Auto-context and Its Application to High-level Vision Tasks

Zhuowen Tu
Lab of Neuro Imaging, University of California, Los Angeles
ztu@loni.ucla.edu

Abstract

The notion of using context *information for solving high-level vision problems has been increasingly realized in the field. However, how to learn an effective and efficient context model, together with the image appearance, remains mostly unknown. The current literature using Markov Random Fields (MRFs) and Conditional Random Fields (CRFs) often involves specific algorithm design, in which the modeling and computing stages are studied in isolation. In this paper, we propose an* **auto-context** *algorithm. Given a set of training images and their corresponding label maps, we first learn a classifier on local image patches. The discriminative probability (or classification confidence) maps by the learned classifier are then used as context information, in addition to the original image patches, to train a new classifier. The algorithm then iterates to approach the ground truth. Auto-context learns an integrated low-level and context model, and is very general and easy to implement. Under nearly the identical parameter setting in the training, we apply the algorithm on three challenging vision applications: object segmentation, human body configuration, and scene region labeling. It typically takes about* $30 \sim 70$ *seconds to run the algorithm in testing. Moreover, the scope of the proposed algorithm goes beyond high-level vision. It has the potential to be used for a wide variety of problems of multi-variate labeling.*

1. Introduction

It has been noted that context and high-level information plays a vital role in object and scene understanding [2, 22]. Yet, a principled way of learning an effective and efficient context model, together with the image appearance, is not available. Context comes into a variety of forms and it can be referred to as Gestalt laws in middle level knowledge, intra-object configuration, and inter-object relationship. For example, a clear horse face may suggest the location of its tail and legs, which are often occluded or not easy to identify. The strong appearance of a car might well suggest the existence of a road, and vice versa [8].

From the Bayesian statistics point view, context is represented by the joint statistics of the multi-variate in the posterior probability, which is decomposed into likelihood and prior. In vision, likelihood and prior are often referred to as appearance and shape respectively. However, there are still many technological hurdles to overcome and the difficulties can be summarized into two aspects: *modeling* and *computing*. (1) Difficulty in modeling complex appearances: Objects in natural images observe complex patterns. There are many factors contributing to the complexity such as texture (homogeneous or inhomogeneous), lighting conditions, and occlusion. (2) Difficulty in learning complicated shapes. (3) Difficulty in computing for the optimal solution: The optimal solution is often considered as the one which maximizes a posterior (MAP), or equivalently, minimizes an energy. Searching for the optimal solution for the combination of the appearance and shape models is an non-trivial task.

From the energy minimization point of view, models like Markov Random Fields (MRFs) [6], conditional Markov Random Fields (CRFs) [9], and computing algorithms such as Belief Propagation (BP) [15, 30], have been widely used in vision [21]. However, these models and algorithms share somewhat similar disadvantages: (1) the choice of functions used are quite limited so far; (2) they usually rely on a fixed topology with very limited neighborhood relation; (3) they are slow in solving many real vision problems, whereas biological vision systems can identify and understand complex objects/scenes very rapidly; (4) they are only guaranteed to obtain the optimal solution for a limited function families. Hidden Markov models (HMM) [12] studies the dependencies of the neighboring states, which is in a way similar to the MRFs. HMM is also limited to short range context information and usually is time consuming in learning the parameters and computing for the solutions.

From the point of view of using context information, there have been a lot of recent work proposed in object recognition and scene understanding [8, 19, 16, 22, 18, 28, 7, 20]. A pioneering work was proposed by Belongie et al. [2] which uses shape context in shape matching.

In this paper, we make an effort to address some of the questions mentioned above by proposing an **auto-context** model. The algorithm targets the posterior distribution di-

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

rectly in a supervised approach. Like in the BP algorithm [30], the goal is to learn/compute the marginals of the posterior, which we also call *classification maps* for the rest of this paper. Each training image comes with a label map in which every pixel is assigned with a label of interest. A classifier is first trained to classify each pixel. There are two types of features for the classifier to choose from: (1) image features computed on the local image patches centered at the current pixel (we use image patches of fixed size 11×11 in this paper), and (2) context information on the classification maps. Initially, the classification maps are uniform, and thus, the context features are not selected by the first classifier. The trained classifier then produces new classification maps which are used to train another classifier. The algorithm iterates to approach the ground truth until convergence. In testing, the algorithm follows the same procedure by applying the sequence of learned classifiers to compute the posterior marginals.

The auto-context algorithm integrates the image appearances (observed data) together with the context information by learning a series of classifiers. Unlike many the energy minimization algorithms where the modeling and computing stages are separated, auto-context uses the same procedures in the two (this is a property of many classifiers since they get close-form solutions). The difference between the results in training and testing is the generalization error of the trained classifiers. Therefore, there is no explicit energy to minimize in auto-context. This alleviates the burden in searching for the optimal solution. Auto-context uses deterministic procedures, but it carries the uncertainties without the need of making any hard decisions. This makes the auto-context algorithm significantly faster than most the existing energy minimization algorithms. Compared to MRFs, CRFs, auto-context no longer works on a fixed neighborhood structure. Each pixel (sample) can have support from a large number of neighbors, either short or long range. It is up to the learning algorithm to select and fuse them. The classifiers in different stages may choose different supporting neighbors to either enhance or suppress the current probability towards the ground truth. Also, the appearance (likelihood) and prior (context and shape) are directly combined in an implicit way and the balance between the two is naturally handled.

