

A boosting regression approach to medical anatomy detection

Shaohua Kevin Zhou[†], Jinghao Zhou[‡], and Dorin Comaniciu[†]

[†]Integrated Data Systems Department, Siemens Corporate Research, Princeton, NJ 08540

[‡]Department of Biomedical Engineering, Rutgers University, Piscataway, NJ 08854

Abstract

The state-of-the-art object detection algorithm learns a binary classifier to differentiate the foreground object from the background. Since the detection algorithm exhaustively scans the input image for object instances by testing the classifier, its computational complexity linearly depends on the image size and, if say orientation and scale are scanned, the number of configurations in orientation and scale. We argue that exhaustive scanning is unnecessary when detecting medical anatomy because a medical image offers strong contextual information. We then present an approach to effectively leveraging the medical context, leading to a solution that needs **only one scan** in theory or several sparse scans in practice and **only one integral image** even when the rotation is considered. The core is to learn a regression function, based on an annotated database, that maps the appearance observed in a scan window to a displacement vector, which measures the difference between the configuration being scanned and that of the target object. To achieve the learning task, we propose an image-based boosting ridge regression algorithm, which exhibits good generalization capability and training efficiency. Coupled with a binary classifier as a confidence scorer, the regression approach becomes an effective tool for detecting left ventricle in echocardiogram, achieving improved accuracy over the state-of-the-art object detection algorithm with significantly less computation.

1. Introduction

Detecting medical anatomic structure is important to medical image understanding. For example, in order to segment the anatomic structure [2], a two-stage algorithm is proposed: anatomy detection followed by database-guided segmentation. The detection result can also serve valuable initialization information for other segmentation techniques such as level set, active contour, etc.

A promising approach to medical anatomy detection is to use the classifier-based object detection approach: It first trains a binary classifier, discriminating the anatomic structure of interest from the background, and then exhaustively

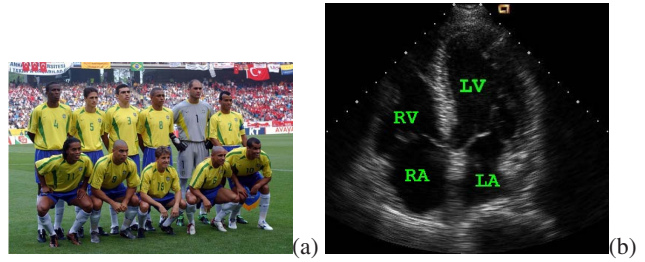


Figure 1. (a) Face detection. There are multiple faces in the image. (b) Left ventricle detection. There is only one LV in the A4C echocardiogram.

scans the query image for anatomy targets. Suppose that the trained classifier is denoted by posterior probability $p(O|I)$, the above scanning procedure mathematically performs one of the following two:

$$\text{find } \{\theta : p(O|I(\theta)) > 0.5; \theta \in \Theta\}, \quad (1)$$

$$\hat{\theta} = \arg \max_{\theta \in \Theta} p(O|I(\theta)), \quad (2)$$

where $I(\theta)$ is an image patch parameterized by θ and Θ is the parameter space where the search is conducted. In (1), multiple objects are detected such as in face detection (Figure 1(a)); in (2), only one object is detected such as in left ventricle (LV) detection from a 2D echocardiogram of an apical four chamber (A4C) view (Figure 1(b)). The 2D echocardiogram is a 2D slice of the human heart captured by an ultrasound imaging device and the A4C view is a canonical slice where all four chambers, namely left ventricle, right ventricle (RV), left atrium (LA), and right atrium (RA), are visible.

In [11], Viola and Jones used the classifier-based approach and reached real time detection of frontal-view face by exhaustively searching all possible translations and a sparse set of scales. They made three contributions: invoking the AdaBoost to do feature selection, using the so-called integral image to enable fast evaluation of the features, and training the boosting cascade to quickly eliminate negatives. Recently, approaches (mostly extended from [11]) that detect objects under in-plane/out-of-plane rotations are proposed [2, 4, 5]; but only a sparse set of orientations and scales are tested in order to meet real time requirement. Either multiple classifiers are learned [4, 5] or one classifier

is learned but multiple integral images according to different rotations are computed [2]. In general, the computational complexity of the classifier-based approach linearly depends on the image size (for the translation parameter), and the number of tested orientations and scales.

The medical anatomy such as LV often manifests an arbitrary orientation and scale. To give an accurate account of orientation and scale, which is required for subsequent tasks like LV endocardial wall segmentation, the detection speed is sacrificed for testing a dense set of orientations and scales. Further, if the one-classifier approach [2], which is shown to perform better than the multiple-classifier approach [4], is used, then rotating the images and computing their associated integral images cost extra computations. Therefore, it is challenging to build a rapid detector for medical anatomy using the classifier-based approach.

In summary, the exhaustive native of the classifier-based approach makes real time computation impossible when a high-dimensional space is searched. In this paper, we take a completely different approach. By leveraging the anatomical structure that manifests regularization and context in geometry and appearance in medical images, we formulate a regression task in section 2 to avoid exhaustive scanning in medical anatomy detection. In theory, we need only *one scan*. Also, we compute *only one* integral image. In addition, we contribute a novel image-based boosting ridge regression (IBRR) algorithm in section 3 for multiple regression with a multidimensional output. We present in section 4 the medical anatomy detection algorithm and report in section 5 experimental results on detecting the LV in the 2D echocardiogram of the A4C view.

2. Overview

We first present the basic idea of the regression approach to medical anatomy detection. Then, we justify our approach by addressing the existence issue and contrasting it with related work. Finally, we define the learning task.

2.1. Basic idea

Figure 2 illustrates the basic idea of the regression-based medical anatomy detection. In Figure 2(a), we are only interested in finding the center position $(t_{x,0}, t_{y,0})$ of the LV in an A4C echocardiogram, assuming that the orientation of the LV is upright and the scale/size of the LV is fixed.

