Auto-Context and Its Application to High-Level Vision Tasks and 3D Brain Image Segmentation

Zhuowen Tu and Xiang Bai

Abstract—The notion of using *context* information for solving high-level vision and medical image segmentation problems has been increasingly realized in the field. However, how to learn an effective and efficient context model, together with an image appearance model, 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 a learning algorithm, auto-context. 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 created 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 until convergence. Auto-context integrates low-level and context information by fusing a large number of low-level appearance features with context and implicit shape information. The resulting discriminative algorithm is general and easy to implement. Under nearly the same parameter settings in training, we apply the algorithm to three challenging vision applications: foreground/background segregation, human body configuration estimation, and scene region labeling. Moreover, context also plays a very important role in medical/brain images where the anatomical structures are mostly constrained to relatively fixed positions. With only some slight changes resulting from using 3D instead of 2D features, the auto-context algorithm applied to brain MRI image segmentation is shown to outperform state-of-the-art algorithms specifically designed for this domain. Furthermore, the scope of the proposed algorithm goes beyond image analysis and it has the potential to be used for a wide variety of problems for structured prediction problems.

Index Terms—Context, object recognition, image segmentation, 3D brain segmentation, discriminative models, conditional random fields.

1 Introduction

Context and high-level information plays a vital role in object recognition and scene understanding [2], [28], [45]. Nevertheless, a principled way of learning an effective and efficient context model, together with an image appearance model, is not available. Many types of information can be referred to as context: Different parts of an object can be context to each other; different objects in a scene can be each other's context. For example, a clearly visible horse's head may suggest the locations of its tail and leg, which are often occluded. A car might suggest the existence of a road and vice versa [18].

From the Bayesian point of view, context information is carried in the joint statistics of multivariate in the posterior probability, which is often decomposed into likelihood and prior. In vision, likelihood and prior often correspond to appearance and shape, respectively. There are many technological hurdles to overcome to build successful vision systems. The difficulties can be summarized into two main aspects: *modeling* and *computing*. 1) Difficulty in modeling complex appearances—objects in natural images observe complex patterns and there are many factors contributing to the complexity such as textures (homogeneous or inhomogeneous), lighting conditions, viewing angles, and occlusions. 2) Difficulty in learning complicated shapes and configurations. 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 is a nontrivial task.

In vision, models like Markov Random Fields (MRFs) [13] and Conditional Random Fields (CRFs) [23], [21] have been used to capture the context information. Energy minimization algorithms, such as Belief Propagation (BP) [29], [56], have been widely adopted. However, these models and algorithms share somewhat similar disadvantages: 1) The choice of functions used is quite limited so far, 2) they usually rely on a fixed topology with very limited neighborhood relations, and 3) many of them are only guaranteed to obtain the optimal solution for limited function families. In Section 3.4, we will provide more insights about why auto-context is effective and compare it against the BP algorithm.

In this paper, we make an effort to address some of the shortcomings of existing methods by proposing a new algorithm, *auto-context*. The algorithm targets the posterior distribution directly in a supervised manner. Like in the BP algorithm [56], 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.

- *Z. Tu is with the Laboratory of Neuro Imaging, Department of Neurology and Department of Computer Science, University of California, 635 Charles E. Young Drive South, Suite 225, Los Angeles, CA 90095-7334. E-mail: ztu@loni.ucla.edu.*
- *X. Bai is with the Department of Electronics and Information Engineering, Huazhong University of Science and Technology, Wuhan, Hubei 430074, P.R. China. E-mail: xbai@hust.edu.cn.*

Manuscript received 14 Aug. 2008; revised 11 Apr. 2009; accepted 15 July 2009; published online 18 Nov. 2009.
Recommended for acceptance by R. Zabih.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2008-08-0502.
Digital Object Identifier no. 10.1109/TPAMI.2009.186.

A classifier is first trained to classify every pixel. There are two types of features for the classifier to choose from: 1) *image features* computed on the local image patches and 2) *context information* from a large number of sites on the classification maps. In this paper, we use image patches of fixed size 21×21 and $11 \times 11 \times 11$ for 2D natural images and 3D MRI images, respectively. The size is fixed for all rounds of the algorithm. For natural images, the initial classification maps are usually uniform since we do not know what objects might appear at where a prior. Context features are typically not selected by the first classifier since they are uninformative. The first trained classifier produces a new classification map, which becomes the input for training the next classifier. The algorithm iterates to approach the ground truth until convergence. In medical imaging, we can often use a probabilistic atlas [35] as the initial classification map since the anatomical structures are roughly positioned. 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 rich image appearance models together with the context information by learning a series of classifiers. The appearance (likelihood) and the high-level context and shape information (prior) are seamlessly combined in an implicit way and the balance between the two is naturally handled. Unlike many energy minimization algorithms, where the modeling and computing stages are separated, auto-context uses the same procedures in both phases. The training and testing results differ in the generalization power of the trained algorithm. Auto-context uses deterministic procedures for computing the marginal distributions. However, it does not make any hard decisions in the process. Uncertainties are propagated through learned closed-form functions in the classifiers, rather than by performing sampling or integration. This makes the auto-context algorithm significantly faster than most existing algorithms. Compared to MRFs and CRFs, auto-context is not limited to a fixed neighborhood structure. Each pixel/voxel 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 to converge toward the ground truth.

We demonstrate the auto-context algorithm on challenging high-level vision tasks for three well-known data sets: horse segmentation in the Weizmann data set [3], human body configuration estimation in the Berkeley data set [26], and scene region labeling in the MSRC data set [38]. The results demonstrate significant improvement over many existing algorithms in terms of both speed and quality. In addition, we apply the algorithm on brain images for both segmenting a single structure (caudate) and performing whole brain segmentation. A thorough comparison is made using many standard metrics and we observe a large improvement over state-of-the-art algorithms across various domains. The proposed auto-context framework is general and easy to implement. Its scope goes beyond high-level vision tasks; indeed, it has the potential to be used for many problems for structured prediction where joint statistics need to be modeled. This is demonstrated on a typical machine learning problem, handwritten character recognition [20], and we observe comparable performance gain with a state-of-the-art algorithm [42].

2 Related Work

We discuss related work in two broad areas: 2D image understanding and 3D medical image segmentation.

2.1 Related 2D Image Understanding Work

There has been a lot of recent work in using context information for object recognition, scene understanding [18], [38], [32], [43], [51], [15], [39], and tracking [55], [53]. A pioneering work was proposed by Belongie et al. [2] which used context in shape matching. Hoiem et al. [18], [17] presented a system combining the interaction between different objects in a loop as mutual support. Auto-context differs from these works in several aspects: 1) It has a single objective function to minimize (classification error), 2) local appearances and context are simultaneously integrated, and 3) the training procedure in auto-context is simpler and more general. Rabinovich et al. [32] recently showed that using context information in the postprocessing stage can boost the overall performance.

Three approaches directly related to auto-context are: Boosted Random Fields (BRFs) [43], Mutual Boosting [9], and SpatialBoost [1], which all used boosting to combine the contextual information. However, these algorithms used contextual beliefs as weak learner in the boosting algorithm. Auto-context is a general algorithm and the classifier of choice is not limited to boosting. It directly targets the posterior through iterative steps, resulting in a simpler and more efficient algorithm. Under nearly the same set of parameters in training, we demonstrate several 2D natural image and 3D medical image applications using the auto-context algorithm which are not available in [43], [9], [1].

A feed-forward way of combining context and appearance was proposed in [51] for object detection. However, their method does not iteratively learn a posterior. More importantly, their findings led to the conclusion that the performance gain from using context is negligible (unless the image quality is really poor). Our experimental results in Fig. 5a suggest otherwise. One possible reason might be due to their specifically designed context features. Compared to other algorithms that use context [32], [18], it learns an integrated model without the need for specifying particular types of context.

2.2 Related 3D Image Segmentation Work

The task of segmenting subcortical and cortical structures is very difficult due to their intrinsic ambiguous patterns. Neuroanatomists often develop and use complicated protocols [27] in guiding the manual delineation process and these protocols may vary from task to task. There have been many medical image segmentation algorithms developed in the past. These algorithms range from shape-driven [54], [30], atlas and knowledge-based [35], Markov Random Fields models [10], [31], to classification/learning-based approaches [24]. They have produced encouraging results, although there are some common drawbacks: 1) Most of them assume very simple appearance patterns, 2) many algorithms are slow with very time-consuming energy minimization steps (e.g., it takes about one day for FreeSurfer [10] to segment a MRI image), and 3) they usually involve heavy algorithm design (e.g., many carefully engineered energy terms) which poses a big hurdle for transporting the systems to other modalities, or even on the same modality but to segment different anatomical structures.

It was shown in [54] that using a joint prior for the shapes of neighboring brain structures can improve the segmentation result. Even though context information might play a more important role in 3D medical image analysis than in 2D natural images, context has been somewhat underexplored in the medical imaging domain. One possible reason is due to the difficulty of deriving explicit context information for 3D objects. The proposed auto-context algorithm has the advantage of fusing a large number of 3D context and implicit shape features, without the need of worrying about explicit 3D shape representations.