Two pieces of work directly related to auto-context are: Boosted Random Fields (BRFs) [22] and SpatialBoost [1], which both use boosting algorithm to combine the contextual information. However, both the algorithms use contextual beliefs as weak learner in the boosting algorithm, which is time-consuming to update. Possibly due to this reason, SpatialBoost was only illustrated on an interactive segmentation task. BRFs learns the message update rules in the belief propagation algorithm, and the main focus of SpatialBoost is to propose an extended algorithm for the boosting algorithm. The auto-context is a general algorithm and the classifier of choice is not limited to boosting algorithms. It directly targets the posterior through iterative steps, resulting in a simpler and more efficient algorithm than BRFs and SpatialBoost. Under nearly the same set of parameters in training, we demonstrate several applications using the auto-context, which are not available in [22, 1]. A feed forward way of including context with appearance was proposed in [28] for object detection. However, their method is not to iteratively learn a posterior. More importantly, their findings lead to the conclusion that the help from context is negligible (unless the image quality is really poor). Our experimental results in Fig. (3.a) suggest it differently. One possible reason might be that the image segmentation/parsing task is different from the object detection problem focused in [28]. Others [16] also showed that explicit context information improves region segmentation/labeling results greatly.

Compared to the traditional Bayesian approach for image understanding [25] , auto-context is much easier to train and it avoids heavy algorithm design. It is significantly faster than many the existing algorithms in this domain. Compared to the algorithms using context [16, 8, 26], it learns an integrated model. There is no hard decision made in the intermediate stages and uncertainties are all carried through posterior marginals.

We demonstrate the auto-context algorithm on challenging high-level vision tasks for three well known datasets: horse segmentation in the Wizemann dataset [3], human body configuration in the Berkeley dataset [13], and scene region labeling in the MSRC dataset [19]. The proposed algorithm is general and very easy to implement. Its scope goes beyond high-level vision tasks. Indeed, it has the potential to be used for many problems for multi-variate labeling where the joint statistics needs to be modeled.

2. Problem formulation

In this section, we give the problem formulation for the auto-context model and briefly discuss some related algorithms.

2.1. Objective

Let the data vector be $X = (x_1, ..., x_n)$. In the case of 2D image, $X = (x_{(i,j)}, (i,j) \in \Lambda)$ where Λ denotes the image lattice. For notational clarity, we do not distinguish the two and call the both 'image'. In training, each image X comes with a ground truth $Y = (y_1, ..., y_n)$ where $y_i \in \{1..K\}$ is the label of interest for each pixel i. The training set is then $S = \{(Y_j, X_j), j = 1..m\}$ where m denotes the number of training images. The Bayes rule says $p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)}$, where $p(X|Y)$ and $p(Y)$ are the likelihood and prior respectively. Often, we look for the optimal solution maximizing a posterior (MAP)

$$Y^* = \arg max\ p(Y|X) = \arg max\ p(X|Y)p(Y).$$

As mentioned before, the main difficulties for the MAP framework come from two aspects. (1) *modeling*: it is very hard to learn accurate $p(X|Y)$ and $p(Y)$ for real-world applications. Both of them have high complexity and usually do not follow independent identical distribution (i.i.d.). (2) *computing*: The combination of the $p(X|Y)$ and $p(Y)$ is often non-regular. Besides many recent advances made in optimization and energy minimization [21], a general and immediate solution yet remains out of reach.

Instead of decomposing $p(Y|X)$ into $p(X|Y)$ and $p(Y)$, we study the posterior directly. Moreover, we look at the marginal distribution $\mathcal{P} = (\mathbf{p}_1, ..., \mathbf{p}_n)$ where $\mathbf{p}_i$, as a vector for discrete labels, denotes the marginal distribution of

$$p(y_i|X) = \int p(y_i, y_{-i}|X)dy_{-i}, \tag{1}$$

where y_{-i} refers to the rest of y other than y_i. This is seemingly a more challenging task as it requires to integrate out all the dy_{-i}. Next, we discuss how to approach this.

2.2. Traditional classification approaches

A traditional way to approximate eqn. (1) is by treating it as a classification problem. Usually, a classifier is considered to be translation invariant. The training set becomes $S = \{(y_{ji}, X_j(N_i)), j = 1..m, i = 1..n\}$. Instead of using the entire image X_j, the training set includes image patch centered at each i, $X_j(N_i)$. N_i denotes all the pixels in the patch. In the context of boosting algorithms, it was shown [5, 4] that one can learn the posterior based on logistic regression

$$p(y = k|X(N)) = \frac{e^{F_k(X(N))}}{\sum_{k=1}^{K} e^{F_k(X(N))}}, \quad \sum_{k=1}^{K} F_k(X(N)) = 0. \tag{2}$$

$F_k(X(N)) = \sum_{t=1}^{T} \alpha_{k,t} \cdot h_{k,t}(X(N))$ is the strong classifier on a weighted sum of selected weak classifier $h_{k,t}$ for label k. The learned posterior marginal, $p(y = k|X(N))$, is a very crude approximation to eqn. (1) and it only uses some context implicitly through image patch $X(N)$. Due to this limitation, the well-known CRFs or Discriminative Markov Random Fields (DRFs) model [9] tries to explicitly include the context information by adding another term $p(y_{i1}, y_{i2}|X(N_{i1}), X(N_{i2}))$. Though CRFs has been successfully applied in many applications [9, 10, 17], it still has the limitations similar to those in the MRFs as discussed in Sect. (1). CRFs still uses fixed neighborhood structure with fairly limited number of connections. The computing complexity explodes on a large neighborhood (clique) structure. This limits their modeling capability and only short-range context is used in most cases (the long-range context model in [10] uses a few connections). Also, it limits their computing capability since the interactions are slowly propagated through pair-wise relations.

