Recognizing Scene Viewpoint using Panoramic Place Representation Supplementary Material: Algorithm Analysis

Jianxiong Xiao jxiao@mit.edu
Krista A. Ehinger kehinger@mit.edu
Aude Oliva oliva@mit.edu
Antonio Torralba torralba@mit.edu
Massachusetts Institute of Technology

Abstract

This document provides further analysis of the algorithm proposed in the paper [5]. We first derive the algorithm in a maximum-likelihood framework. Then, we provide an interpretation of the algorithm as Latent Structural SVM. Finally, we relate the algorithm to k-means and EM.

1. Introduction

There is a trade-off between the level of supervision and the difficulty of obtaining labels. Since category labeling is much easier than viewpoint labeling, we propose a two-stage approach to first train the place category classification model with supervision, then train the viewpoint model in a second, unsupervised stage. The procedure at training time is a two-stage process: first we use photos generated from the panoramas to train a multi-class classifier to predict the place category; then we train a model to predict the viewpoints within each category. The procedure at test time is also a two-stage process. First, the place category of the test image is identified using the place category SVM. Next, the viewpoint is predicted using the trained viewpoint SVM for that place category (Equation 4).

For the viewpoint classification, in each place category, assume there are m different viewpoints uniformly placed on the range -180° to $+180^\circ$. We have N panoramas $\{I_i\}_{i=1...N}$ that must be aligned to train the viewpoint predictor for that place category. We denote their -180° direction as corresponding to L_i when they are aligned.

Furthermore, some places have two types of layouts which are 3D mirror images of each other. For example, in a hotel room the bed may be located to the left of the doorway or to the right of the doorway – the spatial layout of the room may be the same, only flipped. By giving the algorithm the freedom to horizontally flip each panoramic image, these two types of layout can be considered as just one layout, and we can train a better model with better alignment. However, in order to avoid adding an artificial symmetry structure to the data, only one of the original panorama and the flipped panorama is allowed to participate in the alignment. We use a binary variable H_i to denote whether or not to flip the panorama I_i for alignment. For each panorama I_i, we use either the original or flipped version, depending on H_i, and then we generate m views $\{I_{ij}(H_i)\}_{j=1...m}$ uniformly from -180° to $+180^\circ$. Our task is to align all $\{I_i\}$, i.e. find $\{L_i\}$ and $\{H_i\}$, by assigning viewpoint labels $L_{ij} \in \{1 \dots m\}$ to all $\{I_{ij}(H_i)\}$.

To denote the circular ordering constraint, we define the function

$$C(k, j) = r(k + j - 1), \tag{1}$$

where

$$r(x) = \mod(x - 1, m) + 1 \tag{2}$$

is used to ensure circular indices. For a configuration $\langle l_1, l_2, \dots, l_m \rangle$, we say that it satisfies the circular ordering constraint if and only if $\exists k \in \{1, \dots, m\}$, such that $l_1 = k = C(k, 1)$, $l_2 = C(k, 2)$, $\dots$, $l_j = C(k, j)$, $\dots$, $l_m = C(k, m)$. This means that it is equivalent to say that the -180° direction of panorama I_i is aligned at k, and the label for $\{I_{ij}(H_i)\}$ is $L_{i1} = k, L_{i2} = C(k, 2), \dots, L_{im} = C(k, m)$.

2. Maximum-likelihood Interpretation

For a panorama I_i, we define the joint probability of label assignment as

$$\begin{aligned} & P(L_{i1}, H_i, I_i|\mathcal{M}) \\ = & \prod_{j=1...m} P(L_{ij} = C(L_{i1}, j), I_{ij}(H_i)|\mathcal{M}), \end{aligned}$$

where there is only one free variable L_{i1} per panorama for viewpoint orientation, to enforce the circular ordering as a hard constraint.

