LSTM for Image Annotation with Relative Visual Importance

Geng Yan[1]
gyan@zju.edu.cn
Yang Wang[2]
ywang@cs.umanitoba.ca
Zicheng Liao[1]
zliao@zju.edu.cn

[1] College of Computer Science
Zhejiang University
Hangzhou, China

[2] Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada

Abstract

We consider the problem of image annotations that takes into account of the relative visual importance of tags. Previous works usually consider the tags associated with an image as an unordered set of object names. In contrast, we exploit the implicit cues about the relative importance of objects mentioned by the tags. For example, important objects tend to be mentioned first in a list of tags. We propose a recurrent neural network with long-short term memory to model this. Given an image, our model can produce a *ranked* list of tags, where tags for objects of higher visual importance appear earlier in the list. Experimental results demonstrate that our model achieves better performance on several benchmark datasets.

1 Introduction

Consider Fig. 1. This image contains a rich set of objects of different sizes, colors and directions of motion. And humans have the remarkable ability to selectively process very narrow regions of the scene that are important to us. So when asked to annotate this image, we only mention a subset of the objects appearing in the image, and we mention the important objects first. In comparison, current visual recognition systems lack the capability of reasoning about the relative importance of objects in the scene, while a human child naturally casts his/her focus onto the two persons and the boat and leaves the inessentials to the back-channel.

Therefore, image understanding not only means to assign categorical labels to an image but also requires to parse the relative importance of the visual labels, for example, in the form of a ranked tag list where more important items are put on the top. Such a ranked tag list can be useful for various applications including image retrieval, image parsing and image caption generation. For example, if a user searches for images with the query "tree", we probably do not want to return the image in Fig. 1.

To extract the importance information of visual content in a scene is difficult. The difficulty comes from a variety of factors: from the ambiguity of low level pixel representation (e.g., irradiance to digital signal transformations and nonlinearity of color spaces), to the mid-level vision problems (e.g., occlusion, clutter, lighting conditions and camera projection), to the higher level cognitive issues such as the effects of context and perceptual fill-in

Pages 78.1-78.11
DOI: https://dx.doi.org/10.5244/C.30.78

Figure 1: An image may contain a rich set of objects, e.g. person, boat, tree, water, helmet, etc. But when humans are asked to describe an image, they do not enumerate all the objects in the image. Instead, they will choose a few important objects and put them in some order depending on the relative importance of these objects. In this paper, we develop a method for generating such ranked list of object tags that take into account of the relative object importance.

– let alone to reason about the relationship between objects and to understand the on-going events and their impact on an observer.

Most previous work in image annotation treats each tag independently and learns a separate classifier to predict the presence/absence of a tag. There has been some work on relaxing this independence assumption. For example, Qi *et al*. [18] use conditional random fields (CRFs) to model the correlations between pairs of tags. However, CRFs can only capture very simple correlations (e.g. co-occurrence statistics). They cannot model the relative importance of tags implicitly captured by the relative order of tags.

Recently, recurrent neural networks (RNN) have been proved to be a powerful tool for modeling sequential data, e.g. text and speech generation. In computer vision, RNN has been successfully applied to image and video captioning. In this work, we take a step further to use RNN to generate a ranked tag list from an image. While it is naturally the case that RNN can model strongly correlated sequential data (i.e. a sentence) using shared recurrent operators, it is not at all intuitive that this will be the case for a ranked tag list in our case, as the statistical or semantical correlation between the list elements is much less obvious. Our intuition is that there do exist regularities in the context of visual scenes, which provides the "visual syntax" for the elements of an image. So, we are attempting to capture the relative importance of visual objects that are constructed out of such a "visual syntax" by RNN.

The main contributions of this paper are three-fold. First, we introduce a novel problem for image annotation. In our problem formulation, we consider image annotation as producing a ranked list of tags, where the relative order of tags provide implicit cues about the relative importance of objects mentioned in the list of tags. Second, we propose an approach that combines CNN and RNN for this problem. The sequential nature of RNN allows our model to implicitly capture the relative importance of tags. Finally, our experiments demonstrate that our proposed method outperforms other baseline methods for this problem.

