Efficient Scale Space Auto-Context for Image Segmentation and Labeling

Jiayan Jiang and Zhuowen Tu
Lab of Neuro Imaging, Department of Neurology, Department of Computer Science, UCLA
{jet.jiang, zhuowen.tu}@loni.ucla.edu

Abstract

The Conditional Random Fields (CRF) model, using patch-based classification bound with context information, has recently been widely adopted for image segmentation/labeling. In this paper, we propose three components for improving the speed and accuracy, and illustrate them on a recently developed auto-context algorithm [28]: (1) a new coding scheme for multiclass classification, named data-assisted output code (DAOC); (2) a scale-space approach to make it less sensitive to geometric scale change; and (3) a region-based voting scheme to make it faster and more accurate at object boundaries. The proposed multi-class classifier, DAOC, is general and particularly appealing when the number of class becomes large since it needs a minimal number of $\lceil \log_2 k \rceil$ *binary classifiers for* k *classes. We show advantages of the DAOC classifier over the existing algorithms on several Irvine repository datasets, as well as vision applications. Combining DAOC, the scale-space approach, and the region-based voting scheme for auto-context, the overall algorithm is significantly faster* ($5 \sim 10$ *times) than the original auto-context, with improved accuracy over many of the existing algorithms on the MSRC [24] and VOC 2007 [7] datasets.*

1. Introduction

In an image segmentation/labeling task, each training image comes with a label map in which every pixel is assigned a label of interest. Many existing algorithms [11, 24, 23] perform labeling using the Conditional Random Fields (CRF) model [12], in which patch-based classification is bound with context information. A recently proposed auto-context algorithm [28] targets the posterior probability directly. It integrates image appearances together with context information by learning a series of classifiers. 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, and (2) context information on the classification maps. Initially, the classification maps are uniform over class labels, and thus the context features are not selected by the first classifier. The trained classifier then produces new classification maps which are used, along with local patch features, to train a successive classifier.

In this paper, we propose three components and illustrate them on the auto-context algorithm: (1) a new multiclass classifier, DAOC; (2) a scale-space approach; and (3) a region-based voting scheme.

1. The CRF type of algorithms are not bound to any particular choice of classifier. However, the type of classifier used also largely decides the overall performance (accuracy and speed) of the algorithm. We first propose a new multiclass classifier to boost efficiency in training and testing, without sacrificing much accuracy. It is particularly appealing in dealing with large number (tens or even hundreds) of classes, which is typical in many of the recent learning and vision applications.

Traditionally, the design of multi-class classifier has been mostly driven by reducing the test error (small training error + good generalization). When dealing with a large number of classes on large scale datasets, we are also concerned about the training and testing complexity. In this paper, we focus on the error-correcting output code (ECOC) [1, 6, 9] direction due to its potential of having small computational overhead. ECOC combines several binary classifiers (base learners) through an output coding scheme [1] to produce a coding matrix $M^{k \times l}$, whose entries are in $\{1, -1\}$ [1]. Each column of M is a binary partition of the k classes, and each row is an output code for one class. Then, l binary base learners are trained according to the partitions. A test sample is associated with the class label whose output code is "nearest" to the response vector returned by base learners.

Depending on how the coding matrix is designed, the existing approaches can be roughly divided into 3 categories: (1) Code-oriented design [6, 21, 9, 25], which pursues codes with optimal error-correcting capability; (2) Classifier-driven code design [4], which proposes codes given a set of given base learners to achieve a lower empirical loss; (3) Code-classifier joint design [2, 18, 13], which makes an effort to simultaneously design codes and train

[1]It is also possible to incorporate a "don't care" symbol 0.

base learners, at the cost of significant computational overhead.

Although in principle, one needs a minimal number of $\lceil \log_2 k \rceil$ classifiers in dealing with a k-class problem (as opposed to k classifiers in one-vs-all [19]) using ECOC, most of the existing coding methods produce code length l much larger than k. This is mainly because the codes and classifiers are lacking in the awareness of each other, and accordingly are not cooperating in a harmonic way. To bring the advantage of output coding scheme into full play, we propose a data-assisted coding scheme DAOC to better balance code-optimality and classifier-amicability at a very small computational overhead.

2. Existing segmentation and labeling approaches using Conditional Random Fields [11, 10, 24] or the auto-context algorithm [28] classify image patches at a fixed scale, which are sensitive to large scale-change of objects. In this paper, we make an effort to automatically explore the scale information of the image patch.

3. Performing image labeling based on each pixel is not efficient because it has to scan every pixel in the image and it also produces fuzzy labels on the object boundary due to ambiguity. This "border leakage" effect is undesirable and we try to avoid it by considering region affinity and using a region-based voting scheme in classifying each pixel.

Combining all these three components, we are able to improve the efficiency (both in training and testing) and effectiveness of the auto-context algorithm. We obtained encouraging results on the MSRC [24] and VOC 2007 [7] image segmentation/labeling datasets with $5 \sim 10$ times faster than the original algorithm.

