

Neural ranking for automatic image annotation

Weifeng Zhang^{1,2,3} · Hua Hu¹ · Haiyang Hu¹

Received: 22 August 2017 / Revised: 28 March 2018 / Accepted: 4 April 2018 /
Published online: 25 April 2018
© Springer Science+Business Media, LLC, part of Springer Nature 2018

Abstract Automatic image annotation aims to predict labels for images according to their semantic contents and has become a research focus in computer vision, as it helps people to edit, retrieve and understand large image collections. In the last decades, researchers have proposed many approaches to solve this task and achieved remarkable performance on several standard image datasets. In this paper, we propose a novel learning to rank approach to address image auto-annotation problem. Unlike typical learning to rank algorithms for image auto-annotation which directly rank annotations for image, our approach consists of two phases. In the first phase, neural ranking models are trained to rank image's semantic neighbors. Then nearest-neighbor based models propagate annotations from these semantic neighbors to the image. Thus our approach integrates learning to rank algorithms and nearest-neighbor based models, including TagProp and 2PKNN, and inherits their advantages. Experimental results show that our method achieves better or comparable performance compared with the state-of-the-art methods on four challenging benchmarks including Corel5K, ESP Games, IAPR TC-12 and NUS-WIDE.

Keywords Automatic image annotation · Learning to rank · Neural networks · Nearest neighbor

✉ Hua Hu
huhua@hdu.edu.cn

✉ Haiyang Hu
haiyanghu@hdu.edu.cn

Weifeng Zhang
zwf.zhang@gmail.com

¹ School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, China

² Zhejiang Future Technology Institute, Jiaxing, China

³ Science and Technology on Communication Information Security Control Laboratory, Jiangnan Electronic Communication Institute, Jiaxing, China

1 Introduction

With the booming of the number of digital images, effectively and efficiently searching for these images has become an urgent and hot research issue in the multimedia community. Users used to input several key words to the search engine and the search engine returns a number of images who have same or similar annotation words. The search results are largely depend on these annotations. However, annotating images manually requires time and effort, and it is difficult for people to annotate all relevant words for each image. Thus automatic image annotation emerged. Given an input image, the goal of image auto-annotation is to predict a few relevant words to the image which can mainly reflect its semantic contents. Assigning richer, more relevant words to images can help web image search engines improve their fast indexing and retrieval ability [21].

In the past decades, many automatic image annotation methods have been proposed. Most of them focus on mining the image-tag relation [56], image-to-image relation [30, 38], tag-to-tag relation [33] individually or simultaneously [42]. And nearest-neighbor based approaches which predict annotations for testing images by evaluating their relation to training images, have primarily dominated image auto-annotation domain for several years. These approaches can benefit from metric learning techniques [26, 57]. Although these nearest-neighbor based approaches model the image-tag and image-to-image relation, they ignore the tag-to-tag relation which is helpful to improve image annotation performance. On the other hand, learning to rank models which are mainly applied in information retrieval have been adopted to address image auto-annotation [22, 41, 61]. The work in [61] proposed WSABIE to jointly embed images and annotations in a low dimension space and trained WSABIE by learning to rank with the WARP loss and it yielded low memory usage and fast computation time. Li et al. [41] proposed a semi-supervised learning framework called MLRank to simultaneously explore the visual similarity between images and semantic relevance among tags. Inspired by deep learning, Gong et al. [22] designed and trained a deep convolutional ranking network to achieve end-to-end image auto-annotation. However, these ranking models can not achieve better results than traditional nearest-neighbor based models.

In this paper, we design a simple but effective approach to address image auto-annotation problem. Our approach consists of two phases. In the first phase, the tag-to-tag relation is used to train neural ranking models for ranking image's semantic neighbors. Then nearest-neighbor based models propagate annotations from these semantic neighbors to the image. Thus our approach integrates learning to rank algorithms [43] and nearest-neighbor based models and inherits their advantages. Specially, We propose two neural ranking models including pair-wise neural ranking model and list-wise neural ranking model. The pair-wise neural ranking model is achieved by training neural networks using cross-entropy loss function while the list-wise neural ranking model is trained with our proposed visual and semantic neighbors ranking loss function. Both of these loss functions evaluate the similarity error or ranking error based on the images' tag-to-tag relation. We argue that neural ranking models can help images find more semantically similar neighbors in their neighbor list. These ranking models are then integrated with several state-of-the-art nearest-neighbor like models including TagProp [26] and 2PKNN [57]. Our approach is similar to the work in [22, 41, 61]. In contrast to these work which design ranking models to directly rank tags for testing images, our approach trains neural ranking models to rank training images for testing images and tags are then propagated from these ranked training images. We will also detail how to train such neural ranking networks. To evaluate our approach, we conduct several experiments on four image datasets, including Corel 5k, IAPR TC-12, ESP Game

and NUS-WIDE and report comparable or better results, compared with the state-of-the-art methods. Our main contributions are as follows.

- We proposed a visual and semantic neighbors ranking loss function which can minimize the divergence between visual and semantic neighbors ranking.
- We designed and trained both pair-wise neural ranking models and list-wise neural ranking models. Unlike existing learning to rank models for image annotation which directly rank tags for image, our neural ranking models aim to help images find more semantically similar neighbors.
- We integrated our neural ranking models with several nearest-neighbor based models to predict image's annotations on four benchmark datasets and achieved better results than the state-of-the-art models.

The rest of this paper is organized as follows: We first review the development course and latest progress both in automatic image annotation and learning to rank domains in Section 2. Then in Section 3, we briefly introduce our method. Then we present our proposed neural ranking models and visual and semantic neighbors ranking loss function and detailed the network architecture and implementation details in Section 4 followed by demonstrating the nearest-neighbor based label propagation models in Section 5. The experimental setup and results will be given in Section 6. And we conclude our work in the end.

2 Related work

As our work falls both in the areas of automatic image annotation and learning to rank, we review the development course and latest progress both in these two areas.

2.1 Automatic image annotation

Automatic image annotation is a multi-label classification task while most existing works focus on single-label classification where each image is classified into only one category [13, 39, 48, 62]. From this view, image auto-annotation is a much more challenging problem and has become a hot research topic. In last decades, a number of techniques [3, 9, 15, 25, 26, 30, 38, 45, 53, 60, 61, 65] have been proposed and evaluated on standard datasets. In early works in this field, image auto-annotation was considered as special machine translation problem, which tried to establish a relationship between images and annotations. Cross Media Relevance Models(CMRM) [30], Continuous Relevance Model(CRM)[38], and Multiple Bernoulli Relevance Model(MBRM) [15] assume different, non-parametric density representations of the joint word-image space. They approximate the probability of observing a set of blobs and annotations in a given image. Recently, non-parametric nearest neighbor like models [26, 40, 45, 57] have illustrated marked success mainly because patterns in the data can be adapted by the high capacity of these models. These models have two bases. The first is how to design visual features to represent images. These features should be able to reflect the semantic content of the images. Thus low-level features including color, texture *etc.* and mid-level features, such as GIST [50], SIFT [44], which show their power in many computer vision problems, have been tried. The second basis is finding the visual neighbors which are also semantic neighbors. Furthermore, these models can be enhanced by using metric learning to find a more reasonable distance metric for the image features [26, 57, 61]. However, designing and selecting hand-crafted features are usually

difficult. These models suffer from *semantic gap* mainly because visual features can not completely abstract the semantic content of the images.

Several recent embedding methods [23, 24] based on Canonical Correlation Analysis (CCA) [27] have been proposed to bridge the semantic gap by finding linear projections to maximize the correlation between visual features and textual features. Kernel CCA [27], as an extension of CCA, has also been tried, in which visual features and textual features are nonlinearly projected into embedding space [2, 56]. However, CCA or KCCA based embedding methods are hard to scale to large datasets because of the high computational cost to compute the eigenvalues.

