Automatic View Recognition for Cardiac Ultrasound Images

Matthew Eric Otey[1], Jinbo Bi[2], Sriram Krishnan[2], Bharat Rao[2], Jonathan Stoeckel[2], Alan Katz[3], Jing Han[3], and Srinivasan Parthasarathy[1]

[1] Department of Computer Science and Engineering, The Ohio State University
[2] Computer-Aided Diagnosis and Therapy Group, Siemens Medical Solutions USA, Inc.jinbo.bi@siemens.com
[3] St. Francis Hospital, Roslyn, New York

Abstract. Ultrasound images of the heart can be taken from many different angles. Diagnostic analysis of these images requires recognizing the pose of the heart so that important cardiac structures can be identified. We are concerned with the problem of automatically classifying cardiac ultrasound images with respect to what view of the heart they contain. Our solution to this problem has two novel aspects: first, a hierarchical classifier is constructed to classify an unknown view into corresponding windows, and then further classify it into one of the respective subclasses in the window; second, a simple dimension reduction approach is used to compress the information of many image features and to enhance the classification accuracy. Experiments on recognizing apical two-chamber, apical four-chamber, parasternal long axis and parasternal short axis views demonstrate the effectiveness of our approach.

1 Introduction

Many researchers have investigated feature extraction from and classification of ultrasound images. Some work has concentrated on feature extraction only [1]. Other work has involved feature extraction for the purpose of classification of various tissue types. For example, there has been work on classifying breast lesions as being benign or malignant [2]. Other work has been done on classifying ultrasound images of liver tissue in order to distinguish between healthy and diseased tissue [3], detecting liver cirrhosis [4], and differentiating between benign and malignant liver tumors [5].

There exist fifteen basic views of heart in ultrasound images. The problem of automatically distinguishing between such a relatively large number of views simultaneously is very difficult. However, these fifteen views are imaged from four windows: the parasternal, apical, subcostal, and suprasternal windows. We have discovered that there is greater similarity between views in the same window than between views in different windows. This observation leads us to develop a hierarchical classification scheme that first distinguishes between the different windows, and then between the views belonging to a specific window. In this paper, as our data sets did not contain examples from all views, we present our preliminary work on automatically distinguishing between two views in each of two windows. These views contain the apical two chamber (a2c), the apical four chamber (a4c), the parasternal long axis (plax), and the parasternal short axis (psax) views. Example images of each of these four views can be seen in Figure 1.

Previous work closely related to our study include techniques for differentiating between different ultrasound views of an organ, such as the work done by Ebadollahi *et al.* [6] for automatic indexing of ultrasound clips according to their view of the heart. They derive their features using using the Gray-Level Symmetric Axis Transform to detect the chambers of the heart, and Markov Random Fields to model the

constellation of the chambers. They achieve accuracies of up to 88.35% using a support vector machine classifier and leave-one-out cross validation. However, they make the assumption that clinically similar views (views that are not distinguishable by human experts and for clinical purposes are considered identical) are considered to be the same view. Also, they only use frames from the ultrasound clips which have the correct number of distinct chambers for each view and no false or missing chambers as their training/testing data set. When they remove these assumptions, their accuracy falls to as little as 34%, and to 52% when clinically similar views are considered the same. In our approach, we use simpler features and classifiers, and use the entire ultrasound clip instead of just the "best frames." Zhou *et al.* [7] present an approach for view recognition of cardiac ultrasound images. However, they only differentiate between the a2c and a4c views, and they achieve an accuracy of up to 90% for this problem. Also, their approach requires a pre-processing phase in which the left ventricle is identified by a human. With our approach, we can achieve accuracies of up to 87.9% using leave-one-out cross-validation on our training data set and up to 92.7% on our test set, as detailed in Section 5, using a much more automatic pre-processing phase, as described in Section 2 below.

