

Recent Hot Machine Learning Hammers used in Computer Vision

Alireza Fathi
College of Computing
Georgia Institute of Technology
afathi3@cc.gatech.edu

1. Introduction

I will describe a few techniques that are widely used in computer vision these days and I consider them hot methods. There is no standard tool for measuring whether a method is hot or not, and this is solely based on my personal judgement as a researcher who is attending computer vision conferences and is actively publishing in this area. I consider a method to be hot when in one year it appears in tens of papers, and one can observe researchers talking about it during the main conferences of the field. In the following sections I describe three class of methods that I believe are hot these days. Note that hundreds of papers use the method presented in each of these sections and it is impractical to mention all of them in this article. As a result I only mention a few papers that best represent each of the methods.

2. Hidden Mid-level Variables

Using mid-level cues for detecting objects, recognizing activities and classifying images is not a new idea. Back in early 2000s and maybe much earlier, researchers understood that it is crucial to model objects as a constellation of parts. One would detect an object by simultaneously detecting its parts and verifying that their relative spatial location matches the training examples [11, 10].

In these methods either the parts are labeled for training and from these labels a model is learned for each part, or it is assumed that each part corresponds to a single patch or feature in the image. In the former case, the main issue is that it is impractical to get parts annotated for every object example. In the latter case, the part models are not descriptive enough. In addition, these models were generative and one would prefer discriminative models that do not necessarily learn distributions on data generation but rather achieve a better result.

Quattoni et al. [17, 18] introduced Hidden Conditional Random Fields (HCRF) for both object detection and gesture recognition. Their model has three layers. In the bottom layer are the low-level features or patches. In the top layer are the object or activity labels. In the middle layer, there exist latent (hidden) variables such as parts in case of objects or states in case of activities. A latent variable is neither observed during training nor during the test. For example, in case of objects, one is given a set of images with features (bottom level) and object labels (top level), but parts and their labels are unknown. The hypothesis is that assigning sets of low-level features to parts and then describing the objects as a set of parts and their relative pose will improve the detection results. An illustration is shown in Fig 1.

Fathi and Mori [8] used a similar approach for recognizing actions. They use optical flow vectors as low-level features and learn mid-level motion features in small (x, y, t) windows from these features using Adaboost algorithm. In the next stage, they learn action classifiers from mid-level features. Felzenszwalb et al. [12] propose a framework for detecting objects using deformable part models. In their approach, each object consists of a set of latent parts. The scale, location and orientation of the parts are unknown. They use latent-SVM to learn their model. This paper [12] is highly cited and has inspired many computer vision methods in the last couple of years.

It can be shown that the Latent SVM model is identical to HCRF, however, in Latent SVM a max-margin criteria is used for optimization while HCRF is probabilistically optimized [13, 22]. Further, Latent SVM is identical to MI-SVM formulation of Multiple Instance Learning (MIL) in [2].

Here we describe the Max-Margin HCRF [22] model which is identical to Latent SVM. Given the feature vectors \mathbf{x} of an image, its label y , and part labels \mathbf{h} , an HCRF is defined as

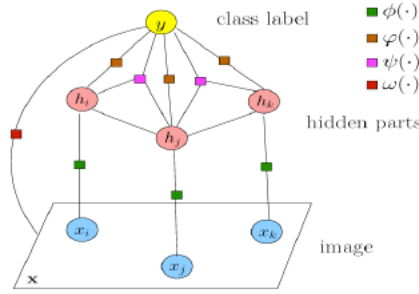


Figure 1. Hidden part model (taken from [22]).

$$p(y, \mathbf{h}|\mathbf{x}; \theta) = \frac{\exp(\theta^T \cdot \Phi(\mathbf{x}, \mathbf{h}, y))}{\sum_{\hat{y} \in \mathcal{Y}} \sum_{\hat{\mathbf{h}} \in \mathcal{H}^m} \exp(\theta^T \cdot \Phi(\mathbf{x}, \hat{\mathbf{h}}, \hat{y}))}$$

where θ is the model parameter, $\Phi(\mathbf{x}, \mathbf{h}, y)$ is a feature vector capturing the unary and binary potentials between nodes, and \mathcal{H}^m is the set of all possible labelings of hidden parts. The hidden part labels \mathbf{h} are unknown both during the training and during the test. For each image we are interested in calculating the $p(y|\mathbf{x}; \theta)$:

$$p(y|\mathbf{x}, \theta) = \sum_{\mathbf{h} \in \mathcal{H}^m} p(y, \mathbf{h}|\mathbf{x}; \theta)$$