2 Related work

Image annotation and label relations: Image tagging and annotation is a very active area of research in computer vision. The simplest approach is to consider it as a multilabel classification problem, and learn a classifier to predict the presence/absence of each tag. But this simple approach ignores the fact that image tags are often correlated, e.g. "water" and "boat" tend to be used together in an image. To address this limitation, probabilistic graphical models are often used. Conditional random fields (CRF) [14] and structural SVM (SVM-struct) [23] are two commonly used models for image annotation. Qi *et al*. [18] use CRF to model the correlative relations of labels for video annotation. Desai *et al*. [5] use SVM-struct to model multi-object layout in object detection. Lan *et al*. [15] develop a SVM-struct model to capture the structured preference among tags for image tag ranking. Deng *et al*. [4] propose a label relation graph to model the hierarchy and exclusion relations among labels.

Predicting visual importance in images: For a given image, the tags associated with it are not equally important. Usually, tags mentioned first correspond to objects that are semantic important in the image. Recent computer vision research starts to understand the relative importance of visual content in an image and exploits it in various applications. Hwang and Grauman [9] discover the relationship between image tags and their visual importance through statistical correlation analysis and use it to improve image retrieval. Hwang and Grauman [10] exploit the order of tags as additional cues to improve object detection. Berg et al. [1] take a human-centric perception perspective to the visual importance problem, and explore a number of handcrafted factors (e.g. composition, semantics, context) and use those factors to predict what will be described about an image by human annotators. Lan and Mori [15] formulate the problem of visual importance prediction as a ranking problem, and learned a discriminative model with a max-margin formalism.

Recurrent neural networks (RNN): Our proposed approach uses recurrent neural networks (RNN) to model the relative order of tags. RNN is a powerful model for sequence modeling. Through recurrent state update, higher order label relations can be implicitly encoded. In recent years, RNN with long short term memory (LSTM) has been successfully applied in a wide variety of applications. For example, Sutskever *et al*. [21] use RNNs to translate sentences from one language to another language. Karpathy *et al*. [12] use RNNs to generate sentence descriptions of images. Similar techniques have been used in video captioning [17], where a hierarchical RNN structure is used to capture nonuniform-length topic transition in videos. Higher dimensional RNNs also exist. For example, Theis and Bethge [16] use a 2-dimensional RNN to encode spatial correlations of image pixels and train the model for texture synthesis.

3 Background

Our proposed approach is based on recurrent neural network with long-short term memory. In this section, we briefly review the background on these models.

3.1 Recurrent neural networks

Recurrent neural network (RNN) [19, 25] is an extension of standard feed-forward neural network for modeling sequences. Let $x_t \in \mathbb{R}^D$ be the input to an RNN at time t. Given a

sequence of T inputs $(x_1, x_2, ..., x_T)$ (where $x_t \in \mathbb{R}^D$), RNN assigns a sequence of hidden states $(h_1, h_2, ..., h_T)$ (where $h_t \in \mathbb{R}^H$) corresponding to the input at each time step. The hidden state h_t each time t is calculated based on the input x_t at current time t and the hidden state h_{t-1} at previous time $t-1$:

$$h_t = \tanh(W_{hh}h_{t-1} + W_{hx}x_t + b_h) \tag{1}$$

where W_{hh}, W_{hx} and b_h are parameters in the RNN model and tanh is a commonly used activation function in RNN.

For many applications, we also have an output (e.g. a prediction label) at each time. Suppose each output is a discrete label with K possible values, the hidden state h_t can be mapped to predict the output as follows:

$$z_t = W_{zh}h_t \tag{2}$$
$$y_t = \text{softmax}(z_t) \tag{3}$$

where $z_t, y_t \in \mathbb{R}^K$ and y_t is a vector indicating the probability of picking each of the K classes.

3.2 Long-short term memory

In principle, RNN can be learned using standard backpropagation in neural networks. But in practice, the learning of RNN suffers from the vanishing gradient problem [2]. This makes it very difficult to train RNN that captures the long-term dependencies of inputs. To address this limitation, the most popular solution is to use a variant of the RNN known as the long-short term memory (LSTM) [8].

