Incremental On-line Semi-supervised Learning for Segmenting the Left Ventricle of the Heart from Ultrasound Data

Gustavo Carneiro[*,†]
Australian Centre for Visual Technologies
The University of Adelaide

Jacinto C. Nascimento[*]
Instituto de Sistemas e Robótica
Instituto Superior Técnico

Abstract

Recently, there has been an increasing interest in the investigation of statistical pattern recognition models for the fully automatic segmentation of the left ventricle (LV) of the heart from ultrasound data. The main vulnerability of these models resides in the need of large manually annotated training sets for the parameter estimation procedure. The issue is that these training sets need to be annotated by clinicians, which makes this training set acquisition process quite expensive. Therefore, reducing the dependence on large training sets is important for a more extensive exploration of statistical models in the LV segmentation problem. In this paper, we present a novel incremental on-line semi-supervised learning model that reduces the need of large training sets for estimating the parameters of statistical models. Compared to other semi-supervised techniques, our method yields an on-line incremental re-training and segmentation instead of the off-line incremental re-training and segmentation more commonly found in the literature. Another innovation of our approach is that we use a statistical model based on deep learning architectures, which are easily adapted to this on-line incremental learning framework. We show that our fully automatic LV segmentation method achieves state-of-the-art accuracy with training sets containing less than twenty annotated images.

1. Introduction

One of the major problems investigated in medical image analysis is the automatic segmentation of the left ventricle (LV) of the heart from ultrasound data. From a clinical setting viewpoint, there are important reasons that justify the interest in solving this problem, such as [4]: 1) it can increase patient throughput; and 2) it can reduce inter-user variation in the LV delineation procedure. From the medical image analysis standpoint, LV segmentation has been intensively investigated over the last years because this problem offers several challenges, including: large appearance and shape variation of the LV, low signal-to-noise ratio of ultrasound data, and edge dropout due to imaging conditions.

Lately, there has been an increasing interest in the study of statistical pattern recognition approaches to solve the automatic LV segmentation problem [2, 3, 5]. These approaches build a classifier by modeling statistically the LV appearance and shape using a set of manually annotated images (i.e., the training set). Essentially, this procedure consists of estimating a large number of parameters of such statistical model, and the robustness of this estimation is directly related to the size and richness of the training set. Actually, it is not uncommon that systems based on such approaches need in the order of hundreds of training images [5, 18]. However, acquiring such large training sets is prohibitively expensive for most of the researchers working on medical image analysis, resulting in insufficient investigation of these models. Therefore, methods that alleviate the dependence on large annotated training sets are of utmost importance for a more extensive exploration of statistical models in medical image analysis.

The problem of training statistical models with few manually annotated data can be addressed via semi-supervised learning methods [20]. The main assumption underlying these methods is that samples belonging to the same class tend to cluster together in the feature space, and if a few annotated examples are given, we can associate unannotated samples of the cluster with the label of annotated samples in that same cluster, as shown in Fig. 1. One class of semi-supervised learning methods is based on incremental or self-training algorithms which uses a small training set to initially estimate the model parameters, and then use this model to classify unannotated samples and retrain the same model [9, 11, 12, 15, 16, 17]. Particularly important for these methods are the type of model used and the way of classifying unannotated samples for re-training. This classification of unannotated samples has shown to be more effective if an external classifier is used [11, 15].

In this paper, we introduce a new incremental on-line semi-supervised approach for the problem of automatic LV segmentation. Initially, a small annotated training set is used to estimate the parameters of a statistical model that automatically learns the LV shape and appearance, and this model is used to build an initial classifier that detects and segments the LV from ultrasound data. Given a new unannotated test sequence, the system uses this trained classi-

[*]This work was supported by the FCT (ISR/IST plurianual funding) through the PIDDAC Program funds and Project HEARTTRACK (PTDC/EEA-CRO/103462/2008). This work was partially funded by EU Project IMASEG3D (PIIF-GA-2009-236173). [†]This work was developed while Gustavo Carneiro was with the Instituto Superior Técnico.

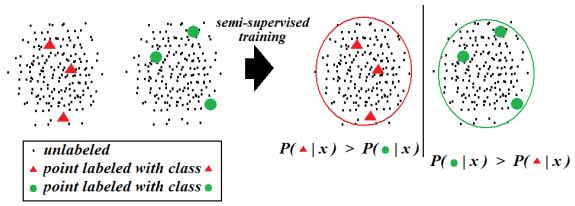


Figure 1. Semi-supervised learning. The graph on the left shows the classification problem where only a small subset of the samples are annotated. The graph on the right displays the result of semi-supervised learning, where $P(o|\mathbf{x})$ is the probability of class o given point $\mathbf{x}$.

fier to detect positive hypotheses (i.e., LV segmentations) in each frame, which are verified by an external classifier that uses a prior LV model of shape and appearance (i.e., this external classifier does not go through any training process). The positive hypotheses that have survived the verification process are then added to the training set, which is used to re-train the statistical model incrementally. The system also produces an LV segmentation using the updated statistical model.

Our approach is innovative in the sense that it yields *on-line* re-training and segmentation, instead of the more common setting adopted in semi-supervised approaches with *off-line* re-training and segmentation. This innovation constrains the annotated samples to be included in the training set for re-training. This means that we have to include samples which have been annotated with relatively low confidence, or the process halts due to lack of new samples for re-training (this relaxation has also been explored by Levin et al. [11]). Another innovation of our approach is that the classifier being trained is based on deep neural networks [7], which, contrary to the more commonly used boosting classifiers [9, 11, 15, 16, 17], is straightforward to be adapted from a batch to an incremental (on-line) learning setting. Finally, we derive the formulation of the incremental on-line semi-supervised approach. The main result of the paper is that we achieved competitive fully automatic LV segmentation results in public databases [13] using less than twenty manually annotated images.