During training, we are looking for the best model $\mathcal{M}^*$ in maximum-likelihood criteria

$$\mathcal{M}^* = \arg\max_{\mathcal{M}} \prod_{i=1...N} P(I_i|\mathcal{M}) \tag{3}$$

where $\{L_i, H_i\}$ are unknown latent variables.

During testing, given a normal view photo I (such as the images in [6, 2, 7]), we obtain the viewpoint

$$L = \arg\max_{j=1...m} P(j, I|\mathcal{M}^*) \quad (4)$$

by using the best learned model $\mathcal{M}^*$.

Given an initial set of assignments or an initial model, we design an iterative refinement technique for training, which alternates between two steps:

Alignment step For each panorama in the training data I_i, we can assign the label to maximize the joint probability

$$\left\langle L_{i1}^{(t)}, H_i^{(t)} \right\rangle = \arg\max_{L_{i1}, H_i} \prod_{j=1...m} P(L_{ij} = C(L_{i1}, j), I_{ij}(H_i)|\mathcal{M}^{(t)})$$

Because we enforce a hard constraint in $P_{\mathcal{C}}$, there are only m non-zero solutions. Therefore, we try all of them and get the best alignment configuration $L_{i1}^{(t)}$ and $H_i^{(t)}$.

Maximization step We obtain the new updated $\mathcal{M}^{(t+1)}$ by maximizing the likelihood

$$\mathcal{M}^{(t+1)} = \arg\max_{\mathcal{M}} \prod_{i=1...N} P(L_{i1}^{(t)}, H_i^{(t)}, I_i|\mathcal{M})$$
$$= \arg\max_{\mathcal{M}} \prod_{i=1...N} \prod_{j=1...m} P(C(L_{i1}^{(t)}, j), I_{ij}(H_i^{(t)})|\mathcal{M}).$$

In practice, we want to make use of a powerful discriminative classifier to train a strong model. We choose to train a m-way classifier for viewpoint, using the photos $\{I_{ij}(H_i^{(t)})\}$ with current label $L_{ij}^{(t)}$. The decision values from the trained classifier can then be transformed into probability values by using a sigmoid function with normalization [4].

The algorithm is deemed to have converged when the alignment no longer changes. To obtain an initial assignment, we could randomly assign an alignment solution to each panorama and then proceed to the training step, thus computing the initial model from randomly-assigned views. However, we have found empirically that, when starting from random alignments, the algorithm usually converges very quickly to a bad local optima. Therefore, we borrow the idea from curriculum learning[3, 1]: start small, learning easier aspects of the task or easier sub-tasks, and then gradually increase the difficulty level. In our case, we can control *curriculum quantity*, by training the model to align just one panorama at first (a trivial case), and adding one more panorama to the training set with every iteration. Since the order in which panoramas are added affects the final model in this scheme, we also control *curriculum difficulty* by greedily choosing the panorama from the remaining training set that has the largest joint probability (as defined in Equation 3), predicted using the current model $\mathcal{M}^{(t)}$. Figure 9(a) in the paper shows an example of different behaviors of these three schemes. With the greedily incrementally scheme, we derive the algorithm presented in the paper in this maximal-likelihood framework.

In practice, instead of using the raw photo I, we use popular image features. For the classifier, we use the One-Vs-Rest Kernelized Support Vector Machine (SVM). We have tested other classifiers, including K-nearest-neighbor, SVM-KNN [10], N-Vs-Rest SVM, and Support Vector Regression (with various methods of formulating the circular nature of the viewpoint label values), but all gave lower empirical performance than the One-Vs-Rest SVM.

3. Interpretation as Latent Structural SVM

Our model and algorithm can be interpretated as Latent Structural SVM and Concave-Convex Procedure (CCCP) [9], with circular ordering as additional constraints. For I_{ij}, denote the scene category as y_{ij}, and $\{L_{ij}, H_{ij}\}$ as latent variables. With this formulation, the CCCP algorithm applied to Structural SVM with latent variables gives rise to a very intuitive algorithm that alternates between imputing the latent variables $\{L_{ij}, H_{ij}\}$ that best explain the training pair (I_{ij}, y_{ij}) and solving the Structural SVM optimization problem while treating the latent variables as completely observed. These correspond to our alignment step and maximization step.