2. Data Assisted Output Code

One aim of this paper is to propose a new data-assisted output code (DAOC) scheme for multiclass classification. The features of our scheme are as follows: (1) As opposed to design codes blind to data, our output codes are incrementally proposed from the underlying data distribution. In this way, we avoid searching in an exponentially large code space and avoid many unnecessarily hard classification problems due to the blindness (see Fig. 1 for an illustration); (2) We pursue each code sequentially based on good data separation, along with good row- and column-separation of the coding matrix. Thus we attain a balance between code-optimality and base-learner-amicability at a small overhead; (3) A probability-based decoding scheme is also proposed in this paper, which further improves the overall classification accuracy. Combining all these features, our scheme can give satisfactory results over a wide range of code lengths, in particular in the cases where $l < k$. This ability is crucial to applications with hundreds of classes. It is noted that in this paper we use AdaBoost as our base learner, but our scheme is general and other classifiers can be used as well.

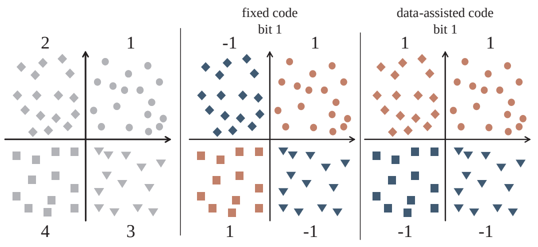

Figure 1. An example of data-assisted output code strategy. The problem is to classify the 4 classes given the two features using binary base learners. Here shows the design for the first bit. A strategy without looking at the data may introduce a XOR problem (in the middle) which is much harder to deal with than the coding by looking at the data (shown in the third column).

2.1. Literature Review

As mentioned in Section 1, we divide the existing output code designs into three categories. Next, we will give a brief review of these methods, along with several possible decoding strategies.

Notations $\{(x_1, y_1), \cdots, (x_m, y_m)\}$ are training samples, where x_i belongs to some input domain X, and $y_i \in Y = \{1, \cdots, k\}$ is the class label. Coding matrix $M \in \{-1, 1\}^{k \times l}$, where l is the code length, column μ_s $(s = 1, \cdots, l)$ is a binary partition of the k classes, and row τ_v^T $(v = 1, \cdots, k)$ is an output code for class v. Base learners give response $g(x) = (g_1(x), \cdots, g_l(x))$ for a test sample x, where $g_(x) \in \{-1, +1\}$.

The error-correcting output codes (ECOC) scheme was first proposed by Dietterich and Bakiri [6]. They suggested to borrow good error-correcting codes from the coding theory. To comply with the independence assumption across bits, they insisted on two criteria in selecting coding matrix: row separation for strong error-correcting property and column separation to decorrelate bits. Although they empirically demonstrated the good performance of their method, the results were obtained under long code bits (for example, 207 bits for a 26-class problem). They concluded that the code-oriented design is robust to particular assignment of codewords to classes. This statement, however, was only validated empirically on the `NETtalk` task. On the contrary, other work showed that the blindness to data *does* have an impact on performance [1, 18, 13]. Another example of code-oriented design is AdaBoost.OC [22] and its variations [9, 25]. AdaBoost.OC combines the benefits of AdaBoost's reweighing and ECOC's relabeling by defining joint weights over samples and incorrect labels. It incrementally generates partitions and trains base learners until a specified round limit is achieved. Unfortunately, finding optimal partition is an instance of MAX-CUT problem, and hence is NP-complete. Another problem is that extremely hard binary problems may arise during the process [13]. For these reasons, researchers usually turn to a random partitioning strategy [22], with very long code bits ($l \geq 500$).

Crammer and Singer [4] made an attempt to learn a coding matrix given a set of binary base learners. They showed the original discrete problem is NP-complete and turned to solving a relaxed problem. Indeed, the design of output codes and the training of base learners are chicken-and-egg problem, and breaking this naturally interleaving process will inevitably make the classifier-driven code design sub-optimal. Our approach is very different from [26], in which the number of classifiers is linear to the class number. Features are naturally shared in our DAOC classifier.

It seems promising to jointly optimize coding matrix and base learners simultaneously. However, as stated before, this problem is NP-hard and generally only local optimal solutions can be obtained. An early attempt in code-classifier joint design was made by Alpaydin and Mayoraz [2], where they showed this strategy is equivalent to a two-layer multilayer perceptron, when base learners are restricted to a linear form and Hamming-decoding is used. More recently, Rätsch et al. [18] proposed an algorithm which utilizes alternating method to achieve the joint optimization of coding matrix and base learners. Therefore, less bits might be needed since codes and base learners are more mutually aware. On the other hand, their optimization is a batch procedure, and thus a slight change of code length will require a re-training from scratch. Li [13] incorporated a similar alternating strategy into each round of AdaBoost.ECC, which is a variant of AdaBoost.OC. The drawbacks of alternating method are (1) only local optimum is guaranteed; and (2) significant overhead is involved.