Suppose that, during running time, we confront an image patch $\mathcal{I}(t_x, t_y)$ centered at position $\theta = (t_x, t_y)$. If there exists an oracle \mathcal{F} that does the following: given an image patch $\mathcal{I}(t_x, t_y)$, it tells the difference vector $d\theta = (dt_x, dt_y)$ between the current position θ and the target position $\theta_0 = (t_{x,0}, t_{y,0})$, i.e., $d\theta = \theta_0 - \theta$ or

$$dt_x = t_{x,0} - t_x, \quad dt_y = t_{y,0} - t_y, \quad (3)$$

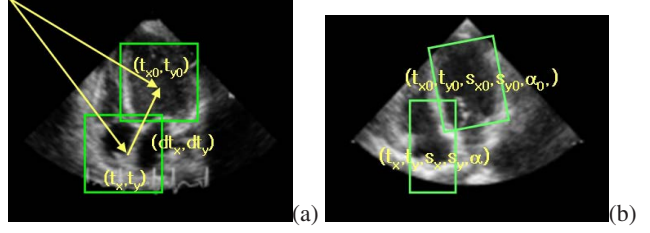


Figure 2. (a) A graphical illustration of regression-based medical anatomy detection based on a 2D translation parameterization. (b) The regression setting of a 5D parameter space: (t_x, t_y) is the LV center; (s_x, s_y) the LV size, and α the LV angle.

then we achieve the detection using *just one* scan. In other words, the oracle gives a mapping $\mathcal{F} : \mathcal{I} \rightarrow (dt_x, dt_y)$, i.e.,

$$(dt_x, dt_y) = \mathcal{F}(\mathcal{I}(t_x, t_y)) \text{ or } d\theta = \mathcal{F}(\mathcal{I}(\theta)), \quad (4)$$

and the ground truth position is estimated as

$$\hat{\theta}_0 = \theta + \mathcal{F}(\mathcal{I}(\theta)). \quad (5)$$

Learning the function $\mathcal{F}(\mathcal{I}(\theta))$ is referred to as *regression* in machine learning.

It is straightforward to extend the 2D case to a higher dimension. For example, to model the unaligned LV in real images, we use a 5D parameterization $\theta = (t_x, t_y, \log(s_x), \log(s_y), \alpha)$: (t_x, t_y) for translation, α for orientation, and (s_x, s_y) for scale (or size) in both x - and y -directions. Due to the multiplicative native of the scale parameter, we take the log operator to convert it to additive. The ‘difference’ vector $d\theta = (dt_x, dt_y, ds_x, ds_y, d\alpha)$ is given as

$$\begin{aligned} dt_x &= t_{x,0} - t_x, dt_y = t_{y,0} - t_y, d\alpha = \alpha_0 - \alpha, \\ ds_x &= \log(s_{x,0}) - \log(s_x), ds_y = \log(s_{y,0}) - \log(s_y), \end{aligned} \quad (6)$$

where $\theta_0 = (t_{x,0}, t_{y,0}, s_{x,0}, s_{y,0}, \alpha_0)$ is the ground truth parameter of the target. Figure 2(b) illustrates the meaning of the five parameters.

2.2. Existence

Does such an oracle \mathcal{F} exist? To answer this, we observe the fundamental differences between generic object detection and medical anatomy detection. Unlike general object detection that needs to detect object instances from unconstrained scenes (see Figure 1(a)), medical anatomy detection applies to more constrained medical images (see Figure 1(b)). As a result, in generic object detection, an unknown number of objects can appear at arbitrary locations in the images with arbitrary background; in medical anatomy detection, since the anatomic structure of interest is tied with human body atlas, there is a known number of objects appearing within geometric and appearance contexts. Often only one object is available. For example, in the echocardiogram shown in Figure 1, there is only one target LV available and its relation with respect to other structures such as

left atrium, right ventricle and right atrium is geometrically fixed (that is why they are called left/right ventricle/atrium). Also there exists a strong correlation among their appearances. By knowing where the LA, RV, or RA is, we can predict the position of the LV quite accurately. In principle, by knowing where we are (*i.e.*, knowing (t_x, t_y)) and then looking up the map/atlas that tells the difference to the target (*i.e.*, telling (dt_x, dt_y) through the oracle), we can reach the target instantaneously in a virtual world.

Medical atlas is widely used in the literature [1, 6, 7, 9]. However, current work uses the atlas as an *explicit* source of prior knowledge about the location, size, and shape of the anatomic structures and matches/deforms it to match the image content for segmentation, tracking, etc. In this paper, we take an *implicit* approach, that is, embedding the atlas in a learning framework. After learning, the atlas knowledge is fully absorbed and the atlas is no longer kept.

Torrallba and Sinha [10] investigate modeling the context information for general object detection. They view the context from a *holistic* perspective and use a mixture of Gaussians to relate the global image content represented by windowed Fourier transform with the object's location and scale. We view the context from a *local* perspective, believing that what is locally observed is a part of a big picture, and develop a regression algorithm to relate the local image content to the target's attributes.

2.3. Learning

How to learn the oracle \mathcal{F} ? We leverage machine learning techniques, based on an annotated database. As in Figure 3, we first collect, from the database, input-output pairs (as many as possible) as training data: By varying the location, we crop out different local image patches; meanwhile recording their corresponding difference vectors. Similarly, for the 5D parameterization, we can extract the training data. We now confront a multiple regression setting with a multidimensional output, which is not well addressed in the machine learning literature. In this paper, we propose the image-based boosting ridge regression (IBRR) algorithm, which builds upon the image-based regression (IBR) algorithm [13], to fulfill the learning task.










| | | | | | |
|-------------------------------------------------------------------------------------|-----------|-------------------------------------------------------------------------------------|----------|-------------------------------------------------------------------------------------|---------|
|  | (-15,-12) |  | (-3,-8) |  | (-4,-6) |
|  | (-5,-17) |  | (-7,-21) |  | (15,16) |
|  | (15,-6) |  | (16,-5) |  | (17,6) |

Figure 3. Examples of training data: image patch I and its associated parameter $d\theta = (dx, dy)$.