2. Incremental On-Line Semi-supervised Learning

For the derivation of our algorithm, assume that an image region is represented by a feature vector $\mathbf{x} \in \Re^D$, and the annotation of an image region is denoted by a vector $\mathbf{y} \in \Re^{2N}$ containing N two-dimensional points. Consider that we have a set of training image regions represented by $\mathcal{X}$ with $\mathcal{Y}$ denoting the respective set of manual annotations, and that we also have a test sequence of unannotated images $\{I_t\}_{t=1..T}$. An image region is a crop of the original image that contains the annotation aligned according to a specific position, scale and rotation of the annotation points (see Fig. 3). The goal of the incremental learning is to estimate the parameters $\boldsymbol{\theta}$ of the classifier $p(\mathbf{y}|\mathbf{x},\boldsymbol{\theta})$ that measures the probability of annotation $\mathbf{y}$ given the feature vector $\mathbf{x}$, and the parameter vector $\boldsymbol{\theta}$. In order to estimate $\boldsymbol{\theta}$, we use the annotated training set $\{\mathcal{X},\mathcal{Y}\}$, the test sequence $\{I_t\}_{t=1..T}$, and an external classifier $p(\mathbf{y}_i^{(e)}|\mathbf{y}_i,\mathbf{x}_i)$, which represents the probability of segmentation $\mathbf{y}_i^{(e)}$ given an initial guess $\mathbf{y}_i$ and feature vector $\mathbf{x}_i$ (note that this external classifier does not have any parameters to estimate). The estimation of $\boldsymbol{\theta}$ can be summarized as [14]:

$$\begin{aligned}\boldsymbol{\theta}^\star &= \arg\max_{\boldsymbol{\theta}} P(\mathcal{Y}|\mathcal{X},\boldsymbol{\theta}) \\ &\propto \arg\max_{\boldsymbol{\theta}} \log \sum_{\widetilde{\mathbf{x}}_i \in I_t} f_i \frac{p(\mathcal{Y},\widetilde{\mathbf{y}}_i^{(e)},\widetilde{\mathbf{y}}_i|\widetilde{\mathbf{x}}_i,\mathcal{X},\boldsymbol{\theta})}{f_i} \\ &= \arg\max_{\boldsymbol{\theta}} \log \sum_{\widetilde{\mathbf{x}}_i \in I_t} f_i \frac{p(\widetilde{\mathbf{y}}_i^{(e)}|\widetilde{\mathbf{y}}_i,\widetilde{\mathbf{x}}_i)p(\mathcal{Y},\widetilde{\mathbf{y}}_i|\widetilde{\mathbf{x}}_i,\mathcal{X},\boldsymbol{\theta})}{f_i},\end{aligned} \tag{1}$$

where f_i is an auxiliary function that has the constraints $\sum_i f_i = 1$ and $f_i \geq 0$, $\widetilde{\mathbf{y}}_i = \arg\max_{\mathbf{y}} p(\mathbf{y}|\widetilde{\mathbf{x}}_i,\boldsymbol{\theta})$, and $\widetilde{\mathbf{y}}_i^{(e)} = \arg\max_{\mathbf{y}} p(\mathbf{y}|\widetilde{\mathbf{y}},\widetilde{\mathbf{x}}_i)$. Using Jensen's inequality, we can find the following lower bound to the objective function (1):

$$\begin{aligned}&\underbrace{\sum_{\widetilde{\mathbf{x}}_i \in I_t} f_i \log \frac{p(\widetilde{\mathbf{y}}_i^{(e)}|\widetilde{\mathbf{y}}_i,\widetilde{\mathbf{x}}_i)p(\mathcal{Y},\widetilde{\mathbf{y}}_i|\widetilde{\mathbf{x}}_i,\mathcal{X},\boldsymbol{\theta})}{f_i}}_{\text{Lower Bound}} \leq \\ &\log \sum_{\widetilde{\mathbf{x}}_i \in I_t} f_i \frac{p(\widetilde{\mathbf{y}}_i^{(e)}|\widetilde{\mathbf{y}}_i,\widetilde{\mathbf{x}}_i)p(\mathcal{Y},\widetilde{\mathbf{y}}_i|\widetilde{\mathbf{x}}_i,\mathcal{X},\boldsymbol{\theta})}{f_i},\end{aligned} \tag{2}$$

which is easier to maximize than the original objective function (1). Therefore, we solve the following optimization problem:

$$\begin{aligned}\boldsymbol{\theta}^\star = \arg\max_{\boldsymbol{\theta}} &\sum_{\widetilde{\mathbf{x}}_i \in I_t} f_i \log \frac{p(\widetilde{\mathbf{y}}_i^{(e)}|\widetilde{\mathbf{y}}_i,\widetilde{\mathbf{x}}_i)p(\mathcal{Y},\widetilde{\mathbf{y}}_i|\widetilde{\mathbf{x}}_i,\mathcal{X},\boldsymbol{\theta})}{f_i} \\ \text{s.t.} &\sum_{\widetilde{\mathbf{x}}_i \in I_t} f_i = 1, f_i \geq 0, \forall i \in 1,...,T.\end{aligned} \tag{3}$$

In order to find the function f_i, we take the derivative of the Lagrangian $\mathcal{L} = \lambda(\sum_i f_i - 1) - \sum_i \gamma_i f_i - \sum_{\widetilde{\mathbf{x}}_i \in I_t} f_i \log \frac{p(\widetilde{\mathbf{y}}_i^{(e)}|\widetilde{\mathbf{y}}_i,\widetilde{\mathbf{x}}_i)p(\mathcal{Y},\widetilde{\mathbf{y}}_i|\widetilde{\mathbf{x}}_i,\mathcal{X},\boldsymbol{\theta})}{f_i}$ with respect to f_i and set it to zero, which produces:

$$f_i = p(\widetilde{\mathbf{y}}_i^{(e)}|\widetilde{\mathbf{y}}_i,\widetilde{\mathbf{x}}_i)p(\widetilde{\mathbf{y}}_i|\widetilde{\mathbf{x}}_i,\boldsymbol{\theta}). \tag{4}$$

Hence, we can formulate an iterative algorithm comprising the following expectation (E) from (4) and maximization (M) steps:

- **E-step**:

$$f_i^{(t)} = p(\widetilde{\mathbf{y}}_i^{(e)}|\widetilde{\mathbf{y}}_i,\widetilde{\mathbf{x}}_i)p(\widetilde{\mathbf{y}}_i|\widetilde{\mathbf{x}}_i,\boldsymbol{\theta}^{(t-1)}) \tag{5}$$

- **M-step**:

$$\boldsymbol{\theta}^{(t)} = \arg\max_{\boldsymbol{\theta}} \mathrm{E}_{f_i^{(t)}}\Big[\log p(\widetilde{\mathcal{Y}}|\widetilde{\mathcal{X}}, \boldsymbol{\theta})\Big], \qquad (6)$$

where the superscript (t) indicates the iteration index, $\widetilde{\mathcal{X}} = \mathcal{X} \cup \{\widetilde{\mathbf{x}}_i\}$, $\widetilde{\mathcal{Y}} = \mathcal{Y} \cup \{\widetilde{\mathbf{y}}_i^{(e)}\}$, and $\mathrm{E}_{f_i^{(t)}}[\log p(.)]$ denotes the expected value of $\log p(.)$ over $f_i^{(t)}$.

