Probabilistic 3D Polyp Detection in CT Images: The Role of Sample Alignment

Zhuowen Tu[1], Xiang Sean Zhou[3], Adrian Barbu[2], Luca Bogoni[3], Dorin Comaniciu[2]
[1] Lab of Neuro Imaging, Dept. of Neurology, UCLA
[2] Integrated Data Systems Dept., Siemens Corporate Research,
[3]CAD Solutions, Siemens Medical Solutions

Abstract

Automatic polyp detection is an increasingly important task in medical imaging with virtual colonoscopy [15] being widely used. In this paper, we present a 3D object detection algorithm and show its application on polyp detection from CT images. We make the following contributions: (1) The system adopts Probabilistic Boosting Tree (PBT) to probabilistically detect polyps. Integral volume and 3D Haar filters are introduced to achieve fast feature computation. (2) We give an explicit convergence rate analysis for the AdaBoost algorithm [2] and prove that the error at each step ϵ_{t+1}. is tightly bounded by the previous error ϵ_t. (3) For a 3D polyp template, a generative model is defined. Given the bound and convergence analysis, we analyze the role of "sample alignment" in the template design and devise a robust and efficient algorithm for polyp detection. The overall system has been tested on 150 volumes and the results obtained are very encouraging. [1]

1. Introduction

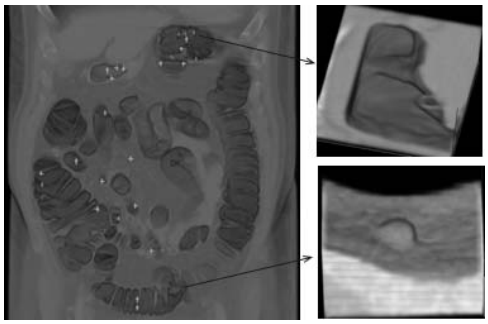


Figure 1. A view of an input CT volume. Crosses show some possible locations of polyps. The top figure on the right is a colon fold, and the bottom one displays a true polyp. More examples can be seen in Fig. 14 and 15

[1]The first author was invovled in this work when he as an empolyee in the integrated data systems department at Siemens Corporate Research, Princeton, New Jersey.

Virtual colonoscopy [15] has become a widely used technology in clinical trials. It takes a 3D volume from CT and reconstructs 3D surfaces of colon wall allowing doctors to virtually look inside the colon to detect polyps and cancers. The size of a typical CT volume ranges from $450\times512\times512$ to $800 \times 512 \times 512$. During the examination, the colon is inflated with air, which appears to be dark. Colon folds and other tissues have gray intensity values. The left figure in Fig. (1) shows a view of a 3D volume. A polyp is a half-hemisphere like structure on the colon wall. If it is left untreated, it will likely develop into a cancer. Detecting polyps using the virtual colonoscopy technology is not so easy since many other objects such as colon folds and residual materials have similar shapes and intensity patterns as those of polyps. Therefore, automatic polyp detection is one of the keys to the further success of the virtual colonoscopy technology [16]. The size of a polyp is measured by its diameter. Usually, a polyp smaller than $6mm$ is not of much clinical significance [12]. Polyps bigger than $9mm$ are very likely to be cancers and can be identified by doctors easily. It is most important for the algorithm to be able to detect polyps in $6 \sim 9mm$ range since they may develop into cancers. Fig. (1) shows a sample volume and the bottom-right figure displays a true polyp.

A number of polyp detection algorithms have been proposed recently [4, 6, 7, 16] with good results reported. The surfaces of polyps are specifically studied in [4] by Jerebko *et al*. Other methods [6, 7, 16] study geometric features and various heuristics such as shape index. But most of them need to perform colon segmentation first. Moreover, the features/heuristics used are very limited and mostly manually designed. This creates a problem when polyps and the backgrounds have large intra-class variation and inter-class similarity. Also, it is not clear how these methods can be directly extended to similar problems, for example, lymph-node or lung cancer detection.

In this paper, we present a learning based approach for 3D object detection and show its application on polyp detection. We define a generative model to capture the underlying generation process. In the corresponding discrimina-

tive model, we show that different choices of *sample alignment* lead to different performances of the classifier. We give an explicit convergence rate analysis for the AdaBoost algorithm [2] and prove that the error at each step ϵ_{t+1} is tightly bounded by the previous error ϵ_t. This matches our intuition that the convergence of AdaBoost becomes slow when all the candidate weak classifiers are not so informative, though asymptotically, the overall error in AdaBoost is decreasing. This gives us a theoretical explanation of why it is often desirable to align the training data to reduce the data complexity. For example, faces are clustered into a few typical angles in [14, 8] in detection. For 3D objects, we design *integral volumes* and 3D Haar filters for fast computation of the features. A cascade of Probabilistic Boosting Tree (PBT) is adopted. The overall algorithm has been tested on 150 volumes and the results obtained are very encouraging. Our learning based algorithm automatically selects around one thousand features from 50,000 candidate features. The approach does not have the pre-segmentation stage. Our method is adaptive and has the potential to be applied in other tasks such as lymph-node detection with no articulation.