3. Image-based boosting ridge regression

In this section, we first review the IBR algorithm [13], which produces better results on several heterogeneous tasks than data-driven regressors such as support vector regressor while running substantially faster. Then, we present our IBRR proposal to further make the IBR generalize better, run faster, and more trainable.

3.1. Image-based boosting regression

We follow the notation used in [13]: a is a scalar, \mathbf{a} a column vector, and \mathbf{A} a matrix. The input is denoted by $\mathbf{x} \in \mathcal{R}^d$, the output by $\mathbf{y}(\mathbf{x}) \in \mathcal{R}^q$, the regression function by $\mathbf{g}(\mathbf{x}) : \mathcal{R}^d \rightarrow \mathcal{R}^q$ and the training data points by $\{(\mathbf{x}_n, \mathbf{y}_n); n = 1, 2, \dots, N\}$. Further, we denote $\mathbf{x}^T \mathbf{A} \mathbf{x} = \|\mathbf{x}\|_{\mathbf{A}}^2$ and $\text{tr}\{\mathbf{x}^T \mathbf{A} \mathbf{x}\} = \|\mathbf{x}\|_{\mathbf{A}}^2$. In medical anatomy detection, $\mathbf{x} = \mathbf{I}$ is the image, $\mathbf{y} = d\theta$ is the difference vector, and the regression function $\mathbf{g}(\mathbf{x}) = \mathcal{F}(\mathbf{I})$ is the oracle.

3.1.1 Boosting regression

IBR minimizes the following cost function, which combines a regression output fidelity term and a subspace regularization term:

$$J(\mathbf{g}) = \sum_{n=1:N} \{\|\mathbf{y}(\mathbf{x}_n) - \mathbf{g}(\mathbf{x}_n)\|_{\mathbf{A}}^2 + \lambda \|\mu - \mathbf{g}_t(\mathbf{x}_n)\|_{\mathbf{B}}^2\}, \quad (7)$$

where λ is a *regularization coefficient*.

IBR assumes that the regression output function $\mathbf{g}(\mathbf{x})$ takes an additive form:

$$\mathbf{g}_t(\mathbf{x}) = \mathbf{g}_{t-1}(\mathbf{x}) + \alpha_t \mathbf{h}_t(\mathbf{x}) = \sum_{i=1:t} \alpha_i \mathbf{h}_i(\mathbf{x}), \quad (8)$$

where each $\mathbf{h}_i(\mathbf{x}) : \mathcal{R}^d \rightarrow \mathcal{R}^q$ is a weak learner (or weak function) residing in a *dictionary* set \mathcal{H} , and $\mathbf{g}(\mathbf{x})$ is a strong learner (or strong function). Boosting is an iterative algorithm that leverages the additive nature of $\mathbf{g}(\mathbf{x})$: At iteration t , one more weak function $\alpha_t \mathbf{h}_t(\mathbf{x})$ is added to the target function $\mathbf{g}_t(\mathbf{x})$. So,

$$J(\mathbf{g}_t) = \sum_{n=1:N} \{\|\mathbf{r}_t(\mathbf{x}_n) - \alpha_t \mathbf{h}_t(\mathbf{x}_n)\|_{\mathbf{A}}^2 + \lambda \|\mathbf{s}_t(\mathbf{x}_n) - \alpha_t \mathbf{h}_t(\mathbf{x}_n)\|_{\mathbf{B}}^2\}, \quad (9)$$

where $\mathbf{r}_t(\mathbf{x}) = \mathbf{y}(\mathbf{x}) - \mathbf{g}_{t-1}(\mathbf{x})$ and $\mathbf{s}_t(\mathbf{x}) = \mu - \mathbf{g}_{t-1}(\mathbf{x})$.

It is shown in [13] that the optimal function $\hat{\mathbf{h}}$ (dropping the subscript t for notational clarity) and its weight coefficient $\hat{\alpha}$, which maximally reduce the cost function or equivalently boost the performance, are given as follows:

$$\hat{\mathbf{h}} = \arg \max_{\mathbf{h} \in \mathcal{H}} \epsilon(\mathbf{h}), \quad \hat{\alpha}(\hat{\mathbf{h}}) = \frac{\text{tr}\{(\mathbf{A}\mathbf{R} + \lambda\mathbf{B}\mathbf{S})\hat{\mathbf{H}}^T\}}{\|\hat{\mathbf{H}}\|_{\mathbf{A}+\lambda\mathbf{B}}^2}, \quad (10)$$

where

$$\epsilon(\mathbf{h}) = \frac{\text{tr}\{(\mathbf{A}\mathbf{R} + \lambda\mathbf{B}\mathbf{S})\mathbf{H}^T\}}{\sqrt{\|\mathbf{H}\|_{\mathbf{A}+\lambda\mathbf{B}}^2} \sqrt{\|\mathbf{R}\|_{\mathbf{A}}^2 + \lambda \|\mathbf{S}\|_{\mathbf{B}}^2}}, \quad (11)$$

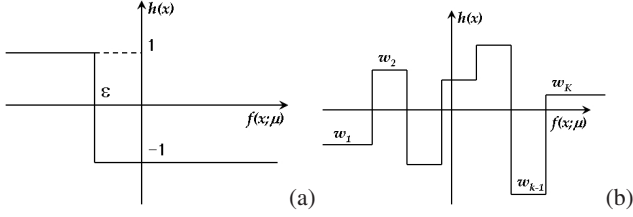


Figure 4. (a) Binary decision stump. (b) Regression stump. The regression stump carries more representational power than the decision stump.

and the matrices $R_{q \times N}$, $S_{q \times N}$, and $H_{q \times N}$ are defined as: $R = [r(x_1), \dots, r(x_N)]$, $S = [s(x_1), \dots, s(x_N)]$, $H = [h(x_1), \dots, h(x_N)]$. Finally, IBR invokes shrinkage (with the shrinkage factor $\eta = 0.5$), leading to a smooth output function: $g_t(x) = g_{t-1}(x) + \eta \alpha_t h_t(x)$.