Algorithm 1 Incremental On-line Semi-supervised Learning

1: **for** t = 1:T **do**

2: E-step: Sample and re-build training set

$$\text{Sample } (\widetilde{\mathcal{Y}}^{(e)}, \widetilde{\mathcal{X}}) \sim \sum_{\widetilde{\mathbf{x}}_i \in I_t} f_i^{(t)} \times \mathcal{N}(\widetilde{\mathbf{y}}_i^{(e)}, \Sigma),$$
$$\text{with } f_i^{(t)} \geq \gamma \text{ defined in (5).}$$
$$\text{Update } \widetilde{\mathcal{Y}} = \mathcal{Y} \cup \widetilde{\mathcal{Y}}^{(e)}, \quad \widetilde{\mathcal{X}} = \mathcal{X} \cup \widetilde{\mathcal{X}} \qquad (7)$$

3: M-step: re-estimate classifier parameters

$$\boldsymbol{\theta}^{(t)} = \arg\max_{\boldsymbol{\theta}} p(\widetilde{\mathcal{Y}}|\widetilde{\mathcal{X}}, \boldsymbol{\theta})$$
$$\text{subject to } f_i^{(t)} \geq \gamma, \ \textstyle\sum_i f_i^{(t)} = 1$$

4: Produce annotation for current image

$$\mathbf{y}^* = \int_{\mathbf{y}} \int_{\mathbf{x} \in I_t} \mathbf{y} p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}^{(t)}) p(\mathbf{x}) d\mathbf{y} d\mathbf{x}, \qquad (8)$$

5: **end for**

Therefore, we propose an iterative on-line EM algorithm in Alg. 1, where T denotes the size of the unannotated test sequence $\{I_t\}_{t=1..T}$. The goal of the algorithm is to maximize (with respect to $\boldsymbol{\theta}$) and generalize (in the data space $\mathbf{x}$) the model $p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta})$ with the constraint that there are no transitions of $p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta})$ on high density regions of $p(\mathbf{x})$ [20]. Both the generalization goal and the constraint are achieved by incrementally incorporating into the training set the samples $(\widetilde{\mathbf{x}}_i, \widetilde{\mathbf{y}}_i^{(e)})$, which produced $f_i^{(t)} \geq \gamma$ (7), where $\gamma > 0$ is a free variable. Notice that Alg. 1 produces on-line segmentation results in (8) while incrementally re-training the classifier. Typically, semi-supervised learning algorithms re-train the classifier incrementally using all training set, and the classification results are produced *off-line* (i.e., after the re-training process is over) [20]. The main consequence of this difference resides in how the training set is updated for the re-training process. Using the off-line classification, one can select the data that produced $f_i^{(t)} \geq \gamma$ for high values of γ [20] (i.e., the data for which the current classifier is highly confident). For the on-line classification, high values of γ may stop the addition of newly annotated samples to $\widetilde{\mathcal{Y}}$ and $\widetilde{\mathcal{X}}$, which can halt the incremental re-training process. On the other hand, low values of γ may cause the addition of false positive samples to the training sets $\widetilde{\mathcal{Y}}$ and $\widetilde{\mathcal{X}}$. Therefore, finding the optimal value for γ requires the study of such trade offs. In Sec. 2.1 we show a toy example demonstrating the importance of γ, and in Sec. 6, we provide an empirical study of the influence of the threshold γ on the performance of the system.

Another important point in Alg. 1 is the size of the initial training set used to estimate $\boldsymbol{\theta}^{(0)}$. For supervised learning algorithms (i.e., *without* re-training), the performance of the classification in unseen data deteriorates significantly with the use of small training sets, while semi-supervised learning approaches tend to be more robust to the size of this initial training set. We compare these two learning algorithms in Sec. 6, where the supervised algorithm is referred to as 'Supervised' and the semi-supervised is denoted by 'Incremental'.

An important note about Alg. 1 is that the samples to be included in the training set are obtained by sampling a Gaussian mixture model (GMM) represented by (7), where $\mathcal{N}(\mu, \Sigma)$ is the Gaussian probability density function with mean μ and covariance Σ (we set Σ to be $10^{-3} \times \mathbf{I}$, with $\mathbf{I}$ the identity matrix). Note that the number of samples drawn from the GMM is the same as the size of the initial training set $|\{\mathcal{X}, \mathcal{Y}\}|$, and that the number of components of (7) is the number of detections in $(\widetilde{\mathbf{x}}_i, \widetilde{\mathbf{y}}_i^{(e)})$, which produced $f_i^{(t)} \geq \gamma$.

2.1. Toy Example

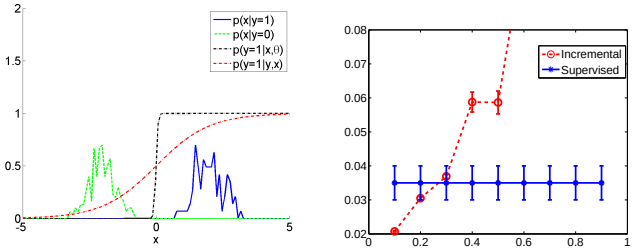
In this section we describe a toy example that compares the algorithms 'Supervised' and 'Incremental'. This example also facilitates the understanding of the role of γ in Alg. 1 (Fig. 2). The setting simulates that of the real experiment as follows: 1) the data space is described by variable $\mathbf{x}$ and $\mathbf{y} \in \{0, 1\}$ denotes the annotation, and 2) there is a hidden generative probabilistic model per class denoted by $p(\mathbf{x}|\mathbf{y}) \sim \mathcal{N}(\mu_{\mathbf{y}}, \sigma)$ with

$$\mu_{\mathbf{y}} = \begin{cases} \mathcal{N}(-2, 1), & \text{if } \mathbf{y} = 0 \\ \mathcal{N}(+2, 1), & \text{if } \mathbf{y} = 1 \end{cases}, \qquad (9)$$