2. Problem Formulation

We start our discussion from a generative model of generic objects. Let $\Psi = \{\Delta_1, \Delta_2, ...\}$ be a dictionary in which Δ_i denotes a 2D/3D template. Each object instance $\mathbf{x}$ we observe is assumed to be generated by a transformation function T on Ψ. A set of typical parameters in T, $\Theta = \{l, s, \theta, \phi, \alpha\}$, can be: l– the number of template used, s– scale, θ–rotation, ϕ–deformation, and others which we summarize into α. Let y be the label of $\mathbf{x}$ and $y = +1$ if $\mathbf{x}$ is an object of interest (positive), and $y = -1$ if it is not of interest (negative). The probability of $\mathbf{x}$ can be modeled by a generative model as

$$p(\mathbf{x}|\Theta, y; \Psi) = \frac{1}{Z} exp\{-||\mathbf{x} - T(\Theta, \Psi)||\}, \qquad (1)$$

where $||\ \ ||$ defines a distance measure between $\mathbf{x}$ and $T()$, and Z is the partition (normalization) function $Z = \sum_{\mathbf{x}} exp\{-||\mathbf{x} - T||\}$. Usually, $p(\mathbf{x}|\Theta, y = +1; \Psi)$ has low entropy since it is focused on a specific class of object. The samples are drawn from a limited number of templates in Ψ. $p(\mathbf{x}|\Theta, y = -1; \Psi)$ often has high entropy because it covers all the objects not of interest.

In a discriminative approach, we want to classify a sample $\mathbf{x}$ based on the posterior

$$p(y|\mathbf{x}). \qquad (2)$$

Or, we can explicitly look for some parameters in Θ, Θ_1 to compute

$$p(y|\mathbf{x}) = \sum_{\Theta_1} p(y|\Theta_1, \mathbf{x}) p(\Theta_1|\mathbf{x}). \qquad (3)$$

Suppose we have a supervised learning algorithm which learns a discriminative model based on a set of training samples $\{(\mathbf{x}_i, y_i), i = 1...N\}$. What forms of the samples we use for training and under what Θ_1 will have a great impact on the performance and speed of the classifier. This is an "alignment" issue and the question is how much aligned we want our sampel to be. This is a common problem we are facing in many object recognition/detection tasks using discriminative approaches. Next, we specifically discuss it in the context of AdaBoost [2, 10].

2.1. Convergence Rate of AdaBoost

Let $\{(x_i, y_i, D_1(i)), i = 1...N\}$ be a set of training samples and $D_1(i)$ is the distribution for each sample x_i. AdaBoost algorithm [2, 10] proposed by Freund and Schapire learns a strong classifier $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$, based on the training set, by combining a number of weak classifiers. We briefly give the general AdaBoost algorithm[2, 10] below:

Given: $(x_1, y_1, D_1(1)), ..., (x_N, y_N, D_1(N))$; $y_i \in \{-1, 1\}$
For $t = 1, ..., T$:

- Train weak classifier using distribution D_t.
- Get weak hypothesis $h_t : \chi \rightarrow \{-1, +1\}$.
- Calculate the error of h_t : $\epsilon_t = \sum_{i=1}^{N} D_t(i) \mathbf{1}_{(y_i \neq h_t(x_i))}$.
- Set $\beta_t = \epsilon_t / (1 - \epsilon_t)$ and $\alpha_t = -\log \beta_t$.
- Update: $D_{t+1}(i) \leftarrow D_t(i) \cdot \beta_t^{\mathbf{1}_{(y_i = h_t(x_i))}}$ with $\sum_i D_{t+1}(i) = 1$.

Output the the strong classifier: $H(x) = sign(f(x))$ and $f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$.

Figure 2. Discrete AdaBoost algorithm. $\mathbf{1}$ is an indicator function.

It is proved in [2] the training error $\epsilon = \sum_i D_1(i) \mathbf{1}_{(y_i \neq h_t(x_i))}$ is bounded by

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

Moreover, it is shown that AdaBoost and its variants are asymptotically approaching the posterior distribution [3].

$$p(y|x) \leftarrow q(y|x) = \frac{exp\{2yf(x)\}}{1 + exp\{2yf(x)\}}. \qquad (5)$$

As we can see in eqn. (4), the overall training error is decided by the error ϵ_t at each round. For each ϵ_t, it is decided by how well the positives and negatives are separated by the candidate weak classifiers. Next, we show the error bound of ϵ_{t+1} w.r.t. ϵ_t in AdaBoost.

Theorem 1 *At step t, let the error for a candidate weak classifier $h^{(j)}$ be $\epsilon_t^{(j)} = \sum_{i=1}^{N} D_t(i)\mathbf{1}_{(y_i \neq h^{(j)}(x_i))}$ and ϵ_t be the best error achieved. Apparently, $\epsilon_t \leq \epsilon_t^{(j)}; \forall j$. $\epsilon_t = 0$ when the classifier perfectly classifies the data and $\epsilon_t = \frac{1}{2}$ when it makes random guesses. $\frac{1}{2} \geq \epsilon_t \geq 0$.*

Therefore, the best weak classifier picked for $t+1$ can best achieve

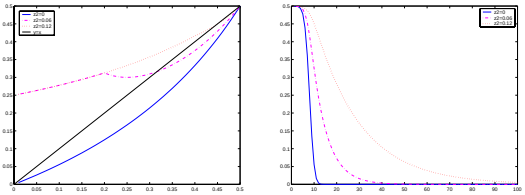
$$\epsilon_{t+1} \geq \frac{\epsilon_t}{2(1-\epsilon_t)}. \tag{6}$$

Proof. See appendix. This theorem says that the error for the next weak classifier can best achieve $\frac{\epsilon_t}{2(1-\epsilon_t)}$. Fig. (3) shows the curve of the function.