3.1.2 Weak function based on decision stump

In [13], the over-complete feature representation based on the local rectangle features [8, 11] is used to construct one-dimensional (1D) decision stumps as primitives of the dictionary set \mathcal{H} . This construction enables robustness to appearance variation and fast computation. Each local rectangle feature has its own attribute μ , namely feature type and window position/size.

A 1D decision stump $h(x)$ is associated with a local rectangle feature $f(x; \mu)$, a decision threshold ϵ , and a binary direction indicator p , i.e., $p \in \{-1, +1\}$.

$$h(x; \mu) = \begin{cases} +1 & \text{if } pf(x; \mu) \geq p\epsilon \\ -1 & \text{otherwise} \end{cases}. \quad (12)$$

Figure 4(a) illustrates a decision stump. Given a moderate image size, one can generate a huge number of features by varying their attributes. Denote the number of features by M . By adjusting the threshold ϵ (say K evenly spaced levels), one can further create K decision stumps per feature. In total, we have $2KM$ 1D decision stumps.

A weak function is constructed as a q -dimensional (q -D) decision stump $h(x)_{q \times 1}$ that stacks q 1D decision stumps.

$$h(x; \mu_1, \dots, \mu_q) = [h_1(x; \mu_1), \dots, h_q(x; \mu_q)]^T. \quad (13)$$

Because each $h_j(x; \mu_j)$ is associated with a different parameter, one can construct a sufficiently large weak function set that contains $(2KM)^q$ weak functions!

3.1.3 Incremental feature selection

Boosting operates as a feature selector: At each round of boosting, the features that can maximally decrease the cost function are selected. However, to transform the boosting algorithm into an efficient implementation, there is a computational bottleneck, that is the maximization task in (10). This task necessitates a *greedy* feature selection scheme,

which is too expensive to evaluate, because it involves evaluating $(2MNK)^q$ decision stumps for each boosting round.

IBR uses an *incremental* feature selection scheme by breaking the q -D regression problem into q dependent 1D regression problems. Using the incremental vector

$$h^i(x)_{i \times 1} = [h_1(x), \dots, h_i(x)]^T = [h^{i-1}(x)^T, h_i(x)]^T, \quad (14)$$

the optimal $h_i(x)$ is searched to maximize the $\epsilon(h^i)$, which is similarly defined as in (11) but based on all i ($i \leq q$) dimensions processed so far. The incremental selection scheme needs evaluating only $2qMNK$ decision stumps with some overhead computation while maintaining the dependence among the output dimension to some extent.

3.2. Image-based boosting ridge regression (IBRR)

The above-reviewed IBR has two major drawbacks. First, it is restrictive to use the subspace regularization term $\|\mu - g(x_n)\|_B^2$ in (7), which amounts to a multivariate Gaussian assumption about the output variable that often manifests a non-Gaussian structure for real data. As a result, the generalization capability is hampered. Second, the weak function $h(x)$ is too “weak” as it consists of several 1D binary decision stumps $h_j(x)$ sharing the same weight coefficient α . Consequently, the training procedure takes a long time and the learned regression function uses too many weak functions, affecting its running speed. To overcome the drawbacks of IBR, we propose to (i) replacing the subspace regularization and (ii) enhancing the modeling strength of the weak function.

3.2.1 Weak function based on regression stump

Instead of using the 1D binary decision stumps as primitives, we propose to use regression stumps. A regression stump $h(x; \mu)$, illustrated in Figure 4(b), is defined as

$$h(x; \mu) = \sum_{k=1:K} w_k [f(x; \mu) \in R_k] = e(x; \mu)^T w, \quad (15)$$

where $[\cdot]$ is an indicator function, $f(x; \mu)$ is the response function of the local rectangle feature with attribute μ , and $\{R_k; k = 1, 2, \dots, K\}$ are evenly spaced intervals¹. In the above, all the weights w_k are compactly encoded by a weight vector $w_{K \times 1} = [w_1, w_2, \dots, w_K]^T$ and the vector $e(x; \mu)$ is some column of the identity matrix: only one element is one and all others are zero. Similarly, we construct the weak function $h(x)_{q \times 1}$ by stacking q different 1D regression stumps, i.e.,

$$h(x; \mu_1, \dots, \mu_q) = [e_1(x; \mu_1)^T w_1, \dots, e_q(x; \mu_q)^T w_q]^T, \quad (16)$$

¹There are two exceptions: the first interval R_1 starts from $-\infty$ and the last one R_K ends at $+\infty$. The interval boundary points are empirically determined. We first find the minimum and maximum responses for the feature and then uniformly divide them.

where w_j is the weight vector for the j^{th} regression stump $h_j(\mathbf{x}; \mu_j)$. We further encode the weights belonging to all regression stumps into a weight matrix $\mathbf{W}_{K \times q} = [w_1, w_2, \dots, w_q]$. Since we now use the weights, we drop the common coefficient α in the regression output function defined in (8) and instead use the following form:

$$\mathbf{g}_t(\mathbf{x}) = \mathbf{g}_{t-1}(\mathbf{x}) + \mathbf{h}_t(\mathbf{x}) = \sum_{i=1:t} \mathbf{h}_i(\mathbf{x}). \quad (17)$$

It is easy to verify that a regression stump can be formed by combining multiple decision stumps. Such a combination strengthens the modeling power of weak functions and consequently accelerates the training process. Our empirical evidence shows that the training time is almost inversely proportional to the number of levels used in the weak function. Because using the regression stump brings the risk of overfitting, we will ameliorate this risk by considering the model complexity of the regression stump.

3.2.2 Boosting ridge regression

Ridge regression, also known as Tikhonov regularization [3], is the most commonly used method of regularization for an ill-conditioned system of linear equations. It also allows analytic derivations. In this paper, we adopt the ridge regression principle into a boosting framework.