3 Problem Formulation

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

3.1 Objective

For a 2D image, the input is $X = (x_{(i,j)}, (i,j) \in \Lambda)$, where Λ denotes the image lattice. For an 1D vector, the input can be denoted as $X = (x_1, \dots, x_n)$. For notational simplicity, we do not distinguish the two and call them both "images." We will use the 1D vector input for illustration. In training, each image X comes with a ground truth $Y = (y_1, \dots, y_n)$, where $y_i \in \{1, \dots, K\}$ is the class label for pixel i. The training set is then $S = \{(Y_j, X_j), j = 1, \dots, m\}$, 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. One possibility is to search for the optimal solution by 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 cluttered images. Both of them have high complexity and usually do not follow independent identical distributions (i.i.d.). 2) *Computing*: The combination of the $p(X|Y)$ and $p(Y)$ is often nonregular. Besides many recent advances made in optimization and energy minimization [41], a general solution still 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 $\mathcal{P} = (\mathbf{p}_1, \dots, \mathbf{p}_n)$, 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 integrating out all of the dY_{-i}. Next, we discuss how to approach this.

3.2 Traditional Classification Approaches

A traditional way to approximate (1) is by treating it as a classification problem. Usually, a classifier is considered to be translation invariant. The training set becomes $S = \{(y_{ji}, X_j(N_i)), j = 1, \dots, m, i = 1, \dots, n\}$, where m is the number of training images and n is the number of pixels in each image. For notational simplicity, we assume one training image since using multiple training images follows the identical procedure.

$$S = \{(y_i, X(N_i)),\ i = 1, \dots, n\}.$$

Instead of using the entire image X, the training set includes an image patch centered at each i, $X(N_i)$. N_i denotes all of the pixels in the patch. In the context of boosting algorithms, it was shown [11], [12] that one can learn the discriminative model based on logistic regression:

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

$F_k(X(N)) = \sum_{t=1}^{T} \alpha_{k,t} \cdot h_{k,t}(X(N))$ is the strong classifier on a weighted sum of selected weak classifier $h_{k,t}$ for label k. Many other classifiers also output a confidence which can be turned into an approximated posterior. It is noted that our algorithm is not limited to any particular choice of classifier and many traditional classifiers can be used, such as CART [4] or SVM [48]. The learned posterior marginal, $p(y = k|X(N))$, is a very crude approximation to (1) and it only uses context through image patch $X(N)$. Due to this limitation, the well-known CRFs algorithms [23], [21] try to explicitly include the context information by adding another term $p(y_{i1}, y_{i2}|X(N_{i1}), X(N_{i2}))$, where $i1$ and $i2$ are the neighbors. Though CRFs have been successfully applied in many applications [21], [22], [33], it still has limitations similar to those in the MRFs as discussed in Section 1. CRFs still use a fixed neighborhood structure with a fairly limited number of connections. The computing complexity explodes given 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 [22] uses only very sparse connections). Also, it limits their computing capability since the interactions are slowly propagated through pairwise relations.

3.3 Auto-Context

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

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

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

$$S_1 = \{(y_i, (X(N_i), \mathcal{P}^{(0)}(i))), i = 1, \dots, n\}, \tag{3}$$

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

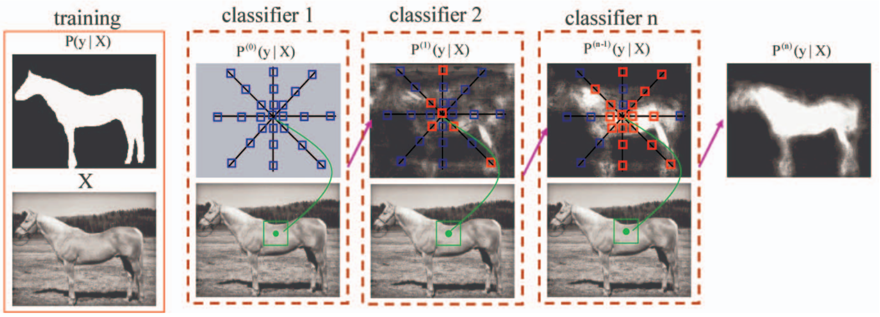


Fig. 1. Illustration of the classification map updated at each round for the horse segmentation problem. The blue rectangles represent candidate context locations and the red rectangles represent selected contexts in training at each stage.

Note that even the first classifier is trained the same way as the others. We simply start the probability map from a uniform distribution. Since the uniform distribution is not informative, the context features are not selected by the first classifier. In certain applications, such as medical image segmentation, the positions of the anatomical structures are roughly known, and one can use a probability atlas [35] as the initial $\mathcal{P}^{(0)}$. Fig. 2 outlines the training process of the auto-context algorithm.

3.3.1 Auto-Context Convergence Analysis

The algorithm iteratively updates the marginal distribution to approach

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

Next, 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.

Theorem 1. *The auto-context algorithm (not tied to any particular classifier type) monotonically decreases the training error, $\epsilon = \sum_i \delta(y_i \neq H(X(i))$, where y_i is the true label and $H(X(i))$ is the output by the classifier.*

Proof. We show it in the context of boosting but the proof holds on other classifiers as well. Again, we consider only one image in the training data and use $X(i)$ to denote $X(N_i)$. In the AdaBoost algorithm [11], one choice of 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 [12]. The multiclass case can be written in a logistic function as well, as in (2).

At different steps, we have

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

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

where

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

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

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

since the equality can be easily achieved by making

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

The boosting algorithm (or almost any valid classifier) chooses 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,

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

□

When other types of classifiers are used on the error measure, $\epsilon = \sum_i \delta(y_i \neq H(X(i))$. The proof also holds as long as the classifier can choose the current classification confidence as input feature. 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 having the feature of its own probability (or classified labels or the error function is measured on the labels) in the candidate pool, which is not hard to achieve. Note that this proof does not guarantee convergence to the global optimal solution. However, by fusing a large amount of context information, the algorithm is shown to be effective in practice, as we demonstrate on many applications. Fig. 1 gives an illustration of the iterations of auto-context. There has been some debate about the probabilistic explanation for the boosting algorithms. Nevertheless, we emphasize that the proposed auto-context framework is not dependent on any particular choice of classifier.

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 $p^{(n)}(y_i|X(N_i), \mathcal{P}^{(n-1)}(i))$

Fig. 2. The training procedure of the auto-context algorithm.

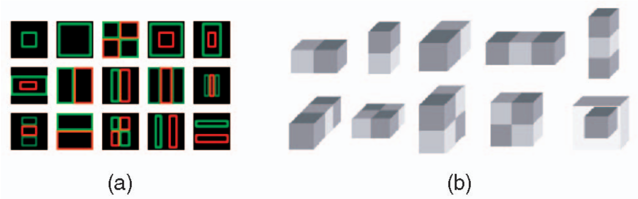


Fig. 3. Illustration of some (a) 2D Haar-like and (b) 3D Haar-like filters.

3.3.2 *Feature Design*

In this section, we discuss the two types of features used: 1) image appearance features and 2) context features.

Image features. The image appearance features include Haar responses on the input image. For the 2D applications shown in this paper, we use a similar set of Haar features as used in [50]. One reason to use Haar is due to their computational efficiency when computed using integral images [50]. For color images, we use the $L^*u^*v^*$ decomposition and compute the Haar features on three channels separately. Complementary features can collaboratively improve the performance. For example, histogram of gradient (HOG) features [7] are shown to be very effective and they are somewhat complementary to the Haar features. In some cases, the absolute position of a pixel is a good feature as well. It is particularly informative for medical images where objects have roughly fixed positions. In scene understanding, it is also useful since sky often appears on the top and road appears on the bottom. In addition, we can obtain filter responses of different Gabor functions and Canny edge maps at different scales.

The size for the basic image patch can vary as well. For appearance-based classification, using a big patch, say, 51×51 will perform slightly better than using a small one, say 21×21. However, the difference diminishes in the later stages of the auto-context algorithm with context features included. We tried three different patch sizes, 21×21, 41×41, and 51×51 in the horse segmentation experiment (Section 4.1), and the F-values at the first stage are, respectively, 0.78, 080, and 0.80. However, they all reach 0.83 at the fourth stage. We have a similar observation on the MSRC data set though the difference in the first stage is bigger, in terms of pixel accuracy: 58.0 percent using 51×51 versus 50.4 percent using 21×21 at the first stage, but they both reach around 77 percent in the end.

In the 3D MRI brain image segmentation, we compute 3D Haar features on the original images directly, and some examples are shown in Fig. 3b.