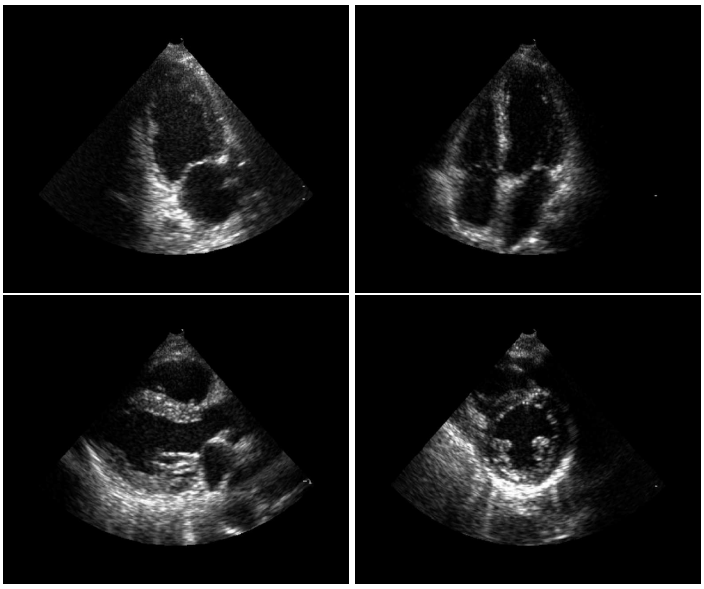


Fig. 1. Four different ultrasound views of the heart. Clockwise from upper left: apical two chamber view, apical four chamber view, parasternal long axis view, and parasternal short axis view.

2 Data Description and Pre-processing

The ultrasound images that we are concerned with come in the form of video clips. The number of frames can vary from clip to clip. The clips have 3 color channels (RGB). Each frame of the clip contains the actual ultrasound image as well as some diagnostic information. The area of a frame displaying the actual ultrasound image is referred to as the fan area due to its shape (see Figure 1). The actual size of this fan area in a given ultrasound clip depends on the machine and its settings. In this work, we assume that all clips have the same resolution, though clips with different resolutions can be resampled to a common resolution.

The first stage of our approach is to detect, in each clip, the fan area, from which image features will be extracted. Detection of the fan area is itself not trivial, as the size of the fan area varies from clip to clip. Although a fan mask can be detected from each

clip, some features are evaluated assuming the fan area is uniform across all clips. We therefore construct a universal fan mask by finding the union of the individual masks identified from each clip in our training set: the masks are overlaid on each other, and any pixel that is marked as "on" in a given percentage of the individual masks will be marked as "on" in the universal mask. Moreover, ultrasound images have a wide range of intensity values dependent on the machine setting, so an important preprocessing step is to normalize the pixel intensities of these images. We first convert each of the clips to a single intensity matrix by averaging over the RGB color channels. Then a standard linear normalization is carried out by dividing the pixel intensities by the interquartile range of the pixel intensity distribution. These interquartile values are chosen to reduce impact of extreme outliers or noise.

3 Feature Extraction

Gradient features In ultrasound images, the walls (and corresponding chambers) are usually quite distinct from the noise in the image. One simple way of measuring the orientation of the walls (and therefore the chambers) is to find the sum of the magnitudes of the gradients in each of the x, y, and z directions. The magnitude of the gradients measure the vertical and horizontal structure in the clips (x and y gradients), and the motion in the clips (z gradients).

The four cardiac views of our concern show different physical structures. For example, the apical classes have a lot of vertical structure, the plax class has a lot of horizontal structure, and the psax class has a circular structure, which will lead to different values for the gradient magnitude in x and y directions. There is a distinct separation between the a4c and plax views with respect to the values of the x and y gradients, as can be seen in the scatter plot in Figure 2.

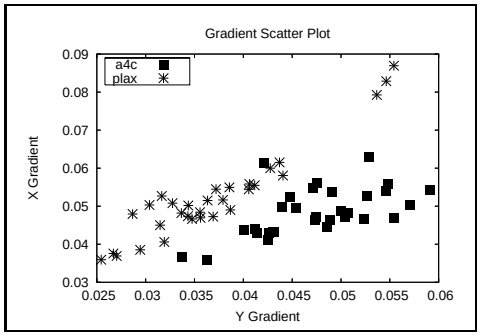


Fig. 2. A scatter plot of the x and y gradient features for the a4c and plax classes.

Another set of features are also derived from gradients. The XZ and YZ sum-gradients are computed by first finding the z-gradients in the volume and then summing across all frames to get a two-dimensional image. From this image we find the x and y gradients in the manner described above. The "real" sum-gradients in the x, y, and z directions are computed in a similar manner as the basic sum-gradients, but take the sign of the gradients into account. The sum-gradient combinations (x+y, x+z, y+z) are computed by just adding the respective basic gradient features together. The sum-gradient ratios (x:y, x:z, y:z) are computed by just dividing one sum-gradient feature by another. Finally, we use the standard deviation of the x and y sum-gradients across all frames, and the standard deviation of the magnitude of the z gradient for each voxel in the fan volume.

