riffled Independence Model

In this section, we introduce the Max-Margin Riffled Independence Model (MMRIM). A graphical representation of the model is shown in Fig. 3. We first run the hierarchical riffle independent decomposition algorithm [11] to discover the structures as well as the latent groupings from the image tags. We model two types of preferences (or interactions) between image tags: preferences among tags (which we call *tag preferences*) and preferences among groups (which we call *group preferences*). We build a Riffled Independence Model (RIM) to capture these preferences. Note that learning a RIM is the same as learning a Markov Random Field (MRF), where a Structured SVM is applied. However, inference in RIM is different from MRF (see Sec. 4).

3.1. Model Formulation

We first describe the notation used in this paper. The input to our learning module is a set of $\langle X, Y, R\rangle$ triplets, where X denotes an image, Y denotes the annotations (tags) associated with the image, and R denotes the tag ranks. Suppose there are V tags in an image, we write x_i for the feature vector for tag i. The entire image can be represented as a collection of feature vectors $X = (x_1, x_2, \ldots, x_V)$. The tags associated with the image are represented as $Y = (y_1, y_2, \ldots, y_V)$, where $y_i \in \mathcal{Y}$ is the label of tag i and $\mathcal{Y}$ is the set of all possible tag labels. Given an image X with annotations Y, the output of our model is tag ranks that arrange the annotations into an ordered list. We map the rank list to a vector $R = (r_1, r_2, \ldots, r_V)$, where $r_i \in \mathcal{R}$ denotes the rank for tag i in the current image and $\mathcal{R}$ denotes the set of ranks. The higher the rank, the more this tag is relevant to the image.

We build a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to capture the preferences among tags. The graph is obtained by running the hierarchical riffle independent decomposition algorithm [11]. An example graph structure is shown in Fig. 3. The graph consists of two types of edges: edges connect nodes in the leaf sets (denoted by $\mathcal{E}_a$) and edges connect nodes in the larger subsets (denoted by $\mathcal{E}_b$), where $\mathcal{E} = \{\mathcal{E}_a, \mathcal{E}_b\}$. In $\mathcal{E}_a$, all pairs of tags within the same leaf set are connected (see the blue lines in Fig. 3). In $\mathcal{E}_b$, all pairs of tags from two different groups are connected (see the red lines in Fig. 3).

Besides obtaining the graph structure $\mathcal{G}$, the hierarchical riffle independent decomposition algorithm also discovers the latent groupings. Take Fig. 3 for example, furry (F), bear (B), stripe (S) and zebra (Z) are grouped into "animal", while tree (T) and grass (G) are grouped into "plant". The group labels of the tags associated with an image are represented as $H = (h_1, h_2, \ldots, h_V)$, where $h_i \in \mathcal{H}$ is the group label of tag i and $\mathcal{H}$ is the set of all possible group labels. Intuitively, *group* is a mid-level representation of tags, and denotes a latent coalition of tags.

We define the score of labeling image X with tags Y in the order of R as:

$$\theta^\top \Phi(X, Y, R) = \alpha^\top \phi(X, Y, R) + \beta^\top \psi(X, Y, R) \quad (1)$$

where

$$\alpha^\top \phi(X, Y, R) = \sum_{(i,j)\in\mathcal{E}, r_i > r_j} (\alpha_{y_i} \cdot x_i - \alpha_{y_j} \cdot x_j) \quad (2)$$

$$\beta^\top \psi(X, Y, R) = \sum_{(i,j)\in\mathcal{E}_b, r_i > r_j} (\beta_{h_i} \cdot x_i - \beta_{h_j} \cdot x_j) \quad (3)$$

The potential functions in Eq. 2 and Eq. 3 capture *tag preferences* and *group preferences* respectively. x_i is the feature vector of tag i, here we adopt a simpler approach by setting x_i to the output score of an independently trained classifier. Intuitively, one can see that both Eq. 2 and Eq. 3 represent the summation of weighted differences of the confidence scores between every pair of tags, under the condition that the i-th tag is ranked higher than the j-th tag ($r_i > r_j$). α_{y_i} and β_{h_i} are standard linear models for ranking tag y_i and group h_i respectively. The potential function on groups (Eq. 3) captures the intuition that the preferences of groups (e.g."animal" $\succ$ "plant") suggests the preferences of tags (e.g. "bear" $\succ$ "grass").