Also,

$$\epsilon_{t+1} \geq \frac{1}{2\epsilon_t}\frac{\epsilon_t^2 + z_2(1-2\epsilon_t)}{1-\epsilon_t}, z_2 \leq \epsilon_t. \tag{7}$$

The bound in eqn. (6) is achieved when $z_2 = 0$. If every time, weak classifier selected at $t+1$ always complements the one at t, then $z_2 = 0$. However, this situation is very ideal. Fig. (3.a) shows the bounds for ϵ_{t+1} w.r.t. to different z_2 and Fig. (3.b) demonstrates the corresponding convergence rates. We can see that the convergence becomes very slow when no weak classifier has a good discrimination power.



(a) Bound in eqn. (6) (b) ϵ w.r.t. t

Figure 3. (a) shows the bounds w.r.t. different z_2 in eqn. (6). (b) displays the corresponding convergence rate ϵ against time t if every time the bound is achieved.

2.2. Design Strategies

AdaBoost and its variants have recently been applied in a variety of problems in machine learning and computer vision [2, 1, 8, 14, 5, 13]. In vision, it has specially shown its promises in object detection/recognition.

From the convergence analysis in sect. (2.1), we see that the convergence of AdaBoost is slow when all weak classifiers are not so effective. There are three possible remedies to this problem:

(I) **Design smart weak classifiers.**

Existing methods in AdaBoost select and combine a set of weak classifiers from an pre-defined pool of candidates. Researchers strive to design informative weak classifiers/features for the specific problems in their domain. It is desirable to have a principled way of automatically "discovering" smart features.

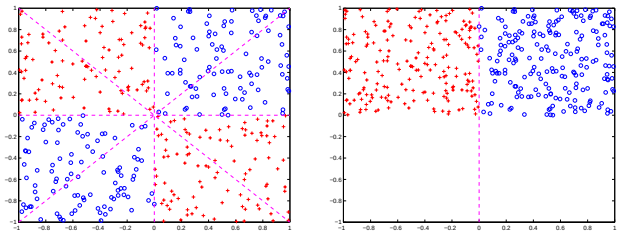
(II) **Change the selection strategy.**

AdaBoost is a greedy method. The error bound discussed in sect. (2.1) is in part due to this aspect. Li and Zhang [8] introduced the FloatBoost algorithm to allow the the algorithm to look back and improve the weak classifiers chosen. The importance of joint statistics of the features were discussed in [1]. Probabilistic Boosting-Tree method [13] utilizes a divide-and-conquer strategy.

(III) **Reduce the complexity of training samples.**

Except for those problems in which the nature of training samples is fixed, we often have a choice of how complicated our training samples can be.

In the problem of object detection, we formulate the generation process of an object in eqn. (1). Since an object is generated by a set of parameters Θ, based on a dictionary Ψ, it may observe a large variation in shape, size, and deformation. One often hopes to train a classifier which precisely computes the discriminative model $p(y|\mathbf{x})$. Usually, it is very hard to do when the training samples have large variation. In the context of AdaBoost, it is hard to design weak classifier well separating the positives from the negatives if they have large variation. The overall classifier is no so effective as shown in sect. (2.1). Instead, we can perform data alignment to reduced the complexity of training samples. This will lead to more effective classifiers. Following eqn. (3), we can see that alignment comes with the price of integrating/searching the parameters in the testing stage. It requires to compute each $p(\Theta_1|\mathbf{x})$, which is usually time consuming.



(a) Original samples $\mathbf{x}$ (b) Aligned samples.

Figure 4. A toy example to show the effectiveness of sample alignment on classifiers. No line can well separate the two classes in a). If we know that the points in the second half are undergoing a rotation of 180 degrees and we rotate these points accordingly, then the new samples in b) can be easily classified.

Fig. (4) illustrates a simple example in which there are two sets of training samples. Fig. (4a) shows the original set of training samples. This is a well-known example that no linear boundary can separate the positives from the negatives. The four dividing lines shown in the figure all have error $\epsilon = \frac{1}{2}$. If we know that there is a rotation parameter θ controlling the generation process, we can align the data.

The positives and negatives in Fig. (4b) then become easy to classify.

3. Training Classifiers for Polyp Detection

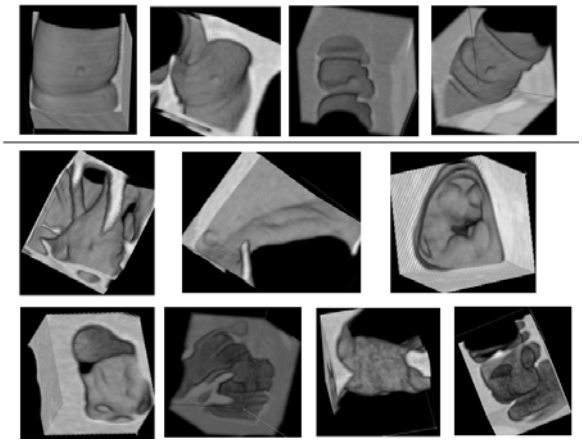


Figure 5. Some polyp examples. The first row shows several small and medium size polyps. The second and the third row shows some polyps of medium and big sizes.