and $\sigma \sim \mathcal{N}(1, 1)$ with $\mathcal{N}(\mu, \sigma)$ representing the Gaussian density function with mean μ and standard deviation σ. The goal is to learn the parameters $[a, b]$ of the logistic model $p(\mathbf{y} = 1|\mathbf{x}, [a, b]) = \frac{\exp\{a\mathbf{x}+b\}}{1+\exp\{a\mathbf{x}+b\}}$ using Alg. 1. This means that at $t = 0$, we generate annotated samples for classes $\mathbf{y} \in \{0, 1\}$ using the model in (9), and learn $[a, b]$, producing the results for the supervised algorithm. For $t > 0$, new unannotated samples for both classes are generated according to (9), which are annotated using (7). This new training set is then used for re-estimating $[a, b]$ (see Alg.1). The parameters resulting from this learning procedure are denoted by $[\hat{a}, \hat{b}]$. We also estimate the parameters for the ideal classifier by generating a training set according to the distributions

$$p(\mathbf{x}|\mathbf{y} = 0) \sim \mathcal{N}(-2, 1), \text{ and } p(\mathbf{x}|\mathbf{y} = 1) \sim \mathcal{N}(+2, 1). \qquad (10)$$

The parameters of the logistic model are then learned with maximum posterior estimation, producing a^* and b^*. Finally, we consider the external classifier $p(\widetilde{\mathbf{y}}_i^{(e)}|\widetilde{\mathbf{y}}_i, \widetilde{\mathbf{x}}_i)$ in (7) as a prior model for the posterior classifier. For this reason, we simulate this external classifier with a logistic model with fixed parameters $a_{ext} = 1$ and $b_{ext} = 0$ (red-dashed curve on left graph in Fig. 2).



a) ideal and external classifiers b) error mean and variance

Figure 2. Toy example. The left graph shows the ideal and external classifiers, and the right graph displays the error mean and variance for each algorithm at different values for γ. Note that 'Supervised' yields the same error values because it does not depend on γ. Please see text for details.

In this experiment, we consider the following sets:

- initial training sets $\mathcal{K} = \{(200 \times 200), (100 \times 100), (50 \times 50), (20 \times 20), (10 \times 10)\}$, with each element representing number of positives $\times$ number of negative samples;
- set of γ values $\mathcal{L} = \{0.1, 0.2, ..., 0.9\}$;
- set of algorithms $\mathcal{M} = \{$'Supervised', 'Incremental'$\}$.

For each element $k \in \mathcal{K}$, $l \in \mathcal{L}$ and $m \in \mathcal{M}$, we run 10 times the Alg.1. Note that when $m =$ 'Supervised', we assume $T = 0$ in Alg. 1, otherwise, $T = 10$. The error is computed as follows: $P_e(k,l,m) = \int p(\text{error}, k, l, m|\mathbf{x})p(\mathbf{x})d\mathbf{x}$ with

$$p(\text{error}, k, l, m|\mathbf{x}) = (p(\mathbf{y} = 1|\mathbf{x}, [a^*, b^*]) - p(\mathbf{y} = 1|\mathbf{x}, [\hat{a}(k,l,m), \hat{b}(k,l,m)]))^2, \quad (11)$$

where $[\hat{a}(k,l,m), \hat{b}(k,l,m)]$ are the parameters learned for each of the indices $k \in \mathcal{K}$, $l \in \mathcal{L}$ and $m \in \mathcal{M}$ above. In Fig. 2, the right graph shows the mean and variance of $P_e(k,l,m)$ over k with respect to l and m. From Fig. 2, we can see that 'Incremental' yields smaller errors than 'supervised' in terms of mean and variance of $P_e(k,l,m)$ for $\gamma < 0.3$.

3. Segmenting the Left Ventricle using Deep Learning Methods

The classifier $p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta})$ is based on deep neural networks [7], which is a type of deep learning classifier. Deep neural networks have been recently explored by Carneiro et al. [2], who showed that this classifier can achieve state-of-the-art LV segmentation results with 400 annotated training images. Moreover, contrary to boosting classifiers [9, 11, 15, 16, 17], the adaptation of deep belief networks from a batch to an on-line learning is straightforward.

The annotated training set (Fig.3) is denoted by $\mathcal{D} = \{(I, \vartheta, \mathbf{y})_i\}_{i=1..M}$, with LV images I_i, rigid transformation parameters $\vartheta_i \in \Re^5$ (position $\mathbf{p} \in \Re^2$, orientation $\xi \in [-\pi, \pi]$, and scale $\sigma \in \Re^2$), and manual annotations $\mathbf{y}_i = [\mathbf{s}_j]_{j=1..N}$ with $\mathbf{s}_j \in \Re^2$. Note that $\mathbf{x}$ is obtained with the rigid transformation ϑ applied to the image I through the function $h : I \times \vartheta \rightarrow X_i$. The rigid transformation is obtained in order to align the base and apical points of the annotation (top-right image of Fig.3) into specific values in a canonical system.

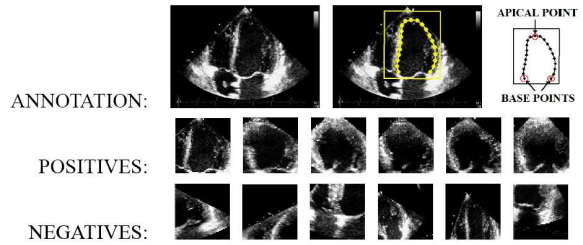


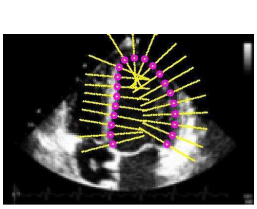
Figure 3. Original training image (top left) with the manual LV segmentation in yellow line and star markers (top middle) with the rectangular patch representing a canonical coordinate system for the segmentation markers. The top-right image shows the reference patch with the base and apical points highlighted and located at their canonical locations within the patch (these points are used to define the rigid transform of the patch). The images on the second and third rows display several positive and negative patches (respectively) used to train the rigid classifier.