2.3. Auto-context

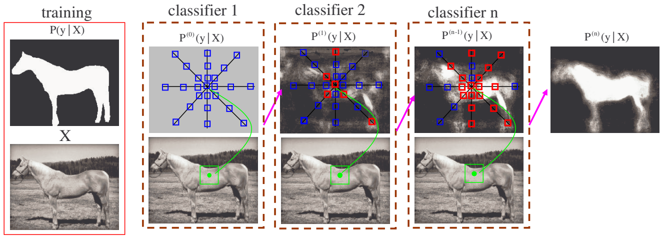


Figure 1. Illustration of the classification map updated at each round for the horse segmentation problem. The red rectangles are those selected contexts in training.

To better approximate the marginals in eqn. (1) by including a large number of context information, we propose an auto-context model. As mentioned above, a traditional classifier can learn a classification model based on local image patches, which now we call

$$\mathcal{P}^{(0)} = (\mathbf{p}_1^{(0)}, ..., \mathbf{p}_n^{(0)})$$

where $\mathbf{p}_i^{(0)}$ is the posterior marginal for each pixel i learned by eqn. (2). We construct a new training set

$$S_1 = \{(y_{ji}, X_j(N_i), \mathcal{P}_j^{(0)}(i)), j = 1..m, i = 1..n\}, \tag{3}$$

where $\mathcal{P}_j^{(0)}(i))$ is the classification map for the jth training image centered at pixel i. We train a new classifier, not only on the features from the image patch $X_j(N_i)$, but also on the probabilities, $\mathcal{P}_j^{(0)}(i))$, of a large number context pixels. These pixels can be either near or very far from i, and Fig. (1) shows an illustration. It is up to the learning algorithm to select and fuse important supporting context pixels, together with features about image appearance. Once a new classifier is learned, the algorithm repeats the same procedure until it converges. The algorithm iteratively updates the marginal distribution to approach

$$p^{(n)}(y_i|X(N_i), \mathcal{P}^{(n-1)}) \rightarrow p(y_i|X) = \int p(y_i, y_{-i}|X)dy_{-i}. \tag{4}$$

In theorem 1 we show that the algorithm is asymptotically approaching $p(y_i|X)$ without doing explicit integration. A more direct link between the two, however, is left for future research.

In fact, even the first classifier is trained the same way as the others by giving it a probability map of uniform distribution. Since the uniform distribution is not informative at all, the context features are not selected by the first classifier. In some particular applications, e.g. medical image segmentation, the positions of the anatomical structures are roughly known. One then can use a probability atlas as the initial $\mathcal{P}^{(0)}$.

Given a set of training images together with their label maps, $S = \{(Y_j, X_j), j = 1..m\}$: For each image X_j, construct probability maps $\mathcal{P}_j^{(0)}$ with uniform distribution on all the labels. For $t = 1, ..., T$:

- Make a training set $S_t = \{(y_{ji}, (X_j(N_i), \mathcal{P}_j^{(t-1)}(i))), j = 1..m, i = 1..n\}$.
- Train a classifier on both image and context features extracted from $X_j(N_i)$ and $\mathcal{P}_j^{(t-1)}(i)))$ respectively.
- Use the trained classifier to compute new classification maps $\mathcal{P}_j^{(t)}(i)$ for each training image X_j.

The algorithm outputs a sequence of trained classifiers for $p^{(n)}(y_i|X(N_i), \mathcal{P}^{(n-1)}(i))$

Figure 2. The training procedures of the auto-context algorithm.

Theorem 1 *The auto-context algorithm monotonically decreases the training error.*

Proof: For notational simplicity, we consider only one image in the training data and use $X(i)$ to denote $X(N(i))$. In the AdaBoost algorithm [5], the error function is taken by $\epsilon = \sum_i e^{-y_i H(X(i))}$ for $y_i \in \{-1, +1\}$, which can be given an explanation as the log-likelihood model [4]. At different steps,

$$\epsilon_t = -\sum_i \log p^{(t)}(y_i|X(i), \mathcal{P}^{(t-1)}(i)),\ and$$

$$\epsilon_{t-1} = -\sum_i \log \mathbf{p}_i^{(t-1)}(y_i),$$

where

$$p^{(t)}(y_i|X(i), \mathcal{P}^{(t-1)}(i)) = \frac{e^{F_k^{(t)}(X(i), \mathcal{P}^{(t-1)}(i))}}{\sum_{k=1}^{K} e^{F_k(t)(X(i), \mathcal{P}^{(t-1)}(i))}}. \tag{5}$$

$F_k^{(t)}(X(i), \mathcal{P}^{(t-1)}(i))$ includes a set of weak classifiers selected for label class k. It is straightforward to see that we can at least make

$$p^{(t)}(y_i|X(i), \mathcal{P}^{(t-1)}(i)) = \mathbf{p}_i^{(t-1)}(y_i)$$

since the equality can be easily achieved by making

$$F_k^{(t)}(X(i), \mathcal{P}^{(t-1)}(i)) = \log \mathbf{p}_i^{(t-1)}(k)).$$

The boosting algorithm (or almost any valid classifier) choose a set of $F_k^{(t)}$ in minimizing the total error ϵ_t, which should at least do better than $\mathbf{p}_i^{(t-1)}(y_i)$ Therefore,

$$\epsilon_t \leq \epsilon_{t-1}. \ \square$$

The convergence rate depends on the amount of error reduced $\epsilon_{t-1} - \epsilon_t$. Intuitively, the next round of classifier tries to select features both from the appearances and the previous classification maps. A trivial solution is to use the previous probability map for the classifier. This also shows that the optimal classifier is at a stable point. Of course, this requires to have the feature of its own probability in the candidate pool, which is not hard to achieve. Fig. (1) gives an illustration of the procedures of the auto-context. Several rays are shot from the current pixel and we sparsely sample the context locations (both individual pixels and windows) to use their classification probabilities as features. Each round of training will select different sets of context pixels, either short range or long-range.