LSTM defines a more complex memory cell at each time step. Each memory cell contains an internal state $\mathbf{c}_t$ that stores information about inputs up to time t. LSTM also has three types of gates (input gate i_t, forget gate f_t, output gate o_t) that control how information enters and leaves each cell. The input gate i_t controls the degree to which LSTM will allow the current input x_t to influence the hidden state h_t. The forget gate f_t modulates the influence of previous hidden state h_{t-1} to current hidden state h_t (i.e. how much to forget about previous hidden state). The output gate o_t controls how much information is transferred from the memory cell to the hidden state at current time. Specifically, the hidden state h_t in a LSTM model is computed as follows:

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1}) \tag{4}$$
$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1}) \tag{5}$$
$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1}) \tag{6}$$
$$\tilde{c}_t = \tanh(W_{\tilde{c}x}v + W_{\tilde{c}h}h_{t-1}) \tag{7}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{8}$$
$$h_t = o_t \odot \tanh(c_t) \tag{9}$$

where σ is the sigmoid function and $\odot$ is the element-wise multiplication. Using these gating mechanism, LSTM can capture longer dependencies by selectively forgetting previous states or ignoring current inputs.

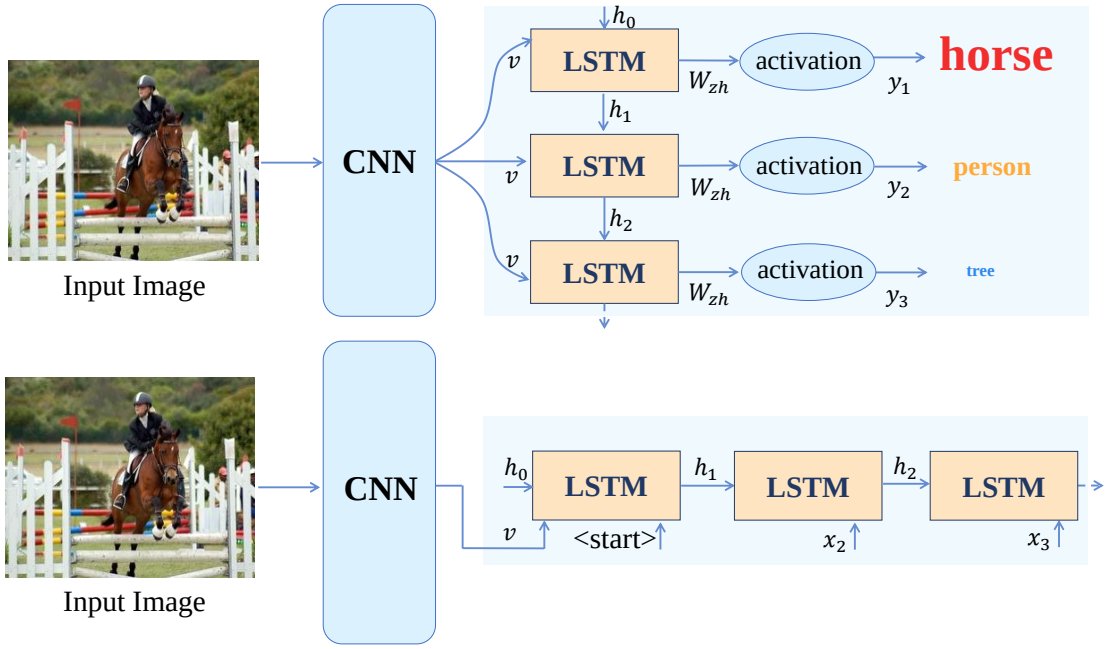


Figure 2: Illustration of the LSTM model. (Top) In our model, the image feature is used as an input to the LSTM at each time step. (Bottom) In the LSTM model used for image captioning (e.g. [12]), the image feature if fed in addition of a default START token to the initial state in the LSTM model, the subsequent states takes the output of the previous state as input. The first hidden state h_0 is initialized to zero.