3.2. Max-Margin Learning

Assume we are given a collection of training images X^n, annotations Y^n and rankings R^n, we want to train the model parameters θ that tends to correctly predict the tag

ranks R^*. We formulate this as an optimization problem.

$$\min_{\theta,\xi\geq 0} \frac{1}{2}||w||^2 + C\sum_n \xi_n$$
$$\theta^\top \Phi(X^n, Y^n, R^n) - \theta^\top \Phi(X^n, Y^n, R^*) \geq \Delta\left(R^n, R^*\right) - \xi_n, \quad \forall n, \qquad (4)$$

where $\Delta\left(R^n, R^*\right)$ is a loss function measuring the cost incurred by predicting R^* when the ground truth is R^n. In order to tractably solve the loss-augmented inference (often called *finding the most violated constraint*) in structured SVM learning, the loss function needs to be decomposable over the graph structure $\mathcal{G}$. Here we introduce a new loss function that is decomposable over the riffled independence graph structure shown in Fig. 3.

Riffled independence tree loss: The loss function decomposes over the riffle independent graph structures in the following form:

$$\Delta\left(R^n, R\right) = \sum_{(i,j)\in\mathcal{E}_a, r_i^n > r_j^n} \gamma_{(r_i^* > r_j^*)} + \sum_{(i,j)\in\mathcal{E}_b, r_i^n > r_j^n} \gamma_{(r_i^* > r_j^*)} \quad (5)$$

where $\gamma(r_i > r_j)$ is 0 if $r_i > r_j$ and 1 otherwise. The first term shows the penalties on the rankings of the leaf sets, and the second term is the penalties on the rankings of the higher levels of the structure. This form of learning problem is known as structural SVM, and many well-tuned solvers can be applied to solve this problem. Here we use the bundle optimization solver in [6].

4. Riffle Independent Tag Rank Prediction

Given the model parameters θ and tags Y, the inference problem is to find the best rank list R^* for an image I. It requires solving the following optimization problem:

$$\max_{R\in\mathcal{R}} \theta^\top \Phi(X, Y, R) \qquad (6)$$

For inference, we use a bottom-up strategy to generate the full ranking R^*. Following the riffled independence property, the inference consists of two stages: 1) Predict the rankings independently for each leaf set. 2) Interleave groups of ranked tags into larger groups recursively in a stagewise fashion to generate the full ranking. An example of the inference procedure is shown in Fig. 2 (b). In order to better explain the inference procedure, we first split Eq. 2 into the following two potential functions:

$$\alpha^\top \phi(X, Y, R, \mathcal{E}_a) = \sum_{(i,j)\in\mathcal{E}_a, r_i > r_j} (\alpha_{y_i} \cdot x_i - \alpha_{y_j} \cdot x_j) \quad (7)$$

$$\alpha^\top \phi(X, Y, R, \mathcal{E}_b) = \sum_{(i,j)\in\mathcal{E}_b, r_i > r_j} (\alpha_{y_i} \cdot x_i - \alpha_{y_j} \cdot x_j) \quad (8)$$

where Eq. 7 and Eq. 8 respectively capture the preferences among tags within the same leaf set, and from different groups. $\mathcal{E}_a$ and $\mathcal{E}_b$ correspond to the blue links and red links in Fig. 3 respectively.

Then we separate the inference problem into the following two subsequent optimization problems:

$$\max_{R_a} \alpha^\top \phi(X, Y, R_a, \mathcal{E}_a) \qquad (9)$$

$$\max_{R} \alpha^\top \phi(X, Y, R, \mathcal{E}_b) + \beta^\top \psi(X, Y, R) \qquad (10)$$

Eq. 9 represents the first stage of inference, where tags in each leaf set are ranked independently, and R_a denotes the local rank lists of all leaf sets. Suppose that there are K leaf sets, then $R_a = \{R_a^1, \ldots, R_a^K\}$, where R_a^k denotes then local rank list of the k-th leaf set. Similarly, $\mathcal{E}_a = \{\mathcal{E}_a^1, \ldots, \mathcal{E}_a^K\}$, where $\mathcal{E}_a^k$ denotes the edges of the k-th leaf set, and here we assume the tags in a leaf set are fully connected. Since we infer the local rank list for each leaf set independently, Eq. 9 can be written as: $\max_{R_a^k} \alpha^\top \phi(X, Y, R_a^k, \mathcal{E}_a^k)$, for $k = 1$ to K. In the example of Fig. 2 (b), this stage corresponds to ranking tags "B, F", "Z, S" and "G, T" independently in the three leaf sets. The leaf sets have group labels "bear", "zebra" and "plant respectively.