Very recently, researchers adopt deep learning algorithms, which have shown its great power in single-label image classification [37] to automatically learning image features extraction and showed significant performance gain in large image datasets [22, 32, 35, 37]. And deep learning has also been adopted to solve image auto-annotation task. Kiros et al. [35] use convolutional neural network (CNN) [37] to build a hierarchical model for learning image representations from the pixel level and then added an autoencoder which has only one hidden layer to learn binary codes for annotations. Literature [56] proposes a method which directly used a pre-trained network to extract visual features and experimental results show the CCA based model gained remarkable benefit from these deep learning features. Furthermore, literature [59] proposes a CNN-RNN framework to learn a joint image-label embedding to exploit the label dependency and image-label relevance. However, these models still can not achieve even comparable results with the models combing CCA or KCCA embedding and nearest-neighbor based label propagation models.

Different from embedding methods, our proposed learning to rank models aim to rank semantically similar training images for a testing image. These models are based on neural networks and can achieve pair-wise or list-wise ranking by using different network architectures and loss functions. These networks can be trained using Stochastic Gradient Descent (SGD) or Adam [34] and easily scale to large scale amount of data. We also use pre-trained CNN to extract visual features. Inspired by [2, 56], we then applied our neural ranking models to TagProp and 2PKNN to finally predict tags for testing images. Thus our approach inherits the advantages of learning to rank and nearest-neighbor based models.

2.2 Learning to rank

Learning to rank models, mainly applied in information retrieval, adopt machine learning algorithms to rank the retrieved documents to response the user's query based on their relevance. For the importance of the ranking problem especially in the field of information retrieval, numerous learning to rank models have been proposed. These models can be categorized into three classes based on their training objectives [43]. The first class is the point-wise model which is trained to estimate the relevance of query-document pair. Linear Regression [63] and Random Forests [4] are typical point-wise learning to rank models. RankSVM [55] and RankNet [5] falls into the second class - pair-wise model, which learns to predict relevance information in the form of preferences between pairs of documents with respect to the same query. And the third class is list-wise model which directly optimizes the entire retrieved list by minimizing ranking loss, such as ListNet[8]. Recently, LambdaRank and LambdaMART are proposed to combine pair-wise with list-wise approaches [6]. Furthermore, active learning methods [64] and semi-supervised learning methods [7, 66, 67] have been employed to learning to rank models. These methods can take advantage of unlabeled training instances and other privileged information to boost the performance of information retrieval system.

Learning to rank has also been used to address automatic image annotation problem in recent years. WSABIE [61] trains an ranking model with the WARP loss and it yields low memory usage and fast computation time. Gong et al. [22] respectively train convolutional ranking networks using three loss functions - Softmax [26], Pairwise Ranking [31], and WARP [61]. The trained CNNs then directly rank the tags for testing images. Training a deep CNN needs not only a large labeled image dataset, but also expensive computational time. And, to date, these SGD trained models are frustratingly difficult to beat CCA based models [36]. Different from them, we designed and trained both pair-wise neural ranking model and list-wise neural ranking model, aiming to rank semantically similar training images for a given testing image. Then nearest-neighbor based models are used to predict testing image's annotations by propagating tags from its neighbors.

3 Model

In this section, we detail the proposed image auto-annotation approach. Figure 1 illustrates the framework of our method which mainly comprises two components: neural ranking and nearest-neighbor based label propagation. The key idea of our method is to learn neural ranking networks to bridge the semantic gap. Specially, We propose two neural ranking models including pair-wise neural ranking model and list-wise neural ranking model. The pair-wise neural ranking model is achieved by training neural networks using cross-entropy loss function while the list-wise neural ranking model is trained with our proposed visual and semantic neighbors ranking loss function which minimizes the divergence between visual and semantic neighbors ranking. Both of these loss functions evaluate the similarity error or ranking error based on the images tag-to-tag relation. In training phase, training instances (pair-wise or list-wise) are used to train neural ranking networks with these tag-to-tag relation as supervision information. In testing phase, these trained neural ranking networks rank testing image's semantic neighbors. Then nearest-neighbor based models, such as TagProp [26], 2PKNN [57], et al., propagate tags from testing image's nearest neighbors appearing in its neighbor ranking list. In our method, image's visual features are extracted using pre-trained convolutional neural networks (CNN) [28] for its success in many computer vision tasks. Semantic features are coded with word embedding and fisher vector. In detail, associated annotations are converted into real-valued vectors by word embedding (Word2Vec) [47] and then word2vec embedding of each tag will be pooled using Fisher vector [36, 52] which provides state-of-the-art results on many vision and natural

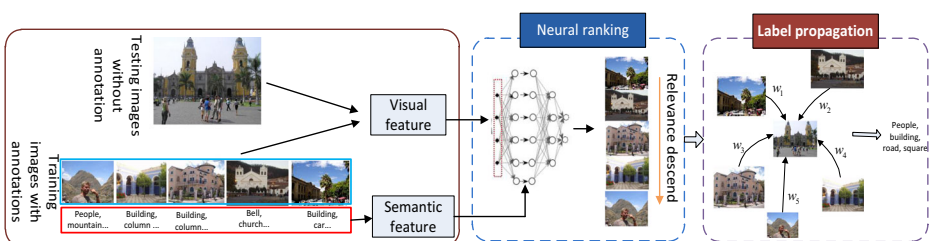


Fig. 1 Framework of our proposed model. Testing image's semantic neighbors are ranked by neural ranking models trained with visual and semantic relation information. Then nearest-neighbor model propagates tags from testing image's nearest neighbors appearing in its neighbor ranking list

language processing tasks [10, 51]. We use the cosine similarity of fisher vectors to rank semantic neighbors.

4 Neural ranking models

In this section, we first introduce our proposed neural ranking networks. Then we detail the architectures of these networks and explain how we train these neural ranking networks.

4.1 Ranking architectures

Pair-wise model Given a query image q and a pair of images $\langle d_1, d_2 \rangle$, such that $d_1 \succ_q d_2$ (d_1 is more semantically similar to q), this pair-wise ranking model (see Fig. 2a) learns a ranking function $\mathcal{R}(q, d_1, d_2; \theta)$ that predicts the probability of image d_1 to be ranked higher than d_2 in the neighbor ranking list of image q . θ is the parameter of the model. To train this model, training instances, each having five elements: $(q, d_1, d_2, s_{q,d_1}^S, s_{q,d_2}^S)$, are sampled which will be introduced in Section 4.3. s_{q,d_i}^S denotes the semantical similarity of image q and d_i , which can be calculated using the cosine similarity between fisher vectors of their annotations:

$$s_{q,d_i}^S = \cos(FV_q, FV_{d_i}) = \frac{FV_q^T \cdot FV_{d_i}}{\|FV_q\| \cdot \|FV_{d_i}\|} \tag{1}$$

where FV_q and FV_{d_i} is the fisher vector representation of annotations of image q and d_i respectively, which will be detailed in Section 4.3. Given a batch of training instances, we define the following cross-entropy loss as the training objective:

$$\mathcal{L} = -\frac{1}{b} \sum_{i=1}^b [P\{d_1 \succ_q d_2\}_i \log(\mathcal{R}(\{q, d_1, d_2\}_i; \theta)) + (1 - P\{d_1 \succ_q d_2\}_i) \log(\mathcal{R}(\{q, d_1, d_2\}_i; \theta))] \tag{2}$$