In this section, we focus on polyp detection and show how to train a classifier to detect them. An input volume is in full 3D and it is isotropic after interpretation. The typical size of a volume is $500 \times 512 \times 512$. There is no occlusion between objects. Our task is to design an algorithm for automatic polyp detection. Fig. (1) shows a sample volume and Fig. (5) displays a number of true polyps. For a small or medium size polyp, it often observes a regular shape as half hemisphere. When it becomes big, it starts to develop a variety of shapes as shown in the second and third row of Fig. (5). This is due to its interaction with the colon wall and other structures. It is especially a problem for those methods based on the analysis of curvatures or Gaussian type generative models.

3.1. Sample Alignment and Data Augmentation

In 3D CT volumes, polyps appear in all possible orientations on the colon wall. According to the discussion in the previous section and eqn. (3), one design principle is to reduce the complexity of the training samples. We choose $\Theta_1 = \{s, \theta, \alpha\}$ where s is the scale, θ is the orientation, and α is the aspect ratios of a polyp w.r.t. its depth, height, and width. We create a template of size $24 \times 24 \times 16$, whose 2D view is shown in Fig. (6a). The orientation sphere is divided into 14 zones, 4 of which are along the major axises. Fig. (6b) illustrates the idea. In training, we are given 130 polyps coming from size of $2mm$ to $25mm$. Fig. (5) shows some of these samples. We align each polyp to the major orientation shown in Fig. (6c).

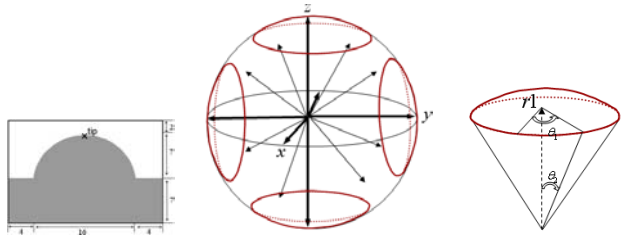


Figure 6. (a) shows the template polyp in 2D. (b) displays the orientation sphere which is divided into 14 zones, 4 of which are along the major axises. (c) illustrates one major orientation to which we align our positive training samples.

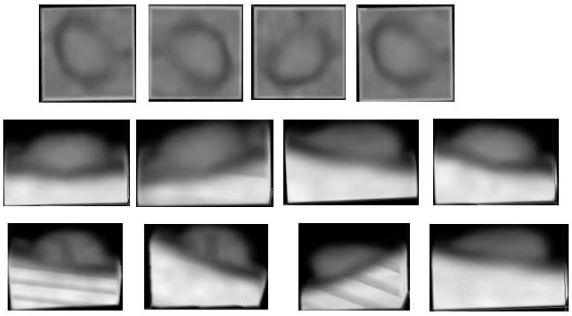


Figure 7. A typical polyp and its augmented data shown in 3D.

As we mentioned before, aligning training data comes at a price of searching for more parameters in the detection stage. The more parameters we explicitly take out during training, the more we need to specifically search for in the detection stage. We do not want to align the data perfectly since it requires a search for every specific value of scale and rotation. We also don't want to train a classifier which computes $p(y|\mathbf{x})$ directly since the classifier will be confused on very complicated $\mathbf{x}$. We observe that positives have a certain degree of regularity once we make them the same size and along the major direction. In the orientation zone $r1$, we sample the possible detailed orientations, (θ_1, θ_2) to augment the training data. This has two effects on the algorithm: We don't need to search for specific values of (θ_1, θ_2); It augments the training samples from 130 to around 13,000. This reduces the chance of overfitting. Fig. (7) shows a typical polyp sample after alignment and its augmented data, which are polyps created based on original training samples. Our goal is to train a classifier based on augmented training data to capture the underlying shape and intensity pattern of objects of interest.

3.2. Features

Extensive efforts have been made in the literature to design features for polyp detection [6, 7, 16] or for generic object detection in natural images [9]. Fig. (8) shows two

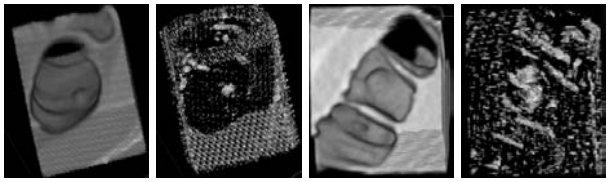


Figure 8. Tow sub-volumes and their shape index features.

sub-volumes and the "shape index" for each voxel, which was introduced by Yoshida *et al*. in [16]. We can see that these features are good at detecting roughly round surfaces, but there are lots of false positives.