HCRF is trained using a probabilistic framework, however, Max-Margin HCRF is trained by maximizing the margin as below:

$$\begin{aligned} \min \quad & \frac{1}{2} \|\theta\|^2 + C \sum_{t=1}^N \xi^{(t)} \\ \text{s.t.} \quad & \max_{\mathbf{h}} \theta^T \Phi(\mathbf{x}^{(t)}, \mathbf{h}, y) - \max_{\mathbf{h}'} \theta^T \Phi(\mathbf{x}^{(t)}, \mathbf{h}', y^{(t)}) \leq \xi^{(t)} - \delta(y, y^{(t)}), \forall t, y \end{aligned}$$

where

$$\delta(y, y^{(t)}) = \begin{cases} 1 & y \neq y^{(t)} \\ 0 & y = y^{(t)} \end{cases}$$

This objective is optimized by coordinate descent:

1. Holding θ, ξ fixed, optimize the latent variables \mathbf{h}' for the pair $(\mathbf{x}^{(t)}, y^{(t)})$:

$$\mathbf{h}_{y^{(t)}}^{(t)} = \arg \max_{\mathbf{h}'} \theta^T \Phi(\mathbf{x}^{(t)}, \mathbf{h}', y^{(t)})$$

2. Holding the $\mathbf{h}_{y^{(t)}}^{(t)}$ fixed, optimize θ, ξ by solving the following optimization problem:

$$\begin{aligned} \min \quad & \frac{1}{2} \|\theta\|^2 + C \sum_{t=1}^N \xi^{(t)} \\ \text{s.t.} \quad & \max_{\mathbf{h}} \theta^T \Phi(\mathbf{x}^{(t)}, \mathbf{h}, y) - \max_{\mathbf{h}'} \theta^T \Phi(\mathbf{x}^{(t)}, \mathbf{h}_{y^{(t)}}^{(t)}, y^{(t)}) \leq \xi^{(t)} - \delta(y, y^{(t)}), \forall t, y \end{aligned}$$

The optimization problem in step 1 can be efficiently solved for certain graph structures like trees.

3. Multiple Instance Learning

Reducing the amount of required supervision is a popular topic in computer vision, given the expense of labeled image data. Recent works have tried to provide different sources of automatic weakly supervised annotations by using web data [3] or cheap human annotation systems such as Amazon’s Mechanical Turk. Others have studied probabilistic clustering methods such as pLSA and LDA for unsupervised discovery of object topics from unlabeled image collections [20].

Unsupervised methods are not necessarily appropriate for learning object categories, since they have no guarantee of finding topics corresponding to object classes. An alternative approach is to expand a small set of labeled data to the unlabeled instances using semi-supervised learning. For example, Fergus et. al [14] leverage the semantic hierarchy from WordNet to share labels between objects.

More recently, Multiple Instance Learning (MIL) has shown great promise as a method for weakly supervised learning in computer vision communities [2, 6, 9, 7, 15, 21]. In MIL, labels are provided for bags containing instances (e.g. images containing objects). These information are then leveraged to classify instances and bags. Buehler et al. [6] localize signs in footage recorded from TV with a given script. Vijayanarasimhan and Grauman [21] learn discriminative classifiers for object categories given images returned from keyword-based search engines. Fathi et al. [9] learn object instances from weakly supervised information provided for daily activities.

In the MIL framework, different than typical learning where labels are assigned to examples, instances are grouped into bags and labels are provided for bags. If a bag has a positive label, it means there is at least one instance in it that belongs to the positive class, but there are usually many instances in the bag that belong to the negative class. If a bag is labeled negative, it means all the examples in that bag are negative. Given the description of the MIL, it is easily possible to observe that it is a good fit to many computer vision problems. For example, in case of object detection, an image can represent a bag, and all the possible bounding boxes in that image represent the instances. If the object of interest does not exist in the image, all the bounding boxes belong to the negative class. If the object of interest exists in the image, one or a few of the bounding boxes will correspond to the object of interest, but still most of the bounding boxes belong to the background region.