Eq. 10 represents the interleaving stage of inference. After the first stage, tags within each leaf set are ordered into local rank lists. So the input to the second stage is the local rank lists R_a, and the output is the full rank list R. In this stage, groups of tags are recursively interleaved to form larger groups in a stagewise fashion. We also refer to the example of Fig. 2 (b): the tags in the groups "bear" and "zebra" are interleaved to form the larger group "animal", then the tags in 'animal" are interleaved with tags in "plant" to form the full rank list.

We now explain how we solve the two-stage inference problem. In the first stage, the problem of finding optimal local rank list R_a^k for the k-th leaf set can be formulated into an integer linear programming (ILP) problem. We introduce variables z_{ijst} for all edges $(i,j) \in \mathcal{E}_a^k$: $z_{ijst} = 1$ if the i-th and j-th tags are assigned with ranks s and t respectively, and zero otherwise. We use ϕ_{ij} to represent the potential function in Eq. 2 that involve the edge $(i,j) \in \mathcal{E}_a^k$. The ILP can be written as:

$$\max_z \sum_{(i,j)\in\mathcal{E}_a^k} \sum_{s=1}^{M}\sum_{t=1}^{M} \phi_{ij} z_{ijst} \qquad (11a)$$

$$\text{s.t. } \forall i \in \mathcal{V}_a^k \sum_{s=1}^{M} z_{is} = 1,\ z_{is} \in \{0,1\} \qquad (11b)$$

$$\forall (i,j) \in \mathcal{E}_a^k\ z_{ijst} \leq z_{is},\ z_{ijst} \leq z_{jt} \qquad (11c)$$

$$\forall (i,j) \in \mathcal{E}_a^k\ z_{ijst} \geq z_{is} + z_{jt} - 1 \qquad (11d)$$

$$\forall (i,j) \in \mathcal{E}_a^k\ z_{ijst} \geq 0,\ \ \forall s > t\ z_{ijst} = 0 \qquad (11e)$$

where M is the total number of ranks. The constraints in Eq. 11b guarantee that every tag is assigned to only one

rank. The constraints in Eq. 11c-11e correspond to the linearization of the quadratic constraint $z_{ijst} = z_{is} \cdot z_{jt}$. We use the GNU Linear Programming Kit (GLPK) to solve the ILP. Note that here we do not enforce the mutual exclusivity constraints which ensure tags i and j map to different ranks. This is because in terms of image tag ranking, it is common that two tags are equally important to an image according to human's perception.

After we obtain the local rank lists R_a for each leaf set, we can solve the optimization problem in Eq. 10, where we interleave the groups of ranked tags into larger groups in a stagewise fashion. In Fig. 2, this corresponds to interleaving the rank list of "bear" and "zebra", and finally interleaving "animal" and "plant" to form the full rank list. For each "interleaving" operation, we simply enumerate all possible rankings and find the optimal one. For example, suppose that two ranked groups have k and $n-k$ tags respectively, and M is the total number of ranks, then the "interleaving" operation involves enumerating over $\binom{M}{k} + \binom{M}{n-k}$ possible states.

The inference is particularly efficient for tag ranking, which takes around 0.1 sec per image with 15 tags in MATLAB on a 2.8GHZ CPU 8GB RAM PC. In contrast, ranking tags over a fully connected graph takes around 5 min per image under the same settings.

5. Applications of Tag Ranking

We have described an algorithm for learning a model (MMRIM) for tag ranking. The most direct application is to use the model to rank the randomly permuted tags associated with each testing image, which we call *tag ranking*. However, not all images in the real world are annotated. Thus we also use the model to simultaneously predict the tag list and rank the tags for an unannotated image. We call this task *image auto-annotation*. Furthermore, we also demonstrate that the predicted rank list of tags help improve higher-level computer vision tasks, such as image retrieval and tag-based image search. We define the four different applications as follows:

- Tag Ranking: Given an image and its associated tag list, our goal is to rank the tags according to their importance or relevance to the image content.

- Image Auto-Annotation: Automatically predict the tag list for a given image without any annotations. Compared to the existing work on image auto-annotation, our method further provides a ranked tag list.