The classifier $p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta})$ is composed of rigid and non-rigid detectors. The rigid detector determines the probability that $\mathbf{x}$ represents an image region containing a left ventricle aligned in the same way as the training set images (see positive patches in Fig. 3). The non-rigid detector determines the probability that the contour $\mathbf{y}$ represents an LV segmentation of $\mathbf{x}$. More specifically, we have:

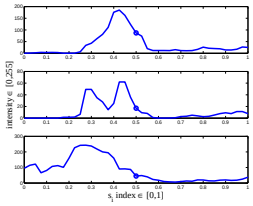
$$p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}) = \sum_{c=0,1} p(\mathbf{y}|c, \mathbf{x}, \boldsymbol{\theta}_{1,n}) p(c|\mathbf{x}_1, \boldsymbol{\theta}_{1,r}), \quad (12)$$

where $\boldsymbol{\theta} = [\boldsymbol{\theta}_n, \boldsymbol{\theta}_r]$ with $\boldsymbol{\theta}_n$ and $\boldsymbol{\theta}_r$ representing the parameters of the non-rigid and rigid classifiers [2], respectively, and c denotes a random variable, where $c = 1$ means that $\mathbf{x}$ represents an LV image region (and $c = 0$ denotes the probability that $\mathbf{x}$ does not represent an LV image region). The parameters of the rigid classifier $\boldsymbol{\theta}_r$ are the following: 1) number of hidden layers, 2) number of nodes per layer, and 3) the parameters of the logistic model of each connection between network nodes. The non-rigid classifier consists of separate a deep neural network where the parameters $\boldsymbol{\theta}_n$ comprises not only the parameters 1-3 above, but also the parameters of the shape model, which is represented by a principal component analysis (PCA) model that reduces the dimensionality of the annotation. The input for the non-rigid classifier are the profiles of perpendicular lines taken across an average LV contour traced in the image region $\mathbf{x}$, and the output is the likelihood of a specific annotation $\mathbf{y}$ indicating the LV border (see Fig. 4).

The parameters of the classifier are learned with maximum a posteriori strategy using the training procedure proposed by Hinton and colleagues [7]. The training consists of two stages: an unsupervised training where an auto-encoder



a) lines perpendicular to annotation points



(b) intensity value profiles with the dot at center representing the LV contour location

Figure 4. Intensity value profiles (from inside to outside the LV) of the lines drawn perpendicularly to annotation points. Those intensity profiles (and respective LV contour location) are used to train the non-rigid classifier. Figure from [2].

is built, and a supervised learning that produces the classifier. In the first stage, several layers of restricted Boltzmann machines (RBM) are greedily learned where the goal is to reconstruct the input data. Then, one last network layer is added, and the network weights are updated (via back-propagation) using a supervised training set, which produces a discriminative classifier.

4. External Model-based Left Ventricle Segmentation

The external LV classifier consists of a model-based method that uses a hand designed model (i.e., no model parameter is estimated from training data). Recall that this external segmentation is denoted by $p(\widetilde{\mathbf{y}}_i^{(e)}|\widetilde{\mathbf{y}}_i, \widetilde{\mathbf{x}}_i)$ used in the definition of f_i in (4). In this model, the left ventricle has a prior shape, its borders are represented by image edges, and the distribution of gray values are consistent inside and outside the chamber. In particular we implemented the Shape Probabilistic Data Association (S-PDAF) algorithm proposed by Nascimento et al. [13], which is an extension of the seminal work by Bar-Shalom [1]. The S-PDAF is used in this paper because of its fast detection, easy implementation and robustness to typical image noise present in medical images.

The algorithm (Fig. 5) has the following major steps. Given an initialization contour $\widetilde{\mathbf{y}}_i$, S-PDAF searches for several coherent hypotheses (strokes) for alternative locations of the LV contour. The search for strokes uses an energy function that is minimized based on the support from the image in terms of texture and edge location. The final LV contour $\widetilde{\mathbf{y}}_i^{(e)}$ is formed by combining a set of strokes according to the prior LV model of shape and appearance [13], where stroke overlaps are suppressed and gaps are filled.

In order to compute the probability $p(\widetilde{\mathbf{y}}_i^{(e)}|\widetilde{\mathbf{y}}_i, \widetilde{\mathbf{x}}_i)$ of the contour $\widetilde{\mathbf{y}}_i^{(e)}$ produced by S-PDAF, we need to have a way to measure whether the strokes used to produce this LV contour has good continuation, few overlaps and few gaps. The qualitative probability (QP) proposed by Jepson and Mann [10] provides a principled way of measuring the likelihood that a set of edges forms a specific shape. We adapt QP to measure the likelihood that a set of strokes represent an LV contour as follows: $Q(\widetilde{\mathbf{y}}_i^{(e)}, \widetilde{\mathbf{y}}_i, \widetilde{\mathbf{x}}_i) = \prod_{i \in strokes} \alpha^{Ls_i} \times \beta^{Lg_i} \times \nu^{Lo_i}$, where Ls_i represents the stroke length, Lg_i denotes the gap length, and Lo_i is the overlap length. The QP parameters have been arbitrarily set as follows: $\alpha = 2$, $\beta = 0.9$, $\nu = 0.2$, which means that we give positive weight to stroke length and negative weight to contour gaps and stroke overlaps (note that variations of these values produce little effect in the final segmentation result, as long as $\alpha \in [1.5, 3.0]$, $\beta \in [0.6, 0.95]$, and $\nu \in [0.1, 0.3]$) Hereafter, assume that $p(\widetilde{\mathbf{y}}_i^{(e)}|\widetilde{\mathbf{y}}_i, \widetilde{\mathbf{x}}_i) = Q(\widetilde{\mathbf{y}}_i^{(e)}, \widetilde{\mathbf{y}}_i, \widetilde{\mathbf{x}}_i)$.

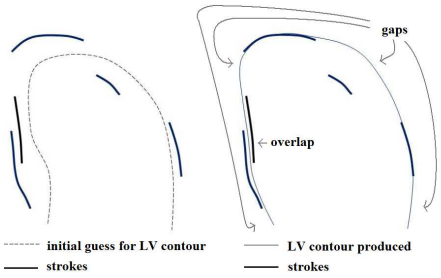


Figure 5. Contour produced by the external model-based LV segmentation (right) given the initial guess (left) and set of strokes.

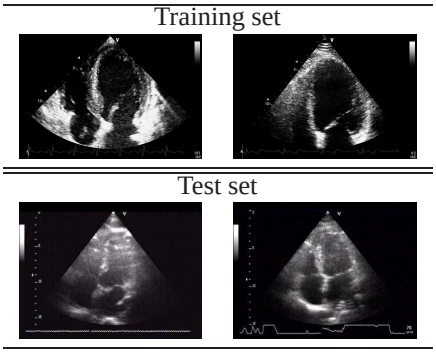


Figure 6. First images of a subset of the training and test sets.

5. Learning and Detection Details

In this section, we first introduce the training and test sets, and then we provide some details of the incremental on-line learning and detection procedures.