The model complexity of the regression output function $\mathbf{g}_t(\mathbf{x})$ depends on its weight matrices $\{\mathbf{W}_1, \mathbf{W}_2, \dots, \mathbf{W}_t\}$. Because boosting regression proceeds iteratively, at the t^{th} boosting iteration, we set up the following ridge regression task that only involves the weight matrix \mathbf{W}_t (dropping the subscript t for notational clarity):

$$\arg \min_{\mathbf{W}} \{J(\mathbf{g}) = \sum_{n=1:N} \{\|\mathbf{r}(\mathbf{x}_n) - \mathbf{h}(\mathbf{x}_n)\|_{\mathbf{A}}^2\} + \lambda \|\mathbf{W}\|_{\mathbf{B}}^2\}. \quad (18)$$

Because the weight vectors $\{w_1, w_2, \dots, w_q\}$ in the weight matrix \mathbf{W} are associated with q different local rectangle features, the optimization in (18) implies two subtasks:

1. Given a set of q features with attributes μ_1, \dots, μ_q , respectively, find the optimal matrix $\hat{\mathbf{W}}(\mu_1, \dots, \mu_q)$ and the minimum cost $\hat{J}(\mu_1, \dots, \mu_q)$;
2. Find the optimal set of q features with respective attributes $\hat{\mu}_1, \dots, \hat{\mu}_q$ that minimizes the cost $\hat{J}(\mu_1, \dots, \mu_q)$. This corresponds to feature selection.

Like [13], the optimization in (18) necessitates a greedy feature selection that is computationally unmanageable; therefore we resort to the suboptimal yet computationally amenable incremental feature selection scheme.

To proceed, we introduce the following “incremental” vectors and matrices:

$$\mathbf{A}^i = \begin{bmatrix} \mathbf{A}^{i-1} & \mathbf{a}^{i-1} \\ \mathbf{a}^{i-1\top} & a_i \end{bmatrix}, \mathbf{h}^i = \begin{bmatrix} \mathbf{h}^{i-1} \\ h_i \end{bmatrix}, \mathbf{r}^i = \begin{bmatrix} \mathbf{r}^{i-1} \\ r_i \end{bmatrix}.$$

1. Initialization $t = 0$.

- (a) Set the fixed \mathbf{A} and \mathbf{B} (the normalization matrices), λ (the regularization coefficient), and η (the shrinkage factor).
- (b) Set the values related to the stopping criteria: T_{max} (the maximum number of iterations) and J_{min} (the minimum cost function).
- (c) Set initial values for $t = 0$: $\mathbf{g}_0(\mathbf{x}) = 0$ and $\mathbf{r}_0(\mathbf{x}) = \mathbf{y}(\mathbf{x})$.

2. Iteration $t = 1, \dots, T_{max}$

- (a) Find the optimal $\hat{\mathbf{h}}_t$ using the feature selection algorithm in Figure 6.
- (b) Form the new function $\mathbf{g}_t(\mathbf{x}) = \mathbf{g}_{t-1}(\mathbf{x}) + \eta \hat{\mathbf{h}}_t(\mathbf{x})$.
- (c) Evaluate the approximation error $\mathbf{r}_t(\mathbf{x}) = \mathbf{y}(\mathbf{x}) - \mathbf{g}_t(\mathbf{x})$ and the cost function $J(\mathbf{g}_t)$.
- (d) Check convergence, e.g., see if $J(\mathbf{g}_t) < J_{min}$.

Figure 5. The proposed image-based boosting ridge regression (IBRR) algorithm.

Assuming that we have found the features up to $i - 1$, that is, the incremental vector $\mathbf{h}^{i-1}(\mathbf{x}; \mu_1, \dots, \mu_{i-1})$ and the weight vectors w_1, \dots, w_{i-1} are known, we aim to find the weak function $h_i(\mathbf{x}; \mu_i) = \mathbf{e}_i(\mathbf{x}; \mu_i)^\top \mathbf{w}_i$ that minimizes the following ridge regression cost $J^i(\mu_i, \mathbf{w}_i)$.

$$J^i(\mu_i, \mathbf{w}_i) = \sum_{n=1:N} \{\|\mathbf{r}^i(\mathbf{x}_n) - \mathbf{h}^i(\mathbf{x}_n)\|_{\mathbf{A}^i}^2 + \lambda \|\mathbf{w}_i\|_{\mathbf{B}}^2\}. \quad (19)$$

It is easy to derive that, for a fixed μ_i , the optimal weight vector is

$$\hat{\mathbf{w}}_i(\mu_i) = \Gamma_i(\mu_i)^{-1} \tau_i \mathbf{e}_i(\mathbf{x}; \mu_i), \quad (20)$$

where

$$\begin{aligned} \Gamma_i(\mu_i) &= \lambda \mathbf{B} + \sum_{n=1:N} \{\mathbf{e}_i(\mathbf{x}_n; \mu_i) a_i \mathbf{e}_i(\mathbf{x}_n; \mu_i)^\top\}, \\ \tau_i &= \sum_{n=1:N} \{(\mathbf{r}^{i-1}(\mathbf{x}_n) - \mathbf{h}^{i-1}(\mathbf{x}_n))^\top \mathbf{a}^{i-1} + r_i(\mathbf{x}_n)^\top a_i\}. \end{aligned} \quad (22)$$

Then, we search the optimal μ_i to minimize the cost function $J^i(\mu_i, \hat{\mathbf{w}}_i(\mu_i))$.

When $\mathbf{A} = \mathbf{B} = \mathbf{I}_q$, the incremental feature selection gives the optimal solution. In this case, the optimal weight $w_{j,k}$ for the j^{th} weak function is the weighted average:

$$w_{j,k} = \frac{\sum_{n=1:N} r_j(\mathbf{x}_n) [f(\mathbf{x}_n; \mu_j) \in R_k]}{\lambda + \sum_{n=1:N} [f(\mathbf{x}_n; \mu_j) \in R_k]}. \quad (23)$$

Similar to [13], we randomly permute the order of the dimension of the output variable in order to improve robustness and remove bias. We also use other tricks to improve computational efficiency, including (i) randomly sampling the dictionary set, *i.e.*, replacing M by a smaller M' ; and (ii) randomly sampling the training data set, *i.e.*, replacing N by a smaller N' .

Figure 5 summarizes the IBRR algorithm and Figure 6 the incremental feature selection scheme inside IBRR.