2.4. Understanding auto-context

We first take a look at the Belief Propagation algorithm [15, 30] since it also works on the marginal distribution. For directed graph, BP is guaranteed to find the global optimal. For loopy graph, BP computes an approximation. For a model on a graph

$$p(Y) = \frac{1}{Z} \prod_{(i,j)} \psi(y_i, y_j) \prod_i \phi_i(y_i)$$

where Z is the normalization constant, $\psi(y_i, y_j)$ is the pair-wise relation between sites i and j, and $\phi_i(x_i)$ is a unary term. The BP algorithm [30] computes the belief (marginal) $p_i(y_i)$ by

$$p_i(y_i) = \frac{1}{Z} \phi_i(y_i) \prod_{j \in N(i)} m_{ji}(y_i), \tag{6}$$

where $m_{ji}(x_i)$ are the messages from j to i,

$$m_{ij}(y_j) \leftarrow \sum_{y_i} \phi_i(y_i) \psi_{i,j}(y_i, y_j) \prod_{k \in N(i) \backslash j} m_{ki}(y_i). \tag{7}$$

Similarly, the auto-context algorithm updates the marginal distribution by eqn. (5). The major difference between BP and auto-context are: (1) On the graphical model, every pair of $\psi_{i,j}(y_i, y_j)$ on all possible labels need to be evaluated and integrated in eqn. (7). Therefore, BP can only work with a limited number of neighborhoods to keep the computational burden under check. For auto-context, it evaluates a sequence of learned classifiers, $F_k^{(t)}(X(i), \mathcal{P}^{(t-1)}(i))$, which is computed based on a set of selected features. Therefore, auto-context can afford to look at a much longer range of support and it is up to the learning algorithm to select and fuse the most supportive contexts and appearance information. Also, there is no integration between the pair y_i and y_j. (2) BP works on a fixed graph structure and the update rule is the same. auto-context learns different classifiers on different set of features at different stages, which allows it to make use of the best available set of information each time. (3) In BP, there is often separate stages to design the graphical model, and learn $\psi(y_i, y_j)$ and $\phi_i(y_i)$. Auto-context is targeted to learn the posterior marginal directly and its inference stage follows the identical steps in the learning phase. The difference between the learning and test stages is the generalization error of the classifiers which can be studied by the VC dimension or margin theory [27]. However, BP has the advantage that it uses the same message passing rule for different forms of $p_i(y_i)$ in eqn. (6), whereas auto-context requires to learn a different set of classifiers for different tasks.

A question one might ask is: "How different is it between learning a recursive model $p^{(t)}(y_i|X_i, \mathcal{P}^{(t-1)}(i))$ and learning $p(y_i|X)$ directly?". A classifier can be learned by using the entire image X rather than the image patch $X(i)$. A major issue is that $p(y_i|X)$ should be a marginal distribution by integrating out the other is as shown in eqn. (1). The correlation between different pixels needs to be taken into account, which is hard by learning one classifier for $p(y_i|X)$. It also builds a feature space too big for a classifier to handle and might lead to severe overfitting. Wolf and Bileschi [28] suggested that using label context might achieve the same effect as using image appearance context, in object detection. Moreover, in both the situations where contexts were used, the improvements were small. We conducted an experiment to train a system with image appearance, instead of the probabilities, for the pixels sparsely sampled on the rays, as suggested in [28]. The results are shown in Fig. (3.a) and the conclusions are different from [28] in two aspects: (1) having appearance context even gives worse result than using features from patches only (due to overfitting); (2) label (in probabilities) contexts greatly improve the segmentation/labeling result.

There have been many algorithms along the line of using context [2, 16, 8, 22, 1]. Auto-context makes an attempt to recursively select and fuse context information, as well as appearance, in a unified framework. The first round of classifier is based purely on the local appearance. Objects with strong appearance cues often achieve high probabilities on their labels. These probabilities then start to make influence to the others, if there are strong correlations between them. Themselves are also getting support from the others. During the iterations, the algorithm learns to suppress the strong probabilities, if wrong, and improve the low ones to the target distribution. Context information not only comes from between-objects, they are also from within-objects (parts), even the parts may be far from each other spatially. Auto-context uses a very general and simple scheme by avoiding heavy manual algorithm design.

3. Experiments

We illustrate the auto-context algorithm on three challenging high-level vision tasks: horse segmentation, human body configuration, and scene parsing/labeling. In these three tasks, the system uses nearly an identical parameter setting, mostly generic ones such as the number of weak classifiers and the stopping criterion. The system can be used for a variety of tasks with given training images and label maps.