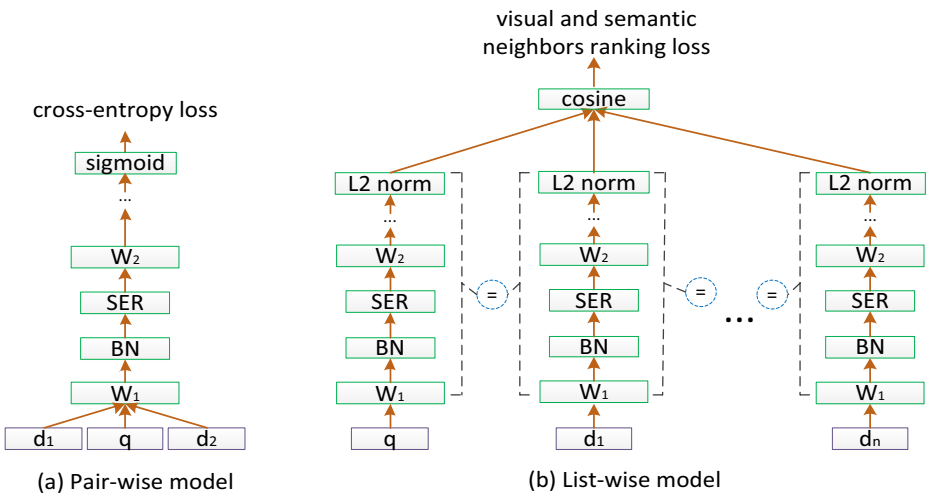


Fig. 2 Our pair-wise and list-wise neural ranking models

$P\{d_1 \succ_q d_2\}$ is the ground-truth probability of image d_1 being ranked higher than d_2 , according to the semantical similarities in the training instance:

$$P\{d_1 \succ_q d_2\} = \frac{s_{q,d_1}^S}{s_{q,d_1}^S + s_{q,d_2}^S} \tag{3}$$

When we use this pair-wise ranking model to rank testing image’s neighbors, we need turn the model’s predictions into a similarity score for each training image. Following [12], we calculate the average of predictions for each training image, against all other candidates, resulting $O(n^2)$ time complexity which is unacceptable in large scale image datasets. Thus, we use Balanced Rank Estimation (BRE) algorithm [14] to efficiently estimate scores.

List-wise model Similar to the previous one, for a given query image q , a list of images $\langle d_1, \dots, d_n \rangle$, such that $d_1 \succ_q d_2 \succ_q \dots \succ_q d_n$, our list-wise ranking model (shown as Fig. 2b) learns to predict the ranking list of q . Our model is inspired by Siamese network [11] which learns two networks sharing the same parameters that map two input images into a low dimensional space such that the Euclidean distance in that space approximates the semantic distance between this pair of images. Following the idea of Siamese network, our list-wise ranking model is composed of n networks that share parameters θ and we train these networks using our proposed visual and semantic neighbors ranking loss introduced in Section 4.2, to minimize the divergence between visual and semantic neighbors ranking lists.

4.2 Visual and semantic neighbors ranking loss

In the task of image auto-annotation, each training image has several annotations to describe the semantic of the image. Thus there are two kinds of image neighbors. The first kind is obtained by comparing images’ annotations (we call them semantic neighbors), and the another is acquired by measuring similarities of images’ visual features (we call them visual neighbors). When we predict the annotations of a testing image, we could only get the testing image’s visual neighbors. To accurately predict tags for testing image, visual features should reflect the image’s semantic contents as far as possible. Unfortunately visual features especially hand-crafted features are not power enough to completely abstract the semantic of the images. Thus traditional annotation models who find neighbors by visual features suffer from semantic gap. In this paper we extract visual features using pre-trained CNN and feed these visual features into our ranking networks. As discussed in previous section, the goal of our list-wise model is to make the ranking of visual neighbors, obtained by measuring images’ visual features’s distances, be similar as far as possible to the ranking of semantic neighbors. In this section we introduce the visual and semantic neighbors ranking loss function that is used to train our list-wise model.

Each training instance is given in the form of $(q, d_1, \dots, d_n, s_{q,d_1}^S, \dots, s_{q,d_n}^S)$ in our list-wise model, where semantical similarity s_{q,d_i}^S can be calculated using (1). Two images having higher semantical similarity have more similar semantics. Then the semantic rank ordering of image d_i with respect to query image q can be computed using the following equation.

$$r_{q,d_i}^S = \sum_{k=1}^n \mathcal{I}(s_{q,d_i}^S < s_{q,d_k}^S) \tag{4}$$

where $\mathcal{I}(\cdot)$ is an indicator function ($\mathcal{I}(true)=1, \mathcal{I}(false)=0$).

Similarly, visual rank of image d_i with respect to query image q can be obtained using the following equation.

$$r_{q,d_i}^V = \sum_{k=1}^n \mathcal{I} \left(s_{q,d_i}^V < s_{q,d_k}^V \right) \tag{5}$$

where s_{q,d_i}^V means the visual similarity between d_i and q . In this paper, we also use cosine similarity to compute this distance.

$$s_{q,d_i}^V = \frac{\psi_\theta(\phi(q))^T \cdot \psi_\theta(\phi(d_i))}{\|\psi_\theta(\phi(q))\| \cdot \|\psi_\theta(\phi(d_i))\|} \tag{6}$$

$\phi(\cdot)$ is the pre-trained CNN [28] without any modification. $\psi_\theta(\cdot)$ is the ranking network with parameter θ . In our method, we aim to train this ranking network to minimize the difference between semantic and visual neighbors ranking lists. Therefore, given a batch of training instances $(q, d_1, \dots, d_n, s_{q,d_1}^S, \dots, s_{q,d_n}^S)_i, i = 1, \dots, b$, we propose the visual and semantic neighbors ranking loss function to train this network.

$$\mathcal{L} = \frac{1}{2b} \sum_{i=1}^b \sum_{j=1}^n \left| r_{\{q,d_j\}_i}^V - r_{\{q,d_j\}_i}^S \right|^2 \tag{7}$$

Unfortunately, the (7) is a non-differentiable loss function because of the indicator function. In this paper, we adopt the solution proposed by [68] in which logistic function $\delta(x) = \log_2(1 + 2^{-x})$ is used to replace indicator function, and the loss function can be rewrote as follows:

$$\mathcal{L} = \frac{1}{2b} \sum_{i=1}^b \sum_{j=1}^n \left| \sum_{k=1}^n \delta \left(s_{\{q,d_j\}_i}^V - s_{\{q,d_k\}_i}^V \right) - \sum_{k=1}^n \delta \left(s_{\{q,d_j\}_i}^S - s_{\{q,d_k\}_i}^S \right) \right|^2 \tag{8}$$

4.3 Network structure and training method

Network structure As shown in Fig. 2, both of our pair-wise and list-wise models are composed of several linear layers with weight matrices $W_l, l = 1, \dots, N$. Each layer is a fully-connected layer computing the following transformation:

$$\mathbf{x}_l = f(BN(W_l \cdot \mathbf{x}_{l-1} + b_l)); l = 1, \dots, N - 1 \tag{9}$$

where W_l and b_l is the weight matrix and the bias of the l^{th} layer respectively, and $f(\cdot)$ is the activation function. Here we use \mathbf{x}_0 to denote the input of the network. Specially, sign expansion root (SER) [16, 17] layers, rather than Rectified Linear Unit (ReLU), are add to separate these successive fully-connected layers,¹ to perform activation function $f(\cdot)$. Typical neural activation ReLU used in CNN denoted as $f(\mathbf{x}) = \max\{\mathbf{0}, \mathbf{x}\}$ only keeps the positive activations and drops the negative activations. Experience in action recognition [19, 20] shows that negative activations should be considered for they contain some useful information when CNN features are used [16, 17]. Therefore, we adopt SER to nonlinearly project the activation of each fully connected layer. Suppose the activation is $\mathbf{x} = [x_1, x_2, \dots, x_n]$, then SER is denoted as

$$f(\mathbf{x}) = \left(\sqrt{\mathbf{x}^+}, \sqrt{\mathbf{x}^-} \right) \tag{10}$$

¹We also tried to use ReLU which perform slightly inferior than using SER (F1 values decrease %4 ~ %7 in our experiments on four datasets.)

where $\mathbf{x}^+ = [x_1^+, x_2^+, \dots, x_n^+]$ is the positive part of feature \mathbf{x} , namely $x_i^+ = \max\{0, x_i\}$, while $\mathbf{x}^- = [x_1^-, x_2^-, \dots, x_n^-]$ is the negative part of feature \mathbf{x} , namely $x_i^- = \max\{0, -x_i\}$. This operation doubles the dimensionality of feature \mathbf{x} allowing us to capture important nonlinear information from both positive and negative activations. BN in (9) is the batch normalization introduced in [29]. For the pair-wise model, we empirically use sigmoid function as the activation function of the last fully connected layer. For the list-wise model, the output of the last fully connected layer go through L2 normalization layer. Furthermore, we add a *cosine* layer to calculate visual similarities (see (6)) between the output vectors of each network to get visual ranking. It is notable that, the size of input vector is $3 \times \text{length}(\phi(\cdot))$ in the pair-wise setting while the list-wise model is feeded with input vector whose size is $\text{length}(\phi(\cdot))$.