4. Medical anatomy detection

We now present how to integrate the learned regression function into a practical algorithm that detects medical

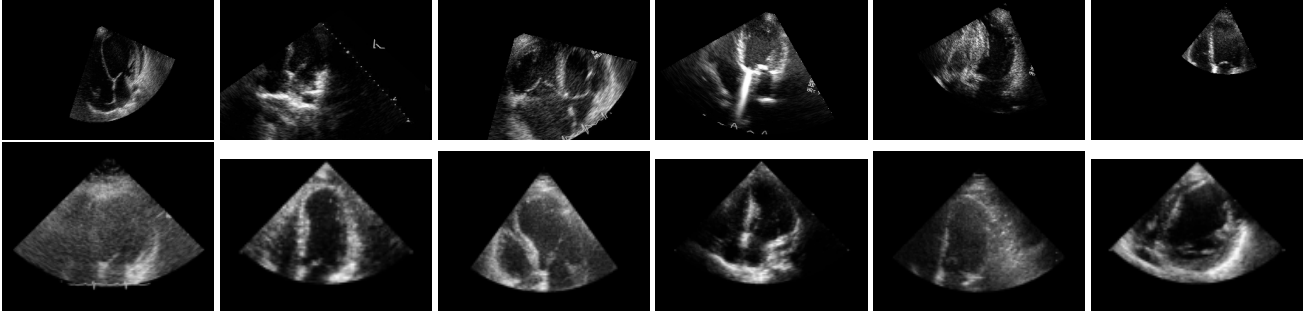


Figure 7. Top row: example images used in Experiment-I where only the LV center is searched. Bottom row: example images used in Experiment-II where the LV center, size, and orientation are searched. Unlike those images in the first experiment, the size of the images in the second experiment varies. Note that is a significant variation in the parameter to be searched as well as in the LV appearance due to ultrasound noise, deformable shape, differences in instrument, sonographer, and patient, etc.

1. Initialization.
 - Create a random permutation of $\{1, 2, \dots, q\}$, yielding $\{[1], [2], \dots, [q]\}$.
2. Iteration over the dimension of the output variable $i = 1, 2, \dots, q$
 - (optional) Sample M' local rectangle features from the dictionary set and form the reduced set of weak functions \mathcal{H}' .
 - (optional) Sample N' data points from the training set.
 - Loop over the feature $\mu_i = 1, 2, \dots, M'$ to find $h_{[i]} = \arg \min_{\mu_i} J^{[i]}(\mu_i, \hat{w}_i(\mu_i))$.
 - Form the new vector $h^{[i]} = [h^{[i-1]T}, h_{[i]}]^T$.

Figure 6. The incremental feature selection algorithm for IBRR.

anatomy. In theory, only one scan is needed to find the target; in practice, we conduct a sparse set of random scans and then estimate the parameter using fusion. Suppose that in total M random samples are scanned at positions $\{\theta^{<1>}, \theta^{<2>}, \dots, \theta^{<M>}\}$. For each $\theta^{<m>}$, we invoke the regressor to predict the difference parameter $d\theta^{<m>}$ and subsequently predict the target parameter $\theta_0^{<m>}$ as follows:

$$d\theta^{<m>} = \mathcal{F}(\mathcal{I}(\theta^{<m>})), \quad m = 1, 2, \dots, M; \quad (24)$$

$$\theta_0^{<m>} = \theta^{<m>} + d\theta^{<m>}, \quad m = 1, 2, \dots, M. \quad (25)$$

We treat the M predictions $\{\theta_0^{<m>}; m = 1, 2, \dots, M\}$ as independent and compute their mean value as the final estimate $\hat{\theta}_0$ for the ground truth parameter:

$$\hat{\theta}_0 = M^{-1} \sum_{m=1:M} \theta_0^{<m>}. \quad (26)$$

One disadvantage of the proposed IBRR algorithm is that it lacks a confidence measure, *i.e.*, the regressor is a black box that tells no confidence about its prediction. In order to provide a confidence score, we learn a binary detector \mathcal{D} specialized for the anatomy of interest. After finding the m^{th} prediction $\theta_0^{<m>}$, we apply the detector \mathcal{D} to the image patch $\mathcal{I}(\theta_0^{<m>})$. If the detector \mathcal{D} fails, then we discard the m^{th} sample; otherwise, we keep the confidence score c^m . This way, we have a weighted set $\{(\theta_0^{<j>}, c^{<j>}); j = 1, 2, \dots, J\}$ (note that $J \leq M$ as samples might be discarded), based on which we calculate the

weighted mean as the final estimate $\hat{\theta}_0$

$$\hat{\theta}_0 = \{ \sum_{j=1:J} c^{<j>} \theta_0^{<j>} \} / \{ \sum_{j=1:J} c^{<j>} \}. \quad (27)$$

In practice, we stop scanning when $J \geq J_{valid}$ in order to further save computation. If there is no sample $\theta_0^{<m>}$ passing the detector \mathcal{D} , then we still use (26) as the final estimate. We find that combining the regressor and binary detector is an effective tool for medical anatomy detection; it only needs a smaller number of scans to reach a better performance than the method using only the regressor.

5. Experimental results and discussions

We tested the boosting regression approach to LV detection from 2D echocardiogram. We had in total 527 sequences of the A4C view. Though we had video sequences, we focused on detecting the LV at the end of diastole (ED) frame, when the LV dilates to its maximum. We randomly selected 450 ED frames for training and the remaining 77 for testing. We conducted two experiments.

5.1. Experiment-I: 2D translation

In the first experiment, we normalized the scale and orientation of the LV; the normalized image size is 200×300 . We left the translation parameter (t_x, t_y) free and empirically found that the range of t_x and t_y is $t_x \sim [120, 213]$ and $t_y \sim [52, 104]$. Figure 7 (the top row) shows six normalized images, which manifest the significant LV appearance variation. For each ED frame, we randomly sampled 20 image patches and recorded their difference vectors; in total, we generated 9,000 training data points.