We use the two sets of data available from [13], which have been annotated by a cardiologist. The first set contains 400 ultrasound images of the left ventricle of the heart, which have been taken from 12 sequences (12 sequences from 12 subjects with no overlap), where each sequence contains an average of 34 annotated frames. Let us denote this set as $\mathcal{D}$. The second set, used exclusively for testing, contains two sequences of 80 images, where each sequence has 40 annotated images (two sequences from two subjects with no overlap). This set is denoted by $\mathcal{T}$ with sequences

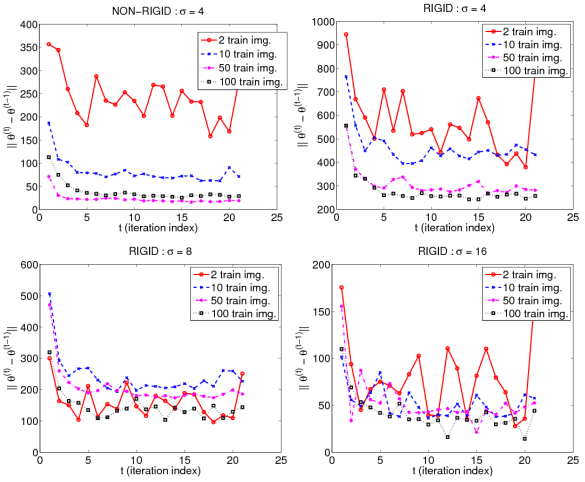


Figure 7. Convergence of the deep belief network parameters for each one of the classifiers by computing the average of the absolute difference of the weights between on-line learning iterations. The legend shows the number of images used during supervised training of the classifier (initial training set size).

A and B. Note that there is no overlap between subjects in sets $\mathcal{D}$ and $\mathcal{T}$. All quantitative comparisons of various algorithms [13] use only the two sequences in the test set $\mathcal{T}$, so we use the same sequences in order to provide a fair comparison with the other methods. The first image of two sequences from $\mathcal{D}$ and two sequences from $\mathcal{T}$ are shown in Fig. 6.

For the training procedure, recall that the parameters of the discriminative classifier $p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta})$ presented in (1) consists of the parameters $\boldsymbol{\theta}_r$ and $\boldsymbol{\theta}_n$ of the rigid classifier and non-rigid regressor, respectively. This classifier is initially trained (*supervised* training) with a subset of $\mathcal{D}$ (in this paper, we consider subsets of sizes $\{2, 6, 10, 20, 50, 100\}$ that are formed by uniformly sampling $\mathcal{D}$) to maximize $p(\mathcal{Y}|\mathcal{X}, \boldsymbol{\theta})$, which builds $\boldsymbol{\theta}^{(0)}$ in Alg. 1. Given a test sequence in $\mathcal{T}$ the classifier is iteratively trained (*incremental* training), according to the description of Alg. 1.

We follow the same multi-scale training procedure for the rigid classifier and non-rigid regressor described by Carneiro et al. [2]. The image scale space is built as follows: $L(\mathbf{p}, \sigma) = \mathcal{N}(\mathbf{p}, \sigma) * I(\mathbf{p})$, where $\mathcal{N}(\mathbf{p}, \sigma)$ is the Gaussian kernel, $*$ is the convolution operator, $I(\mathbf{p})$ is the input image, σ is the image scale parameter, and $\mathbf{p}$ is the image coordinate. The rigid classifier is trained at three scales $\sigma = \{4, 8, 16\}$ in order to form a coarse-to-fine detection approach. For the rigid classifier, we build positive and negative training sets that are defined based on a scale-dependent margin, which increases by a factor of two after each octave. The positive set is built using samples that are within the margin explained above using the manual annotations of the training set and the detections in (7) of Alg. 1, while the negative set consists of samples taken from random locations, scales and orientations that have a distance from the samples in the positive set of at least the margin explained above (Fig. 3). The non-rigid regressor is trained only at $\sigma = 4$, where the training sample consists of a line of 41 pixels extracted perpendicularly from the LV contour points (see Fig. 4) and the label to learn is the pixel index in $\{1, ..., 41\}$ that is closest to the LV contour. Fig. 7 displays the evolution of the average of $\|\boldsymbol{\theta}^{(t)} - \boldsymbol{\theta}^{(t-1)}\|$ as a function of the iteration parameter t for the rigid classifier in at scales $\sigma \in \{4, 8, 16\}$ and non-rigid regressor in at $\sigma = 4$. It is worth noticing that as the number of initial training images increases, the convergence of the incremental on-line training improves.

The detection procedure consists of running the rigid classifier at scale $\sigma = 16$ on the K_{coarse} initial hypotheses [2] (here, $K_{\text{coarse}} = 1000$), by sampling a distribution in the space of rigid transformations (the parameters of this distribution are learned from the training set). From this detection, cluster the hypotheses (using k-means algorithm) and select the top K_{fine} clusters (here, $K_{\text{fine}} = 10$) in terms of the best hypothesis within each cluster. Then run the rigid classifier at scale $\sigma = 8$ on these hypotheses and repeat the procedure for scale $\sigma = 4$. Finally, run the non-rigid classifier over the final top K_{fine} hypotheses. The final segmentation contour points are then projected to the principal component analysis (PCA) space built with the respective subset of the training set $\mathcal{D}$ [2]. The PCA space transforms the 41-dimensional vector (representing the contour) to a 5-dimensional vector, which is back projected onto the original contour space, producing a less noisy final contour. All hypotheses found are then averaged (this is an estimation of the decision function in Eq. 8 in Alg. 1) using the result of the classifier $p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta})$ as weights.

6. Experiments

In this section, we show empirical evidence of the importance of two key parameters in Alg.1, which are: 1) the confidence threshold γ, and 2) the number of images used to estimate $\boldsymbol{\theta}^{(0)}$. We also show a quantitative comparison between the algorithms 'Supervised' and 'Incremental'. Furthermore, we compare quantitatively the performance of our algorithm and of state-of-the-art LV detectors recently proposed [2, 5, 13]. The performance of the detectors is assessed by comparing the contour estimates with manual reference contours (see Sec. 5) using the error measures defined below.