It is noteworthy that our networks are different from that used in recent hot research topics, such as image question answering [1] and video question answering [70]. A common approach to image or video question answering is to map both the input image/video and question to a common embedding space. Visual feature extract from image or video and textual feature extracted from question are combined in an output stage, which can take the form of a classifier (e.g. a multilayer perceptron) to predict short answers from predefined set or a recurrent network (e.g. an LSTM) to produce variable-length phrases. However our neural ranking models are not trained based on two channels. The input of the neural network only includes the visual feature extracted from image and the semantic feature is used to generate ground-truth probability of one image being ranked higher than another with respect to a query image. These ground-truth probabilities will supervise the training of neural networks.

Algorithm 1 One epoch of list-wise ranking network training

Input: \mathbb{T} : Training images with associated tags.

- 1 Randomly initialize ranking network parameter θ ;
 - 2 Randomly partition training set into N mini-batches;
 - 3 **foreach** *mini-batch* **do**
 - 4 Use training instance sampling to generate b instances \mathcal{B} in the form of $(q, d_1, \dots, d_n, s_{q,d_1}^S, \dots, s_{q,d_n}^S)_i, i = 1, \dots, b, s_{q,d_i}^S$ is obtained using 1
 - 5 **foreach** *instance I_i in \mathcal{B}* **do**
 - 6 Forward propagate preprocessed images q and $d_i, i = 1, \dots, n$, through pre-trained CNN, getting $\phi(q)$ and $\phi(d_i)$
 - 7 Map $\phi(q)$ and $\phi(d_i)$ into $\psi_\theta(\phi(q))$ and $\psi_\theta(\phi(d_i))$ by ranking network
 - 8 Calculate visual similarity of each d_i with respect to q using equation and 6
 - 9 Calculate loss \mathcal{L} using 8
 - 10 Update network parameter θ
-

Semantic features For each tag of a training image, we use Word2Vec [47] to encode it as a 300 dimensional real valued vector. Then we use Fisher vector representation of Hybrid Gaussian-Laplacian mixture model (HGLMM) proposed by Klein et al. [36] to encode the Word2Vec vectors of a training image's all tags into one vector. Following literature [36], Independent Component Analysis (ICA) is applied to construct a codebook with 50 centers using first and second order information. Then HGLMM and Fisher vector are used

to produce a $300 \times 50 \times 2 = 30000$ dimensional vector to represent one training image's tags. Furthermore, we apply Principal Component Analysis (PCA) on these 30000 dimensional vectors to reduce them to 10000 dimensions. This operation help us save memory and training time. Finally, semantical similarity between training images can be calculated based on these 10000 dimensional fisher vectors using (1).

Training instance sampling Our loss function of pair-wise model involves all triplets consisting of a query image q and two relevant images d_1 and d_2 . Given a mini-batch having m images, we can obtain a training set consisting of C_m^3 training instances. It is computationally infeasible to optimize over all such instances. Hence, we first randomly select t query images from the mini-batch, then randomly select k pairs of images from the rest of the mini-batch, resulting $b = t \cdot k$ training instances within each mini-batch. For our list-wise model, we also first select t query images randomly. And then we select k lists of images, the length of each list is L ($L < m - 1$), from the remaining $m - 1$ images in the mini-batch. We will analyze the effectiveness of this training instance sampling method in our experiments.

Training details To train our list-wise model, we are given a number of training images with tags, denoted as \mathbb{T} . We use Adam [34] to update the network's parameters. For one training epoch, we randomly partition the training set into N mini-batches. Then we use training instance sampling method to generate b training instances in each mini-batch. For each image, we resize it and put it into a pre-trained ResNet [28] and the get the 2048 dimensional activations of the layer right before the last fully-connected layer as the CNN feature $\phi(I)$ of the image. Our list-wise ranking network nonlinearly projects $\phi(I)$ into $\psi_\theta(\phi(I))$. Semantical and visual similarities then can be obtained to rank neighbors. Although our neural ranking networks can have many layers in general, we select the number of layers and the size of each layer from [1,2,3,4] and [256, 512, 1024, 2048, 4096] respectively. These hyper-parameters are tuned on the validation set using batched GP bandits with an expected improvement acquisition function [54]. In our work, Adam with initial learning rate 0.001 is used to update the parameters of ranking networks. The size of mini-batch is fixed to be 128. Algorithm 1 gives the pseudo-code of our list-wise ranking network training (The training process of our pair-wise model is similar and more simple, and we do not give the pseudo-code here). In our experiments, we first train both of these two networks on the large-scale dataset NUS-WIDE, and annotate the testing images in this dataset. When we predict labels for testing images in the other smaller datasets, we fine-tune these networks using their training images, after initializing the networks with the weights learned from NUS-WIDE.

5 Label propagation

For each testing image, both of our pair-wise and list-wise neural ranking networks can generate ranking list in which semantically similar training images are ranked top. This property is one of the basis of many nearest-neighbor based models and can effectively improve their performance. Following this key idea, we apply our ranking networks to several popular nearest-neighbor based image auto-annotation models including Tagprop [26], 2PKNN [57] because they report the state-of-the-art results on several datasets and their source codes are opened.

5.1 TagProp

TagProp² is proposed by Guillaumin [26] which learns weighted nearest neighbor model or word-specific logistic discriminant model by maximizing the log-likelihood of the tag predictions in the training set. When using weighted nearest neighbor model, the probability of tag t appearing in the annotations of testing image I is:

$$P_{TagProp}(I, t) = \sum_{J \in \mathcal{N}(I)} w_{IJ} \mathcal{I}(J, t) \quad (11)$$

where $\mathcal{I}(J, t)$ is an indicator function who equals one if image J has label t , zero otherwise. w_{IJ} is a learned weight of image J for propagating tags to image I . $\mathcal{N}(I)$ is the neighbors of image I . This weight can be learned from image rank, referred to as the TagProp-RK, or image distance, referred to as the TagProp-SD for single distance and TagProp-ML for multiple distances combined by metric learning. When using word-specific logistic discriminant model, tag t annotates image I with probability:

$$P_{TagProp}(I, t) = \sigma \left(\beta_t + \alpha_t \sum_{J \in \mathcal{N}(I)} w_{IJ} \mathcal{I}(J, t) \right) \quad (12)$$

where $\sigma(\cdot)$ is sigmoid function and $\{\alpha_t, \beta_t\}$ can be trained per tag. The resulting variants are referred to as TagProp- σ RK, TagProp- σ SD, TagProp- σ ML, respectively. When we integrate pair-wise model with TagProp, the normalized estimated similarity score can be considered as w . And the visual similarity (see (6)) can be considered as w when we build TagProp on list-wise model. Because both the estimated similarity score and visual similarity can be considered as a single distance between images, we finally integrate our models with TagProp- σ SD in our experiments, denoted as PNR+TagProp- σ SD and LNR+TagProp- σ SD.

5.2 2PKNN

Verma and Jawahar [57] proposed 2PKNN³ model to address image auto-annotation task. 2PKNN is composed of two phases. Given a testing image I , the first phase is to identify its neighbors $\mathcal{N}(I)$ from each semantic group which is a subset of training images with one common label. The second phase of 2PKNN is to predict labels for testing image by weighted summing over the reference images in $\mathcal{N}(I)$:

$$P_{2PKNN}(I, t) = \sum_{J \in \mathcal{N}(I)} \exp \left(- \sum_{i=1}^n \omega_i d_{I,J}^i \right) \mathcal{I}(J, t) \quad (13)$$

where $d_{I,J}^i$ is the distance between the i^{th} feature of image I and J . The default value of weight ω_i is $1/n$. Furthermore, this distance can be improved by metric learning which learns the weight ω_i , leading to 2PKNN-ML model. When we integrate our models with 2PKNN, we use (14) to predict tags for testing image, where $s_{I,J}^V$ is the normalized similarity score estimated by pair-wise model or visual similarity generated by list-wise model.