- Image Retrieval: Given a novel query image, we first use our model to predict an ordered list of tags for it. Then we use the χ^2 kernels described in [12] to compute the similarities between the features of the query image and all images in the database. Images are ranked based on the similarities. Now we describe the details of extracting features from an image. We construct the feature vector by concatenating the image features and the rank features. The details of image features will be introduced in the next section. The rank features are computed in the following:

$$\eta = \frac{1}{\sum_i r_i} [r_1, \ldots, r_V] \tag{12}$$

  where r_i denotes the rank for tag i in the query image, and V is the total number of tags associated with the query image. We expect the top ranked images share similar importance levels of objects and attributes with the query image rather than only similar categories.

- Relative Tag-based Image Search: Existing keyword based image search methods restrict queries to categorical labels (e.g. "ocean, mountain, tree, ship"), and thus fail to capture the semantic relationships between tags. We propose to use *relative tags* as queries. As opposed to the presence of tags, relative tags indicate the strength of a tag w.r.t. the other tags, e.g. "more ocean than mountain, more ship than tree". Relative tags are more informative and descriptive compared to the traditional keywords. Given relative queries, e.g. "more ocean than mountain, more ship than tree", we will be able to depict a picture in mind with detailed importance levels of tags, e.g. "a ship in an ocean, backed by trees and mountains".

6. Experiments

We test our model on two datasets: SUN Attribute [16] and LabelMe [18]. The SUN Attribute dataset contains 14,340 images and 102 scene attributes spanning from materials, surface properties, lighting, functions and affordances, to spatial envelope properties. The LabelMe dataset consists of mostly office and street scenes of 3825 images with an average of 23 tags per image. Different from SUN Attribute where tags correspond to attributes, the tags in LabelMe are objects. For both datasets, we evaluate our method on the four different scenarios outlined above.

Implementation details. We first show how to construct the ground truth tag ranks. Tags for both datasets were collected in previous work. In SUN Attribute, the number of positive labels (votes) each attribute received from AMT workers is provided [16]. The number of votes indicates how confident an attribute presents in an image, and we use it as the ground truth tag ranks. In this way, each tag in an image is labeled with one of four importance levels: Most Relevant (3 votes), Relevant (2 votes), Less Relevant (1 vote) and Irrelevant (0 votes). We construct the tag list for an image by using the attributes receive more than zero votes. In order to mimic real annotations from

the Internet, we further add three noisy tags to the tag list of each image. The noisy tags are randomly sampled from the tags that receive zero votes. For LabelMe, we use the 3825 images compiled in [13], where the tag rank list associated with each image are also provided. We quantize the tag rank list associated with each image into three levels: Most Relevant, Relevant and Less Relevant. We also randomly sample three irrelevant tags and add them to the tag list of each image. Thus tag lists for both datasets are labeled with four importance levels, from Irrelevant to Most Relevant ($|\mathcal{R}| = 4$).

Following [16], we extract four types of image features: Gist, HOG 2×2, self-similarity, and geometric context color histogram for the SUN Attribute Dataset. We train an SVM classifier for each attribute. Our SVM classifiers use a combination of kernels generated from the four types of features (see [23] for feature and kernel details). For LabelMe, we extract three types of image features: Gist, color histograms, and bag-of-words histograms following [12]. We train SVM classifiers with χ^2 kernels on these features for each tag (see [12] for feature and kernel details). We use the output score of an independently trained classifier as the tag's feature vector of our model (x_i in Eq. 2).

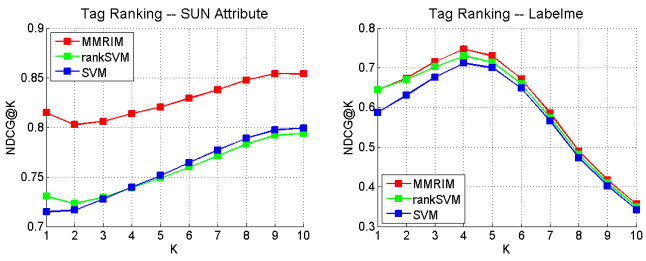
We use the Normalized Discounted Cumulative Gains (NDCG) [4] to measure the performance of the tag ranking approaches. It is defined as: $NDCG@K = \frac{1}{Z}\sum_{i=1}^{K}\frac{2^{rel(i)}-1}{\log(1+i)}$. Where K is called truncation level, Z is the normalization constant to make sure the optimal ranking gets an NDCG score of 1, and $rel(i)$ is the relevance of the i-th ranked instance.