There are several tuning parameters in the IBRR algorithm. First, we set $A = B = I_q$ for simplicity, which means that the weighted average formula in (23) is applicable. For the number of threshold levels K for a weak function, the regularization coefficient λ and the shrinkage coefficient η , we empirically tested different combinations and decided to use the following: $K = 64$, $\lambda = 0.1/K$, and

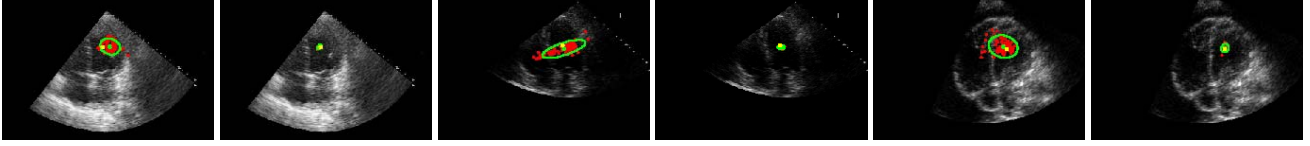


Figure 8. The odd-indexed images show the 100 predicted target outputs (red) and the even-indexed images show only the predicted target outputs (red) passing the detector. The green point is the final estimate of the target position, the green curve is the 95% confidence curve, and the yellow point indicates the ground truth position. Note that the region bounded the 95% confidence curve on the even-indexed images is significantly smaller than that on the odd-indexed images.

$\eta = 0.1$. After learning, we ended up with a regressor with 500 weak functions or 1,000 local rectangle features (as one weak function has two features). The whole IBRR training procedure takes about 15 hours on a high-end workstation with four Xeon 3GHz CPUs and 3GB RAM. We followed [2] to train a LV detector of three cascades, each with 8, 35, and 86 local rectangle features, respectively.

We implemented three scanning methods: “IBRR”, “IBRR+Det”, and “Det”. The “IBRR” means that we randomly scanned the image within the range of t_x and t_y using the learned IBRR function and used (26) as the final estimate of the target position. The “IBRR+Det” means that we further equipped the “IBRR” method with the trained detector and used (27) as the final estimate. We also set $J_{valid} = 5$ to enable early exit when scanning. The “Det” means that we exhaustively scanned the image within the same range using the detector and used the location that maximizes the detector response as the final estimate.

Given a testing ED frame, we randomly scanned M positions to obtain M predictions for the target location. Figure 8 shows $M = 100$ predicted target positions (the red points) on three example images: The majority of the prediction is close to the ground truth position (the yellow point) although there are outliers. In Figure 8, we also showed only the predicted points that pass the binary detector: All the outliers are eliminated, thereby significantly improving the precision of the estimate. This improvement is evidenced by the smaller region bounded by the confidence curves.

Figure 9 shows the effect of the number of features and the number of random scans to the detection error, which measured as $t_{detected} - t_{groundtruth}$ in t_x and t_y separately. Here we reported the bias and standard deviation of the detection error for the training and testing sets, respectively. From Figure 9, we observe that (i) There is no conspicuous difference between the training and testing detection error for all three methods, implying that the learning generalizes well; (ii) For the “IBRR” method, the localization uncertainty in the t_x is much higher than in t_y ; (iii) The increase in the number of features and the number of random scans does not significantly decrease the detection error and only marginally improves the detection precision; and (iv) The “IBRR+Det” method significantly reduces the detection uncertainty to a level similar to the “Det” method.

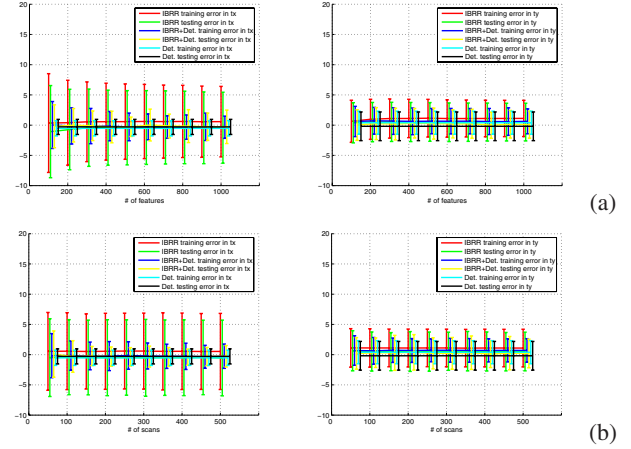


Figure 9. (a) The detection error versus the number of local rectangle features with the number of scans fixed at 100. (b) The detection error versus the number of scans with the number of selected features fixed at 400.

| Method | IBRR | IBRR+Det | Det |
|------------------------|------------------|------------------|------------------|
| # of features | 400 | 400+129 | 129 |
| Training err. in t_x | 0.58 ± 6.36 | -0.09 ± 2.96 | -0.53 ± 1.26 |
| Training err. in t_y | 1.10 ± 3.17 | 0.64 ± 2.39 | 0.40 ± 1.92 |
| Testing err. in t_x | -0.42 ± 6.21 | -0.26 ± 2.41 | -0.29 ± 1.25 |
| Testing err. in t_y | 0.51 ± 3.24 | 0.24 ± 2.90 | -0.19 ± 2.38 |
| # of eff. scans | 100 | 58 | 3174 |
| Avg. speed | 20ms | 16ms | 122ms |

Table 1. Performance comparison of the three methods for the 2-parameter case.

In Table 1, we profiled the three methods². In terms of localization performance, there is no systematic detection bias observed in all three methods. The “Det” method gives the best localization precision and the “IBRR” method is the worst while the “IBRR+Det” is comparable to the “Det”. In terms of speed, the “Det” method is the slowest due to its exhaustive native and the “IBRR+Det” is the fastest, more than 7.5 times faster than the “Det”, while the “IBRR” is slower than the “IBRR+Det”. The speed, which can be further tuned, was recorded on a laptop with a Pentium 2.1GHz CPU and 2GB RAM. In short, the “IBRR+Det” achieves appealing detection performance while running the fastest.

²To count the number of effective scans in Table 1, we excluded those scans if their associated image patches have less than 40% of their pixels inside the known fan.