Context features. Context features are obtained from the classification maps from the previous iterations. Ideally, the marginals (classification probabilities) of every pixel could be put into the feature candidate pool for selection. However, this would create a very large feature space, making the training process slow. Therefore, we only sparsely sample some locations, which we found to be effective. It gives a good balance of training efficiency and classification power. In the horse segmentation experiment, we used dense context features (20,000) and relatively sparse contexts (5,000) and obtained the same F-value of 0.83. For each pixel of interest, eight rays in 45-degree intervals are extended out from the current pixel and we sparsely sample the context locations on these rays. Their classification probabilities are used as features (both individual probabilities and the mean probability within a 3×3 window). Fig. 4 gives an illustration. All locations within 3 pixels away from the current pixel are in the candidate feature pool. This makes sure that local contexts will not be missed, if they are indeed informative. A radius sequence (5, 7, 10, 12, 15, 20, 25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100, 125, 150, 175, 200) is used for choosing other context locations on each ray. In MRI brain image segmentation, the situation is similar except that the context locations are in 3D. In each round, the algorithm will automatically select different sets of context locations, both short-range and long-range (some selected features are shown in Table 4 providing an intuitive understanding of what features have been learned). These context features implicitly represent the shape and configuration information.

Additional features. On one hand, the image and context features we talked about so far are very general, and they can be directly applied to many applications. On the other hand, in image understanding and medical imaging, there are often domain specific variables (sometimes hidden) to the solutions. The understanding and explicit inferences of these variables are likely to further improve the performance of a system. For example, using

Object Class	first feature	fourth feature
building	$\mathbf{p}(building)$ in $[(3,-3),(5,-1)]$	$\mathbf{p}(building)$ at $(-20,36)$
grass	$\mathbf{p}(grass)$ in $[(1,-3),(3,-1)]$	$\mathbf{p}(grass)$ at $(16,27)$
tree	$\mathbf{p}(tree)$ in $[(1,-3),(3,-1)]$	gradient in L^* channel of color
sheep	$\mathbf{p}(sheep)$ in $[(0,-2),(2,0)]$	$\mathbf{p}(building)$ at $(-5,10)$
face	$\mathbf{p}(face)$ in $[(6,10),(8,12)]$	mean of $\mathbf{p}(body)$ at $[(61,36),(63,38)]$
car	$\mathbf{p}(car)$ in $[(6,10),(8,12)]$	$\mathbf{p}(water)$ at $(-2,-3)$
bike	$\mathbf{p}(bike)$ in $[(1,-3),(3,-1)]$	$\mathbf{p}(building)$ in $[(0,-71),(2,-69)]$
chair	$\mathbf{p}(chair)$ in $[(-1,-3),(1,-1)]$	$\mathbf{p}(car)$ at $(0,-1)$
road	$\mathbf{p}(road)$ in $[(2,-2),(4,0)]$	$\mathbf{p}(tree)$ at $(-27,16)$
boat	$\mathbf{p}(boat)$ in $[(2,-2),(4,0)]$	$\mathbf{p}(water)$ at $(-27,16)$

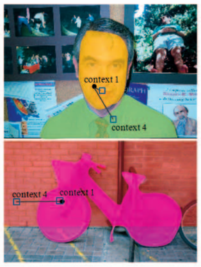


Fig. 4. Description of some features selected by the auto-context algorithm in the second stage for the MSRC image labeling task. Since there are multiple labels, each image has n discriminative probability maps on every label. For example, $\mathbf{p}(building)$ denotes the probability map for each pixel being of the building class. In the second column, the first selected feature for some classes is given where $[(),()]$ denotes a rectangle of the top-left and bottom-right corner w.r.t. the current pixel of interest. We give the description of the fourth selected features, many of which show contextual information. Some contexts are used to do enhancement, e.g., the body context for the face class; some others might be doing suppression, e.g., the building context for the sheep class. The figures on the right demonstrates the features for face and bike, respectively.

the geometric cues [16] about the 3D world facilitates a better understanding of 2D images [17]. In the 3D brain image segmentation task, we engaged an explicit generative model to extract an adaptive atlas first. We observe around 10-20 percent performance gain on a large MRI data set (> 500 images) over the baseline algorithm (no context); using the auto-context algorithm on top of all of these features gives further 5-10 percent improvement (the 20 percent performance gain is due to the complementariness of generative and discriminative models and a detailed discussion can be found in [25]). It is evident that using informative cues improves the performance of the auto-context algorithm. We leave the study of exploring other cues about domain specific variables for future research.

3.4 Understanding Auto-Context

We first take a look at the Belief Propagation algorithm [29], [56] since it also works on the marginal distribution. For certain directed graphs, BP can find the global optimal. For graphs with loops, 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 pairwise relation between sites i and j, and $\phi_i(y_i)$ is a unary term. The BP algorithm [56] 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,

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

Similarly, the auto-context algorithm updates the marginal distribution by (5). The major differences between BP and auto-context are: 1) In BP, every pair of $\psi_{i,j}(y_i, y_j)$ on all possible labels needs to be evaluated and integrated in (7). Therefore, BP can only work with a limited number of neighborhoods to keep the computational burden under check. For auto-context, we evaluate a sequence of learned classifiers, $F_k^{(t)}(X(i), \mathcal{P}^{(t-1)}(i))$, which are computed discriminatively 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 informative context and appearance information. Note that, 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 sets of features at different stages, which allows it to make use of the best available information each time. In the experiments, we will compare different choices of learning classifiers, e.g., using a fixed one or separating the context prior from the likelihood. We show that the auto-context setting works the best. 3) In BP, there are often separate stages to design the graphical model and to learn $\psi(y_i, y_j)$ and $\phi_i(y_i)$. Auto-context is designed to learn the posterior marginal directly and its inference stage follows identical steps to the learning phase. However, BP has the advantage that it uses the same message passing rule for different forms of $p_i(y_i)$ in (6), whereas auto-context learns a different set of classifiers for different tasks.

A question one might ask is: *"How different is 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 trained by using the entire image X rather than an 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 (1). The correlation between different pixels needs to be taken into account, which is done by learning one classifier for $p(y_i|X)$. A key concept here is about knowledge representation and propagation. An image is composed of many different objects. Objects and their parts often locally observe certain degrees of regularity, and it is much more effective to gather information locally and propagate it than trying to solve everything in one shot. The possible configurations of different objects or even the same object with different parts are too numerous to learn effectively. It would also result in a feature space too big for a classifier to handle, which would lead to over-fitting.

Wolf and Bileschi [51] suggested that using label context might achieve the same effect as using image appearance context in object detection; moreover, for both types of context their 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 [51]. Our results are shown in Fig. 5a and the conclusions differ significantly from [51] in two aspects: 1) Having a much enlarged appearance context pool actually degrades performance (as opposed to using only local appearance features) and 2) label context, computed using our auto-context algorithm, greatly improves the segmentation/labeling result.

There have been many algorithms that have attempted to integrate context [2], [32], [18], [43], [1]. The auto-context algorithm makes an attempt to recursively select and fuse context information, as well as appearance, in a unified framework. The first trained classifier is based purely on the local appearance; objects with strong appearance cues are often correctly classified even after the first round. These probabilities then start to influence their neighbors, especially if there are strong correlations between them.

4 Experiments

We perform experimental studies in three areas: 1) 2D natural image understanding, 2) handwritten OCR recognition, and 3) 3D MRI brain image segmentation. For 2D image understanding, we illustrate the auto-context algorithm on three challenging tasks: horse segmentation, human body configuration estimation, and scene parsing/labeling. In these three tasks, the system uses a nearly identical parameter setting, including the number of weak classifiers and the stopping criterion. For brain imaging applications, we show both single structure (caudate) segmentation and whole brain segmentation.

The procedures described in the auto-context algorithm are generic. However, there are several important implementation issues and a detailed discussion will help to

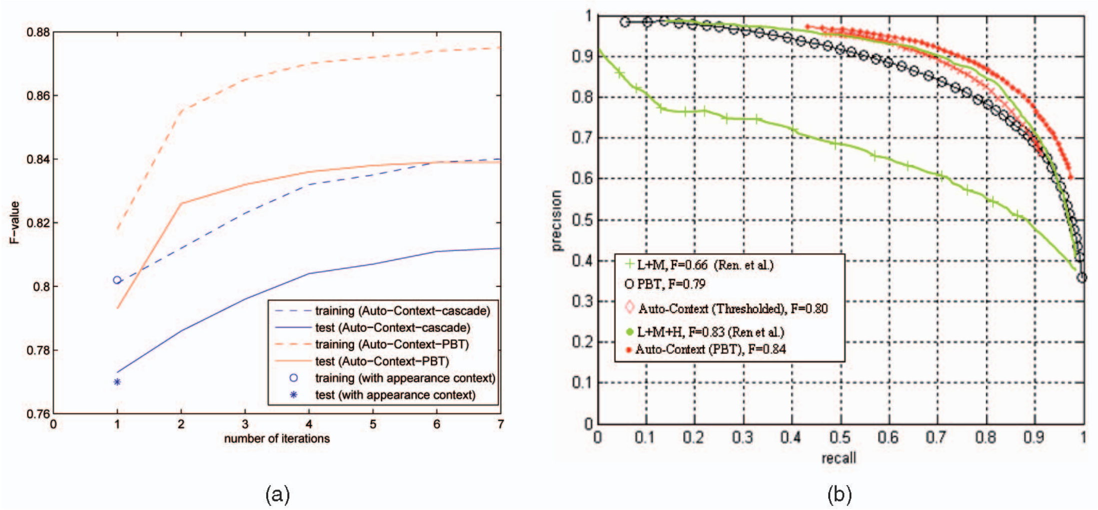


Fig. 5. (a) The training and test errors at different stages of auto-context for horse segmentation. (b) The precision-recall curves by different algorithms and the auto-context algorithm achieves the best result, particularly in the high-recall area. There are no position features used in this experiment in (a). For the appearance context case, only one stage is needed and that is why it has a single dot.

better understand the algorithm. Next, we highlight some critical points and empirical observations, which apply to all experiments reported in this section.