There are various approaches developed for Multiple Instance Learning. Here we describe the approach developed by Fathi et al. [9] which extends the framework of [7] to handle multiple-classes simultaneously.

Chen et. al [7] extend ideas from the diverse density framework to solve the MIL problem. Fathi et al. [9] further extend their method to (1) handle multiple instance labels simultaneously and (2) apply mutual equality constraints among some instances in each bag. They find a similarity measure between every instance x_i and every bag B_j , using the following equation

$$\begin{aligned} Pr(x_i|B_j) &= s(x_i, B_j) \\ &= \max_{x_k \in B_j} \exp\left(-\frac{\|x_i - x_k\|^2}{\sigma^2}\right) \end{aligned}$$

where σ is a pre-defined scaling factor. Given m instances in total and l bags, the following $m \times l$ matrix is built:

$$\begin{bmatrix} s(x_1, B_1) & \dots & s(x_1, B_l) \\ s(x_2, B_1) & & s(x_2, B_l) \\ \vdots & \ddots & \vdots \\ s(x_m, B_1) & \dots & s(x_m, B_l) \end{bmatrix} \quad (1)$$

In this matrix each row corresponds to similarities of an instance to all the bags, and each column captures the similarity of each bag to all the instances. Their objective function is to find a sparse set of instances corresponding to each object category which have a high similarity to positive bags and a low similarity to negative bags.

Fathi et al. [9] extend the formulation of [7] to infer multiple instance classes simultaneously. Instead of minimizing for a single vector w , they look for r sparse vectors w_1, w_2, \dots, w_r where each w_c is m dimensional and is positive at a few representative instances of object class c and is zero every where else.

They further add equality constraints $w_c(p) = w_c(q)$ between a pair of elements (p, q) in all $w_c, c = \{1, \dots, r\}$ if there is a temporal link between regions corresponding to instances p and q . Finally, they optimize the following linear program which minimizes the L1-norm of w_c vectors and returns sparse vectors:

$$\min_{w,b,\xi} \left\{ \sum_{c=1}^r |w_c| + C \sum_{j=1}^l \xi_j \right\} \quad (2)$$

$$w_c \cdot s(:,j) \geq w_{c'} \cdot s(:,j) + \delta_{c,c',B_j} - \xi_j \text{ s.t. } \forall c, c' = \{1, \dots, r\}$$

$$w_c(p) = w_c(q) \text{ s.t. } \forall c = \{1, \dots, r\}, (p, q) \in \mathcal{C}$$

$$\xi_j \geq 0 \text{ s.t. } \forall j = \{1, \dots, l\}$$

where ξ_j is slack variable for bag j , C is a constant, $s(:,j)$ contains the similarity vector of instances to bag j , δ_{c,c',B_j} is 1 if bag B_j is positive for class c and negative for class c' and 0 otherwise, and \mathcal{C} contains the set of equality constraints between instances.

4. Random Forests

The idea behind random forest was first introduced in Amit and Geman [1]. In this work, all the details for random forests as is used these days is described. Breiman [5] shows that the generalization error of random forests converges to a limit when the number of trees get big. The generalization error is dependent on the strength of the trees and the correlation between them. Even though these papers are heavily cited, random forests were not widely used in computer vision until recently. In 2005, Lepetit et al. [16] used randomized trees for keypoint recognition. Bosch et al. [4] use random forests for image classification. However, this method became popular after it was used in Shotton et al. [19] for human pose estimation from depth images. This paper describes the machinery behind Microsoft's new Xbox and Kinect game system. Their framework performs faster than real-time and basically shows the capability of random forests in learning from huge amount of data and running very fast at classification. We provide the algorithm for building a random tree below. A random forest is a collection of random trees.

1. Randomly propose a set of splitting candidates $\phi = (\theta, \tau)$ where θ contains feature parameters and τ are thresholds
2. Partition the set of examples Q into left and right subsets by each ϕ :

$$\begin{aligned} Q_l(\phi) &= \{x | f_\theta(x) < \tau\} \\ Q_r(\phi) &= \{x | f_\theta(x) \geq \tau\} \end{aligned}$$

3. Compute the best ϕ leading to the minimum entropy:

$$\begin{aligned} \phi^* &= \arg \max G(\phi) \\ G(\phi) &= H(Q) - \frac{|Q_l(\phi)|}{|Q|} H(Q_l(\phi)) - \frac{|Q_r(\phi)|}{|Q|} H(Q_r(\phi)) \end{aligned}$$

where $H(Q)$ is the Shannon entropy.