Peak Features The peak features estimate the number of horizontal and vertical edges or walls in the images. We use this feature to try to discriminate between the a2c and a4c classes, since the a2c images have only two walls while the a4c images have 3

walls. We extract the peak features by viewing the image as a matrix and finding the sum of each row (column). This value is then normalized by the number of pixels in each row that are in the mask area. The resulting vectors are then smoothed to remove noise. The resulting feature is then the number of maxima in the vector.

Other Statistical Features We also derive several statistics in an attempt to characterize the distribution of pixel intensities in an ultrasound clip. To derive these features we first average across all frames in a clip. We then extract the mean, standard deviation, and the second through fourth statistical moments of the pixel intensities in the average frame. Since images of different views different numbers of walls and chambers, each view presents a different distribution of intensities.

Raw Pixel Intensity Features Another set of features we consider are the values of raw pixel intensities after doing normalization and taking the average across all frames in the clip. There is a constant number of features (pixels) for each clip because we use a universal mask, even though this may cover more or less area than the actual fan area. We perform resampling to reduce the image size, smoothing the image in the process to make the feature amount scalable. We use Naïve Bayes classifiers to determine classification accuracy for various values of the smoothing and resampling parameters. We find that we achieve maximum accuracy with a height h of 16 and 24 pixels in the resampled image and a standard deviation of 0.25 for the Gaussian kernel. One drawback of using these features is that they are not translation invariant; structures may appear at different places in different images. Hence, it may not be very meaningful to compare corresponding attributes between two images as a means of classification. Another drawback is that, even with resampling, there is still a large number of features, which can degrade both the speed and quality of the classifier. These drawbacks make the features difficult to be efficiently used in the classification (or recognition) task. We hence design the "principal features" to compress information from the raw intensity features, which are described in Section 4.

4 Principal Feature Integration

The raw pixel intensity features generate an input space of very high dimensionality. Even if there exists information useful for the separation of views, it is vague and hidden in this high dimensional space. Hence we design features that aim to project high dimensional data into a lower-dimensional space where the scatter of the data is reshaped to enhance class separability. We call these features "principal features."

These new features are generated using the output of a basic classifier, which we call "mean model classifier". Consider a data set of n-dimensional feature vectors belonging to c classes, respectively. The model M_i for the ith class is the mean of all feature vectors belonging to the ith class. Now for a given feature vector u, we want to approximate it by a convex combination of the M models. In other words,

$$u = \alpha_1 M_1 + \alpha_2 M_2 + ... + \alpha_c M_c \tag{1}$$

or in the matrix format,

$$M\alpha = u \tag{2}$$

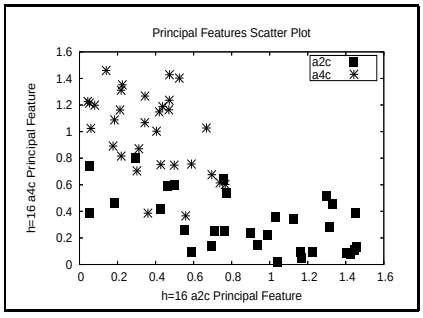
where M is a n-by-c matrix with the ith column equal to M_i. We further require that

$$\Sigma_i \alpha_i = 1 \tag{3}$$

and

$$\alpha_i \geq 0, \ \ \forall i = 1, \cdots, n. \tag{4}$$

We solve a least squares problem for (1) to obtain α. The mean model classifier then classifies an unknown feature vector u according to the index of the largest component of α.



Procedure: Generate Principal Features
T = Training data
T_α = {}
for all points $u \in T$
  Construct M from $T - \{u\}$
  Solve $M\alpha = u$ for α
  $T_\alpha = T_\alpha \cup \{\alpha\}$
end for
return T_α

Fig. 3. (a) A scatter plot of the a2c and a4c principal features derived from the $r = 16$ raw pixel intensities; (b) Algorithm for generating principal features.