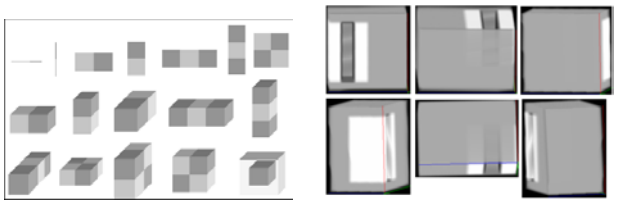
Our approach classifies a sub-volume based on a template of $24 \times 24 \times 16$. As we mentioned in Sect. (3.1), we align and augment positive samples to one of the major directions. According to our approach, we want our features to have the following properties:

(I) They should be scale and aspect ratio invariant to a certain degree. We need to search for different possible scales and aspect ratios in the detection stage.

(II) They should be easy and fast to compute.

(III) They should be informative for the objects of interest.

These rule out mean curvatures and Gaussian curvatures [11] which are shown to be effective in polyp detection. We will show how to make use of them later.



(a) (b)

Figure 9. (a) shows 1D, 2D, and 3D Haar filters used. (b) shows rotated Haar filter according to the 6 major orientations.

Viola and Jones proposed an efficient algorithm for face detection [14]. Similar to the integral image and 2D Haar filters used in their method, we design integral volume and 3D Haar filters for polyp detection. At each location (x_1, y_1, z_1), the integral volume is computed $\int_{x_1}\int_{y_1}\int_{z_1} V(x,y,z)dxdydz$. The computational cost of computing Haar filters is largely reduced since every time we only need to sum up the values of corners of the Haar in the integral volume. Since the procedure of aligning samples and training classifiers is time-consuming, we also want to make the features to be "semi-rotation invariant". This is to say that once the classifier for one major direction, $r1$, is trained, we can automatically derive the classifiers for the other orientations. Haar filters meet this requirement for the 6 major directions. For the other 8 orientations, we will rotate the 3D data in the detection stage rather than having another set classifiers for them. This will likely reduce the training efforts and also, the chance of overfitting since virtually there is only one classifier trained.

3.3. Probabilistic Boosting-Tree

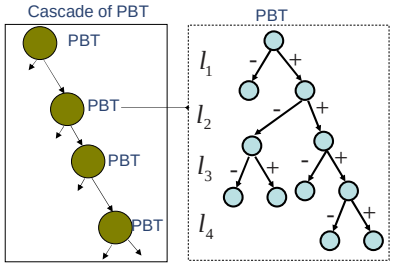


Figure 10. A cascade of Probabilistic Boosting-Tree (PBT).

AdaBoost and its variants have shown their promises in many vision problems. In the first attempt, we employ a cascade of AdaBoost approach [14] to train our classifier in several stages of bootstrapping. Due to the large variation of polyps in the training data even after alignment, the training error remains high, 0.3, after 4 levels of cascade. The error hardly goes down further. It is because positives and negatives become similar to each other and hard to distinguish after several layers of cascade. Probabilistic Boosting Tree (PBT) is a recently proposed method to improve the cascade approach and explicitly computes the discriminative model. Detailed discussion of PBT can be found in [13]. As we can see in eqn. (5) that each strong classifier learned approaches the target distribution $p(y|x)$. PBT does training and testing in a divide-and-conquer manner and outputs the overall discriminative model as

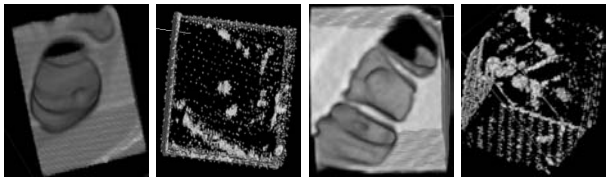
$$\begin{aligned}\tilde{p}(y|x) &= \sum_{l_1} \tilde{p}(y|l_1,x)q(l_1|x) \\ &= \sum_{l_1,l_2} \tilde{p}(y|l_2,l_1,x)q(l_2|l_1,x)q(l_1|x) \\ &= \sum_{l_1,..,l_n} \tilde{p}(y|l_n,...,l_1,x),...,q(l_2|l_1,x)q(l_1|x)\end{aligned}$$

Each tree level l_i is an augmented variable. To facilitate the process of bootstrapping and reduce the number of negatives for PBT, we find out that a cascade approach is still useful here. Fig (10) shows a diagram of a cascade of PBT. At each stage of PBT, the probability is used as a threshold. Those samples whose $\tilde{p}(y|x)$ is bigger than the probability will pass into the next stage.

3.4. Training Procedures

We use 80 volumes for training in which there are 130 polyps annotated and segmented by radiologists.

First, we train an AdaBoost classifier to classify whether a voxel is on the surface of polyp or not. This classifier is used to quickly screen out majority of the voxels that are on flat surface. The features used are intensity, gradient, Gaussian curvatures, mean curvatures, etc. We train a general AdaBoost algorithm to combine these features. Some results are shown in Fig. (11).



(a) (b) (c) (d)

Figure 11. Tow sub-volumes are shown in a) and c), and b) and d) display the results using trained shape classifier. In b) and d), the brighter the voxel, the higher the probability it is on a polyp.