1. **Choice of classifier**: Auto-context uses a sequence of classifiers, and thus the classifier quality influences overall performance. Since we use a large number of candidate features (around 10,000) in the image understanding case, boosting appears to be a good choice. It has many appealing properties: a natural feature selection and fusion process, can deal with a large number of features on a considerable amount of training data, features do not have to be normalized, and it can be efficiently trained and used. SVMs can also be used, as shown in the OCR case in Section 4.2, but they are better suited when the number of features is relatively small. In the horse segmentation example, we try SVM classifier also. The F-values by the first and second stages of a SVM-based auto-context are, respectively, 0.54 and 0.75, whereas a boosting-based auto-context achieves 0.78 and 0.82, respectively. This confirms that auto-context is not tied to any specific choice of classifier; however, as we can see, due to the feature selection and fusion capability of different types of classifier, different choices of the base classifier do have an impact on the overall performance. Using decision tree (typically 2 or 3-level) as the weak classifier significantly outperforms decision-stump-based boosting [12]. In the experiments, each boosted classifier selects and fuses 100 weak classifiers of 2-level decision tree. Boosting typically converges when 500 weak classifiers are combined [36]; in practice, it varies from task to task. We found that combining 100 weak classifiers gives a good balance between efficiency and effectiveness. Other ensemble learning algorithms, e.g., random forest [37], are also good choices. A thorough empirical comparison of various classifiers can be found in [5]. Since each pixel is a training sample, the training data consist of millions of positive and negative patches. It is often not efficient for a single node boosting algorithm to perform the classification. We adopt the probabilistic boosting tree (PBT) algorithm [44]. PBT learns and computes a discriminative model in a hierarchical way by

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

where $p(l_i|.)$ is the classification model learned by boosting node in the tree. The details can be found in [44].

2. **Multiclass classifier**: We use PBT to deal with both the two-class and multiclass classification in this paper. A typical parameter in PBT is the depth, which is set to 5 in most cases. One can also use one-versus-all [34] to directly combine two-class classifier into multiclass classifier, though it is less efficient than PBT in testing. The one-versus-all strategy is easy to implement; however, it loses efficiency in both training and testing when the number of class becomes large. Other options for the multiclass classifier include random forests or error correcting output codes [8].
3. **Appearance and context features**: We have described the features in Section 3.3.2. Once a large number of features are used in the candidate pool, adding more gives very small improvement unless the features are really complementary to the existing ones. The sparsely sampled context features described in Section 3.3.2 are quite effective. Both short-range and long-range contexts are important, though they might play different roles in different applications.
4. **Number of stages**: The second stage of the auto-context algorithm often gives the most gain and performance levels off typically at stage 4 or 5.

4.1 Horse Segmentation: A Running Example

We show an application of object segmentation on the Weizmann data set consisting of 328 gray scale horse images [3]. The data set also contains manually annotated label maps. We split the data set randomly into half for training and half for testing. The training stage follows the steps described in Fig. 2. The conclusions from this study are general:

- Using classification maps as context always improves performance over the patch-based classification algorithm.
- One can train a separate classifier based on classification maps only. This allows the likelihood and prior to be learned separately, though the overall result will be a bit worse than putting them together.
- One can even learn a classifier at stage 2 and apply it to the later stages. This option is useful in cases where training time is also a major concern.

Next, we discuss the details of the algorithm. The images and context features have been described in the previous sections. One can choose to use or not use the spatial coordinates of each pixel as a feature. Section 3.3.2 discusses how the context features are designed; they are the probabilities directly on these pixels and the mean probability around them. The training algorithm starts from probability maps of uniform distribution, and then it recursively refines the maps until convergence. The first classifier does not choose any context features as they are uninformative. Starting from the second classifier, nearly 90 percent 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. An illustration of the features selected by the algorithm are shown in Fig. 1. In Section 4.4, we give detailed descriptions of some selected context features to help clarify what has been learned in scene understanding.

Fig. 5a shows the $F\text{-}value = \frac{2\times Precision\times Recall}{Precision+Recall}$ [33] for the different stages of the auto-context algorithm. We use two types of classifiers, a cascade of boosted classifiers [50] and PBT [44]. No position features are used in this experiment, meaning that the learned classifier is translation invariant. Moreover, we conduct an experiment, as suggested in [51], to train the system with the appearance, rather than probabilities, of the context pixels. We use the cascade classifier in this case. Stage one can be considered as a traditional patch-based classification approach where no probability context is used. Several observations can be made from Fig. 5:

1. the auto-context algorithm significantly improves the results over patch-based classification methods;
2. the auto-context model is effective on both types of classifiers;
3. using appearance context does not improve the result in testing (sometimes making it slightly worse);
4. the second stage of the auto-context usually gives the biggest improvement.

Fig. 5b 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. 5b. Training takes about half a day for auto-context using cascade and a couple of days for auto-context using PBT, with both having five stages of classifiers. In testing, it takes about 40 seconds on an image size around 320×260 to compute the final probability maps. Fig. 6 shows some results and the bottom two are the images with the worst scores. Though our purpose is not to design a specific horse segmentation algorithm, our algorithm outperforms many of the existing algorithms reported so far [33], [3].

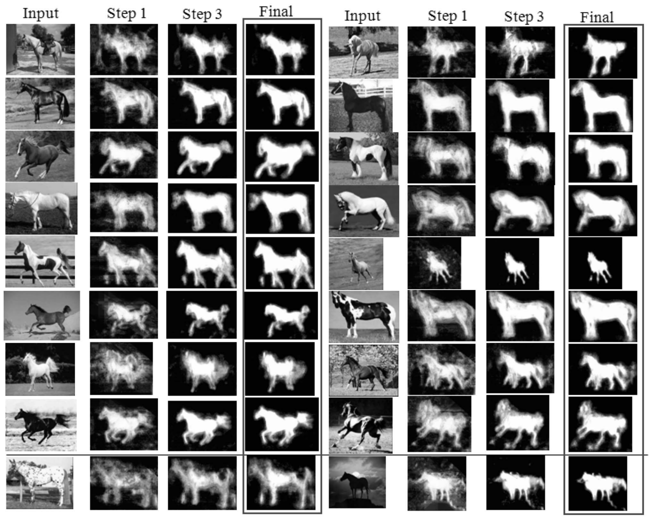


Fig. 6. The first and the fifth column display some test images from the Weizmann data set [3]. Other columns show probability maps at different stages of the auto-context algorithm. The last row shows two images with the worst scores.

Starting from stage 2, the majority of the features selected by the classifiers are the context features, and this motivates us to further explore the role of context. Here, we use cascade of AdaBoost [50] as the basic classifier. We conduct a comparative study using different settings (position features are also used here):

1. a regular auto-context algorithm using cascade classifiers;
2. using only context features in the feature candidate pool starting from the second stage;
3. at stage 2, doing the same thing as in case 1, but repeat this learned classifier for the later stages;
4. at stage 2, doing the same thing as in case 2, but repeat this learned classifier for the later stages.

Fig. 7a shows the F-values in testing for the four cases, and Fig. 7b illustrates the overall precision-recall curves. All four cases start from the same point since they share the same classifier at stage 1. Case 1 serves as the baseline algorithm. Case 2 tests how important it is to have appearance features together with the context features. As we can see, the algorithm is not performing as well as the regular auto-context algorithm. This shows the importance of seamlessly integrating appearance and context features. However, it is still an improvement over the patch-based classification algorithm. Case 3 answers the question of how important it is to learn different classifiers after stage 2. If one would repeat the same classifier learned at the second stage, the algorithm does not generalize well.