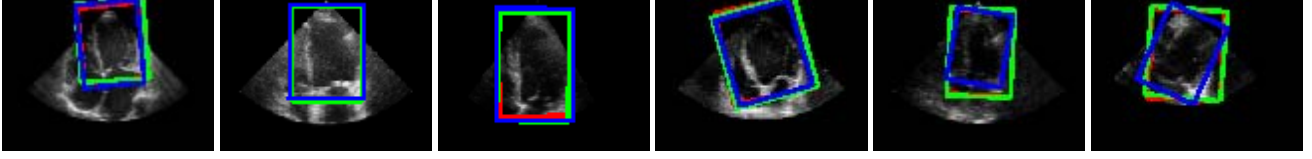


Figure 10. The inferred LV box versus the ground truth. The red box is from the “IBRR” method, the green is from the “IBRR+Det” method, and the blue is the ground truth.

5.2. Experiment-II: 5D parameter

In the second experiment, we evaluated the 5-parameter setting for the unaligned LV in the A4C view. Figure 7 (the bottom row) shows six ED images with the unaligned LV present. The range of the five parameters is empirically found as: $t_x \sim [43, 118]$, $t_y \sim [24, 70]$, $s_x \sim [26, 86]$, $s_y \sim [37, 92]$ and $\alpha \sim [-25, 35]$. We scanned the image following the above range. The average image size is 111×151 .

Using the same set of training parameter in Experiment-I, we trained the regressor based on 450 randomly selected ED frames, each yielding 30 image patches; in total we had 13,500 training data. It takes more than two days to train the regressor, which consists of 10,000 local rectangle features (or 200 weak functions). Training the detector is not as straightforward as in Experiment-I because here the image rotation is considered. We followed [12] to train the detector, which is able to simultaneously detect the object as well as infer its rotation yet using only one integral image. We experimented the same three methods: “IBRR”, “IBRR+Det”, and “Det”. We set $J_{valid} = 10$ in the “IBRR+Det” method. For the “Det” method, we exhaustively scanned the image every 4 pixels in translation and every 4 pixels in scale.

Table 2 compares the three scanning methods. The error in scale is measured as $s_{detected}/s_{groundtruth} - 1$. Because we did not observe significant performance difference between training and testing, we pooled them together and jointly reported the results. We note that the “IBRR+Det” runs the fastest, about 7 times faster than the “IBRR” method and more than 50 times faster than the “Det” method, while yielding comparable performance to the “IBRR” in terms of bias and improving the localization precision. The slowest “Det” method does not yield the best performance any more in terms of either bias or variance because it does not exhaust all possible configurations. Figure 10 shows example images with estimated and ground truth boxes overlaid.

6. Conclusion

We have presented an image-based boosting ridge regression (IBRR) approach to fast medical anatomy detection. The IBRR, a general learning algorithm for multiple regression with multidimensional output, exhibits

| Method | IBRR | IBRR+Det | Det |
|------------------|------------------|------------------|-----------------|
| # of features | 1000 | 1000+1201 | 1201 |
| err. in t_x | 0.32 ± 3.13 | 0.65 ± 2.07 | 1.69 ± 3.40 |
| err. in t_y | 0.67 ± 2.40 | 1.25 ± 1.95 | 0.84 ± 3.73 |
| err. in s_x | 0.02 ± 0.12 | 0.04 ± 0.12 | 0.05 ± 0.17 |
| err. in s_y | 0.01 ± 0.08 | 0.02 ± 0.08 | 0.04 ± 0.15 |
| err. in α | -1.76 ± 7.17 | -0.98 ± 6.39 | 0.22 ± 6.74 |
| # of eff. scans | 200 | 38 | 29383 |
| Avg. speed | 704ms | 118ms | 6300ms |

Table 2. Performance comparison of the three methods for the 5-parameter case.

good generalization capability and training efficiency. The regression-based detection algorithm replaces the exhaustive scanning of the query image required by the classifier-based detector by a sparse scanning and reaches improved accuracy with significantly less computation and no need for image rotation. In the future, we will investigate regression approaches with prediction confidence and extend the same approach to other medical applications.

References

- [1] T. Cootes, C. Beeston, G. Edwards, and C. Taylor. A unified framework for atlas matching using active appearance models. In *IPMI*, 1999. 3
- [2] B. Georgescu, X. S. Zhou, D. Comaniciu, and A. Gupta. Database-guided segmentation of anatomical structures with complex appearance. In *Proc. CVPR*, 2005. 1, 2, 7
- [3] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001. 5
- [4] M. Jones and J. Viola. Face recognition using boosted local features. In *Proc. ICCV*, 2003. 1, 2
- [5] S. Li and Z. Zhang. FloatBoost learning and statistical face detection. *PAMI*, 26(9):1112–1123, 2004. 1
- [6] M. Lorenzo-Vald, G. Sanchez-Ortiz, R. Mohiaddin, and D. Rueckert. Segmentation of 4D cardiac MR images using a probabilistic atlas and the EM algorithm. In *MICCAI*, 2003. 3
- [7] J. Mazziotta, A. Toga, A. Evans, J. Lancaster, and P. Fox. A probabilistic atlas of the human brain: Theory and rational for its development. *Neuroimage*, 2:89–101, 1995. 3
- [8] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *Proc. ICCV*, 1998. 4
- [9] G. Subsol, J. Thirion, and N. Ayache. A general scheme for automatically building 3D morphometric anatomical atlases: application to a skull atlas. *Medical Image Analysis*, 2:37–60, 1998. 3
- [10] A. Torralba and P. Sinha. Statistical context priming for object detection. In *Proc. ICCV*, 2001. 3
- [11] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. of CVPR*, 2001. 1, 4
- [12] J. Zhang, S. Zhou, L. McMillan, and D. Comaniciu. Joint real-time object detection and pose estimation using probabilistic boosting network. In *Proc. CVPR*, 2007. 8
- [13] S. Zhou, B. Georgescu, X. Zhou, and D. Comaniciu. Image-based regression using boosting method. In *Proc. ICCV*, 2005. 3, 4, 5