4 Our Approach

The goal of our work is to generate an ordered tag list given an image. The tags correspond to objects appearing in the image. The order of the tags captures the relative visual importance of the objects. That is, more visually salient objects appear on the top of the list.

Our proposed model combines the convolutional neural network (CNN) for images representation and the LSTM for sequential tag list modeling (Figure 2). This model is inspired by recent work of image captioning using RNN (e.g. [6, 12, 24, 26]). The distinction is that in image captioning, the image feature only directly modulates the starting state of the RNN for caption generation. Once the first word of the caption is generated, the remaining words are generated purely based on previous hidden states of the RNN. For sentence generation, this makes more sense since the words in a sentence tend to have strong dependencies, so it is reasonable to feed the image feature only to the initial state. Notably, Karpathy et al. [12] reported it performs better to feed the image feature to the first state only than to every time step.

However, we find this is not the case for our application. One explanation is that while the words in a sentence tend to have strong dependencies – so it is possible to predict the next word based on previous words in a sentence, the words in a tag list have much looser dependencies, so it turns to be insufficient for the traditional RNN architecture to predict the next tag in the list purely based on previous tags. To address this issue, we modify the RNN model so that the memory cell takes the image feature as one of its inputs at each time step. Another modification is that we have dropped the output of the previous state (y_{t-1}) as one of the inputs to the hidden state at current time step. So every state takes input from the image and the previous hidden state. Fig. 2 shows an illustration of our model and that of a traditional RNN model for the image captioning application.

Image representation: Following prior work (e.g. [12]), we represent an image as a 4096-dimensional CNN feature vector using pre-trained VGGNet. We then use a fully connected layer to reduce the dimension to h. In other words, given an input image I, we represent it as a h-dimensional feature vector as:

$$v = W_I \cdot CNN(I) + b_I \tag{10}$$

where $W_I \in \mathbb{R}^{h \times 4096}$ and $b_I \in \mathbb{R}^h$ are the parameters to be learned. $CNN(I)$ is the 4096-dimensional CNN feature extracted on the image I.

LSTM for tag list prediction: We modify the standard LSTM, so that the hidden state at each time step considers the image feature I as one of the inputs. In other words, our LSTM model is defined as follows:

$$i_t = \sigma(W_{ix}v + W_{ih}h_{t-1}) \tag{11}$$
$$f_t = \sigma(W_{fx}v + W_{fh}h_{t-1}) \tag{12}$$
$$o_t = \sigma(W_{ox}v + W_{oh}h_{t-1}) \tag{13}$$
$$\tilde{c}_t = \tanh(W_{\tilde{c}x}x_t + W_{\tilde{c}h}h_{t-1}) \tag{14}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{15}$$
$$h_t = o_t \odot \tanh(c_t) \tag{16}$$

At each time step t, we need to predict a tag from a vocabulary of size V. We use another linear layer to project the hidden state h_t into a vector of dimension V, followed by a softmax operator. This will give us the probability of choosing each of the V possible tags as the predicted tag at time t:

$$z_t = W^{(z)}h_{(t)} + b^{(z)} \tag{17}$$
$$p_{t,l} = \frac{\exp(z_{t,l})}{\sum_{k=1}^{V}\exp(z_{t,k})} \tag{18}$$

where $\mathbf{z}_t \in \mathbb{R}^{\Omega}$, and $p_{t,l}$ denotes the probability of picking the l-th tag in the vocabulary as the predicted tag at time t.

Model learning: Let I be an image in the training set, and $\mathbf{y} = [y_1, y_2, .., y_T]^\top$ be the corresponding order tag list of length T. We define the following loss function on this training instance:

$$\ell(I, y) = -\sum_{t=1}^{T}\sum_{l=1}^{V}\mathbb{1}(y_t = l)\cdot\log(p_{t,l}) \tag{19}$$

The loss on the whole training set is simply the summation of the loss on each training instance. The parameters of the model is learned by minimizing the loss function using stochastic gradient descent. We do not explicitly use a regularization term in Eq. 19.