²The source code of TagProp is available at: <http://lear.inrialpes.fr/people/guillaumin/code.php#tagprop>

³The source code of 2PKNN is available at: <http://researchweb.iit.ac.in/yashaswi.verma/eccv12/2pknn.zip>

We denote our models integrated with 2PKNN as PNR+2PKNN and LNR+2PKNN in our experiments.

$$P_{2PKNN}(I, t) = \sum_{J \in \mathcal{N}(I)} s_{I,J}^V \mathcal{I}(J, t) \quad (14)$$

6 Experiments

6.1 Datasets

We conduct a series of experiments to examine the performance of our proposed image auto-annotation method on four widely used standard image datasets. Statistics of these four datasets are shown in Table 1.

Corel 5k This image set is the first and has become a basic dataset for image auto-annotation [9, 15, 26, 38, 45, 57]. It contains 5000 images, falling into 50 categories and each category has 100 images on the same topic. 1 to 5 tags are manually annotated for each image and there are 260 tags in total in the dataset. We extract 4500 images for training (10% of the training images are used as validation set for hyper-parameter tuning) and others for testing.

ESP game It is collected from a computer game, which need two players to predict same tags for a given image to gain points. This dataset is very challenging for its wide variety of image contents. We follow [26] and split this image set into training set with 18689 images and testing set with 2081 images. We also randomly select 10% of the training images as validation set.

IAPR TC12 It is originally used for cross-lingual retrieval. Researchers extract common nouns using natural language processing techniques to generate tags for each image which has been used in [45]. And now it is also a benchmark for image auto-annotation task [15, 45]. This dataset has 17665 images for training (10% of this set are chosen as validation set) and 1962 images for testing. Each image is annotated with an average of 5.7 labels.

NUS-WIDE This dataset has 269648 images collected from Flickr. Most of these images are manually annotated with several labels, leading to 81 labels in total for the whole dataset. Following [22], we have 209347 images after discarding images with no labels and we select a subset of 150k images for training and the others are left for testing. 15k images in the training set are selected as validation set.

Table 1 Statistics of the four image sets. The "Tags per image" row is in the format "mean, median"

	Corel 5K	ESP Game	IAPR TC-12	NUS-WIDE
Total number of images	5000	20770	19627	209347
Total number of tags	260	268	291	81
Training/testing images	4500/500	18689/2081	17665/1962	150000/59347
Tags per image	3.4, 4	4.7, 5	5.7, 5	2.4, 2

6.2 Baselines and evaluation protocols

In this paper, we propose neural ranking networks including pair-wise model and list-wise model, to help testing images find more semantically similar neighbors. And we adopt nearest-neighbor based label propagation models to annotate the testing image by weighted propagating tags from these neighbors. Thus we consider the following models as baselines: (1) TagProp- σ SD based on bare CNN features, and (2) 2PKNN based on bare CNN features

For fair comparison, we follow previous works [2, 26, 35, 57], using the following popular protocols to evaluate our method:

Mean word recall With a given tag, word recall calculates the number of correctly annotated images, divided by the number of images whose ground-truth annotations include this tag. Suppose c_w is the number of images correctly annotated with tag w and g_w denotes the number of images having tag w in the ground truth. Then word recall measures the completeness of annotating images with word w . We separately calculate the word recall for each tag and then report the mean word recall:

$$R = \frac{1}{|W|} \sum_{w \in W} \frac{c_w}{g_w} \quad (15)$$

where W is the fixed vocabulary of annotations and $|W|$ means the size of this vocabulary.

Mean word precision Word precision is the number of correctly annotated images with a given tag w , divided by the total number of images annotated with tag w including correctly annotated ones and wrong ones. We use t_w to denote the number of images annotated with tag w by the model (correctly or not). Follow the mean word recall calculation, we report the mean word precision as follows:

$$P = \frac{1}{|W|} \sum_{w \in W} \frac{c_w}{t_w} \quad (16)$$

F1 It is the trade-off between mean word precision and recall which combined them for easier comparability. We also report this measure of our method and several state-of-the-art methods. It can be obtained by using the following equation:

$$F1 = \frac{2 * P * R}{P + R} \quad (17)$$

N+ It is the number of tags with non-zero recall value. This parameter aims to measure the ability of the model to annotate images with rare tags which are hard to predict due to the low frequency of occurrence in the training set.

Note that, for Corel5k, ESP-Game, and IAPR TC12, we predict 5 labels for each testing image, while 3 labels are predicted for each testing image in NUS-WIDE since images in this dataset only have averagely 2.4 labels (see Table 1).

6.3 Effectiveness of neural ranking models

In this paper, we argue that both of our pair-wise and list-wise neural ranking networks can generate ranking list in which semantically similar training images are ranked top. And this property makes our neural ranking models able to boost the annotation performance of

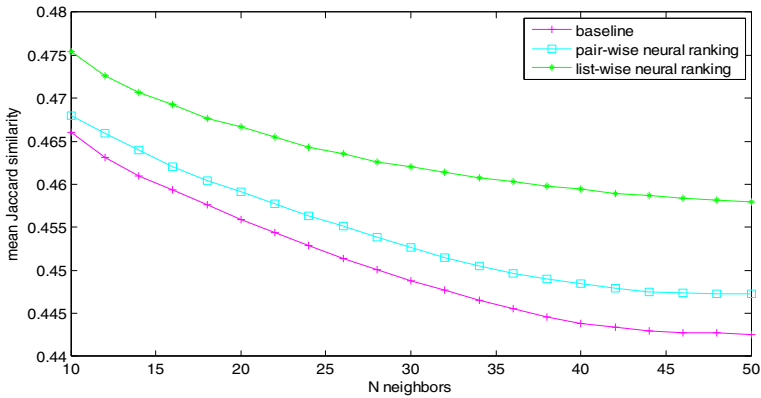


Fig. 3 Mean Jaccard similarity between testing images and their neighbors build using bare CNN features and our ranking models respectively.

nearest-neighbor based models. To validate this idea, we conduct an experiment on NUS-WIDE here. For each image I in the testing set, we respectively use the similarity scores estimated by our pair-wise ranking model, visual similarities generated by our list-wise ranking model and visual similarities based on bare CNN features, to retrieve its top N most similar images $\{I_1, \dots, I_N\}$. We then average the Jaccard similarity between image I and I_i as follows:

$$\frac{1}{N} \sum_{i=1}^N J(I, I_i) = \frac{1}{N} \sum_{i=1}^N \frac{|T_I \cap T_{I_i}|}{|T_I \cup T_{I_i}|} \tag{18}$$

where T_I and T_{I_i} denote the set of labels of image I and I_i respectively. Finally we compute this measure for all the testing images and report the average values using different N in Fig. 3. Experimental results show that both of our neural ranking models achieve higher mean Jaccard similarity, compared with baseline which uses visual similarities of bare CNN features. It means that our neural ranking models can help testing images retrieve more semantically similar neighbors.

6.4 Overall comparison of image annotation models

In this experiment, we analyse the performance of our methods. We apply our neural ranking models to TagProp- σ SD and 2PKNN respectively, resulting four models: Pair-wise neural ranking + TagProp- σ SD (PNR+TagProp- σ SD), List-wise neural ranking + TagProp- σ SD (LNR+TagProp- σ SD), Pair-wise neural ranking + 2PKNN (PNR+2PKNN), List-wise neural ranking + 2PKNN (LNR+2PKNN). For each method, we use the activations of the last second layer as visual features of images and feed them to neural ranking models. To demonstrate the effectiveness of our methods, we also compare our results with several state-of-the-art image auto-annotation models, including TagProp- σ SD, TagProp- σ ML [26], JEC [46], GS [69], RF-opt [18], 2PKNN [57], 2PKNN-ML [57], KSVM-VT [58], MLRank [41], SKL-CRM [49], CMM [2], CCA-KNN [56]. The features used by these methods are all hand-crafted⁴ and these methods span the wide range of model types discussed in Section 1.