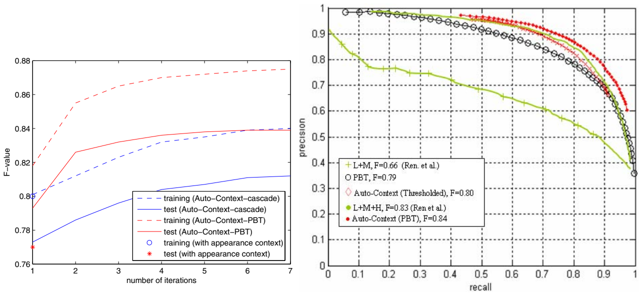
3.1. Horse segmentation

We use the Weizmann dataset consisting of 328 horse images [3] of gray scale. The dataset also contains manually annotated label maps. We split the dataset randomly into half for training and half for testing. The training stage follows the steps described in Fig. (2). Two main implementation issues we have not talked about are the choices of: (1) classifiers, (2) features. Boosting algorithms appear to be a natural choice for the classifier as they are able to select and fuse a set of features from a large candidate pool to approximate the target posterior marginal. However, our algorithm is not tied to any specific classifier and one can choose others such as SVM [27]. Since each pixel is a training sample, there consists of millions of positive and negative ones. It is hard to build a single node boosting algorithm to perform the classification. We adopt the probabilistic boosting tree (PBT) algorithm [23] as it learns and computes a discriminative model in a hierarchical way by

$$\begin{aligned} p(y|X) &= \sum_{l_1} p(y|l_1, X)p(l_1|X) \\ &= \sum_{l_1,..,l_n} p(y|l_n, ..., l_1, X), ..., p(l_2|l_1, x)p(l_1|X), \end{aligned}$$

where $p(l_i|)$ is the classification model learned by boosting node in the tree. The details can be found in [23]. In our implementation of the PBT, we further improve it by using a criterion in [14] for the choice of tree branches. We choose not to discuss the details as it is not the major focus of this paper. A nice thing about PBT is that two-class and multi-class classification problems are treated in an identical way with two-class being a special case. Therefore, our algorithm handles two-class and multi-class labeling problems naturally without any change.

Image features are computed on the patch (size of 21 × 21) centered on each pixel i and we use around 8,000 including Canny edge results at a low scales (1.5), Haar filter responses, and gradients. The context features are the probability values of different pixels relative to the current pixel. For each current pixel, we shoot out many rays and sparsely choose the pixels on these rays. Fig. (1) shows an example. The features can be probability directly on these pixels or the mean probability around them, or even other high order statistics. In total, we use around $4,000$ of the context features. The training algorithm starts from probability maps of uniform distribution, and then it recursively refines the maps until it converges. The first classifier does not choose any context features as they are not informative at all. Starting from the second classifier, nearly 90% of the features selected are context features with the rest being the image features. This demonstrates the importance of using the context information in clarifying the ambiguities.



(a) (b)

Figure 3. (a) shows the training and test errors at different stages of auto-context for horse segmentation. (b) gives the precision-recall curves by different algorithms and PBT based auto-context algorithm achieves the best result, particularly in the high-recall area.

Fig. (3.a) shows the $F - value = \frac{Precision \times Recall}{2(Precision+Recall)}$ [17] for the different stages of the auto-context algorithm. Since cascade of AdaBoost algorithm is widely used in computer vision, we illustrate the auto-context algorithm using a cascade of AdaBoost and PBT classifiers. Moreover, we conduct an experiment, as suggested in [28], to train the system with the appearance, rather than probabilities, of the context pixels. Several observations we can make from this figure: (1) the auto-context algorithm significantly improve the results over patch-based classification methods; (2) auto-context model is effective on both types of classifiers; (3) using appearance context does not improve the result (even slightly worse); (4) The second stage of the auto-context usually gives the biggest improvement.

Fig. (2) gives the procedures of the auto-context algorithm in which all the uncertainties are carried out in a probabilistic fashion with no hard thresholding. One might think that if the previous classifier has already made a firm decision on a particular sample, then it is probably a waste to include this sample in the next step. We design a variation to the auto-context algorithm, called auto-context-th which is similar to auto-context. The only difference is that if any pixel for i for any label k if $p(y_i = k|X) \geq th$ where th is a threshold, then this pixel will not be used in the next round of training. The training time of auto-context-th does appear to be less, but the performance is slightly worse than auto-context. Fig. (3.b) gives the full precision-recall curves for various algorithms. The final version of PBT based auto-context achieves the best result. It significantly outperforms the CRFs model based algorithm (shown as L+M (Ren et al.)) in Fig. (3.b) and it also shows improvement than hybrid model algorithm [11]. Training takes about half a day for auto-context using cascade and a couple of days for auto-context using PBT, with both having 5 stage of classifiers. In testing, it takes about 40 seconds to compute the final probability maps. Fig. (4) shows some results and the bottom two are the images with the worst scores. As we can see, even these results are not too bad. Though our purpose is not to design a specific horse segmentation algorithm, our algorithm outperforms many the existing algorithms reported so far [17, 11, 3, 26]. Also, auto-context model is very general and easy to implement. There is no need to design specific features, which makes the system directly protable to a variety of other applications. However, pursing feature design by human intelligence is still a very interesting and promising direction.