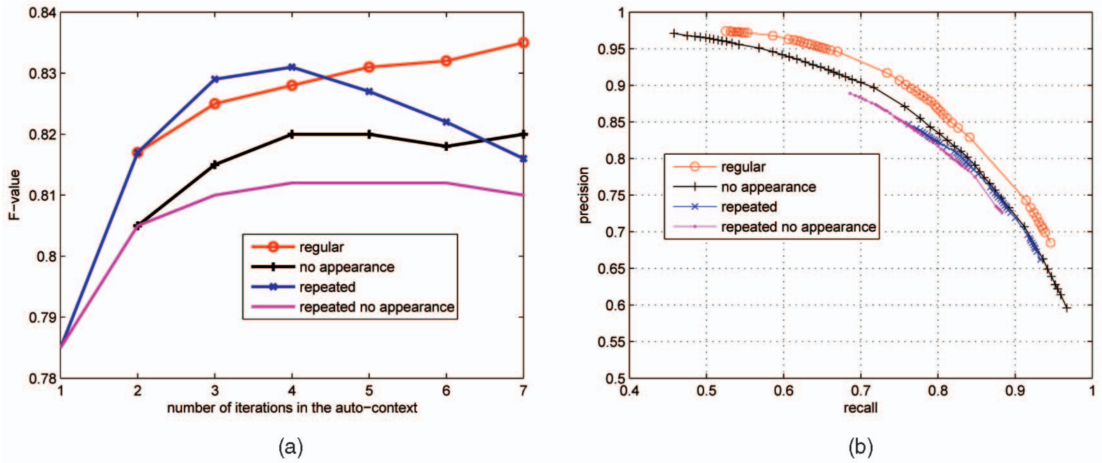


Fig. 7. (a) The F-values of different settings of the auto-context algorithm on the Weizmann horse data set. There is an overfitting effect when the classifiers are repeated. (b) The precision-recall curves of the corresponding algorithms.

The Weizmann data set 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 collected some images from Google in which there are multiple horses at various scales. Fig. 8 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. 8 is labeled as the background.

4.2 Handwriting Recognition Using SVM-Based Auto-Context

In Section 4.1, the auto-context algorithm is illustrated on an object (horse) segmentation task using a variation of the boosting algorithm. As we emphasized before, the formulation of auto-context does not depend on the choice of classifier and it works on the general multivariate labeling problem. Here, we show how it can be used to recognize handwritten words. Fig. 9 shows some examples from an OCR data set [20]. Each input word is composed of a number of characters, each of which is represented by an 16×8 binarized image. Each character has 26 possible labels, $y_i \in \{a, b, \ldots, z\}$. Each input word has n characters. We choose an SVM [48] implementation [6]. We adopt a one-versus-all strategy for dealing with the multiclass classification problem. There are two types of features: data features and context features. For each character, we directly use its input vector as the appearance features (128 binary values). In this regard, the inputs are not treated as images and we approach this handwritten recognition problem as a machine learning problem. In the auto-context algorithm, the first stage uses data features only. For the context features, we look at a neighborhood window of 15 characters resulting in a total number of $15 \times 26 = 390$ context features (26 classification maps). Each entry is the classification probability of character i being of label l at one of its neighboring characters. Except for the feature set and the choice of classifier, the algorithm is otherwise the same as the auto-context algorithm described in Section 4.1. We do 10-fold cross validation on a set of 6,100 words, and use 610 for training and 5,490 for testing. We just use two stages in the auto-context and the classification error for the first (SVM on data features of 128 dimension) and second stage are 0.261 and 0.195, respectively. This shows a 25 percent improvement. It is comparable to the max-margin Markov networks algorithm (M^3N) [42] (we used the same data features as in [42] and the slight difference on the results by SVM is probably due to implementation details).

4.3 Human Body Configuration

To further illustrate the effectiveness of the algorithm, we apply it on another problem, human body configuration estimation. Each body part is labeled into 14 classes, and Fig. 10a shows the template. We use 5-level PBT as the basic classifier, which will produce 15 classification maps, with each corresponding to a part label and an additional label for the background. We use the same set of context features on each classification map, and Fig. 10c shows an illustration.

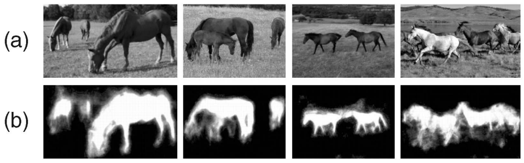


Fig. 8. (a) Some images returned by Google by typing key word "horses." (b) The final probability map by the auto-context algorithm trained on the Weizmann data set [3].

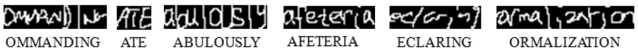

OMMANDING
ATE
ABULOUSLY
AFETERIA
ECLARING
ORMALIZATION

Fig. 9. Some example words from an OCR data set [20].

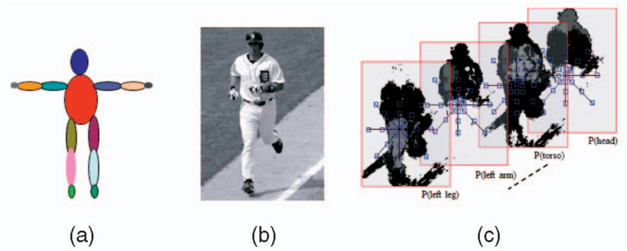


Fig. 10. (a) A template in which body parts are colored into 14 labels. (b) A test image (input) and (c) context features on the discriminative probability maps with each corresponding to a class label.

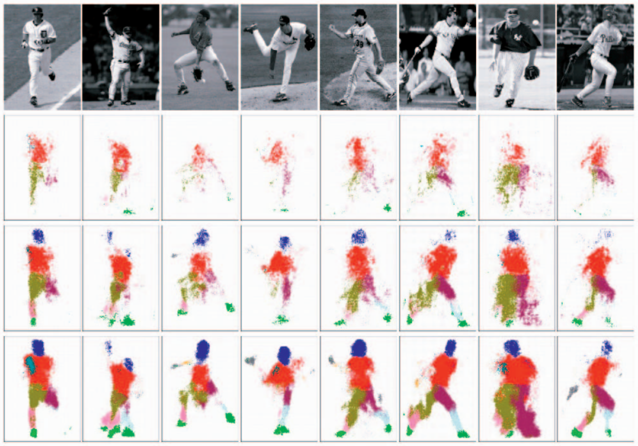
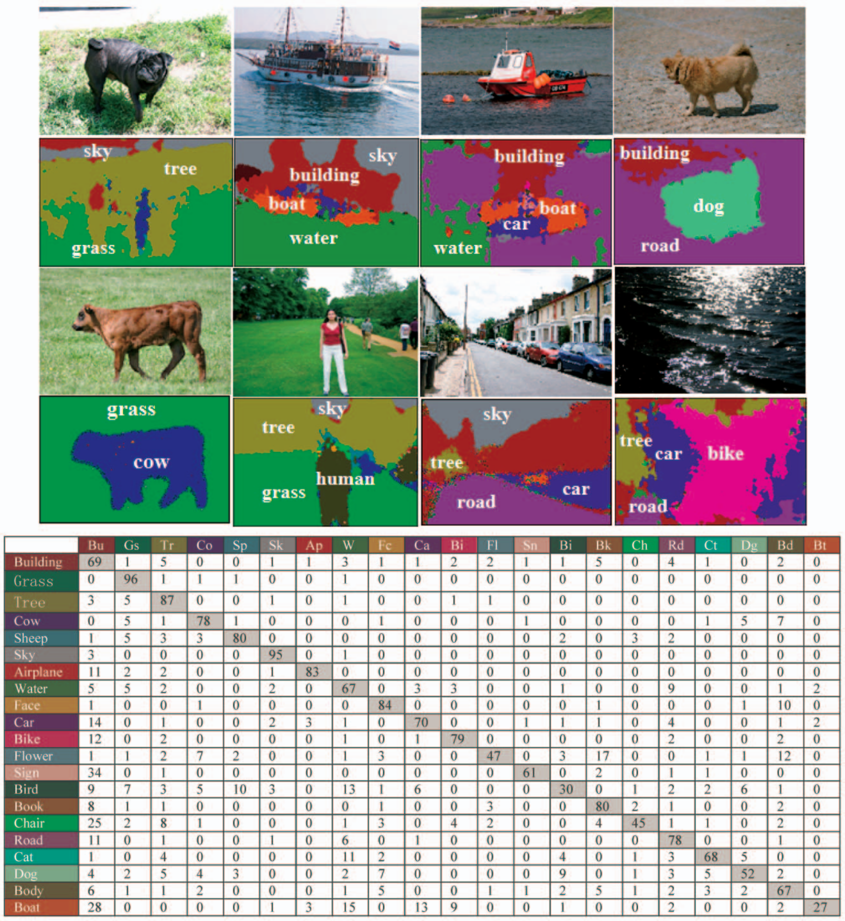


Fig. 11. The first row displays some test images. The second, third, and fourth rows show the classification maps by the first, third, and fifth stage of the trained auto-context algorithm.

We collected around 130 images for training, and used the same set of features as in the horse segmentation problem on image patch of size 21 × 21. Fig. 11 shows the results at different stages of the auto-context on the test images in [26]. As we can see, the torso, 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 is about the same as in the horse segmentation case. We illustrate our algorithm on gray scale images in [26] (they used color images instead). The criterion using "fraction correct" in Mori et al. [26] is different from the accuracy measure here. Nevertheless, we achieve around 90 percent accuracy for the torso, which is comparable to the 91 percent fraction correct rate reported in [26].