Baselines. In order to comprehensively evaluate the performance of the proposed model, we define the following baseline methods to compare with. For the first baseline (called "SVM"), we ignore the tag relations and directly use the classifier scores for ranking. To obtain the tag rank list for an image, we directly sort the output scores of SVM classifiers for each tag. For the second baseline (called "rankSVM"), we use the rankSVM [14] solver for learning. To obtain the rank list for an image, we sort the output scores obtained from the individual tag rankers. The potential function for an individual tag ranker is: $\alpha_{y_i} \cdot x_i$, where x_i is the feature vector of the i-th tag and y_i is the tag category label.

In the following, we show the experimental results of our method for each of the four scenarios outlined above.

6.1. Tag Ranking

We compare the NDCG scores of our method and the baselines in Fig. 4. Here we use four relevance levels for computing NDCG: Most relevant, Relevant, Less relevant and irrelevant. We can see that our method significantly outperforms the baselines on both datasets. At the truncation level of 4 (NDCG@4), we see our method yields around



(a) (b)

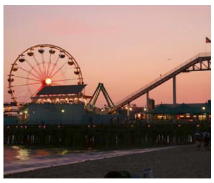
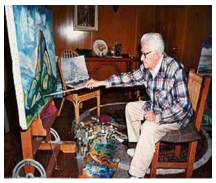
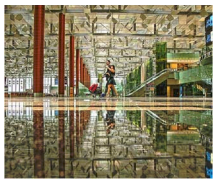
Figure 4: Comparison of tag ranking results of different methods on SUN Attribute dataset and Labelme dataset respectively.

7.5% improvement over both rankSVM and SVM on SUN Attribute; a 1.7% improvement over rankSVM and 3.5% improvement over SVM on LabelMe. In comparison, the improvements on LabelMe are relatively smaller than SUN Attribute. This is because the ground truth ranks on two datasets are provided according to different strategies. The ground truth ranks in SUN Attribute are obtained by voting the presence of the attributes, the process is controlled and important attributes normally receive more votes. But for LabelMe, the tag ranks provided in [13] are obtained in a less controlled setting, by the order in which tags are added to the image. The order usually reflects the importance of tags but is subjective to the AMT workers. Some visualizations of the tag ranking results for MMRIM and rankSVM are shown in Fig. 5.

6.2. Image Auto-Annotation

In the first experiment, we assume each image is associated with a list of annotations. Now we demonstrate that our method is also capable of predicting an ordered list of tags for an unannotated image. In this scenario, rather than reordering the given annotations during testing, we assume each image is associated with the whole tag vocabulary, and predict an ordered list of the whole vocabulary for each testing image. We expect to rank the most relevant tags to the top and irrelevant tags to the bottom of the rank list. In this case, inference is carried out over the whole vocabulary for each image. In practice, in order to increase the efficiency, we reduce the search space ($\mathcal{R}$ in Eq. 6) by the following strategy. For each testing image, we calculate the χ^2 similarities between the testing image and all training images based on their image features, and use the annotations from the top k neighbors to construct the search space $\tilde{\mathcal{R}}$ for the testing image. Here we set k to 5. This procedure greatly reduces the running time of inference.

Fig. 6 (a),(b) shows the comparison of our method and the baselines in terms of Precision at K. Our method outperforms all baselines, and its Prec@4 score is 6.0% and 1.0% better than the second best method (SVM) on SUN Attribute and LabelMe respectively. In order to compare with [12], we also report the auto-annotation accuracy using F1 score on LabelMe with the same experimental settings.

high

low

Ground Truth	MMRIM	rankSVM
man-made	man-made	open area
driving	driving	man-made
vocation	open area	natural light
natural light	vocation	driving
electric lighting	natural light	metal
open area	electric lighting	pavement
metal	metal	electric lighting
using tools	using tools	congregating
pavement	pavement	vocation

Ground Truth	MMRIM	rankSVM
cloth	cloth	enclosed area
enclosed area	enclosed area	no horizon
wood	no horizon	cloth
electric lighting	cluttered space	cluttered space
no horizon	working	working
glossy	electric lighting	electric lighting
cluttered space	paper	wood
working	aged/worn	metal
paper	metal	aged/worn

Ground Truth	MMRIM	rankSVM
man-made	man-made	enclosed area
no horizon	enclosed area	man-made
stressful	no horizon	no horizon
cluttered space	stressful	metal
metal	metal	glass
enclosed area	cluttered space	stressful
gaming	gaming	cluttered space
glass	glass	building
building	building	gaming

Figure 5: Examples of tag ranking results using our method and rankSVM on SUN Attribute. The tags are ordered according to the relevance to the image content. We use red, orange and blue to denote three ground truth ranking levels: most relevant, relevant and less relevant, respectively. Due to space constraint, we only visualize the top 9 tags for each image.