⁴These features are available at: <http://lear.inrialpes.fr/people/guillaumin/data.php>

Table 2 shows the annotation performance of our proposed methods on Corel5k, ESP Game, and IAPR TC-12 datasets, against with two baselines and some state-of-the-art methods proposed in recent years. The results of our experiments demonstrate that our best method LNR+2PKNN outperforms all these state-of-the-art methods we have reviewed while our other three methods also achieve comparable performance with a number of recently proposed methods. Our method LNR+2PKNN achieves P, R, F1 and N+ scores of 43, 52, 47 and 192 on the Corel 5K dataset, 48, 34, 40 and 258 on the ESP Game dataset, 44, 38, 41 and 276 on the IAPR TC-12 dataset. Comparing the results of our methods and the baselines, we also find that our proposed ranking networks supervised by semantic features can significantly boost the annotation performance, probably because of the effectiveness of our neural ranking networks and image’s semantic representation using fisher vector of HGLMM. We also observe that our list-wise ranking network is more effective than pair-wise ranking network, for both TagProp-σSD and 2PKNN.

Table 3 shows the comparison of our methods and the state-of-the-art methods on the large-scale imageset NUS-WIDE. The 634D visual features used by MLRank is a set of

Table 2 Comparison of annotation performance of our methods with the state-of-the-art methods and baselines in terms of mean word recall (R), mean word precision (P), F1 and N+ on Corel 5K, ESP Game, IAPR TC-12 dataset

Method	Feature		Corel5k				ESP Game				IAPR TC-12			
	Visual	Textual	R@5	P@5	F1	N+	R@5	P@5	F1	N+	R@5	P@5	F1	N+
State of the art:														
TagProp-σSD[26]	H	–	35	28	31	145	24	39	30	232	30	41	35	259
TagProp-σML[26]	H	–	42	33	37	160	27	39	32	239	35	46	40	266
JEC[46]	H	–	32	27	29	139	25	22	23	224	29	28	29	250
GS[69]	H	–	33	30	31	146	–	–	–	–	29	32	30	252
RF-opt[18]	H	–	40	29	34	157	26	41	32	235	31	44	36	253
2PKNN[57]	H	–	40	39	40	177	23	51	32	245	32	49	39	274
2PKNN-ML[57]	H	–	46	44	45	191	27	53	36	252	37	54	44	278
K SVM-VT[58]	H	–	42	32	36	179	32	33	33	259	29	47	36	268
MLRank[41]	H	–	32	37	34	151	–	–	–	–	32	38	35	259
SKL-CRM[49]	H	–	46	39	42	184	26	41	32	248	32	51	39	274
CMM[2]	H	BOW	46	42	44	179	–	–	–	–	30	59	40	259
CCA-KNN[56]	V	BV	51	39	44	192	32	44	37	254	34	41	37	273
Baselines:														
TagProp-σSD	R	–	34	29	31	141	28	34	31	237	28	39	33	251
2PKNN	R	–	44	38	41	167	32	43	37	246	30	40	35	261
Our methods:														
PNR+TagProp-σSD	R	FV	35	32	33	168	28	36	32	240	29	40	34	255
LNR+TagProp-σSD	R	FV	49	39	43	181	30	48	37	252	32	46	38	270
PNR+2PKNN	R	FV	45	38	41	182	32	44	37	246	32	45	37	274
LNR+2PKNN	R	FV	52	43	47	192	34	48	40	258	38	44	41	276

H: Hand-crafted features, R: ResNet activations, V: VGG16 activations, BOW: Bag of Words, BV: Binary vector, FV: Fisher vector of HGLMM

Table 3 Comparison of annotation performance of our methods with the state-of-the-art methods and baselines in terms of mean word recall (R), mean word precision (P), F1 and N+ on NUS-WIDE dataset

Method	Feature		NUS-WIDE			
	Visual	Textual	$R@3$	$P@3$	F1	N+
State of the art:						
MLRank[41]	634D	–	28	24	26	–
CNN+Softmax[22]	AlexNet	–	31	32	31	80
CNN+WARP[22]	AlexNet	–	36	32	34	78
CNN+KNN[32]	CaffeNet	–	31	44	36	–
CNN+Logistic[32]	CaffeNet	–	43	41	42	–
CNN-RNN[59]	VGG16	LSTM	30	41	35	–
Baselines:						
TagProp- σ SD	ResNet	–	35	42	38	77
2PKNN	ResNet	–	39	42	40	79
Our methods:						
PNR+TagProp- σ SD	ResNet	FV	40	42	41	79
LNR+TagProp- σ SD	ResNet	FV	44	44	44	80
PNR+2PKNN	ResNet	FV	41	43	42	79
LNR+2PKNN	ResNet	FV	46	44	45	80

FV: Fisher vector of HGLMM

hand-crafted features.⁵ CNN+Softmax and CNN+WARP models train convolutional neural networks from scratch using softmax and WARP loss to achieve end-to-end image annotation, obtaining an inferior performance with respect to the models, including CNN+KNN and CNN+Logistic which use pre-trained CNN features. Wang et al. attempted to learn joint image-label embedding using CNN-RNN framework [59], getting unsatisfactory performance. The best performance is observed in our model LNR+2PKNN, improving P , R , $F1$, and $N+$ to 44, 46, 45 and 80. This suggests that nearest-neighbor based model integrated with our neural ranking networks is effective to address image annotation task for such a large image set.

6.5 Impact of neighborhood size

One of the most important parameters need to be set manually in our methods is K which is the size of the neighborhood \mathcal{N} (see (12) and (13)). To evaluate the impact of K on annotation performance of our methods, we also conduct a number of experiments. For Corel 5K, ESP Game and IAPR TC-12 datasets, We set the parameter K to be 50, 70, 100, 200, 300, 500, 700, 900, 1000 respectively and record the performance of our methods, while K is set to be 100, 200, 300, 500, 600, 700, 800, 900, 1000 for NUS-WIDE dataset. Figure 4 shows how the parameter K affects the performance of our methods and baselines on these four datasets. It is obviously that small value of K leads to low annotation performances for all the methods on all these datasets. It is because we get a small number of neighbors

⁵These features can be find at: <http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm>

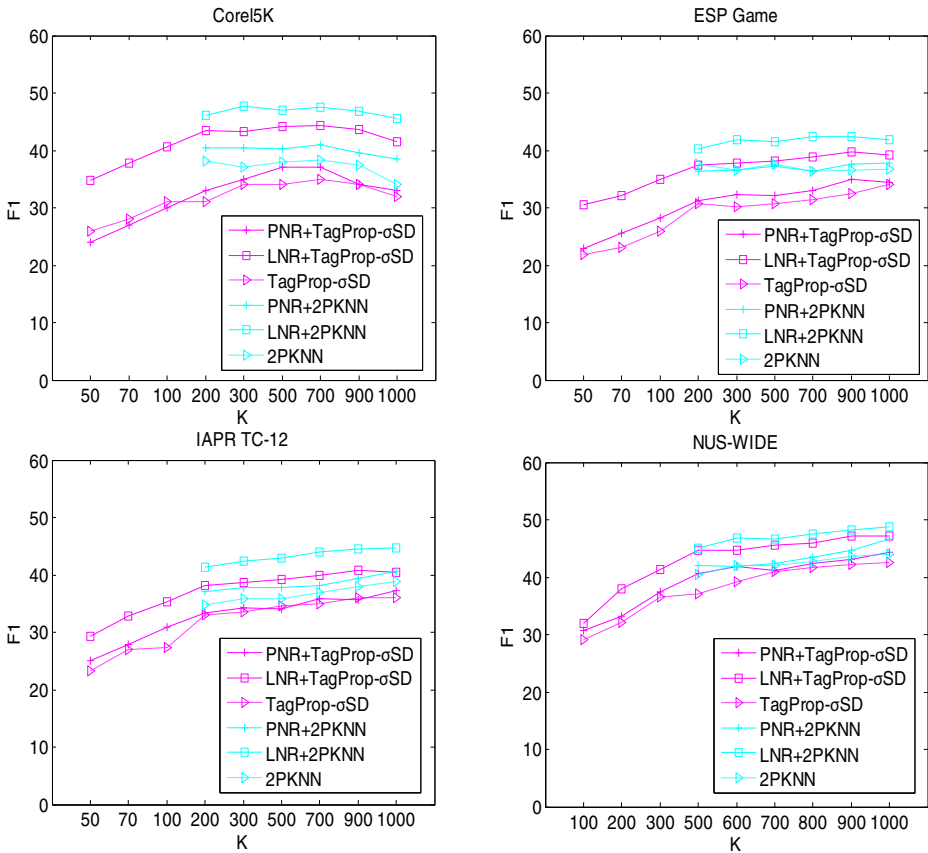


Fig. 4 F1 scores of our models (solid lines) and baselines (dash lines) on Corel5K, ESP Game, IAPR TC-12 and NUS-WIDE datasets varying the neighborhood size K