Along the line of decoding/testing of an unseen sample x, there are also several methods in the literature. Perhaps the simplest method is to choose the class label v^* which has the minimum Hamming distance to the base learners' responses:

$$d_H(\tau_v, g(x)) = \sum_{s=1}^{l} \left(\frac{1 - \text{sign}(\tau_v(s) g_s(x))}{2} \right) \qquad (1)$$

The disadvantage of Hamming decoding is that it ignores the confidence of each base learner. One remedy is to incorporate a weighted Hamming strategy:

$$d_{WH}(\tau_v, g(x)) = \sum_{s=1}^{l} \alpha_s \left(\frac{1 - \text{sign}(\tau_v(s) g_s(x))}{2} \right) \qquad (2)$$

where weight α_s reflects sth base learner's error rate on the training set. AdaBoost.OC [21] and Maximum Likelihood Decoding [9] adopt this method. Another way is to take into account the margin information of each base learner if possible, which gives the loss-based decoding strategy [1]:

$$d_{Loss}(\tau_v, g(x)) = \sum_{s=1}^{l} \exp(-\tau_v(s) g_s(x)) \qquad (3)$$

2.2. The DAOC algorithm

Given a training set $\{(x_1, y_1), \cdots, (x_m, y_m)\}$ with $y_i \in Y = \{1, \cdots, k\}$, the task is to find the best classification with coding matrix M and l binary classifiers. Ideally, one would exhaustively search through all possible coding strategies with the corresponding binary classifiers trained. This requires to train a total number of $2^{k \times l} \times l$ classifiers, which is computationally infeasible. Our data-assisted output code (DAOC) performs training in the following way: (1) Each code bit is assigned sequentially; (2) For each bit we look at the data feature and choose the best solution to balance data-, row-, and column-separation (we also keep an option to use designed code when no data feature meets the requirement of row- and column-separation).

2.2.1 Data-Assisted Code Pursuit

As discussed in Section 2.1, one of the major difficulties in existing designs is that codes and base learners cannot cooperate in a harmonic way at a small computational overhead. Next we tackle this problem using a data-assisted approach. In this paper, we use AdaBoost as base binary learner. Nevertheless, the approach is general and one can use other classifiers as well.

In general, test error for a classifier is bounded by its training error and VC dimension, and the training error ϵ in AdaBoost is bounded by

$$\epsilon \leq 2^T \prod_{t=1}^{T} \sqrt{\epsilon_t (1 - \epsilon_t)}, \qquad (4)$$

where T is the total number of weak classifiers. Indeed, we can further show that ϵ_{t+1} is bounded by the error in the previous round, ϵ_t [29].

At step t, let the error for a candidate weak hypothesis $h^{(j)}$ be $\epsilon_t^{(j)} = \sum_{i=1}^{m} D_t(i) \left[\mu(y_i) \neq h^{(j)}(x_i)\right]$ and ϵ_t be the best error achieved. Apparently, $\epsilon_t \leq \epsilon_t^{(j)}; \forall j$. $\epsilon_t = 0$ when the hypothesis perfectly classifies the data and $\epsilon_t = \frac{1}{2}$ when it makes random guesses: $\frac{1}{2} \geq \epsilon_t \geq 0$. Then, the best weak hypothesis picked for $t+1$ can best achieve:

$$\epsilon_{t+1} \geq \frac{\epsilon_t}{2(1 - \epsilon_t)}. \qquad (5)$$

The above fact shows that for AdaBoost, the error for the next round is lower-bounded by the error for the current round. This implies that to facilitate the training of AdaBoost, one effective way is to reduce the data complexity so that the error rate for its first weak hypothesis is as low as possible (and consequently subsequent weak hypotheses are less bounded). This is possible in the context of output coding scheme by feeding AdaBoost a well chosen partition μ. Another observation is that marginal class separability along each feature is a good indicator for the difficulty of the resulting binary classification problem. Our experimental results show that similar improvement can be observed for AdaBoost using decision stump and decision tree as weak classifiers. We expect that a similar conclusion might still hold for other types of classifiers.

Based on these observations, our data-assisted approach examines two marginal histograms:

$$q_j^L(v) = \sum_{i=1}^{m} [v = y_i] [f_j(x_i) \le \theta] \quad (6)$$

$$q_j^R(v) = \sum_{i=1}^{m} [v = y_i] [f_j(x_i) > \theta] \quad (7)$$

and the weighted marginal entropy measures the class separability along feature f_j at a particular threshold θ:

$$\begin{aligned} WEnt(f_j, \theta) &= Z_j^L \times Entropy(q_j^L / Z_j^L) \\ &+ Z_j^R \times Entropy(q_j^R / Z_j^R) \end{aligned} \quad (8)$$

where Z_j^L and Z_j^R are normalization constants for q_j^L and q_j^R respectively. Intuitively, the smaller $WEnt$ is, the better classification accuracy a decision-stump could achieve, and this in turn implies a better overall performance of AdaBoost according to the above fact. This criterion is inspired by the information gain used in decision tree [16]. It is noted that this search is linear to the number of data features and can be done very efficiently. The candidate output code proposed by f_j and θ is:

$$\mu(v) = \begin{cases} -1 & \text{if } q^L(v) \ge q^R(v) \\ +1 & \text{otherwise} \end{cases} \quad \forall v = 1, \cdots, k \quad (9)$$