Method	F1 score
SVM	0.4621
Rank SVM	0.4352
Our method	**0.5162**
best result in [12]	0.4585

Table 1: Comparison of image auto-annotation accuracies (F1 score) on LabelMe dataset.

We can see that our method is significantly better than the best result reported in [12].

6.3. Image Retrieval

We use all test images as queries and all training images as the database. The ground truth ranking is obtained by sorting the images in the database according to the agreement between the ground truth rank features of the database image and the query image. In SUN Attribute, we compute the agreement as: $rel(i) = \frac{\langle \eta(i), \eta(q) \rangle}{\|\eta(i)\| \|\eta(q)\|}$, where $rel(i)$ represents the relevance of the i-th ranked instance and η is the rank feature (Eq. 12). In LabelMe, we use the strategy introduced in [12] (called "tag rank similarity") to compute the ground truth ranking.

Fig. 6 (c),(d) plots the comparison of NDCG scores of our method and the baselines. Again, our method outperforms the baselines noticeably on both datasets. In order to show that the rank features η in Eq. 12 improve image retrieval, we develop a baseline "visual" by only using image features to compute the χ^2 similarities [1]. Our approach clearly improves over the baseline "visual". SVM and rankSVM are similar to "visual" on SUN Attribute, and slightly better than "visual" on LabelMe. This is likely due to the inaccuracies of rank features predicted by SVM and rankSVM.

[1]For image retrieval, we use the features provided by the authors of [12] and the baseline "visua" defined in this paper is the same as the baseline "visual-only" defined in [12]. In discussions with the authors of [12] we were unable to clarify the source of the difference in results.

6.4. Relative Tag based Image Search

A relative tag query consists of M pairs of tags with preferences, e.g. {"ocean $\succ$ mountain", "natural $\succ$ man-made"}. We consider queries with double and triple pairs of tags, and we generate the query set by randomly sampling from the tags in the training set. In the end, we obtain 200 queries for each query type.

Fig. 6 (e)-(h) shows the comparison of our method and the baselines in terms of NDCG scores. From the figure, it is clear that MMRIM is better than the other methods for both types of queries, at all values of K. At a truncation level of 40 (NDCG@40) for double and triple pairs of queries, MMRIM is respectively, 2.1% and 2.2% better than SVM, the second best method in SUN Attribute, and 5.1% and 3.1% better than rankSVM, the second best method in LabelMe.

7. Conclusion

We have presented the Max-Margin Riffled Independence Model (MMRIM) that integrates the max-margin criterion and riffled independence partitions within the same framework for image tag ranking. Furthermore, our approach models the correlations between different tags leading to improved tag ranking performance. Besides tag ranking, we also apply our model to three higher-level computer vision applications: image auto-annotation, image retrieval and relative tag based image search. Our experimental results on two benchmark datasets demonstrate that our method makes consistent improvements over the baselines.