During the maximization step, when solving SVM while treating latent variables as observed, viewpoint estimation within a category could be interpreted as hierarchical scene categorization, where scenes in a cluster only differ through their viewpoints. There are two different ways to train the SVM. Denote that we have n scene categories and m viewpoints in each category. If we train one SVM to be a n-way classifier to classify the scene category first, and then train n SVMs to be an m-way classifier to classify the viewpoints in each category, we obtain our proposed two-stage approach. If we train the SVM to be a $n \times m$-way classifier, we can obtain a one-stage approach to train scene categories and view points simultaneously.

In general, a one-stage approach may be preferred because it has the benefit of performing viewpoint alignment and category classification in the same framework. However, in our case, the two-stage approach has several advantages over the one-stage approach. First of all, the two-stage approach is several orders of magnitude faster[1] than

[1] The one-stage requires training $n \times m$-way classifier in each iteration. Using One-VS-All SVM, we need to train $n \times m \times T$ SVMs with $(n \times N \times m)^2$-size kernel matrices, where T is the number of iterations before convergence, and N is the number of panoramas in each category. But in the two-stage approach, the scene category is independent of latent

the one-stage approach, which would take several months to run on our dataset using a computer cluster with 12 desktop machines. Secondly, because we use non-linear kernelized SVM, the SVM is usually able to represent different categories quite well with non-linear decision boundaries. Finally, the visual differences between place categories are generally high in our dataset, while the visual differences between different viewpoints in the same place category are low. Therefore, in a One-Vs-Rest SVM, if we use the one-stage approach to train a binary SVM to classify one view versus all other views in all categories, the support vectors for the negatives near the decision boundaries are usually photos with the same place category as the positive example. This means that the training is almost equivalent to the two-stage approach, which uses only photos from the same category for viewpoint training.

4. Relation to k-means and EM

Our algorithm is related to k-means, Expectation-Maximization (EM) and Affinity Propagation [8] algorithms, where the "alignment step" corresponds to the assignment or expectation step, and the "maximization step" corresponds to the update in k-means or the maximization step in EM. However, our algorithm is significantly more powerful than k-means and EM. In contrast to these, same as [6], we use a powerful SVM as the discriminative classifier, a state-of-the-art non-linear histogram intersection kernel with vector quantization to get visual words, multiple kernel combination with linear weighted sums, and a carefully designed learning scheme to greatly boost the performance.

References

[1] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *International Conference on Machine Learning, ICML*, 2009. 2

[2] K. A. Ehinger, J. Xiao, A. Torralba, and A. Oliva. Estimating scene typicality from human ratings and image features. In *CogSci*, 2011. 2

[3] J. L. Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 48:71–99, 1993. 2

[4] J. C. Platt. *Probabilities for SV Machines*, pages 61–74. MIT Press, 2000. 2

[5] J. Xiao, K. A. Ehinger, A. Oliva, and A. Torralba. Recognizing scene viewpoint using panoramic place representation. In *CVPR*, 2012. 1

[6] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010. 2, 3

[7] J. Xiao and L. Quan. Multiple view semantic segmentation for street view images. In *ICCV*, 2009. 2

[8] J. Xiao, J. Wang, P. Tan, and L. Quan. Joint affinity propagation for multiple view segmentation. In *ICCV*, 2007. 3

[9] C.-N. J. Yu and T. Joachims. Learning structural svms with latent variables. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 1169–1176, New York, NY, USA, 2009. ACM. 2

[10] H. Zhang, A. C. Berg, M. Maire, and J. Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *In CVPR*, 2006. 2

variables, and only needs to be trained once. Therefore, we only need to train n SVMs with $(n \times N \times m)^2$-size kernel matrices, and $n \times T$ SVMs with $(N \times m)^2$-size kernel matrices.