The above data-assisted candidate codes reflect the notion of classifier-amicability. To further reflect the optimality of codes, we consider the following criteria when incrementally augmenting the sth column μ_s to coding matrix M:

$$CSep(\mu_s) = \min_{s' < s} \{d_H(\mu_s, \mu_{s'}), d_H(\mu_s, \bar{\mu}_{s'})\} \quad (10)$$

$$RSep(\mu_s) = \min_{v, v'} d_H(\tau_v(\mu_s), \tau_{v'}(\mu_s)) \quad (11)$$

where $\mu_{s'}$ $(s' < s)$ are the previously chosen columns and $\bar{\mu}_{s'}$ denotes the bitwise complement of $\mu_{s'}$. $CSep$ measures the minimum Hamming distance of μ_s to any previously selected code and its complement, and $RSep$ measures the minimum Hamming distance among all pairwise augmented class codes τ_v $(v = 1, \cdots, k)$ at round s. These two measures reflect the column- and row-separation requirements for a strong error-correcting code design. Finally, the optimal partition is chosen by:

$$\begin{aligned} \mu_s^* &= \arg\min_{\mu_s} (\lambda \cdot WEnt - CSep(\mu_s) - RSep(\mu_s)) \\ s.t. &\quad CSep(\mu_s) > 0 \text{ and } RSep(\mu_s) > 0 \end{aligned} \quad (12)$$

where λ is a parameter to balance classifier-amicability and code-optimality. The hard constraints are to keep us away from particularly bad code designs.

The data-assisted approach effectively returns good output codes without invoking base learners by focusing on more promising area in the exponentially large code space. The down side is that due to the incompleteness of data-assisted code space, occasionally the hard constraints in (12) may rule out every proposed code. In this case, we turn to a random partitioning strategy as in [1]. Our experiment, however, shows that this situation seldom happens in practice.

2.2.2 Probability-Based Decoding

In this part, we introduce our probability-based decoding method. Compared with other methods reviewed in Section 2.1, this method interprets the output of AdaBoost as estimates of posterior probabilities [8], and combines their responses within the more principled probability framework.

Based on the assumption that code bits are independent of each other, which is partly validated by the constraints in (12) during training, our predicted class label is:

$$v^* = \arg\max_{v} \prod_{s=1}^{l} \left(\frac{\exp(2\tau_v(s) g_s(x))}{1 + \exp(2\tau_v(s) g_s(x))} \right) \quad (13)$$

Like (3), our decoding considers the margin information given by AdaBoost. The major difference lies in that in (13), each AdaBoost's confidence is normalized, and thus different voices are combined in a more fair way. In the next section, we will show that this approach does improve accuracy over other decoding methods.

2.3. Experiments

To better evaluate the DAOC scheme, we did five devoted experiments on UCI repository. The descriptions of used datasets are summarized in Table 1. For the encoding part, we compare DAOC with traditional data-blinded ECOC [21, 1, 19]. We use a scheme to chooses μ_s at random, but ensuring an approximately half/half partition of all class labels. Although this scheme targets good error-correcting property, it does not look at the data and suffers from the blindness problem. Also, we compare DAOC with one-vs-all strategy. It is noted that the latter is only applicable for $l = k$, and becomes unaffordable when k is large. Finally, to have a comparison with fine-tuned error-correcting codes, we include a fixed ECOC scheme for the 10-class problem `optdigits`. The 15-bit fixed code is borrowed from Table 3 of [6]. For the decoding part, we compare Hamming-based, loss-based, and probability-based decoding strategies. All methods use AdaBoost as base learner, with 100 weak hypotheses (both stump and decision-tree are tested, and conclusions are consistent). The parameter λ is fixed to 0.2 for all the experiments.

Ten trials are conducted for each algorithm and dataset. In each trial, we randomly sample a training set and use the remaining data as test set. Results on `optdigits` and `isolet` with stump and decision-tree as weak hypotheses are shown in the first three columns of Fig. 3. It can be seen that DAOC produces satisfactory results over a wide span of code lengths. It performs consistently better than the data-blinded ECOC scheme, and its error rate is comparable to the fine-tuned codes at its fixed code length. One-vs-all performs slightly better than DAOC at its fixed code length. We emphasize that the advantage of DAOC lies in the region $l < k$, which will be shown later. It is also observed that the probability-based decoding achieves lower error than loss-based decoding, which in turn outperforms simple Hamming decoding. For the limit of space, only probability-based decoding results are plotted for `satimage`, `ecoli`, and `vowel`, in the fourth column of Fig. 3. The observations on these three datasets are consistent with the above experiments.

Since the performance metric error rate does not reflect the computational cost (the number of binary classifiers used, i.e. l) in training and testing, here we design a different performance score as follows:

$$S(l, \epsilon_l) = \sigma(\epsilon_l) + \sigma(\beta \times (l/k)) \tag{14}$$

where l denotes the current code length, ϵ_l is the error rate under l, and $\sigma(x) = 1/(1 + e^{-x})$ is the logistic function. Note that score S (the smaller the better) reflects both effectiveness (first term) and efficiency (second term) and β is a balancing parameter (we use $\beta = 0.5$ in this paper). In the last column of Fig. 3, results on `optdigits`, `vowel`, and `isolet` are plotted based on score S. Under this metric, it can be seen that DAOC maintains a good balance between effectiveness (low test error) and efficiency (short code length, or fewer binary classifiers).