5 Experiment

This section describes the experimental evaluation of our proposed method for image tag list prediction on two standard datasets. We first describe our experimental setup (Sec. 5.1). We then introduce the two datasets: the PASCAL2007 dataset and the LabelMe dataset (Sec. 5.2). Lastly we show results and discussion.

5.1 Experiment setup

Implementation details: We use the VGGNet pretrained on ImageNet to extract a 4096-dimensional CNN feature on an input image. The CNN feature is projected to $d = 100$ dimensions by a fully connected layer before being fed into each LSTM cell. The dimension of a hidden state in LSTM is 300 and the output size is the same as the vocabulary size of tags. Our model is implemented in Torch [22]. We train the model using a batch size of 16 and a dropout rate of 0.5. We use the Adam method [13] to optimize the net, with alpha 0.8, beta 0.999 and epsilon 10^{-8} in the Adam method. The initial learning rate is set to be $4 * 10^{-4}$ and is reduced by half when the loss reaches a plateau. After 7500 iterations, we start to do fine-tuning the VGGNet with the learning rate 10^{-5} and the weight decay 10^{-3}.

Evaluation metrics: We evaluate the image annotation results using the two evaluation metrics used in previous work [15]. The first is the Normalized Discounted Cumulative Gain at top k ($NDCG@k$) [3] defined as follows:

$$NDCG@k = \frac{1}{Z} \sum_{p=1}^{k} \frac{2^{s(p)} - 1}{log(1+p)} \tag{20}$$

where k is called the truncation level, Z is the normalization term, varying based on query, $s(p)$ is the relevance at the position p. Due to the normalization term Z, the score ranges from 0 to 1. $NDCG$ is a widely used performance metric in information retrieval. Intuitively, the truncation level k is the number of instances that users will look through before giving up. $NDCG@k$ measures the ranking quality of the top-k instances by a system, while emphasizing the instances ranked higher.

Another metric we use is the $Precision@k$ [15] defined as:

$$Precision@k = \frac{\text{\# of true positives in first k instances}}{k} \tag{21}$$

Baselines: We compare our proposed approach with several baseline methods.

1. SVM: This method learns a binary linear SVM classifier for each tag. Then the tags are ranked based on their SVM scores.

2. SVM-rank: SVM-rank [11] uses a max margin formulation to learn a ranking function to rank all the possible tags.

3. Naive-RNN: This is the same RNN model used in most image captioning work. The hidden state of the RNN at time t is only influenced by the previous hidden state, and does not take the image feature as the input (except for the 1st time stamp).

method	NDCG	Precision
SVM	0.6698	0.4631
SVM-rank	0.6731	0.4471
Naive-RNN	0.6367	0.4056
Ours	**0.7061**	**0.5081**

(a) PASCAL2007

method	NDCG	Precision
SVM	0.4650	0.5277
SVM-rank	0.4715	0.5172
Naive-RNN	0.4204	0.5048
Ours	**0.4804**	**0.5463**

(b) LabelMe

Table 1: Averaged *NDCG@k* and *Precision@k* numbers for $k = 1, 2, ..., 10$ on the two datasets by our method (bottom row) and the three baseline methods. Our method consistently produces the best results.

5.2 Dataset

We test our method on the following two benchmark datasets, the ground-truth ranked tag lists of the two datasets are provided by Hwang *et al*. [10]:

PASCAL2007: This dataset contains 9963 images from the PASCAL2007 dataset [7]. We remove all duplicate tags in the dataset. We also ignore tags appearing less than 15 times on the training set. In the end, we end up with 217 tags in total. We split the dataset as follows: 8500 for training, 500 for validation and 963 for testing.

LabelMe: This dataset contains 3825 images from the LabelMe dataset [20]. Again, we remove duplicate tags. We also ignore tags appearing less than 10 times on the training set. In the end, we have 97 tags on this dataset. We split the dataset as follows: 1800 for training, 200 for validation and 1825 for testing.