Although this simple classifier itself may not produce good classification accuracy, the coefficient vector α, in a c-dimensional space, becomes very promising features to discriminate views. Geometrically, If the mean model classifier works reasonably well, there should be good separation between the classes in the α space, as the points will cluster around the axes of their respective classes. In other words, if u belongs to class i, we expect its projection to the ith model, α_i, to be larger than any other coefficients. These α features are referred to as "principal features". In Figure 3(a) we plot the a2c and a4c classes in the plane formed by the a2c and a4c principal features derived from the raw pixel intensities for $h = 16$. Note that we have added a small amount of jitter in order to minimize the point overlap, since many points in the same class tend to have very similar values for their principal features. As expected, the points cluster around the axes corresponding to their respective classes.

The principal features can then be used in place of or in addition to the original features to form another vector that is fed into another classifier. This process helps to enhance the final classification, since we are using the output of one classifier as the input to another in order to increase the accuracy. We derive the principal features for the training data using the leave-one-out approach presented in Figure 3(b). Later, we can derive the principal features for each sample in the testing data set by utilizing mean models M calculated from the training data set. We generate principal features using several different feature subsets, such as the raw pixel intensities for $h = 16$, those for $h = 24$, and the x, y, and z sum-gradients, among others.

A benefit of using principal features is that they allow us to compress a large number of regular features into only four features. For example, there are 124 raw pixel intensity features for $h = 16$, and 282 features for $h = 24$. Reducing these two feature sets down to 4 principal features each represents a fifty-fold reduction in the number of features. Also, by using the principal features we are implicitly doing two stages of classification: in the first we generate the principal features using the mean model classifier, and in the second we use the principal features in conjunction with other features to classify the ultrasound images. If the mean model classifier works well, then these principal features will provide good (or better) separation between classes for the second classifier compared to the original features. In some cases, the principal features may not provide better separation between the classes than the original features. However, the principal features may still increase the classification accuracy when used in conjunction with the original features. For example, if we run a Naïve Bayes classifier on just the the x, y, and z gradient features, we attain an accuracy of 46.8%, but when we run it on the principal features derived from these gradient features, the accuracy is only 42.7%. However, when we perform classification using both the gradient features and the principal features, the accuracy increases to 54.8%.

5 Classification Approach

Data Sets The ultrasound video clips we use in our classification experiments were collected from Siemens ACUSON^{TM}ultrasound systems at St. Francis Hospital in Roslyn, New York. Our complete data set from which we draw our training and test sets contains clips from 23 different patients, but the training and test set are disjoint. Each patient has a different number of clips for each view. Our training data set contains 31 ultrasound clips for each of the four views (a2c, a4c, plax, and psax), with some patients contributing two or more clips. The test set contains 14 clips each for the a2c, a4c, and psax classes, and 13 clips for the plax class. All clips have a resolution of 640×480, and have a varying number of frames. The pre-processing techniques described in Section 2 and the feature extraction approaches described in Section 3 are implemented in Matlab using the Image Processing toolbox and the Optimization toolbox (for calculating the principal features). We utilize the implementation of several classifiers available in Weka [8].

Hierarchical Classifier The Weka software package provides implementations of many different classification algorithms. We ran experiments using several different classification approaches with leave-one-out cross-validation on the training data set including naïve Bayes, support vector machine [9, 10] and logistic model trees (LMT) methods. We find that Logistic Model Trees (LMT) classifiers [11] perform consistently well as shown in Table 1(a), where we show the confusion matrix of the LMT classifier on the four-view recognition problem. These are the leave-one-out cross-validation results on the training data.

LMT constructs a tree-structured classifier with logistic regression functions at the leaves. The classic logistic regression approach models $log(p/(1-p)$ as a linear function of the features where p represents the probability of a feature vector x belonging to class i. It can be written as

$$log(p/(1-p) = \beta_0 + \beta^T x$$

where the β vector and the scalar β_0 are parameters to be determined and x denotes the feature vector for each clip. Consequently, the probability of x is

$$p(x) = \frac{\exp(\beta_0 + \beta^T x)}{(1 + \exp(\beta_0 + \beta^T x))}. \quad (5)$$

LMT is motivated by the principle of "divide and conquer". That is, a complex set of data is divided into sufficiently many subsets such that a simple linear logistic regression model adequately fits the data in each subset.