Table 1. Description of the datasets

name	k	# attribute	# train	# test
satimage	6	36	4000	2435
ecoli	8	8	200	136
optdigits	10	64	3500	2120
vowel	11	10	500	490
isolet	26	617	6000	1797

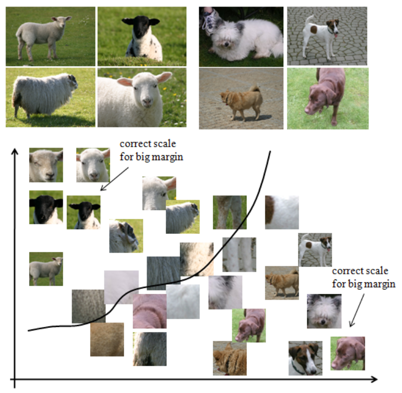

Figure 2. Illustration of the scale space image patches in training.

3. Scale Space Image Patch

The scale-space problem in image analysis has been a long standing problem [14]. Existing image segmentation/labeling approaches using patch-based classification [11, 10, 24, 28] works on fixed size, which are sensitive to big scale changes. To achieve scale-invariance is not a trivial task, since even in the training images, no scale "ground truth" is provided explicitly for each pixel. Recent popular feature extraction algorithms [15] try to find the most informative scale automatically and another method [5] studies the best scale for a particular task through empirical studies. We aim to build an efficient automatic algorithm to explore the scale space, both in training and testing. Since we do not know what the right scale is at each pixel, one interesting approach to try is the multiple instance learning (MIL) methods. At each pixel, we may collect a bag in which there are many patches at different scales. A MIL method then should be able to successfully find the right scale by minimizing the error for all the bags. In our task of image segmentation/labeling, we found it not very effective, probably due to the large intra-class variation. In this exercise, we simply collect image patches at several scales (3 scales: 21×21, 41×41, 61×61) as the training samples. According to the margin theory of classifier, samples with high confidence should have large margin and those confusing ones are mostly on the decision boundary. Therefore, by simply augmenting the volume of training samples, we let the classifier automatically explore the right scale. Fig. 2 shows an illustration. In testing, one can either assign the probability to each pixel with the most confident one, or average the confidences across all the scales. For the MSRC dataset, we see an improvement from 61% to 64% by using the scale space image patch with both using otherwise the same setting (the same DAOC classifier on the same set of features and measured on patch-based classification only with no context information).

4. Region Based Voting Scheme

Performing image labeling based on each pixel is not efficient because it has to scan every pixel in the image and it also produces fuzzy labels on the object boundary due to ambiguity. In other systems [17, 30], classification is performed on interest points/features extracted from segmented regions. However, segmentation is often not stable (though using multiple segments [20] might help a bit) and feature extraction methods are not always satisfactory. In this paper, we take a different approach, which is a region-based voting scheme. Given an image, we use the mean-shift algorithm [3] to perform over-segmentation. A set of pixels (usually $5\% \sim 8\%$) are then randomly selected in each region to perform classfication. The overall discriminative probability (often a vector) is simply the average of the probabilities of these sampled pixels. We achieve three goals by doing this: (1) speed up testing since only