4.4 Scene Parsing/Labeling

We also applied our algorithm on the task of scene parsing/region labeling. We used the MSRC data set [38] of 591 images with 21 types of objects manually segmented and labeled (there are two additional types in the new data set). There is a nuisance category labeled as 0. The setting for this task is similar to 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 [38]. Fig. 12 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. Note that Shotton et al. [38] did not model the 0 class. There are two additional



	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

Fig. 12. The first row shows some difficult test images and a few typical ones, with their corresponding classified labels. The second row displays the legend and confusion matrix. The overall pixel-wise accuracy is 74.5 percent. The result by image patch-based classification achieves 50.4 percent. The number reported in [38] was 72.2 percent, and using auto-context with a postprocessing stage achieves 77.7 percent.

classes, "horse" and "mountain" which were not included in [38]. The accuracy by the first stage of auto-context, classification method PBT only, achieves 50.4 percent. The overall pixel-wise accuracy by four stages of auto-context is 74.5 percent which is better than 72.2 percent reported in [38].

Starting from the second stage of the auto-context algorithm, spatial relationships of the labels (both same and different) at different locations are fused implicitly through classifiers. For example, a pixel classified as being a confusing pattern between boat and building will be clarified as boat, if some context pixels (top, bottom, left, and right) in a range have high probability of being water. In this regard, contextual relationships are maintained in auto-context in an implicit way through individual pixels. To understand how the context features are explicitly playing the role in the algorithm, we give a description of some selected features in the second stage of the algorithm. For a multiclass labeling problem, after each round, n discriminative probability (classification) maps $\mathbf{p}$, corresponding to each class label, are created. In the learning process (after the first stage), the algorithm can choose both appearance and context features to support the decision making. Usually, the first three features selected are still from each class's own probability map. The second column in Fig. 4 gives a description to the fourth features for some typical classes. Each class picks the mean value of a small window surrounding or near the current pixel. The third column of Fig. 4 describes the fourth selected feature. Clearly, context information are being selected and it can be understood intuitively. For example, a pixel on a face looks for context support from body, and a boat pixel looks somewhere upright for water. Sometimes, appearance features are still selected for some classes, e.g., the tree class.

The final probability maps observe certain degree of "noisiness" since no explicit constraints were used in the classification stage. Therefore, enforcing smoothness and region-based consistency can further improve the results. Here, we simply employ a postprocessing process to encourage the neighboring pixels to have the same label as in (8):

$$Y^* = \arg min - \sum_i \log p(y_i|X) + \alpha \sum_{(i,j)} \delta(y_i \neq y_j), \tag{8}$$

where $\alpha = 2.0$ for the results in this paper. Equation (8) essentially combines the classification map by auto-context with a Potts model. Based on the probability maps output by the auto-context algorithm, we simply perform a Iterated Conditional Modes (ICM) method to perform energy minimization, which requires 0.1 seconds. Qualitatively, the segmentation/labeling results did not change too much. Quantitatively, the accuracy improves to 77.7 percent, which is probably due to the sensitiveness of the accuracy measure on the object boundaries. Ideally, smoothness between the neighboring pixels can also be captured by the context features and this postprocessing seems to be redundant. However, the auto-context algorithm fuses many other context features and thus, the consistency is maintained *implicitly*. Equation (8), on the other hand, maintains the local smoothness *explicitly* (in [19], we can see that this postprocessing can be removed by a voting-based scheme).

TABLE 1
Comparison to Other Algorithms on the MSRC Data Set

Algorithm	TextonBoost [38]	[54]	Auto-Context	AC+post
Accuracy	72.2%	75.1%	74.5%	77.7%

AC+post refers to the result by auto-context with a postprocessing for smoothing (which takes about 0.1 seconds per image).

The recognition rate averaged over all the classes is 68.7 percent, whereas it was reported as 64 and 67 percent in [49] and [37], respectively. Also, 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 misclassified to many other classes in [38], such as water, bike, and tree. Our algorithm is more general and easier to implement. The speed reported in [38] was 3 minutes per image, whereas ours is around 70 seconds. A significantly improved algorithm in speed has been proposed in [37] with nearly real-time performance. However, the average accuracy is 72 percent.

It is noted that almost all of the algorithms we compared to, on the horse segmentation, human body configuration, and scene labeling, use context or high-level information. CRF models are indeed context-based. A direct comparison to the algorithms reported on the MSRC data set is given in Table 1. Rabinovich et al. [32] gave the accuracy measure on segmented regions rather than pixels with a score 68.4 percent.

4.5 Single Structure Segmentation in 3D Brain Images

As previously stated, context information plays an important role in medical image analysis where the anatomical structures are roughly positioned and constrained. Segmenting subcortical structures from 3D brain images is of significant practical importance.

We first show our algorithm on a recently established caudate segmentation data set [47]. There are four sets of data provided in this grand challenge competition, two for training and two for testing. As described in the documents from the organizers: "All MRI images are scanned with an Inversion Recovery Prepped Spoiled Grass sequence on a variety of scanners (GE, Siemens, Philips, mostly 1.5 Tesla). Some data sets have been acquired in axial direction, whereas others in coronal direction. All data sets have been reoriented to axial RAI-orientation, but have not been aligned in any fashion." The two training sets are: 1) MRIs and structural segmentations from the Internet brain segmentation repository (IBSR) at Mass General Hospital, Boston, and 2) MRIs and caudate segmentations from the Psychiatry Neuroimaging Laboratory at the Brigham and Women's Hospital Boston (BWH). The two testing sets are: 1) MRIs from different disease study at the UNC Neuro Image Analysis Laboratory, Chapel Hill and 2) 14 MRIs from the Psychiatry Neuroimaging Laboratory at the Brigham and Womens Hospital, Boston. These data are from the same study as the BWH data sets in the training set.

The training sets BWH and IBSR are given as different forms. We use a popular tool, BET [40], to perform automatic skull stripping, followed by a widely used 3D image registration algorithm, AIR [52], to perform 12-parameter nonrigid transformation. A typical image in

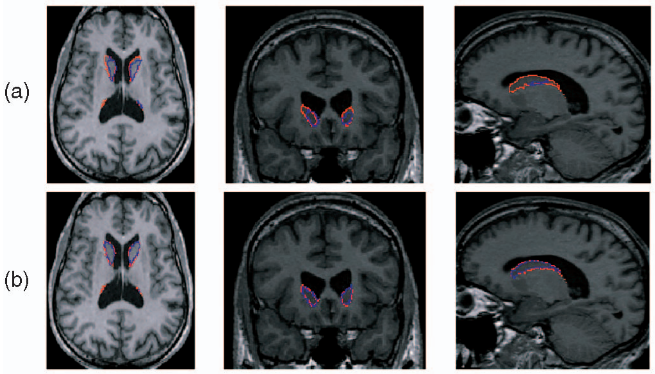


Fig. 13. (a) The results by the Hybrid model [46], and (b) the results on the same slices by the auto-context algorithm. The red lines are the boundaries by the neuroanatomists and blue lines are the boundaries by the algorithm.

TABLE 2
Error Metrics on the Caudate Segmentation by Hybrid Model [46] and Auto-Context: The Overall Scores Are, Respectively, 59.71 and 73.38

Case	OE	Score	VD	Score	AD	Score	RMSD	Score	MD	Score	Total
UNC Ped	40.4	74.6	-23.2	59.5	0.86	68.3	1.21	78.4	5.64	83.4	72.82
UNC Eld	38.8	75.6	-17.2	69.8	0.75	72.2	1.14	79.6	6.79	80.0	75.44
BWH PNL	41.8	73.7	-26.6	53.8	1.51	49.1	3.50	42.1	25.27	28.4	49.42
Average All	40.8	74.3	-23.9	58.3	1.22	57.9	2.53	57.5	17.33	50.6	**59.71**

(a)

Case	OE	Score	VD	Score	AD	Score	RMSD	Score	MD	Score	Total
UNC Ped	33.4	79.0	-12.05	76.5	0.68	74.8	1.09	80.5	12.1	64.4	75.03
UNC Eld	36.8	76.9	-0.69	80.0	0.72	73.4	1.31	76.5	17.6	48.2	71.00
BWH PNL	32.1	78.5	-13.62	74.4	1.17	76.6	1.75	76.5	12.8	62.3	73.64
Average All	33.3	78.3	-10.60	76.0	0.97	75.5	1.52	77.3	13.7	59.8	**73.38**

(b)

the IBSR training image is used as the template. Based on these 25 training images with the left and right caudate manually delineated by experts, we train auto-context to learn six stages of classifiers. Since the images are roughly registered, position is also a very informative cue. Therefore, the feature candidate pool includes positions (x, y, z coordinates) and various 3D Haar responses. We use image patch (subvolume) of $11 \times 11 \times 11$ in this case and there are around 6,000 candidate features. Given a test volume, we perform skull stripping using BET followed by image registration using AIR [52], and then run the sequences of classifiers to segment out the left and right caudate. The segmentation part is similar to the 2D case discussed before. It typically takes 5 minutes to perform the segmentation on a modern PC. The results on the two test data sets were uploaded to the benchmark server with the number measured by the benchmark test organizers.