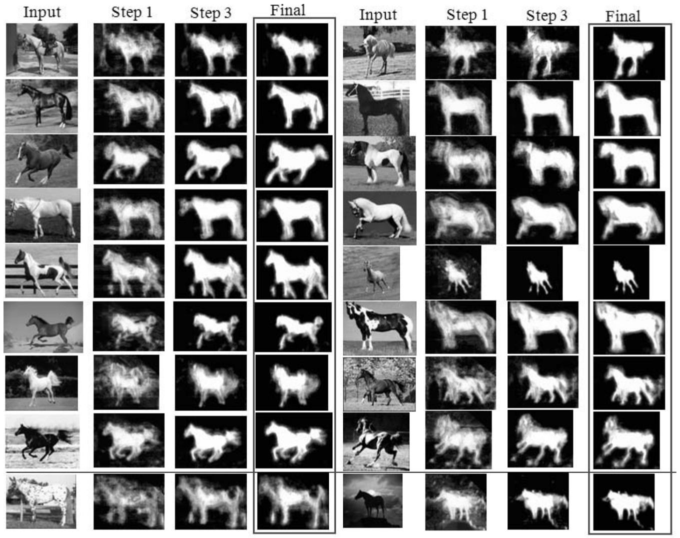
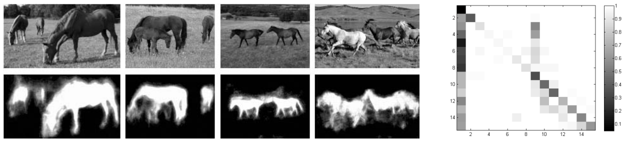


Figure 4. The first and the fifth column displays some test images from the Weizmann dataset [3]. Other columns show probability maps by the different stages of the auto-context algorithm. The last row shows two images with the worst scores.

The Weizmann dataset contains one horse in each image and they are mostly centered. To further test the performance of a trained auto-context on other images, we collect some images from Google in which there are multiple horses at various scales. Fig. (5) shows the input images and the results by auto-context. Notice that the small horse next to the big one in the second figure of Fig. (5.a) is labeled as the background.



(a) (b)

Figure 5. The first row in (a) shows some images searched by Google by typing key word "horses". The second row displays the final probability map by the auto-context algorithm trained on the Weizmann dataset [3]. (b) is the confusion matrix on the test images from the Berkeley human body dataset [13]. The head, main body, left thigh and right thigh can be mostly detected correctly.

3.2. Human body configuration

To further illustrate the effectiveness of the auto-context algorithm, we apply it on another problem, human body configuration. Each body part is assigned with a label.

This is now a multi-class labeling problem rather than foreground-background segregation. As stated before, PBT based auto-context algorithm treats two-class and multi-class problem in an identical way. We collect around 130 images for training, and use the same set of features as in the horse segmentation problem on image patch of size 21×21 (designing some specific features for this task might further improve our result).

Fig. (6) shows the results at different stages of the auto-context on the test images in [13]. Fig. (5.b) gives the confusion matrix. As we can see, the main body, the head, the left thigh, the right thigh and the feet can be labeled robustly in most cases. The arms appear to be confused with the main body and the background. The speed on these test images are about the same as in the horse segmentation case. The existing algorithms in the problem often have a pre-segmentation stage [13, 24], which is prone to errors by the low-level segmentation algorithms. For example, the procedures described by [13, 24] can merge segmented regions into big ones but can not break them. This may cause problem where there is no clear boundary between the parts. The auto-context algorithm computes the posterior marginals for each pixel directly.

We illustrate our algorithm on gray scale images, and color images used in [13]. The results are better than those shown in [24] in which a BP algorithm was implemented. BP computes the marginals by propagating messages through local connected neighbors, whereas auto-context can take long-range context information directly. This results in significant speed improvement. Similar to the argument made in the horse segmentation case, our algorithm is more general than [13, 24]. However, a thorough comparison of the auto-context algorithm with the state of art algorithms (BP, Graph-Cuts, CRFs), in terms of both quality and speed, remains as future research.

Further procedures are still required to explicitly extract the body parts since the auto-context algorithm only outputs probability maps. This is probably the place where more explicit shape information can be used in the Bayesian framework.

3.3. Scene parsing/labeling

We also applied our algorithm on the task of scene parsing/region labeling. We used the MSRC dataset [19] in which there are 591 images with 21 types of objects manually segmented and labeled (there are two additional types in the new dataset). There is a nuisance category labeled as 0. The setting for this task is similar as before, and the only difference is that we use color images in this case. Shotton et al. did not have the background model to learn the regions of 0 label, whereas it is not a problem in our case. However, to obtain a direct comparison to their result, we also exclude the 0 label both in training and testing. We use the identical training and testing images as in [19]. Fig. (7) shows some results and the confusion matrix. The results by auto-context are the marginal probabilities for each pixel belonging to a specific class. We simply assign the label with the highest probability to each pixel. The accuracy by the first stage of auto-context, classification method PBT only, achieves 50.4%. The overall pixel-wise accuracy by 4 layers of auto-context is 74.5% which is better than 72.2% reported in [19]. However, a careful reading at the confusion matrices by both the algorithms shows that our result is more consistent and the mistakes made are more “reasonable”. For example, boat is mostly confused with car and building whereas boat was mis-classified to many other classes in [19] such as water, bike, and tree.