Second, we train a classifier which consists of a cascade of PBT classifiers. Based on the annotation of a polyp and its annotation on the tip, we can precisely locate the bounding box of it. As discussed in the previous section, we align and augment the training samples to $13,000$ positives of size $24 \times 24 \times 16$. In the 80 training volumes, we randomly sample those voxels with gradient along the major $r1$ orientation, and having passed the first basic shape classifier. Also, these voxels should not be on the surface of any annotated polyps. We then cropped out 3D sub-volumes of size $24 \times 24 \times 16$ by aligning these voxels with the tip position in the template. There are in total $30,000$ negative samples obtained. Using these positives and negatives, we train a PBT with 9 levels and 22 weak classifiers for each AdaBoost node. Once a PBT is trained, we then use it to run through the training volumes to bootstrap more negatives samples. Five cascade levels are used. Since each PBT computes the discriminative model $\tilde{p}(y|\Theta_1, x)$, we can easily adjust the threshold to balance between detection rate and number of false positives. The first two levels are set to have nearly 100% detection rate. Each PBT consists of around 1,000 weak classifiers on the Haar filters. Based on the trained cascade of PBT, we rotate the Haar filters shown in Fig. (9b) to generate classifiers for detecting polyps on the other 5 directions.

3.5. Detecting Polyps

Next, we discuss the detection procedure for our algorithm. Jerebko *et al*. [4] use a candidate generation method to obtain a set of candidate locations. An example of them are shown in Fig. (1). We adopt their algorithm to generate candidates. In average, there are 50 polyps obtained per volume.

Fig. (12) gives the details of our polyp detection algorithm after candidate generation. For each candidate location, we create 9 sub-volumes: 3 sub-volumes at different scales with each rotated at 3 orientations. As we discussed in Sect. 2.2, we spend a bit more time in the detection stage with the gain of reduced complexity of training samples. Also, we only train an overall classifier for orientation $r1$ and derive classifiers for all other directions based on it. This also helps to improve the generality of the detector and efficiency. It can be seen that features are much easier to obtain if samples are in the upper-right position than slanted. For each voxel in the sub-volume, we run shape classifier and cascade PBT at different scales and aspect ratios. The best bounding box is outputted if it is found to be a polyp.

- For each candidate location in the volume, we crop out sub-volumes at three different scales and resize them to volumes of $60 \times 60 \times 60$. An example of such three volumes can be seen in the first row of Fig. (13).
- For each of the above volume, we obtain three additional sub-volumes of the current, rotating from $r2$ to $r1$, and rotating from $r3$ to $r1$. The second row of Fig. (13) shows these volumes. It is to cover the other 8 orientation zones whose classifiers can not be easily obtained by simply rotating Haar filters.
- For each the above sub-volume, we run the first layer shape classifier to rule out those voxels that are not on the surface of a polyp.
- For each the voxel passing the shape classifier, compute its gradient direction. If if falls into one of the 6 major orientation, fire the corresponding cascade of PBT classifier.
- We try 96 possibilities of combinations of difference sizes and aspect ratios of templates by considering the current voxel being the tip of the template. (Haar filters corresponding to the 96 possibilities are computed offline).
- If it passes the cascade, remember the bounding box and use the probability outputted in the last layer in the cascade as confidence measure.
- Repeat for all the 9 sub-volumes generated. If there exists bounding box, use the bounding box which has the highest confidence and output it as a polyp.

Figure 12. Outline of the polyp detection algorithm.

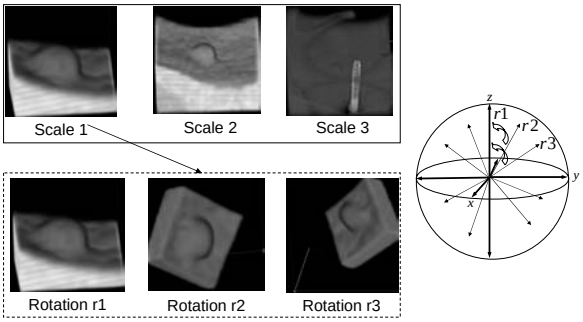
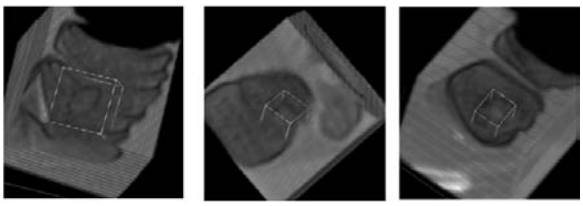


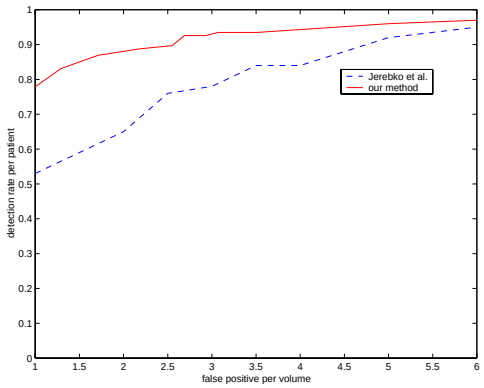
Figure 13. The first row shows sub-volumes at three different scales. This is to cover the possibility of different sizes at a large scale. The second row shows three sub-volumes at different orientations. This is to cover polyps whose orientation may fall into the 8 none-major orientation. The right figure shows the definition of $r1$, $r2$, $r3$, and how the rotated volumes are obtained.