The evaluation process was designed by the organizers of the MICCAI'07 "Segmentation Challenge."[1] The segmentation results are evaluated by assigning a score to each test case on a variety of metrics [14]. The minimum and maximum scores for each metric are normalized to 0 and 100. The total score of a metric is obtained by averaging the individual scores of all test cases. The detailed descriptions of how the measures are obtained can be found in the summary paper of this workshop. The caudate segmentation challenge consists of multiple groups of subjects. One group of test cases is scans of the same subject, performed on different scanners. For these cases, no reference segmentations are available, and these cases will therefore not contribute to the total score. These cases are included to test if a method is reproducible.

A hybrid model was proposed in [46] in which a patch-based classification method is used to learn the discriminative models based on local appearances and a PCA generative model on the shape of the anatomical structures are combined. The hybrid model algorithm gave the best score in the 2007 competition. Using the identical set of features, the auto-context algorithm applied on this task is shown to significantly outperform [46]. Some views for a typical volume are shown in Fig. 13. The red lines show the boundaries delineated by the experts, and the blue lines are the results by the algorithm. Also, the detailed scores are summarized in Table 2. The overall score by the auto-context algorithm is 73.38 which is a significant improvement over the number 59.71 by [46].

4.6 Whole 3D Brain Image Segmentation

In brain imaging, many algorithms [47] were designed for segmenting a specific anatomical structures. They are often hard to extend to segment multiple structures. Our learning-based algorithm has the particular advantages of being general, robust, and computationally efficient. It was shown that our algorithm outperforms many existing algorithm specifically designed for a decided anatomical structure. Some existing algorithms designed to perform multiple brain subcortical structure segmentation include [54], [31], [10]. However, there is heavy algorithm design in [54], [31]. The only one performing whole brain segmentation is the widely used FreeSurfer algorithm [10]. With the identical setting as in the previous caudate segmentation, we train the auto-context algorithm to segment 56 brain structures using 25 training images. All of the volumes are skull stripped by BET [40] followed by 12 parameter nonrigid registration using [52].

It typically takes 25 minutes to segment an MRI image using auto-context, whereas it takes about one day for the FreeSurfer algorithm. We obtain manual delineation of 56 structures by neuroanatomists, such as caudate, hippocampus, putamen, cerebellum, insular cortex, and gyrus rectus. An example test volume is shown in Fig. 14 in which the first row shows the manual delineation and the second row shows the result by the final stage. We use 15 test volumes and repeat the training and testing images a couple of times. It was shown in [46] that FreeSurfer produces a worse result than using the learning-based hybrid algorithm. The average F-value for all 56 anatomical structures is 78.0 percent which improves the hybrid algorithm with score 75.8 percent.

5 Conclusions and Discussions

In this paper, we have introduced the auto-context algorithm, which learns the low-level appearance, implicit shape, and context information through a sequence of discriminative models. Our goal is to design an integrated

[1] http://mbi.dkfz-heidelberg.de/grand-challenge2007.

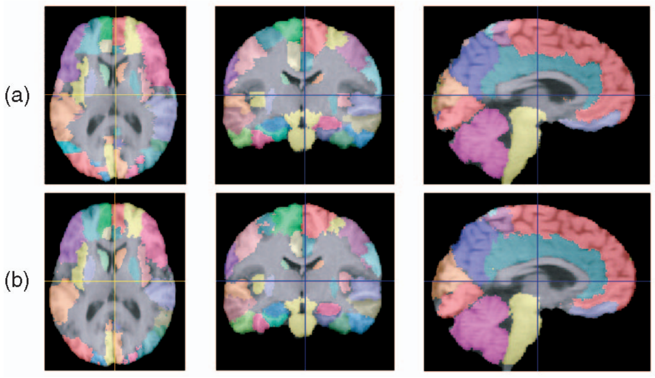


Fig. 14. (a) A typical test images with 56 structures annotated by neuroanatomists. (b) The results by the third stage of the auto-context algorithm.

framework to include both appearance and context information in a principled way. We 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 of supporting contexts, which allow it to rapidly propagate the information. We introduce iterative procedures into traditional classification algorithms to refine the classification results by using context information effectively.

The proposed algorithm is general. We illustrate the auto-context algorithm on three challenging vision tasks. The results are shown to significantly improve the results by patch-based classification algorithms and demonstrate improved results over many existing algorithms using CRFs and MRFs. It typically takes about 30-70 seconds to run the algorithm on an image of size around 300×200. We also demonstrated the auto-context algorithm on two important brain imaging tasks and showed improved results over state-of-the-art algorithms. Moreover, the scope of the auto-context model goes beyond vision applications and it can be applied in other problems of structured prediction in machine learning and AI.

In terms of the advantages, the auto-context algorithm improves the modeling capability of existing methods based on MRFs and CRFs. It does not depend on any particular type of classifier, is very general and easy to implement, and avoids heavy algorithm design (various energy terms and procedures). In terms of the disadvantages, shape and context information in auto-context are utilized in an implicit way. There is a certain limit to this type of implicit information going through discriminative learning. More explicit shape information and object configuration obtained through top-down reasoning, e.g., the silhouette of a shape, can further clarify certain ambiguities, though a more time-consuming inference step may be required. The other limitations for the auto-context model are: 1) The features on the context information are still somewhat limited, 2) different auto-context models need to be trained for different applications, and 3) the algorithm is a supervised method and thus requires a set of well-annotated ground truth data, which might not always be available or can be difficult to obtain. We are also exploring using weakly supervised and semisupervised learning to alleviate the burden on obtaining ground truth data.

Acknowledgments

This work is supported by US Office of Naval Research (ONR) Award No. N000140910099. This work is also funded in part by US National Science Foundation CAREER Award No. IIS-0844566 and US National Institutes of Health Grant U54 RR021813 entitled Center for Computational Biology. The authors also thank China grant 863 2008AA01Z126.The authors thank Yingnian Wu and Piotr Dollár for many stimulating discussions. The authors also thank the anonymous reviewers for providing many constructive suggestions.

References

[1] S. Avidan, "Spatialboost: Adding Spatial Reasoning to Adaboost," *Proc. European Conf. Computer Vision,* pp. 386-396, 2006.

[2] S. Belongie, J. Malik, and J. Puzicha, "Shape Matching and Object Recognition Using Shape Contexts," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 4, pp. 509-522, Apr. 2002.

[3] E. Borenstein, E. Sharon, and S. Ullman, "Combining Top-Down and Bottom-Up Segmentation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition Workshop Perceptual Organization in Computer Vision,* June 2004.

[4] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, *Classification and Regression Trees.* Wadsworth Int'l, 1984.

[5] R. Caruana and A. Niculescu-Mizil, "An Empirical Comparison of Supervised Learning Algorithms," *Proc. Int'l Conf. Machine Learning,* pp. 161-168, 2006.

[6] C.C. Chang and C.J. Lin, *LIBSVM: A Library for Support Vector Machines.* http://www.csie.ntu.edu.tw/cjlin/libsvm, 2001.

[7] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 886-893, June 2005.

[8] T.G. Dietterich and G. Bakiri, "Solving Multiclass Learning Problems via Error-Correcting Output Codes," *J. Artificial Intelligence Research,* vol. 2, pp. 263-286, 1995.

[9] M. Fink and P. Perona, "Mutual Boosting for Contextual Inference," *Proc. Neural Information Processing Systems Conf.,* 2003.

[10] B. Fischl, D. Salat, E. Busa, M. Albert, M. Dieterich, C. Haselgrove, A. van der Kouwe, R. Killiany, S.K.D. Kennedy, A. Montillo, N. Makris, B. Rosen, and A. Dale, "Whole Brain Segmentation: Automated Labeling of Neuroanatomical Structures in the Human Brain," *Neuron,* vol. 33, pp. 341-355, 2002.

[11] Y. Freund and R.E. Schapire, "A Decision-Theoretic Generalization of Online Learning and an Application to Boosting," *J. Computer and System Sciences,* vol. 55, no. 1, pp. 119-139, 1997.

[12] J. Friedman, T. Hastie, and R. Tibshirani, "Additive Logistic Regression: A Statistical View of Boosting," *Annals of Statistics,* vol. 38, no. 2, pp. 337-407, 2000.

[13] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 6, no. 6, pp. 721-741, Nov. 1984.

[14] G. Gerig, M. Chakos, and M. Valmet, "A New Validation Tool for Assessing and Improving 3D Object Segmentation," *Proc. Medical Image Computing and Computer-Assisted Intervention,* pp. 516-523, 2001.