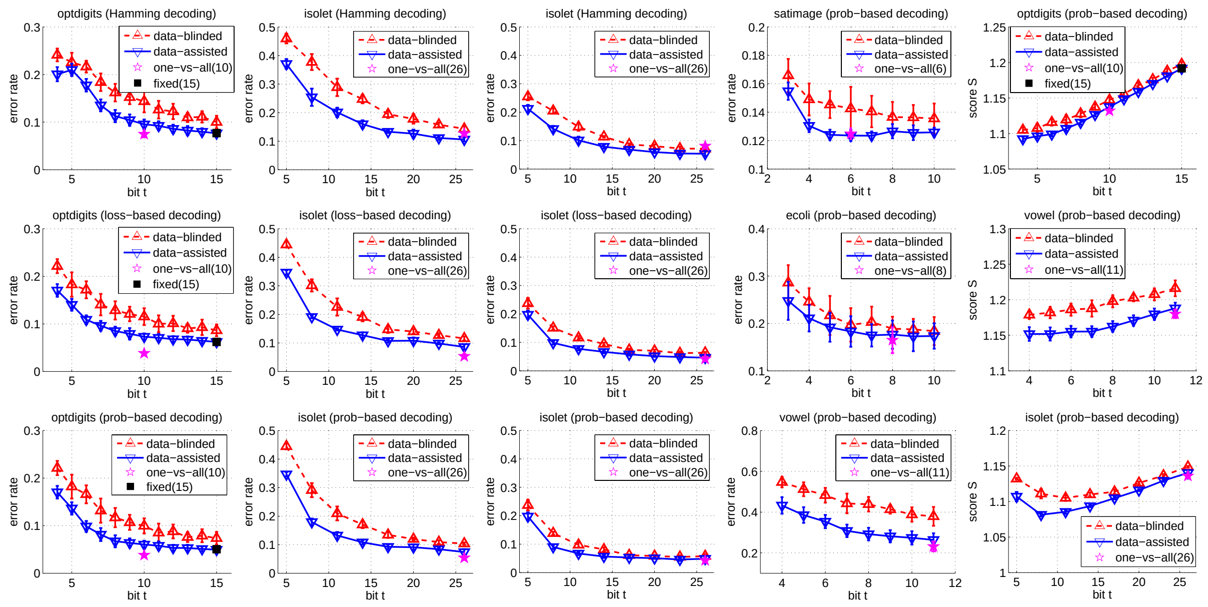

Figure 3. Results on UCI datasets. The first two columns show the results on `optdigits` and `isolet` using different decoding strategies with decision-stump as weak hypothesis. The third column shows results on `isolet` using decision-tree as weak hypothesis. The fourth column shows results on `satimage`, `ecoli`, and `vowel` using probability-based decoding. The last column shows the results on `optdigits`, `vowel`, and `isolet` based on score S, which reflects both effectiveness and efficiency of a classifier under different code lengths. DAOC achieves the best overall performance.

a fraction of pixels are tested; (2) obtain improved accuracy on the boundaries since the final result is complied with intensity-based segmentation; (3) have the possibility to use region-based features such as shape, in addition to features computed from the image patch. Compared to the approaches directly classifying segmented regions, our approach is less dependent on the segmentation result and the robustness of the feature extraction algorithms. Patch-based features are still able to be used. It is noted that we still use the full set of pixels in the training stage to leave the maximum amount of information for the classifiers. Fig. (4) shows an illustration of our approach.

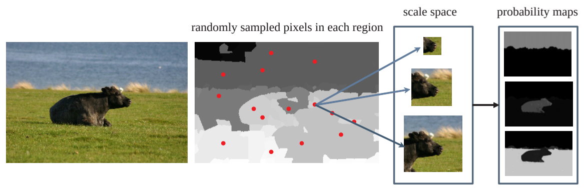

Figure 4. Pixels are randomly sampled in segmented regions by the mean-shift algorithm in testing.

5. Outline of The Algorithm

Here we give the outline of our algorithm.

In training, given a set of images with the corresponding label maps of k classes: (1) Start with initial probability maps of uniform distribution for all the images. (2) Collect image patches of different scales (typically 3), and these are the training samples. (3) Compute both the appearance features, from the image patch, and the context features, from the probability maps [28]. (4) Train a DAOC multi-class classifier (typically $\lceil log_2 k \rceil$ plus 2 additional bits for the error correction). (5) Update the probability maps using trained classifier. (6) Go back to step 2 and repeat the procedure (auto-context).

In testing: given a test image: (1) Start with initial probability maps of uniform distribution. (2) Perform segmentation using the mean-shift algorithm (takes about $0.1s$). (3) For each region, randomly select a set of pixels ($5\% \sim 8\%$). (4) Perform classification on different scales of image patches centered at the pixel. (5) Compute the average probability vector from all scales (one could also use the probability vector which has the smallest entropy but averaging gives better result). (6) Assign each region with the discriminative probability vector averaged over all sampled pixels. (7) Go back to step 3 and repeat the procedure for the number of iterations trained for the auto-context.

6. Experimental Results

We illustrate the proposed algorithm on two widely used and tested datasets: the MSRC dataset [24] and the VOC 2007 dataset [7]. In both the cases, we use the identical set of training and test images provided by the dataset designer. The accuracy of the labeling results are shown in Fig. (7) and Fig. (8). As we stated before, three components are proposed to improve both efficiency and accuracy: (1) a new multi-class classifier (DAOC); (2) a scale-space approach for classification; and (3) a region-based voting scheme. Compared with the original auto-context algorithm with PBT [27] as binary classifiers, DAOC effectively reduces the training and testing time by a significant level (about $5 \sim 10$ times), without sacrificing much accuracy. The scale-space approach positively affects the final results and the region-based voting scheme not only leads to a faster testing stage, but also sharpens the object boundaries. It takes around 10 seconds to segment/label an image in testing.

For the MSRC dataset, the overall per-pixel accuracy is 78%, which is the same as the original auto-context algorithm. However, there was a post-process smoothing stage in the auto-context and the result before smoothing was 74.5%. In addition, the segmentation by the proposed algorithm has much more accurate boundaries than those by the auto-context algorithm. After a careful examination, we found that many of the ground truth labels in the MSRC dataset are not accurate, which may bring about unfair comparison between algorithms. There are mainly two types of inaccuracies in the ground truth labels: (1) Humans are biased towards "more important objects" and tend to favor a high false positive and an extremely low false negative rate of these objects. The result is a lot of background pixels (e.g. grass, sky) are labeled as foreground objects (e.g. cow, bird); (2) Humans are sometimes not consistent themselves. The result is sometimes an area of background pixels is not labeled or the "void" region between different objects is fairly wide. To rectify these two effects, we take an effort to refine the labels of the test images. Some examples are shown in Fig. 5. Performing measurement based on the refined labels, our new algorithm achieves a per-pixel accuracy of 81%, with the original auto-context achieving 76% and 79% before and after post-processing respectively.