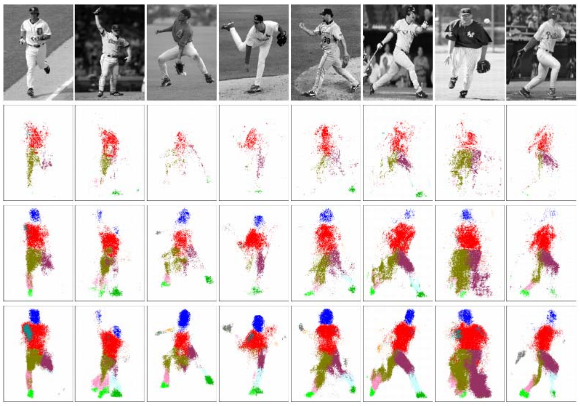


Figure 6. The first row displays some test images. The second, third and forth row shows the classification map by the first, third and fifth stage of the trained auto-context algorithm.

Our algorithm is more general and easier to implement. The speed reported in [19] was 3 minutes per image whereas ours is around 70 seconds. Notice that the classification results are a bit scattered. Another level of algorithm to enforce region-based consistency is still needed. With a post-processing stage to encourage the neighboring pixels to have the same lable, the accuracy improves to 77.7%.

$$Y^* = \arg min - \sum_i \log p(y_i|X) + \alpha \sum_{(i,j)} \delta(y_i \neq y_j),$$

where $\alpha = 2.0$ for the results in this paper.

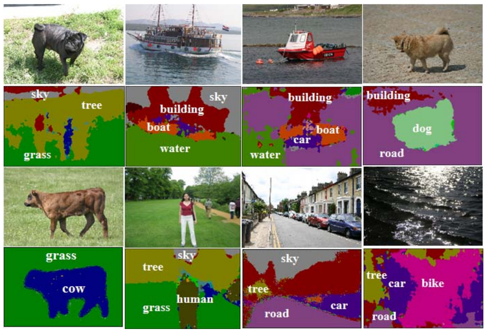
It is noted that almost all the algorithms we compare to, on the horse segmentation, human body configuration, and scene labeling use the context or high-level information. CRF models are indeed context based. A direct comparison to the algorithms reported on the MSRC dataset is given in table (1). [16] gave the accuracy measure on segmented regions rather than pixels with a score 68.4%. Using auto-context with a post-processing achieves 77.7%.

Algorithm	TextonBoost [19]	[29]	Auto-Context	AC+post
Accuracy	72.2%	75.1%	74.5%	77.7%

Table 1. Comparison to other algorithms on the MSRC dataset. AC+post refers to the result by auto-context with a post-processing for smoothing and the post processing takes about 0.1 second.

3.4. Conclusions

In this paper, we have introduced an auto-context algorithm to learn a unified low-level and context model. We



(a)

	Bu	Gs	Tr	Co	Sp	Sk	Ap	W	Fc	Ca	Bi	Fl	Sn	Bi	Bk	Ch	Rd	Ct	Dg	Bd	Bt
Building	69	1	5	0	0	1	1	3	1	1	2	2	1	1	5	0	4	1	0	2	0
Grass	0	96	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Tree	3	5	87	0	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0
Cow	0	5	1	78	1	0	0	0	1	0	0	0	1	0	0	0	0	1	5	7	0
Sheep	1	5	3	3	80	0	0	0	0	0	0	0	0	2	0	3	2	0	0	0	0
Sky	3	0	0	0	0	95	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Airplane	11	2	2	0	0	1	83	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Water	5	5	2	0	0	2	0	67	0	3	3	0	0	1	0	0	9	0	0	1	2
Face	1	0	0	1	0	0	0	0	84	0	0	0	0	0	1	0	0	0	1	10	0
Car	14	0	1	0	0	2	3	1	0	70	0	0	1	1	1	0	4	0	0	1	2
Bike	12	0	2	0	0	0	0	1	0	1	79	0	0	0	0	0	2	0	0	2	0
Flower	1	1	2	7	2	0	0	1	3	0	0	47	0	3	17	0	0	1	1	12	0
Sign	34	0	1	0	0	0	0	0	0	0	0	0	61	0	2	0	1	1	0	0	0
Bird	9	7	3	5	10	3	0	13	1	6	0	0	0	30	0	1	2	2	6	1	0
Book	8	1	1	0	0	0	0	0	1	0	0	3	0	0	80	2	1	0	0	2	0
Chair	25	2	8	1	0	0	0	1	3	0	4	2	0	0	4	45	1	1	0	2	0
Road	11	0	1	0	0	1	0	6	0	1	0	0	0	0	0	0	78	0	0	1	0
Cat	1	0	4	0	0	0	0	11	2	0	0	0	0	4	0	1	3	68	5	0	0
Dog	4	2	5	4	3	0	0	2	7	0	0	0	0	9	0	1	3	5	52	2	0
Body	6	1	1	2	0	0	0	1	5	0	0	1	1	2	5	1	2	3	2	67	0
Boat	28	0	0	0	0	1	3	15	0	13	9	0	0	1	0	0	2	0	0	2	27

(b)

Figure 7. (a) shows some difficult test images (same as shown in [19]) and a couple of typical ones, with their corresponding classified labels. (b) displays the legend and confusion matrix. The overall pixel-wise accuracy is 74.5%. The result by PBT only achieves 50.4%. The number reported in [19] was 72.2%, and using auto-context with a post-processing stage achieves 77.7%.

target the posterior distribution directly, and thus, the test phase shares the same procedures as those in the training. The auto-context algorithm selects and fuses a large number supporting contexts which allow it to perform rapid message propagation. We introduce iterative procedures into the traditional classification algorithms to refine the classification results by taking effective context information.