and some important information may be lost when we propagate tags from weighted neighbors to testing image using small K . Furthermore, we see that annotation performance will remain almost the same or rise slowly with increasing the value of K after some point (200 for ESP Game and IAPR TC-12, 500 for NUS-WIDE). Specially, annotation performance will deteriorate slightly with setting K greater than about 500 for Corel5K dataset, probably because selecting too many neighbors from Corel5K’s small training set brings in irrelevant images to the testing image. Big neighborhood size K will increase the computational cost. Hence, we set K to be 200 for Corel 5K, ESP Game, IAPR TC-12 datasets and 500 for NUS-WIDE dataset, obtaining the results in Tables 2 and 3. Furthermore, as expected, we observe that annotation performance of each of our methods is higher than the corresponding baseline. This again confirms that our neural ranking networks can boost annotation performance of nearest-neighbor based models.

6.6 Effectiveness of training instance sampling

In this section, we analyze the effectiveness of our training instance sampling method on the large scale imageset NUS-WIDE. In our experiments, we first randomly select k query

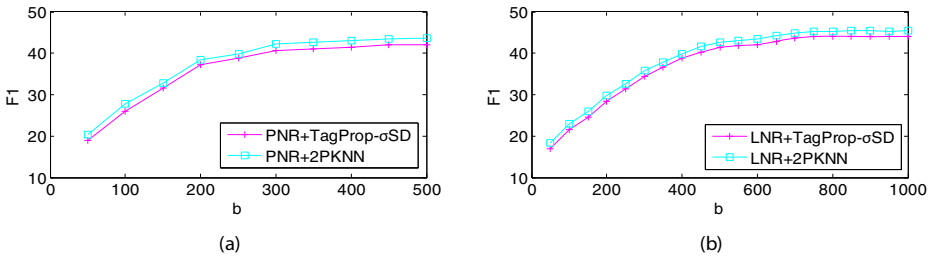


Fig. 5 Analysis of training instance sampling. **a** The curve of $F1$ of pair-wise models on NUS-WIDE. **b** The curve of $F1$ of list-wise models on NUS-WIDE

images, and then select t pairs of images to generate b ($b = k \cdot t$) training instances to train our pair-wise models. For our list-wise models, we also randomly choose k query images, and then select t lists of images (the length of each image list is fixed to be 64). We vary the value of parameter b for both pair-wise and list-wise models, and report the performance in Fig. 5. It is observed that 2PKNN based on neural ranking is slightly better than TagProp- σ SD using same number of training instances. The experimental results also suggest that increasing training instances can boost annotation performance, but the improvement saturates finally. Therefore, we set b to be 250 and 500 for pair-wise model and list-wise model respectively, to achieve compromise between annotation performance and training time cost. The results in Table 3 is obtained using this parameter setting. For the other datasets, we also use this parameter setting to fine-tune the networks.

7 Conclusion and future work

In this paper, we formulate automatic image annotation into a learning to rank framework. Explicitly, we propose pair-wise and list-wise neural ranking networks and apply them to nearest-neighbor based models to address image auto-annotation task. These networks share the similar architecture which is composed of several fully-connected layers. The pair-wise ranking network is trained to minimize cross-entropy loss, while the list-wise ranking network is trained using our proposed visual and semantic neighbors ranking loss. These networks can help images find more semantically similar neighbors. This property improves the performance of the nearest-neighbor based models lying on our embedding network. Our experimental results on four standard datasets including Corel 5K, ESP Game, IAPR TC-12, and large-scale NUS-WIDE demonstrate that our methods make remarkable improvements on all of these datasets, compared with existing state-of-the-art methods.

In our future work, we will consider to build a new image dataset in which each image has privileged information, such as sounding text that describing the image. And only part of the training images have labels. Then active learning [64] or other semi-supervised learning methods [7, 67] can be employed to use these privileged information and instances without labels to boost the performance of automatic image annotation.

Acknowledgements This work is supported by the Natural Science Foundation of China (No. 61572162) and the Zhejiang Provincial Key Science and Technology Project Foundation (No. 2018C01012).

Compliance with ethical standards

Conflict of interests The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Agrawal A, Lu J, Antol S (2015) Vqa: Visual question answering. *Int J Comput Vis* 123(1):4–31
2. Ballan L, Uricchio T, Seidenari L, Bimbo AD (2014) A cross-media model for automatic image annotation. In: *ACM ICMR*, pp 73–80
3. Blei D, Jordan M (2003) Modeling annotated data. In: *ACM SIGIR*, pp 127–134
4. Breiman L (2001) Random forests. *Mach Learn* 45(1):5–32
5. Burges C (2005) Learning to rank using gradient descent. In: *ICML*, pp 89–96
6. Burges C (2010) From ranknet to lambdarank to lambdamart: An overview. In: *Technical report, Microsoft Research*
7. Cai D, He X, Han J (2007) Semi-supervised discriminant analysis. In: *ICCV*
8. Cao Z, Qin T (2007) Learning to rank: from pairwise approach to listwise approach. In: *ICML*, pp 129–136
9. Carneiro G, Chan A, Moreno P, Vasconcelos N (2007) Supervised learning of semantic classes for image annotation and retrieval. *IEEE Trans Pattern Anal Mach Intell* 29(3):394–410
10. Chatfield K, Lempitsky V, Vedaldi A, Zisserman A (2011) The devil is in the details: an evaluation of recent feature encoding methods. In: *BMVC*, pp 1–12
11. Chopra S, Hadsell R, LeCun Y (2005) Learning a similarity metric discriminatively, with application to face verification. In: *CVPR*, pp 539–546
12. Dehghani M, Zamani H, Severyn A, Kamps J, Croft WB (2017) Neural ranking models with weak supervision. In: *ACM SIGIR*, pp 65–74
13. Deng J, Dong W, Socher R, Li L, Li K, Fei-Fei L (2009) Imagenet: A large-scale hierarchical image database. In: *CVPR*, pp 248–255
14. Fabian L, Michael J, Nebojsa J (2013) Efficient ranking from pairwise comparisons. In: *ICML*, pp 109–117
15. Fenga S, Manmatha R, Lavrenko V (2004) Multiple bernoulli relevance models for image and video annotation. In: *CVPR*, pp 1002–1009
16. Fernando B, Anderson P, Hutter M, Gould S (2016) Discriminative hierarchical rank pooling for activity recognition. In: *CVPR*, pp 1924–1932
17. Fernando B, Gawes E, Oramas J, Ghodrati J, Tuytelaars T (2017) Rank pooling for action recognition. *IEEE Trans Pattern Anal Mach Intell* 39(4):773–787
18. Fu H, Zhang Q, Qiu G (2012) Random forest for image annotation. In: *ECCV*, pp 86–99
19. Gao Z, Nie W, Liu A (2016) Evaluation of local spatial-temporal features for cross-view action recognition. *Neurocomputing* 173(1):110–117
20. Gao Z, Zhang H, Liu A (2016) Human action recognition on depth dataset. *Neural Comput Applic* 27(7):2047–2054
21. Gao Z, Zhang L, Chen M (2014) Enhanced and hierarchical structure algorithm for data imbalance problem in semantic extraction under massive video dataset. *Multimedia Tools Appl* 68(3):641–657
22. Gong Y, Jia Y, Leung T, Toshev A, Ioffe S (2014) Deep convolutional ranking for multilabel image annotation. [arXiv:1312.4894](https://arxiv.org/abs/1312.4894)
23. Gong Y, Ke Q, Isard M, Lazebnik S (2014) A multi-view embedding space for modeling internet images, tags, and their semantics. *Int J Comput Vis* 106(2):210–233
24. Gong Y, Wang L, Hodosh M, Hockenmaier J, Lazebnik S (2014) Improving image-sentence embeddings using large weakly annotated photo collections. In: *ECCV*, pp 529–545
25. Gu Y, Xue H, Yang J (2016) Cross-modal saliency correlation for image annotation. *Neural Process Lett* 45(3):777–789
26. Guillaumin M, Mensink T, Verbeek J, Schmid C (2009) Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In: *ICCV*, pp 309–316