4. If the largest gain $G(\phi^*)$ is big enough and the depth of the tree is below a maximum, then recurse by partitioning left and right subsets of the data.

At each node n of the tree, the $p(y|n, \mathbf{x})$ is stored, which contains probability of data point \mathbf{x} belonging to class $y = c$ if feeding \mathbf{x} to the tree will arrive at node n . Usually a histogram is stored at each node by counting how many times a data point belonging to class c visits that node. The final probability is computed by averaging over nodes of the tree, and is computed for the whole forest by averaging over all the trees.

Random forests are one of the most accurate learning algorithms available, run efficiently on large datasets, give an estimate of what variables are important in the classification and can run very fast. On the other hand, random forests have been observed to overfit on some datasets with noisy classification tasks.

5. Conclusion

One can notice all the mentioned methods have one theme in common and that is their capability in handling either huge amount of data or little amount of annotation. In computer vision, these days millions of images are available for each recognition domain. One big challenge is annotating all the data. As a result, methods that use weak supervision such MILs and Latent SVM are becoming popular. These methods use machine learning methods for recovering the label of missing data. On the other hand, methods such as SVM and Adaboost are not efficient enough for learning from millions of examples and in addition, it is not easy to parallelize them. This has led the researchers to use random forests.

References

- [1] Y. Amit and D. Geman. Shape quantization and recognition with randomized trees. In *Neural Computation*, 1997. 4
- [2] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *NIPS*, 2003. 1, 3
- [3] T. L. Berg and D. A. Forsyth. Animals on the web. In *CVPR*, 2006. 3
- [4] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *ICCV*, 2007. 4
- [5] L. Breiman. Random forests. In *Machine Learning*, 2001. 4
- [6] P. Buehler, M. Everingham, and A. Zisserman. Learning sign language by watching tv (using weakly aligned subtitles). In *CVPR*, 2009. 3
- [7] Y. Chen, J. Bi, and J. Z. Wang. Miles: multiple-instance learning via embedded instance selection. In *PAMI*, 2006. 3
- [8] A. Fathi and G. Mori. Action recognition by learning mid-level motion features. In *CVPR*, 2008. 1
- [9] A. Fathi, X. Ren, and J. M. Rehg. Learning to recognize objects in egocentric activities. In *CVPR*, 2011. 3
- [10] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In *CVPR 2004 Workshop on Generative-Model Based Vision*, 2004. 1
- [11] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. In *IJCV*, 2005. 1
- [12] P. Felzenszwalb, D. McAllester, and D. Ramaman. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008. 1
- [13] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and Deva Ramanan. Object detection with discriminatively trained part based models. In *PAMI*, 2010. 1
- [14] R. Fergus, H. Bernal, Y. Weiss, and A. Torralba. Semantic label sharing for learning with many categories. In *ECCV*, 2010. 3
- [15] N. Ikinler-Cinbis and S. Sclaroff. Object, scene and actions: combining multiple features for human action recognition. In *ECCV*, 2010. 3
- [16] V. Lepetit, P. Laguerre, and P. Fua. Randomized trees for real-time keypoint recognition. In *CVPR*, 2005. 4
- [17] A. Quattoni, M. Collins, and T. Darrel. Conditional random fields for object recognition. In *NIPS*, 2005. 1
- [18] A. Quattoni, S. Wang, L-P. Morency, M. Collins, and T. Darrell. Hidden-state conditional random fields. In *PAMI*, 2007. 1
- [19] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth image. In *CVPR*, 2011. 4
- [20] J. Sivic, B. Russell, A. A. Efros, A. Zisserman, and B. Freeman. Discovering objects and their location in images. In *ICCV*, 2005. 3

- [21] S. Vijayanarasimhan and K. Grauman. Keywords to visual categories: multiple-instance learning for weakly supervised object categorization. In *CVPR*, 2008. 3
- [22] Y. Wang and G. Mori. Hidden part models for human action recognition: probabilistics vs. max-margin. In *PAMI*, 2011. 1, 2