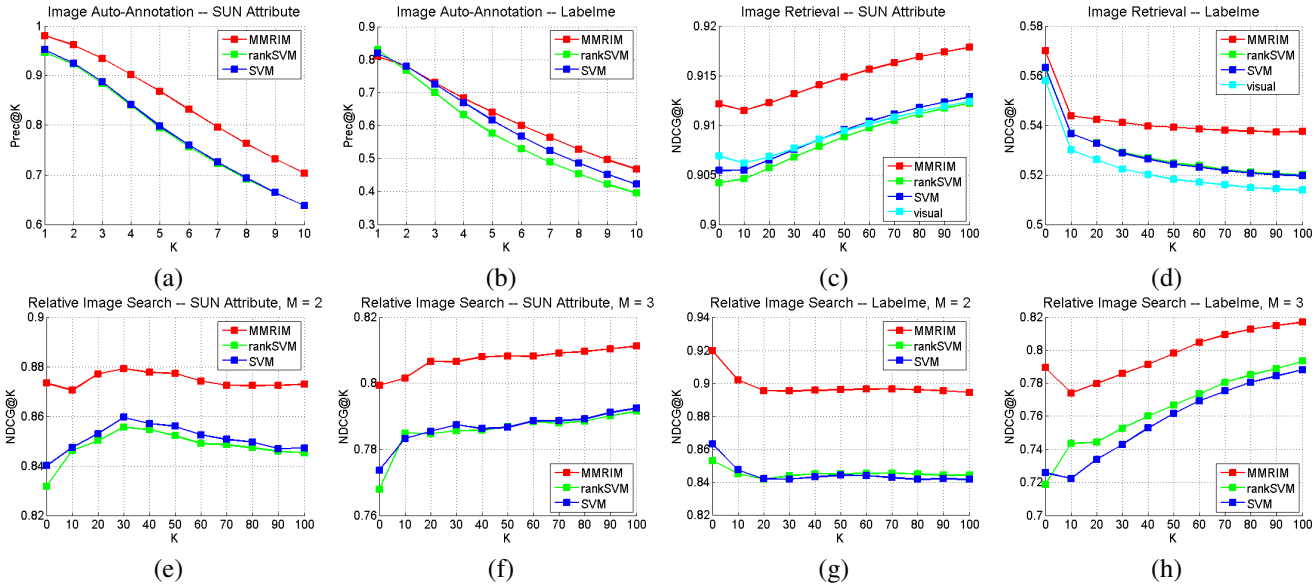


Figure 6: Performance of different methods in higher-level computer vision applications including: image auto-annotation ((a),(b)), image retrieval ((c),(d)), relative image search ((e)-(h)).

References

[1] A.C.Berg, T. Berg, H. D. III, J. Dodge, A. Goyal, X. Han, A. Mensch, M. Mitchell, A. Sood, K. Stratos, and K. Yamaguchi. Understanding and predicting importance in images. In *CVPR*, 2012.

[2] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. M. Blei, and M. I. Jordan. Matching words and pictures. *JMLR*, 2003.

[3] T. L. Berg, A. C. Berg, J. Edwards, and D. Forsyth. Who's in the picture. In *NIPS*. 2004.

[4] O. Chapelle, Q. Le, and A. Smola. Large margin optimization of ranking measures. In *NIPS Workshop on Learning to Rank*, 2007.

[5] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. In *ICCV*, 2009.

[6] T.-M.-T. Do and T. Artieres. Large margin training for hidden markov models with partially observed states. In *ICML*, 2009.

[7] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, 2009.

[8] C. Galleguillos, A. Rabinovich, and S. Belongie. Object categorization using co-occurrence, location and appearance. In *CVPR*, 2008.

[9] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *ICCV*, 2009.

[10] J. Huang and C. Guestrin. Riffled independence for ranked data. In *NIPS*, 2009.

[11] J. Huang and C. Guestrin. Learning hierarchical riffle independent groupings from rankings. In *ICML*, 2010.

[12] S. J. Hwang and K. Grauman. Accounting for the relative importance of objects in image retrieval. In *BMVC*, 2010.

[13] S. J. Hwang and K. Grauman. Reading between the lines: Object localization using implicit cues from image tags. In *CVPR*, 2010.

[14] T. Joachims. Training linear SVMs in linear time. In *SIGKDD*, 2006.

[15] D. Liu, X.-S. Hua, L. Yang, M. Wang, and H.-J. Zhang. Tag ranking. In *WWW*, 2009.

[16] G. Patterson and J. Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *CVPR*, 2012.

[17] R. Rensink, J. ORegan, and J. Clark. To see or not to see: the need for attention to perceive changes in scenes. *Psychol. Sci.*, 1997.

[18] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. LabelMe: A database and web-based tool for image annotation. *IJCV*, 2008.

[19] B. Siddiquie, R. S. Feris, and L. S. Davis. Image ranking and retrieval based on multi-attribute queries. In *ICCV*, 2011.

[20] R. Socher and L. Fei-Fei. Connecting modalities: Semi-supervised segmentation and annotation of images using unaligned text corpora. In *CVPR*, 2010.

[21] M. Spain and P. Perona. Measuring and predicting object importance. *IJCV*, 2010.

[22] Y. Wang and G. Mori. A discriminative latent model of image region and object tag correspondence. In *NIPS*. 2010.

[23] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.