On the VOC2007 and VOC2008 datasets, we use the identical parameter setting for the training and testing, as those in the MSRC dataset. The overall average segmentation accuracy across all classes is 30%, which is better than 24% reported in [23], and comparable to the best reported TKK algorithm [7], which is specifically designed for this task. Also, our per-pixel accuracy is much higher, 41% than that in TKK, 24%. The per-pixel accuracies for Shotton et al., Brookes, TKK, and ours are respectively 58%, 21%, 24%, and 41%. However, the background has 71.5%, and therefore a default algorithm which simply assigns all the pixels is not bad on per-pixel measure. Nevertheless, our algorithm achieves the-state-of-the-art performance averaged based on all classes, while having good per-pixel performance. In Fig. (8), we also report the result on the VOC2008 dataset using 4336 training images and 512 test images provided by the organizers (without using external data). The accuracy measures in VOC2007 and VOC2008 are different.

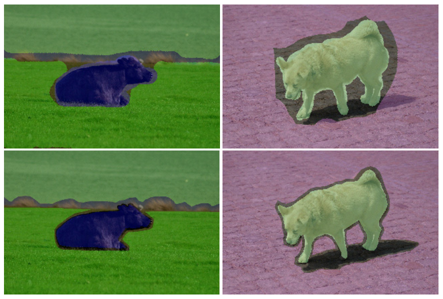
Figure 5. Illustration of manually refined ground truth for the test images.

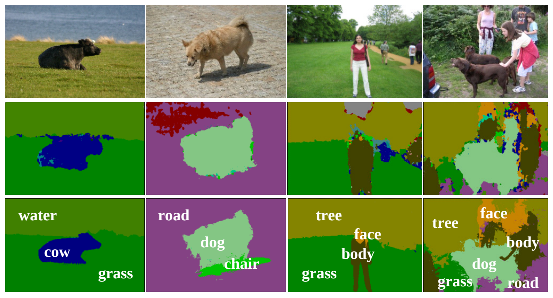

Figure 6. Results on some test images of MSRC. The first row shows the input images. The second shows the results reported in [28] and the third row shows the results by our algorithm. As we can see, the segmentation boundaries are much more accurate with improved recognition rates.

7. Conclusion

In this paper, we have proposed three components which improved efficiency, as well as accuracy, for the auto-context algorithm. We anticipate the same performance gain for many CRF-based vision applications. The new data-assisted output coding (DAOC) scheme effectively reduces the training and testing complexity of the existing multi-class classifiers, by seeking a balance between code-optimality and classifier-amicability at a small overhead. DAOC is general and can be used as a generic multi-class classifier. We have illustrated its advantages on both the traditional machine learning datasets and vision applications. A scale-space approach was also included to make patch-based classification more robust to scale-change of objects. Finally, a region-based voting scheme was used to improve the accuracy and drastically reduce the testing time. Equipped with all these components, our new system attains state-of-the-art segmentation/labeling results on the MSRC and VOC2007, with significant reduction ($5 \sim 10$ times) in training and testing time than the original auto-context algorithm. It typically takes about 10 seconds per-image to perform segmentation/labeling in testing. The algorithm by Shotton et al. [23] using random forest achieves nearly real-time in testing, and is also faster in training than

	building	grass	tree	cow	sheep	sky	airplane	water	face	car	bicycle	flower	sign	bird	book	chair	road	cat	dog	body	boat	**Global**	**Average**
Shotton et al.	49	88	79	97	97	78	82	54	87	74	72	74	36	24	93	51	78	75	35	66	18	72	67
Auto-context	69	96	87	78	80	95	83	67	84	70	79	47	61	30	80	45	78	68	52	67	27	78	69
Ours	53	97	83	70	71	98	75	64	74	64	88	67	46	32	92	61	89	59	66	64	13	78	68
	On refined ground truth labels																						
Auto-context	71	93	88	81	83	94	89	67	89	72	78	48	64	31	80	48	78	70	55	70	28	79	70
Ours	58	96	86	83	78	98	81	66	80	66	91	70	50	36	92	67	90	61	71	69	14	81	72

Figure 7. MSRC segmentation/labeling results.

	background	airplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motorbike	person	plant	sheep	sofa	train	tv/monitor	Average
Brookes	78	6	0	0	0	0	9	5	10	1	2	11	0	6	6	29	2	2	0	11	1	9
Shotton et al.	20	66	6	15	6	15	32	19	7	7	13	44	31	44	27	39	35	12	7	39	23	24
TKK	23	19	21	5	16	3	1	78	1	3	1	23	69	44	42	0	65	30	35	89	71	30
Ours (voc2007)	51	15	14	29	0	25	25	19	17	35	15	63	29	33	45	20	44	29	8	46	40	30
Ours (VOC2008)	68	20	0	9	10	7	11	9	12	3	3	20	5	13	20	20	11	17	3	15	17	14

Figure 8. VOC2007 and VOC 2008 segmentation/labeling results. Note that the accuracy measures in VOC2007 and VOC 2008 are different. For each class, in VOC2007, accuracy=(true positive)/(ground truth), whereas in VOC2008, accuracy=(true positive)/(ground truth + false positive + false negative).

our algorithm. However, we have significantly better accuracies. Our result on the VOC2007 and VOC2008 datasets is still far from being fully satisfactory. This suggests that future research is required to probably study more explict models.