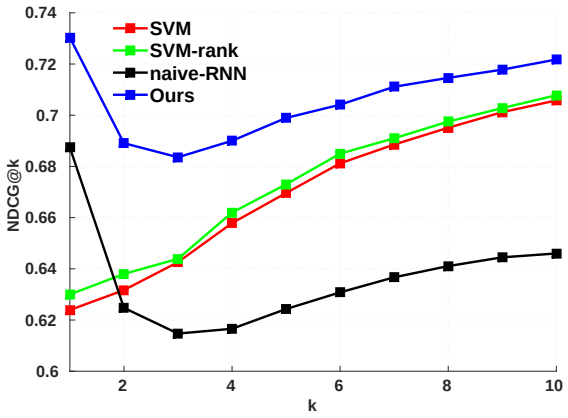
5.3 Results and discussions

Table 1 shows the average *NDCG@k* and *Precision@k* numbers for the four methods on the two datasets. And Fig. 3 (a,c) shows the results in terms of *NDCG@k* on the two datasets for different values of k. Fig. 3 (b,d) shows the results in term of *Precision@k*. The results show that our method outperforms the baseline methods. We can see that SVM and rank-SVM perform similarly to each other. Naive-RNN performs much worse than other methods. This is probably because Naive-RNN is designed for image captioning, where the words in a caption tend to have very strong dependencies. In comparison, the dependencies of tags in a tag list are not as strong as words in a sentence, so the RNN architecture (i.e. Naive-RNN) used for image captioning does not work well in our application.

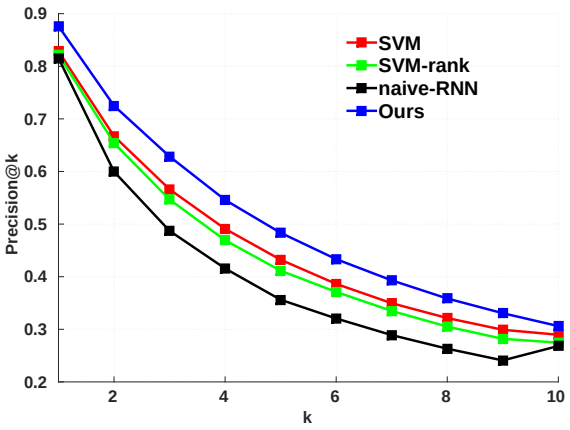
We show some qualitative results in Fig. 4. Compared with SVM, our result is better at putting more important objects in the scene at the top of the predicted tag list. For example, the horse of the first image in the top row, and the cat of the third image top row. Interestingly, for the last image in the bottom row, our method incorrectly classifies the cow as a horse, but still recognizes its visual importance and puts the object at the top of the tag list.

6 Conclusion

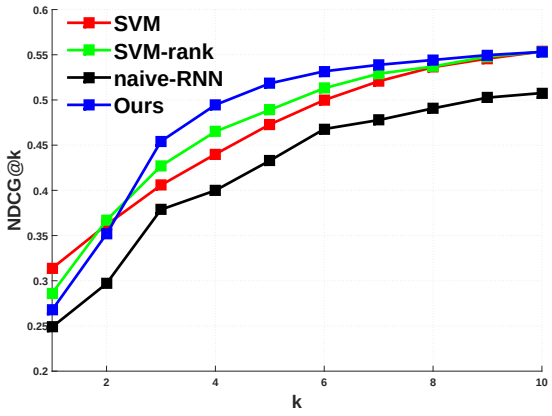
We have introduced a method to produce image annotations and rank them based on their relative visual importance using RNN with LSTM. The RNN model naturally exploits the implicit visual importance cues to boost image annotations, and the resultant ranked tag list is much more informative and useful than traditional order-less image annotation methods. We have evaluated our method on two benchmark dataset. Our experimental results show the superiority of our method over alternative approaches.