[15] X. He, R. Zemel, and M. Carreira-Perpinan, "Multiscale Conditional Random Fields for Image Labelling," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 695-702, June 2004.

[16] D. Hoiem, A. Efros, and M. Hebert, "Geometric Context from a Single Image," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 654-661, June 2005.

[17] D. Hoiem, A. Efros, and M. Hebert, "Closing the Loop on Scene Interpretation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* June 2008.

[18] D. Hoiem, A. Efros, and M. Hebert, "Putting Objects in Perspective," *Int'l J. Computer Vision,* vol. 80, no. 1, pp. 3-15, Oct. 2008.

[19] J. Jiang and Z. Tu, "Efficient Scale Space Auto-Context for Image Segmentation and Labeling," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2009.

[20] R. Kassel, "A Comparison of Approaches to On-Line Handwritten Character Recognition," PhD thesis, MIT Spoken Language Systems Group, Massachusetts Inst. of Technology, 1995.

[21] S. Kumar and M. Hebert, "Discriminative Random Fields: A Discriminative Framework for Contextual Interaction in Classification," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1150-1159, Oct. 2003.

[22] S. Kumar and M. Hebert, "A Hierarchical Field Framework for Unified Context-Based Classification," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1284-1291, Oct. 2005.

[23] J. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," *Proc. 10th Int'l Conf. Machine Learning,* pp. 282-289, 2001.

[24] Z. Lao, D. Shen, A. Jawad, B. Karacali, D. Liu, E. Melhem, N. Bryan, and C. Davatzikos, "Automated Segmentation of White Matter Lesions in 3D Brain MR Images, Using Multivariate Pattern Classification," *Proc. Third IEEE Int'l Symp. Biomedical Imaging,* pp. 307-310, Apr. 2006.

[25] C.B. Liu, A. Toga, and Z. Tu, "Fusing Adaptive Atlas and Informative Features for Robust 3D Brain Image Segmentation," technical report, Lab of Neuro Imaging, Univ. of California, Los Angeles, 2009.

[26] G. Mori, X. Ren, A. Efros, and J. Malik, "Recovering Human Body Configurations: Combining Segmentation and Recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 326-333, June 2004.

[27] K.L. Narr, P.M. Thompson, T. Sharma, J. Moussai, R. Blanton, B. Anvar, A. Edris, R. Krupp, J. Rayman, M. Khaledy, and A.W. Toga, "Three-Dimensional Mapping of Temporo-Limbic Regions and the Lateral Ventricles in Schizophrenia: Gender Effects," *Biological Psychiatry,* vol. 50, no. 2, pp. 84-97, 2001.

[28] A. Oliva and A. Torralba, "The Role of Context in Object Recognition," *Trends in Cognitive Sciences,* vol. 11, no. 12, pp. 520-527, Dec. 2007.

[29] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.

[30] S. Pizer, T. Fletcher, Y. Fridman, D. Fritsch, A. Gash, J. Glotzer, S. Joshi, A. Thall, G. Tracton, P. Yushkevich, and E. Chaney, "Deformable m-Reps for 3D Medical Image Segmentation," *Int'l J. Computer Vision,* vol. 55, no. 2, pp. 85-106, 2003.

[31] K. Pohl, J. Fisher, R. Kikinis, W. Grimson, and W. Wells, "A Bayesian Model for Joint Segmentation and Registration," *NeuroImage,* vol. 31, no. 1, pp. 228-239, 2006.

[32] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie, "Objects in Context," *Proc. IEEE Int'l Conf. Computer Vision,* Oct. 2007.

[33] X. Ren, C. Fowlkes, and J. Malik, "Cue Integration in Figure/Ground Labeling," *Proc. Neural Information Processing Systems Conf.,* 2005.

[34] R. Rifkin and A. Klautau, "In Defence of One-vs-All Classification," *J. Machine Learning Research,* vol. 5, pp. 101-141, 2004.

[35] T. Rohlfing, D.B. Russakoff, and J.C.R. Maurer, "Performance-Based Classifier Combination in Atlas-Based Image Segmentation Using Expectation-Maximization Parameter Estimation," *IEEE Trans. Medical Imaging,* vol. 23, no. 8, pp. 983-994, Aug. 2004.

[36] R.E. Schapire, R.E. Freund, P. Bartlett, and W.S. Lee, "Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods," *Annals of Statistics,* vol. 26, pp. 1651-1686, 1998.

[37] J. Shotton, M. Johnson, and R. Cipolla, "Semantic Texton Forests for Image Categorization and Segmentation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2008.

[38] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "TextonBoost: Joint Appearance, Shape and Context Modeling for Multi-Class Object Recognition and Segmentation," *Proc. European Conf. Computer Vision,* pp. 1-15, 2006.

[39] A. Singhal, J. Luo, and W. Zhu, "Probabilistic Spatial Context Models for Scene Content Understanding," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* June 2003.

[40] S. Smith, "Fast Robust Automated Brain Extraction," *Human Brain Mapping,* vol. 17, no. 3, pp. 856-876, 2001.

[41] 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. European Conf. Computer Vision,* 2006.

[42] B. Taskar, C. Guestrin, and D. Koller, "Max-Margin Markov Networks," *Proc. Neural Information Processing Systems Conf.,* 2003.

[43] A. Torralba, K.P. Murphy, and W.T. Freeman, "Contextual Models for Object Detection Using Boosted Random Fields," *Proc. Neural Information Processing Systems Conf.,* 2004.

[44] Z. Tu, "Probabilistic Boosting Tree: Learning Discriminative Models for Classification, Recognition, and Clustering," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1589-1596, Oct. 2005.

[45] Z. Tu, X. Chen, A. Yuille, and S. Zhu, "Image Parsing: Unifying Segmentation, Detection, and Object Recognition," *Int'l J. Computer Vision,* vol. 63, no. 2, pp. 113-140, July 2005.

[46] Z. Tu, K. Narr, P. Dollar, P. Thompson, and A. Toga, "Brain Anatomical Structure Parsing by Hybrid Discriminative/Generative Models," *IEEE Trans. Medical Imaging,* vol. 27, no. 4, pp. 495-508, Apr. 2008.

[47] B. van Ginneken, T. Heimann, and M. Styner, "3D Segmentation in the Clinic: A Grand Challenge," *Proc. Medical Image Computing and Computer-Assisted Intervention Workshop,* 2007.

[48] V. Vapnik, *Estimation of Dependences Based on Empirical Data.* Springer-Verlag, 1982.

[49] J. Verbeek and B. Triggs, "Region Classification with Markov Field Aspect Models," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* June 2007.

[50] P.A. Viola and M.J. Jones, "Robust Real-Time Face Detection," *Int'l J. Computer Vision,* vol. 57, no. 2, pp. 137-154, 2004.

[51] L. Wolf and S. Bileschi, "A Critical View of Context," *Int'l J. Computer Vision,* vol. 69, pp. 251-261, 2006.

[52] R.P. Woods, J.C. Mazziotta, and S.R. Cherry, "MRI-PET Registration with Automated Algorithm," *J. Computer Assisted Tomography,* vol. 17, pp. 536-546, 1993.

[53] Y. Wu and J. Fan, "Contextual Flow," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2009.

[54] J. Yang, L.H. Staib, and J.S. Duncan, "Neighbor-Constrained Segmentation with Level Set Based 3D Deformable Models," *IEEE Trans. Medical Imaging,* vol. 23, no. 8, pp. 940-948, Aug. 2004.

[55] M. Yang, G. Hua, and Y. Wu, "Context-Aware Visual Tracking," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 31, no. 7, pp. 1195-1209, July 2009.

[56] J. Yedidia, W. Freeman, and Y. Weiss, "Generalized Belief Propagation," *Proc. Neural Information Processing Systems Conf.,* 2000.

Zhuowen Tu received the PhD degree from Ohio State University and the ME degree from Tsinghua University. He is an assistant professor in the Laboratory of Neuro Imaging (LONI), Department of Neurology, with a joint appointment in the Department of Computer Science, University of California, Los Angeles (UCLA). He is also affiliated with the Bioengineering IDP program and Bioinformatics IDP program at UCLA. Before joining LONI, he was a member of the technical staff at Siemens Corporate Research. He received the US National Science Foundation (NSF) CAREER award in 2009 and was awarded the David Marr Prize (with collaborators) in 2003. His research has been on the interface of medical imaging, machine learning, statistical modeling/computing, and computer vision. More specifically, he studies the representation problems and develops learning algorithms for medical imaging and computer vision.

Xiang Bai received the BS and MS degrees in electronics and information engineering from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2003 and in 2005, respectively. He is currently working toward the PhD degree at HUST. From January 2006 to May 2007, he worked in the Department of Computer Science and Information, Temple University. From October 2007 to October 2008, he worked in the Laboratory of Neuro Imaging, University of California, Los Angeles, as a joint PhD student. His research interests include shape analysis, computer graphics, computer vision, and pattern recognition.