Acknowledgments This work is supported by Office of Naval Research Award, No. N000140910099. Any findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Office of Naval Research.

References

[1] E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *J. of Mac. Learn. Res.*, 1:113–141, 2000. 1, 2, 3, 4

[2] E. Alpaydin and E. Mayoraz. Learning error-correcting output codes from data. In *Proc. of ICANN*, 1999. 1, 3

[3] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Analysis Machine Intell.*, 24(5), 2002. 5

[4] K. Crammer and Y. Singer. On the learnability and design of output codes for multiclass problems. *Machine Learning*, 47:201–233, 2002. 1, 3

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conf. on Comp. Vis. and Patt. Recog., CVPR'2007*, June 2005. 5

[6] T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *J. of Art. Int. Res.*, 2:263–286, 1995. 1, 2, 4

[7] M. Everingham, L. V. Gool, C. K. I.Williams, J.Winn, and A. Zisserman. The pascal voc challenge 2007. In *http://www.pascalnetwork.org/challenges/VOC/voc2007/workshop/index.html*. 1, 2, 6, 7

[8] J. Friedman, T. Hastie, and R. Tibshirani. *Additive logistic regression: a statistical view of boosting*. Dept. of Stat., Stan. Univ. Tech. Rep., 1998. 4

[9] V. Guruswami and A. Sahai. Multiclass learning, boosting, and error-correcting codes. In *Proc. of COLT*, 1999. 1, 2, 3

[10] X. He, R. Zemel, and M. Carreira-Perpinan. Multiscale conditional random fields for image labelling. In *Proc. of CVPR*, June 2004. 2, 5

[11] S. Kumar and M. Hebert. Discriminative random fields: a discriminative framework for contextual interaction in classification. In *Proc. of ICCV*, Oct. 2003. 1, 2, 5

[12] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *Proc. of 10th Int'l Conf. on Machine Learning*, pages 282–289, San Francisco, 2001. 1

[13] L. Li. Multiclass boosting with repartitioning. In *Proc. of ICML*, 2006. 1, 2, 3

[14] T. Lindeberg. *Scale-space theory in computer vision*. Dordrecht: Kluwer Academic, 1994. 5

[15] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int'l J. of Comp. Vis.*, 60(2):91–110, 2004. 5

[16] J. Quinlan. Improved use of continuous attributes in c4.5. *J. of Art. Intell. Res.*, 4:77–90, 1996. 4

[17] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *Proc. of ICCV*, Oct. 2007. 5

[18] G. Rätsch, A. J. Smola, and S. Mika. Adapting codes and embeddings for polychotomies. In *Proc. of NIPS*, 2002. 1, 2, 3

[19] R. Rifkin and A. Klautau. In defence of one-vs-all classification. *J. Mach. Learn. Res.*, 5:101–141, 2004. 2, 4

[20] B. C. Russell, W. T. Freeman, A. A. Efros, J. Sivic, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *Proc. of CVPR*, 2006. 5

[21] R. E. Schapire. Using output codes to boost multiclass learning problems. In *Proc. of ICML*, 1997. 1, 3, 4

[22] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999. 2, 3

[23] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *Proc. of CVPR*, 2008. 1, 7

[24] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for mult-class object recognition and segmentation. In *Proc. of ECCV*, 2006. 1, 2, 5, 6

[25] Y. Sun, S. Todorovic, J. Li, and D. Wu. Unifying the error-correcting and output-code adaboost within the margin framework. In *Proc. of ICML*, 2005. 1, 2

[26] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Tran. on Patt. Ana. and Mach. Inte.*, 29(5):854–869, May. 3

[27] Z. Tu. Probabilistic boosting tree: Learning discriminative models for classification, recognition, and clustering. In *Proc. of Int'l Conf. on Comp. Vis. (ICCV)*, pages 1589–1596, Beijing, Oct. 2005. 7

[28] Z. Tu. Auto-context and its application to high-level vision tasks. In *Proc. of Comp. Vis. and Pat. Recog. (CVPR)*, June 2008. 1, 2, 5, 6, 7

[29] Z. Tu, X. S. Zhou, A. Barbu, L. Bogoni, and D. Comaniciu. Probabilistic 3d polyp detection in ct images: The role of sample alignment. In *Proc. of Comp. Vis. and Pat. Recog. (CVPR)*, June 2006. 3

[30] L. Yang, P. Meer, and D. J. Foran. Multiple class segmentation using a unified framework over mean-shift patches. In *Proc. of CVPR*, June 2007. 5