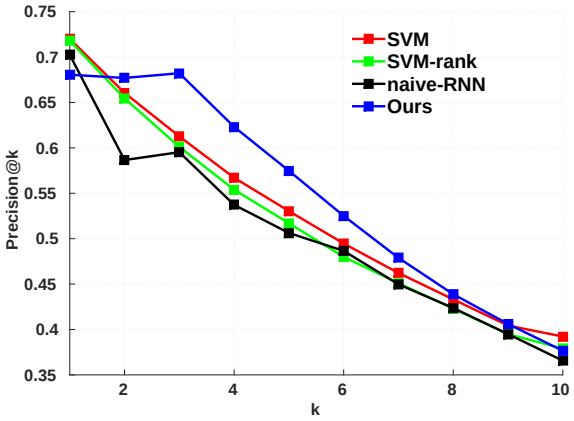
(a) *NDCG@k* on PASCAL



(b) *Precision@k* on PASCAL



(c) *NDCG@k* on LabelMe



(d) *Precision@k* on LabelMe

Figure 3: Comparison of our proposed approach with several baselines. On each dataset, we show the following results: (left column) the *NDCG@k* metric for different *k* values; (right column) the *Precision@k* metric for different *k* values.

Ground-truth: horse, person, tree
Ours: horse, person, hurdle
SVM: hurdle, house, person

Ground-truth: clothes, dog, person
Ours: person, dog, tree
SVM: person, tree, pottedplant

Ground-truth: cat, building, tree
Ours: cat, tree, building
SVM: tree, fence, cat

Ground-truth: bird, tree
Ours: bird, tree, grass
SVM: bird, pottedplant, tree

Ground-truth: train, railway, person
Ours: train, person, tree
SVM: tree, person, train

Ground-truth: train, railway, building
Ours: train, railway, building
SVM: train, railway, grass

Ground-truth: car, tree, grass
Ours: car, tree, grass
SVM: tree, car, road

Ground-truth: cow, person, fence
Ours: horse, person, tree
SVM: grass, tree, fence

Figure 4: Qualitative examples of predicted tag list on the PASCAL2007 dataset by our method and the SVM baseline. Compared with the baseline, our proposed method is better at capturing the relative importance of objects mentioned by the tags.

7 Acknowledgement

We thank all the anonymous reviewers. GY and ZL are supported in part by China 973 program under grant No. 2012CB316405, NSFC grant No. U1509206, and ZJNSF grant No. Q15F020006. YW is supported in part by NSERC.

References

[1] A. C. Berg, T. L. Berg, H. Daume 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 *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[2] Y. Bngio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 1994.

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

[4] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam. Large-scale object classification using label relation graphs. In *European Conference on Computer Vision*, 2014.

[5] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. In *IEEE International Conference on Computer Vision*, 2009.

[6] J. Donahue, L. A. Hendrikcs, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[7] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88 (2):303–338, 2010.

[8] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 1997.

[9] S. J. Hwang and K. Grauman. Accounting for the relative importance of objets in image retrieval. In *British Machine Vision Conference*, 2010.

[10] S. J. Hwang and K. Grauman. Reading between the lines: Object localization using implicit cues from image tags. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[11] T. Joachims. Optimizing search engines using clickthrough data. In *ACM SIGKDD*, 2002.

[12] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[13] D. Kingma and J. Ba. Adam: A method for stochastic optimization. arXiv:1412.6980.

[14] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, 2001.

[15] T. Lan and G. Mori. A max-margin riffled independence model for image tag ranking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[16] P. Pan, Z. Xu, Y. Yang, F. Wu, and Y. Zhuang. Generative image modeling using spatial lstms. In *Advances in Neural Information Processing Systems*, 2015.

[17] P. Pan, Z. Xu, Y. Yang, F. Wu, and Y. Zhuang. Hierarchical recurrent neural encoder for video representation with application to captioning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[18] G. Qi, X. Hua, Y. Rui, J. Tang, T. Mei, and H. Zhang. Correlative multi-label video annotation. In *ACM Multimedia*, 2007.

[19] D. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 1986.

[20] B. C. Russell, A. Torralba, K. P. Murphy, and W. Freeman. LabelMe: A database and web-based tool for image annotation. *International Journal of Computer Vision*, 77 (1-3):157–173, 2008.

[21] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Proc. CVPR*, 2014.

[22] Torch. http://torch.ch/.

[23] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Largin margin methods for structured and interdependent output variables. In *Journal of Machine Learning Research*, 2005.

[24] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[25] P. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of IEEE*, 1990.

[26] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, 2015.