Let $\mathbf{y}_1 = [\mathbf{s}_i^\top]_{i=1..N}$, and $\mathbf{y}_2 = [\mathbf{t}_i^\top]_{i=1..N}$, with $\mathbf{s}_i, \mathbf{t}_i \in \Re^2$, be two vectors of points representing the estimated and reference LV contours, respectively. The smallest distance from a point $\mathbf{s}_i$ to the curve $\mathbf{y}_2$ is $d(\mathbf{s}_i, \mathbf{y}_2) = \min_j ||\mathbf{t}_j - \mathbf{s}_i||_2$, which is known as the distance to the closest point (DCP). We use the error measures below for the experiments. The average error (AV) is defined as [13]:

$$d_{\text{AV}}(\mathbf{y}_1, \mathbf{y}_2) = \frac{1}{N} \sum_{i=1}^{N} d(\mathbf{s}_i, \mathbf{y}_2). \tag{13}$$

The Hausdorff distance (HDF) [8] is described as:

$$d_{\text{HDF}}(\mathbf{y}_1, \mathbf{y}_2) = \max\Big(\max_i\{d(\mathbf{s}_i, \mathbf{y}_2)\}, \max_j\{d(\mathbf{t}_j, \mathbf{y}_1)\}\Big). \tag{14}$$

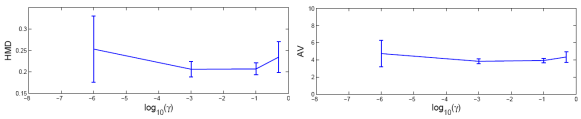


Figure 8. Mean and standard deviation of error measures (13) and (15) as a function of γ for several initial training sets of variable sizes. Other error measures have been omitted due to lack of space.

The definition of the Hammoude distance (HMD) [6] is as follows:

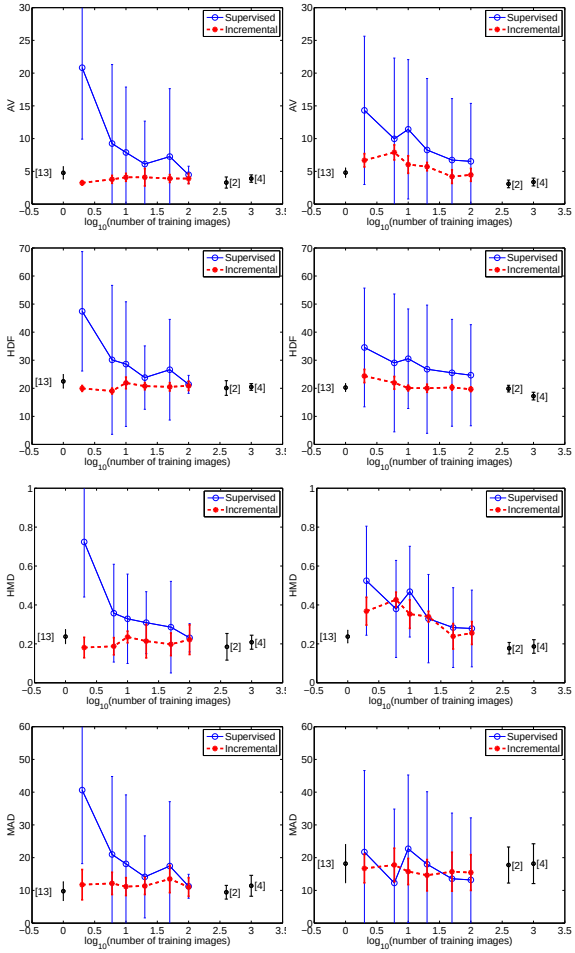
$$d_{\text{HMD}}(\mathbf{y}_1, \mathbf{y}_2) = \frac{\#((R_{\mathbf{y}_1} \cup R_{\mathbf{y}_2}) - (R_{\mathbf{y}_1} \cap R_{\mathbf{y}_2}))}{\#(R_{\mathbf{y}_1} \cup R_{\mathbf{y}_2})}, \quad (15)$$

where $R_{\mathbf{y}_1}$ represents the image region delimited by the contour $\mathbf{y}_1$ (similarly for $R_{\mathbf{y}_2}$), and $\#(.)$ denotes the number of pixels within the region described by the expression in parenthesis. Finally, the mean absolute distance (MAD) [19] is defined by:

$$d_{\text{MAD}}(\mathbf{y}_1, \mathbf{y}_2) = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{s}_i - \mathbf{t}_i\|_2. \quad (16)$$

Figure 8 uses $\mathcal{T}(A)$ (i.e., the sequence A of the test set $\mathcal{T}$) to show how the error measures (13) and (15) vary as a function of γ. The results in Fig. 8 are shown using the average and standard deviation results after running Alg. 1 with initial training sets of sizes $\{2, 6, 10, 20, 50, 100\}$ (recall that each initial training set is formed by sampling $\mathcal{D}$ uniformly). From Fig. 8, we see that 'Incremental' produces smaller and more stable results for $\gamma \in \{0.001, 0.1\}$, so we use $\gamma = 0.001$ for the experiments below.

The final experiment shows how the incremental on-line training method improves the performance of the system initially trained with small training sets (this initial system is denoted by 'Supervised'). We also compare the results with the performance of the following methods: 1) the supervised training method of Carneiro et al. [2] that uses 400 training images; 2) the supervised training approach by Georgescu et al. [5] that uses hundreds of training images; and 3) the model-based method by Nascimento et al. [13] that does not use any training set, but requires elaborate strategies for producing the initial guess for the optimization function. It is important to mention that, different from the method proposed in this paper, the two competing approaches [5, 13] also use a dynamical model of the heart motion, which is usually associated with more precise LV segmentation results. For this experiment, we build three different training sets of sizes $\{2, 6, 10, 20, 50, 100\}$ (that is, we have $6 \times 3 = 18$ distinct training sets) and show the results using mean and standard deviation for each error measure (Fig. 9). Compared to 'Supervised', note that the 'Incremental' reduces the standard deviation and mean errors for almost all error measures in both test sequences. Finally, compared to the state-of-the-art, notice that 'Incremental' produces competitive results with training sets with less than twenty images for most of error measures. Figure 10 displays four cases comparing the LV segmentations



a) seq. $\mathcal{T}(A)$ b) seq. $\mathcal{T}(B)$

Figure 9. Comparison of the performance of the proposed incremental on-line methods and the supervised approach using the error measures (13)-(16) (each row represents one error measure, and each column denotes a different test sequence). We also show the detection results on the same test sets of the supervised training methods [2] and [5] and the unsupervised model-based method [13].

produced by 'Incremental' and 'Supervised' using an initial training set containing 10 annotated images.