27. Hardoon D, Szedmak S, Shawe-Taylor J (2004) Canonical correlation analysis: An overview with application to learning methods. *Neural Comput* 16(12):2639–2664
28. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: *CVPR*, pp 770–778
29. Ioffe S, Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: *ICML*, pp 448–456
30. Jeon J, Lavrenko V, Manmatha R (2003) Automatic image annotation and retrieval using cross-media relevance models. In: *ACM SIGIR*, pp 119–126
31. Joachims T (2002) Optimizing search engines using clickthrough data. In: *ACM SIGKDD*, pp 133–142
32. Johnson J, Ballan L, Fei-Fei L (2015) Love thy neighbors: Image annotation by exploiting image metadata. In: *ICCV*, pp 4624–4632
33. Kang F, Sukthankar R (2006) Correlated label propagation with application to multi-label learning. In: *CVPR*, pp 1719–1726
34. Kingma D, Ba J (2014) Adam: A method for stochastic optimization. arXiv:[14126980](https://arxiv.org/abs/1412.6980)
35. Kiros R, Szepesvari C (2015) Deep representations and codes for image auto-annotation. In: *NIPS*, pp 917–925
36. Klein B, Lev G, Sadeh G, Wolf L (2015) Fisher vectors derived from hybrid gaussian-laplacian mixture models for image annotation. arXiv:[14117399](https://arxiv.org/abs/1411.7399)
37. Krizhevsky A, Sutskever I, Hinton G (2012) Imagenet classification with deep convolutional neural networks. In: *NIPS*, pp 1106–1114
38. Lavrenko V, Manmatha R, Jeon J (2004) A model for learning the semantics of pictures. In: *NIPS*, pp 553–560
39. Lazebnik S, Schmid C, Ponce J (2006) Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: *CVPR*, pp 2169–2178
40. Li X, Snoek C, Worring M (2007) Learning social tag relevance by neighbor voting. *IEEE TMM* 11(7):1310–1322
41. Li Z, Liu J, Xu C, Lu H (2013) Mlrank: Multi-correlation learning to rank for image annotation. *Pattern Recogn* 46(10):2700–2710
42. Liu J, Li M, Liu Q, Lu H, Ma S (2009) Image annotation via graph learning. *Pattern Recogn* 42(2):218–228
43. Liu T (2009) Learning to rank for information retrieval. *Found Trends Inf Retr* 3(3):225–331
44. Lowe D (2004) Distinctive image features from scale-invariant keypoints. *IJCV* 60(2):91–110
45. Makadia A, Pavlovic V, Kumar S (2008) A new baseline for image annotation. In: *ECCV*, pp 316–329
46. Makadia A, Pavlovic V, Kumar S (2010) Baselines for image annotation. *Int J Comput Vis* 90(1):88–105
47. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv:[13013781](https://arxiv.org/abs/1301.3781)
48. Montazer G, Giveki D (2017) Scene classification using multi-resolution waholb features and neural network classifier. *Neural Process Lett* 46(2):681–704
49. Moran S, Lanvrenko V (2014) Sparse kernel learning for image annotation. In: *ACM ICMR*, p 113
50. Oliva A, Torralba A (2001) Modeling the shape of the scene: a holistic representation of the spatial envelope. *IJCV* 42(3):145–175
51. Peng X, Zou C, Qiao Y, Peng Q (2010) Action recognition with stacked fisher vectors. In: *ECCV*, pp 581–595
52. Perronnin F, Sanchez J, Mensink T (2010) Improving the fisher kernel for large scale image classification. In: *ECCV*, pp 143–156
53. Song Y, Zhuang Z, Li H, Zhao Q, Li J, Lee W, Giles CL (2008) Real-time automatic tag recommendation. In: *ACM SIGIR*, pp 515–522
54. Thomas D, Andreas K, Joel W (2014) Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. *J Mach Learn Res* 15(1):3873–3923
55. Thorsten J (2006) Training linear svms in linear time. In: *KDD*, pp 217–226
56. Venkatesh N, Subhransu M, Manmatha R (2015) Automatic image annotation using deep learning representations. In: *ACM ICMR*, pp 603–606
57. Verma Y, Jawahar C (2012) Image annotation using metric learning in semantic neighbourhoods. In: *ECCV*, pp 836–849
58. Verma Y, Jawahar C (2013) Exploring svm for image annotation in presence of confusing labels. In: *British Machine Vision Conference*, pp 1–11
59. Wang J, Yang Y, Mao J, Huang Z, Huang C, Xu W (2016) Cnn-rnn: A unified framework for multi-label image classification. In: *CVPR*, pp 2285–2294

60. Wang L, Liu L, Khan L (2004) Automatic image annotation and retrieval using subspace clustering algorithm. In: ACM International Workshop Multimedia Databases, pp 100–108
61. Weston J, Bengio S, Usunier N (2011) Wsabie: Scaling up to large vocabulary image annotation. In: IJCAI, pp 2764–2770
62. Wu F, Jing X, Yue D (2017) Multi-view discriminant dictionary learning via learning view-specific and shared structured dictionaries for image classification. *Neural Process Lett* 45(2):649–666
63. Yan X, Su XG (2009) *Linear regression analysis: Theory and computing*. World Scientific Publishing Co, Inc, River Edge
64. Yan Y, Nie F, Li W, Gao C, Yang Y, Xu D (2016) Image classification by cross-media active learning with privileged information. *IEEE Trans Multimedia* 18(12):2494–2502
65. Yang C, Dong M, Hua J (2007) Region-based image annotation using asymmetrical support vector machine-based multiple-instance learning. In: CVPR, pp 2057–2063
66. Yang Y, Xu D, Nie F, Yan S, Zhuang Y (2010) Image clustering using local discriminant models and global integration. *IEEE Trans Image Process* 19(10):2761–2773
67. Yang Y, Nie F, Xu D, Luo J, Zhuang Y, Pan Y (2012) A multimedia retrieval framework based on semi-supervised ranking and relevance feedback. *IEEE Trans Pattern Anal Mach Intell* 34(4):723–742
68. Yun H, Raman P, Vishwanathan S (2014) Ranking via robust binary classification. In: NIPS, pp 2582–2590
69. Zhang S, Huang J, Huang Y (2010) Automatic image annotation using group sparsity. In: CVPR, pp 3312–3319
70. Zhu L, Xu Z, Yang Y, Hauptmann AG (2017) Uncovering the temporal context for video question answering. *Int J Comput Vis* 124(3):409–421



Weifeng Zhang received his B.S degree in Electronic Information Engineering from Beijing University of Technology in 2009, and got his M.S degree in Pattern Recognition from Beihang University in 2012. Now he is a Ph.D student of Hangzhou Dianzi University. His research interests include machine learning, multimedia modeling.



Hua Hu is a full professor of Hangzhou Dianzi University, China. He received the Ph.D., M.S. and B.S. degree in Computer Science from Zhejiang University, China in 1998, 1992 and 1989, respectively. His research interests mainly include parallel computing and distributed system and pervasive computing. His research results have been published in more than 50 papers in international journals and conference proceedings.



Haiyang Hu received the Ph.D., M.S. and B.S. degrees in Computer Science from Nanjing University, China in 2006, 2003 and 2000, respectively. Currently, he is a professor of the Hangzhou Dianzi University, China. His research interests include mobile computing and distributed computing. His research results have been published in more than 20 papers in international journals and conference proceedings.