The proposed algorithm is very general. Under nearly a same set of parameters, we illustrate the auto-context algorithm on three challenging vision tasks. The results show to significantly improve the results by patch-based classification algorithms and demonstrate improved results over almost all the existing algorithms using CRFs and BP. It typically takes about 30 $\sim$ 70 seconds to run the algorithm of size around 300 $\times$ 200. However, a full scale of comparison with various choices of the algorithm, like that conducted in [21], is needed. The scope of the auto-context model goes beyond vision applications and it can be applied in other problems of multi-variate labeling in machine learning and AI.

The limitations for the auto-context model are: (1) the features on the context information are still somewhat limited and more explicit shape information is still required; (2) different auto-context models need to be trained for different applications; (3) the training time takes a bit long (a few days) for the scene parsing.

Acknowledgments ZT is funded by NIH Grant U54 RR021813 entitled Center for Computational Biology. We thank Yingnian Wu for many stimulating discussions.

References

[1] S. Avidan, "SpatialBoost: Adding Spatial Reasoning to AdaBoost", *Proc. of ECCV*, 2006.

[2] S. Belongie, J. Malik, and J. Puzicha, "Shape Matching and Object Recognition Using Shape Contexts", *IEEE Trans. on PAMI*, 24(4):509-522, April 2002.

[3] E. Borenstein, E. Sharon and S. Ullman, "Combining top-down and bottom-up segmentation", *Proc. IEEE workshop on Perc. Org. in Com. Vis.*, June 2004

[4] J. Friedman, T. Hastie and R. Tibshirani, "Additive logistic regression: a statistical view of boosting", *Ann. stat.*, vol. 28, no. 2, pp. 227-407, 2000.

[5] Y. Freund and R. E. Schapire, "A Decision-theoretic Generalization of On-line Learning And An Application to Boosting", *J. of Comp. and Sys. Sci.*, 55(1), 1997.

[6] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Trans. PAMI*, vol. 6, pp. 721-741, Nov. 1984.

[7] X. He, R. Zemel, and M. Carreira-Perpinan "Multiscale conditional random fields for image labelling", *Proc. of CVPR*, June, 2004.

[8] D. Hoiem, A. Efros, and M. Hebert, "Putting Objects in Perspective", *Proc. of CVPR*, 2006.

[9] S. Kumar and M. Hebert, "Discriminative random fields: a discriminative framework for contextual interaction in classification", *ICCV*, 2003.

[10] S. Kumar and M. Hebert, "A Hierarchical Field Framework for Unified Context-Based Classification", *Proc. of ICCV*, 2005.

[11] A. Levin and Y. Weiss, "Learning to combine bottom-up and top-down Segmentation", *Proc. of ECCV*, 2006.

[12] J. Li, A. Najmi, and R.M. Gray, "Image Classification by a Two-Dimensional Hidden Markov Model", *IEEE Trans. on Sig. Proc.*, vol. 48, no. 2, pp. 517-533, 1989.

[13] G. Mori, X. Ren, A. Efros, and J. Malik, "Recovering Human Body Configurations: Combining Segmentation and Recognition", *Proc. of CVPR*, June, 2004.

[14] J.R. Quinlan, "Improved use of continuous attributes in C4.5", *J. of Art. Intell. Res.*, 4, pp. 77-90, 1996.

[15] J. Pearl, "Probabilistic reasoning in intelligent systems: networks of plausible inference", *Morgan Kaufmann*, 1988.

[16] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie "Objects in Context", *Proc. of ICCV*, 2007.

[17] X. Ren, C. Fowlkes, and J. Malik, "Cue integration in figure/ground labeling", *Proc. of NIPS*, 2005.

[18] S. Savarese, J. Winn and A. Criminisi, "Discriminative Object Class Models of Appearance and Shape by Correlatons", *Proc. of CVPR*, June 2006.

[19] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "TextonBoost: Joint App., Shape and Context Modeling for MultiClass Object Recognition and Segmentation", *Proc. of ECCV*, 2006.

[20] A. Singhal, J. Luo, and W. Zhu, "Probabilistic spatial context models for scene content understanding", *Proc. of CVPR*, June, 2003.

[21] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother, "A Comparative Study of Energy Minimization Methods for Markov Random Fields", *Proc. of ECCV*, 2006.

[22] A. Torralba, K. P. Murphy, and W. T. Freeman, "Contextual Models for Object Detection Using Boosted Random Fields", *Proc. of NIPS*, 2004.

[23] Z. Tu, "Probabilistic boosting tree: Learning discriminative models for classification, recognition, and clustering", *Proc. of ICCV*, 2005.

[24] Z. Tu, "An Integrated Frmework for Image Segmentation and Perceptual Grouping", *Proc. of ICCV*, Oct., Beijing, 2005.

[25] Z. Tu, X. Chen, A. Yuille, and S.C. Zhu, "Image parsing: unifying segmentation, detection, and object recognition", *IJCV*, 2005.

[26] S. Zheng, Z. Tu, and A. Yuille, "Detecting Object Boundaries Using Low-, Mid-, and High-Level Information", *Proc. of CVPR*, June, 2007.

[27] V. Vapnik, "Statistical Learning Theory", Wiley-Interscience, New York, 1998.

[28] L. Wolf and S. Bileschi, "A Critical View of Context", *Int'l J. on Com. Vis.*, 2006.

[29] L. Yang, P. Meer, and D. J. Foran, "Multiple Class Segmentation Using A Unified Framework over Mean-Shift Patches", *Proc. of CVPR*, June, 2007.

[30] J. Yedidia, W. Freeman, and Y. Weiss, "Generalized Belief Propagation", *NIPS*, 2000.