7. Discussion and Conclusions

In this paper, we presented a novel incremental on-line semi-supervised learning methodology applied to the fully automatic segmentation of the left ventricle of the heart from ultrasound data. The main novelty resides in the formulation of the on-line learning and segmentation algorithm that keeps adding training images and producing LV segmentation as frames of a new test sequence are presented to the system. This contrasts with the off-line learning and detection commonly found in similar semi-supervised learning approaches. This novelty restricts the set of samples that can be introduced into the training set, so the selection

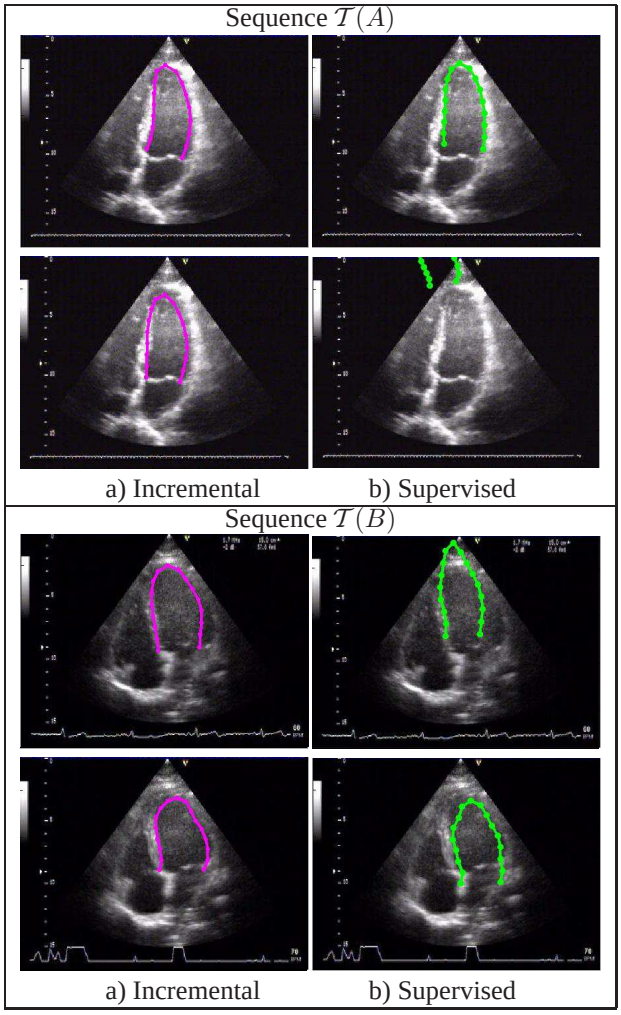


Figure 10. Examples of the detection on test sequences $\mathcal{T}(A)$ and $\mathcal{T}(B)$ produced by the 'Incremental' (first column) and the 'Supervised' (second column) models, where the initial training set contained 10 annotated images.

criterion to add unannotated images to the training set becomes a critical aspect of the algorithm, and we provide an empirical study on this issue. Another novelty lies in the use of deep neural network which is easily adapted to the semi-supervised learning framework. The results show that it is possible to have state-of-the-art results with training sets containing less than twenty annotated training images. We plan to extend this work for the LV segmentation in 3D ultrasound, and also in the joint detection of the LV endocardium and epicardium.

References

[1] Y. Bar-Shalom and T. Fortmann. Tracking and data association. *New York: Academic*, 1988. 5

[2] G. Carneiro and J. Nascimento. Multiple dynamic models for tracking the left ventricle of the heart from ultrasound data using particle filters and deep learning architectures. In *CVPR*, 2010. 1, 4, 5, 6, 7

[3] D. Comaniciu, X. Zhou, and S. Krishnan. Robust real-time myocardial border tracking for echocardiography: An information fusion approach. *IEEE Trans. Med. Imag.*, 23(7):849–860, 2004. 1

[4] H. R. et al. Clinical utility of automated assessment of left ventricular ejection fraction using artificial intelligenceassisted border detection. *American Heart Journal*, 155(3):562–570, 2008. 1

[5] B. Georgescu, X. S. Zhou, D. Comaniciu, and A. Gupta. Databased-guided segmentation of anatomical structures with complex appearance. In *Conf. Computer Vision and Pattern Rec. (CVPR)*, 2005. 1, 6, 7

[6] A. Hammoude. *Computer-assited Endocardial Border Identification from a Sequence of Two-dimensional Echocardiographic Images*. PhD thesis, University Washington, 1988. 7

[7] G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. 2, 4

[8] D. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using hausdorff distance. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(9):850–863, 1993. 6

[9] O. Javed, S. Ali, and M. Shah. Online detection and classification of moving objects using progressively improving detectors. In *CVPR*, 2005. 1, 2, 4

[10] A. Jepson and R. Mann. Qualitative probabilities for image interpretation. In *ICCV*, 1999. 5

[11] A. Levin, P. Viola, and Y. Freund. Unsupervised improvement of visual detectors using co-training. In *ICCV*, 2003. 1, 2, 4

[12] V. Nair and J. Clark. An unsupervised, online learning framework for moving object detection. In *CVPR*, 2004. 1

[13] J. C. Nascimento and J. S. Marques. Robust shape tracking with multiple models in ultrasound images. *IEEE Trans. Imag. Proc.*, 17(3):392–406, 2008. 2, 5, 6, 7

[14] R. Neal and G. Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998. 2

[15] C. Rosenberg, M. Hebert, and H. Schneiderman. Semi-supervised selftraining of object detection models. In *Seventh IEEE Workshop on Applications of Computer Vision*, 2005. 1, 2, 4

[16] P. Roth, H. Grabner, D. Skocaj, H. Bischof, and A. Leonardis. Online conservative learning for person detection. In *VS-PETS*, 2005. 1, 2, 4

[17] B. Wu and R. Nevatia. Improving part based object detection by unsupervised, online boosting. In *CVPR*, 2007. 1, 2, 4

[18] Y. Zheng, A. Barbu, B. Georgescu, M. Scheuering, and D. Comaniciu. Four-chamber heart modeling and automatic segmentation for 3-d cardiac ct volumes using marginal space learning and steerable features. *IEEE Trans. Med. Imaging*, 27(11):1668–1681, 2008. 1

[19] X. S. Zhou, D. Comaniciu, and A. Gupta. An information fusion framework for robust shape tracking. *IEEE Trans. Pattern Anal. Machine Intell.*, 27(1):115–129, 2005. 7

[20] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005. 1, 3