4. Experiments

There yet does not exist a common data set on which people can use to test their polyp detection algorithm. We test our algorithm on the 80 training volumes and 70 test volumes. Among the 70 test volumes, 33 of which have polyps confirmed by radiologists with 53 polyps. We have tested our algorithm on both the training and testing data and have observed similar performances. Fig. (14a) shows some polyps detected with the bounding boxes highlighted. Detection rate for the candidate generation process is 91.8%. Fig. (14a) shows a comparison between our results and that by [4] after the generation process. We can observe much improved results. The algorithm runs about 4 minutes for 50 candidates. At the rate of 3.01 false positives per volume, the overall detection rate per volume for medium size polyps ($6mm \sim 9mm$), and big size polyps > 9 are respectively 98%, 84% after candidate generation. To compare with PBT, we implement a cascade of boosting and it gives rates of 85% and 70% respectively. Kiss et al. reported a system with slightly better performance, 85% detection rate and 2.74 false positives per volume. The number of volumes is less (50 datasets) than ours. We have also tested out algorithm on an unseen hold-out data set of 235 volumes. Out of 82 medium and big size polyps, 73 of them are detected. The sensitivity per patient is 92% with 5.8 false positives per volumes. (Generally it requires the sensitivity to be higher than 90% with less than 6 false positives.)



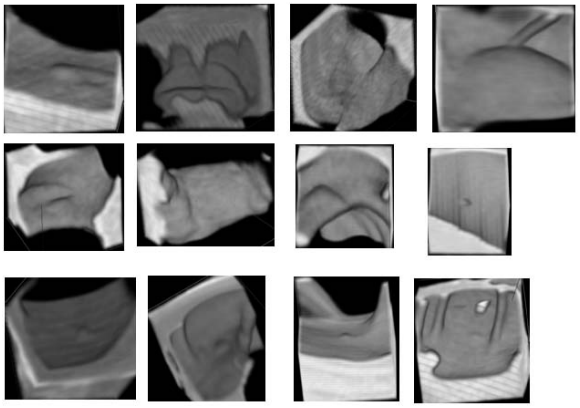
(a) Detected polyps in some of the sub-volumes.



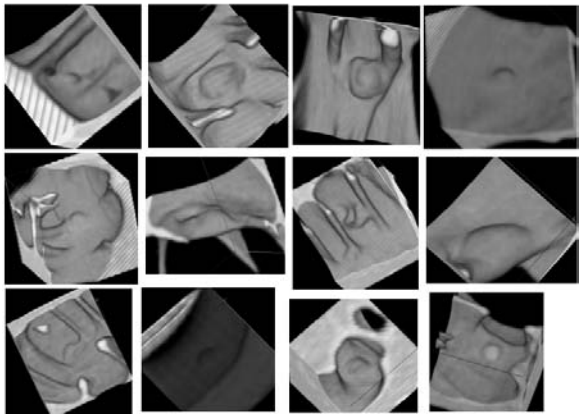
(b) Comparison of our result with the one in [4].

Figure 14. Performance of our algorithm.

Fig. (15a) shows some missed polyps and false positives detected. As we can see, most misses are either because polyps are too small or due to their irregular shapes. Fig. (15b) shows many false positives detected. In fact, most of these false positives observe pulmonary structure and it is very hard to distinguish them from the true polyps. They are yet remain to be further verified by radiologists.



(a) Some polyps missed by our algorithm.



(b) Some false positives detected by our algorithm.

Figure 15. Some failed examples. Most misses are due to small size or irregular shapes. Those false positives really look like polyps and they remain to be verified further by experts in the field.

5. Discussions

In this paper, we have introduced a learning based algorithm for 3D polyp detection in CT images. We formulate the problem by a generative model to facilitate the understanding of discriminative models. In the context of AdaBoost, we prove an error bound for the weak classifiers leading to a principled analysis of sample alignment. We adopt the probabilistic boosting-tree for our classifier and design integral volume and 3D Haar filters for fast computation. We have tested our algorithm on 150 volumes and the results obtained are very good. The proposed system is very adaptive and can be applied in detecting other

pulmonary nodules such as lymph-nodes and lung cancers. Also, it does not require the stage of colon segmentation. Therefore, it is also suitable for detecting polyps in colon without undergoing physical cleansing procedure. Due to the residual materials left inside colon, explicit polyp detection algorithm based on surface shapes requires to remove the residual materials before polyp detection. On the down side of the method, sample alignment is a time-consuming task. Though we have a heterogeneous number of candidate features, $50,000$, these features are pre-defined rather than automatically learned. The candidate generation needs to be directly incorporated into our cascade classifiers.

Acknowledgements We thank Robert Schapire and Yaov Freund for verifying our result on the AdaBoost error bound analysis shown in this paper.

References

[1] X. Chen and A. Yuille, "Detecting and Reading Text in Natural Scenes", *Proc. of CVPR*, 2005. 3

[2] Y. Freund and R. E. Schapire, "A Decision-theoretic Generalization of On-line Learning And An Application to Boosting", *J. of Comp. and Sys. Sci.*, 55(1), 1997. 1, 2, 3

[3] J. Friedman, T. Hastie and R. Tibshirani, "Additive logistic regression: a statistical view of boosting", Dept. of Statistics, Stanford Univ. Technical Report. 1998. 2

[4] A. Jerebko, L. Bogoni, and A. Krishnan, "Symmetric Curvature Patterns for Colonic Polyp Detection", *Siemens Med-Cad Report*, 2005. 1, 6, 7

[5] B. Georgescu, S. Zhou, D. Comaniciu, A. Gupta, "Database-Guided Segmentation of Anatomical Structures with Complex Appearance", *Proc. of CVPR*, 2005. 3

[6] S.B. Gktrk, C. Tomasi, B. Acar, C.F. Beaulieu, D.S. Paik, R.B. Jeffrey, Jr., J. Yee, and S. Napel, "A Statistical 3-D Pattern Processing Method for Computer-Aided Detection of Polyps in CT Colonography", *Trans. on Medical Imaging*, vol. 20, no. 12, Dec. 2001. 1, 4

[7] G. Kiss, J.V. Cleynenbreugel, G. Marchal, and P. Suetens, "Computer Aided Detection in CT Colonography via Spin Images", *Proc. of Medical Image Computing and Computer Assisted Intervention*, 2004. 1, 4

[8] S. Z. Li and Z. Zhang, "FloatBoosting Learning and Statistical Face Detection", *IEEE Trans. PAMI*, vol. 26, no. 9, Sept. 2004. 2, 3

[9] D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints ", *Int'l J. of Computer Vision*, vol. 60, no. 2 , (2004). 4

[10] R. E. Schapire and Y. Singer, "Improved Boosting Algorithms Using Confidence-rated Predictions", *Machine Learning*, 37(3):297-336, 1999. 2

[11] J.P. Thirion and A. Gourdon, "Computing the Differential Characteristics of Isointensity Surfaces", *Computer Vision and Image Understanding*, vol. 61, no. 2, 1995. 5

[12] R. F. Thoeni and I. Laufer, "Polyps and Cancer", *Textbook of Gastrointestinal Radiology* Philadelphia, PA: Saunders, 1994. 1

[13] Z. Tu, "Probabilistic Boosting-Tree: Learning Discriminative Models for Classification, Recognition, and Clustering", *Proc. of ICCV*, 2005. 3, 5

[14] P. Viola and M. Jones, "Fast Multi-view Face Detection", *Proc. of CVPR*, 2001. 2, 3, 5

[15] D. J. Vining, "Virtual colonoscopy", *Gastrointest. Endosc. Clin. N. Amer.*, vol. 7, no. 2, 1997. 1

[16] H. Yoshida and J. Nppi, "Three-Dimensional Computer-Aided Diagnosis Scheme for Detection of Colonic Polyps", *Trans. on Medical Imaging*, vol. 20, no. 12, 2001. 1, 4, 5

Appendix

Let h_t be the selected weak classifier at step t ant it achieves ϵ_t.

$$\epsilon_t = \sum_i D_t(i)\mathbf{1}_{(y_i \neq h_t(x_i))} = \sum_{y_i \neq h_t(x_i)} D_t(i).$$

$$\beta_t = \frac{\epsilon_t}{(1-\epsilon_t)}.$$

Then the partition function (normalization) is

$$\begin{aligned} Z_t &= \sum_i D_t(i) \cdot \beta_t^{\mathbf{1}_{(y_i = h_t(x_i))}} \\ &= \sum_{i:y_i = h_t(x_i)} D_t(i)\beta_t + \sum_{i:y_i \neq h_t(x_i)} D_t(i) \\ &= 2\epsilon_t \end{aligned} \tag{8}$$

Then the error ϵ_{t+1} for any h at $t+1$ is

$$\begin{aligned} \epsilon_{t+1} &= \sum_i D_{t+1}(i)\mathbf{1}_{(y_i \neq h(x_i))} \\ &= \sum_i \frac{D_t(i)}{Z_t}\beta_t^{\mathbf{1}_{(y_i = h_t(x_i))}}\mathbf{1}_{(y_i \neq h(x_i))} \\ &= \frac{1}{2\epsilon_t}\sum_{y_i = h_t(x_i)} D_t(i)\beta_t\mathbf{1}_{(y_i \neq h(x_i))} + \\ &\quad \frac{1}{2\epsilon_t}\sum_{y_i \neq h_t(x_i)} D_t(i)\mathbf{1}_{(y_i \neq h(x_i))} \end{aligned} \tag{9}$$

We know $\sum_i D_t(i)\mathbf{1}_{(y_i \neq h(x_i))} \geq \epsilon_t$. Let $z_1 = \sum_{y_i = h_t(x_i)} D_t(i)\mathbf{1}_{(y_i \neq h(x_i))}$, and $z_2 = \sum_{y_i \neq h_t(x_i)} D_t(i)\mathbf{1}_{(y_i \neq h(x_i))}$, then we have

$$z_1 \geq \epsilon_t - z_2.$$

Therefore,

$$\begin{aligned} \epsilon_{t+1} &= \frac{1}{2\epsilon_t}\left(\frac{\epsilon_t}{1-\epsilon_t}z_1 + z_2\right) \geq \frac{1}{2\epsilon_t}\frac{\epsilon_t^2 + z_2(1-2\epsilon_t)}{1-\epsilon_t} \\ &\geq \frac{\epsilon_t}{2(1-\epsilon_t)} \quad \square \end{aligned} \tag{10}$$