From Table 1(a), we achieve an accuracy of 83.1%. On the testing data, we achieve an accuracy of 89.1%. The corresponding confusion matrix can be seen in Table 1(b). These results, while not poor, still leave room for improvement. We observe that apical views tend to be confused more often with each other than with the parasternal views, and vice versa. From Table 1(a) one can see that 8 clips containing apical views are classified as parasternal views, and 2 clips containing parasternal views are classified as apical views. Similarly, in Table 1(b), five of the clips containing parasternal views are classified as containing apical views. In Figure 4 we can see that for the x and y sum-gradients, there is very good separation between the apical and parasternal *superclasses*, more so than between all four subclasses. This increased distinction between the superclasses extends to the other attributes as well.

We next use Weka to search for classifiers that perform well on the two-class problem for our training data set. Many of the classifiers we try give accuracies in the 90% range. Again, the best classifier we find is an LMT classifier which gives us 95.2%

(a) Training Data				
	a2c	a4c	plax	psax
a2c	23	2	1	5
a4c	2	27	1	1
plax	0	0	29	2
psax	2	0	5	24

(b) Testing Data				
	a2c	a4c	plax	psax
a2c	14	0	0	0
a4c	0	14	0	0
plax	0	0	12	1
psax	5	0	0	9

Table 1. Confusion matrix for a LMT classifier for the four-class problem run on (a) the training data with leave-one-out cross-validation; (b) the testing data.

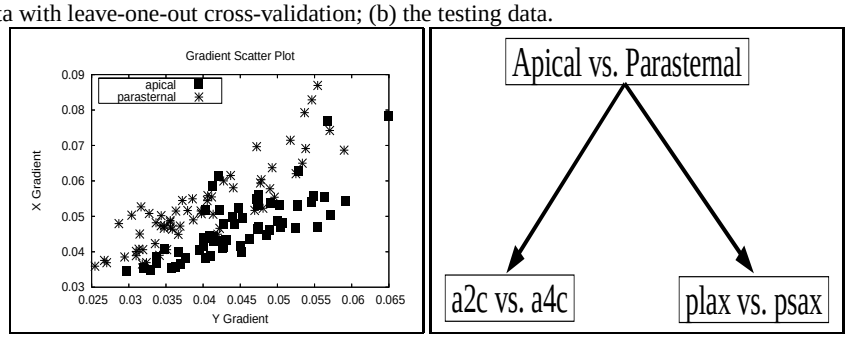


Fig. 4. (a) A scatter plot of the x and y gradient features showing separation between the apical and parasternal superclasses; (b) diagram of the hierarchical classifier.

accuracy. In this case, we find that on the training data set, using leave-one-out cross-validation, only 5 clips containing apical views are classified as having a parasternal view (compared to 8 clips in Table 1(a)), and only 1 clip containing a parasternal view is classified as having an apical view (compared to 2 clips in Table 1(a)).

This observation leads us to develop a classifier strategy that exploits the behavior we note above, namely that the misclassifications tend to be within the apical and parasternal classes, not across them. Hierarchical classification techniques have been used before. Marsolo *et al.* [12] use hierarchical multi-level classification to classify proteins based on their structure. Proteins are first classified according to their Structural Classification of Proteins (SCOP) database class. At the next level, a classifier is used to distinguish between the folds in the corresponding SCOP class.

Our hierarchical approach first tries to classify a feature vector into either the apical or parasternal class. Then it attempts to further classify the vector into the respective subclasses. Hence, there are three total classifiers: one at the top level, and one each for the apical and parasternal branches. A diagram of this classifier can be seen in Figure 4(b). The same feature vector are used at both levels of classification. At the top level, the classifier is trained only to distinguish between the apical and parasternal superclasses. On the left-hand branch, the second-level classifier is trained only to distinguish between the apical two chamber and apical four chamber views, and similarly, on the right-hand branch, the classifier is trained only to distinguish between the parasternal long axis and parasternal short axis views.

(a) Training Data				
	a2c	a4c	plax	psax
a2c	26	1	2	2
a4c	1	29	1	0
plax	0	0	27	4
psax	1	0	3	27

(b) Testing Data				
	a2c	a4c	plax	psax
a2c	14	0	0	0
a4c	0	14	0	0
plax	0	0	12	1
psax	2	0	2	10

Table 2. Confusion matrix for a hierarchical LMT classifier for the four-class problem run on (a) the training data using leave-one-out cross-validation; (b) the testing data.

When we apply this hierarchical classifier to our training data set, our classification accuracy improves. In our implementation, we use LMT classifiers at both levels

(superclass and subclass), our accuracy improves to 87.9%, as can be seen in the confusion matrix in Table 2(a). When we apply the hierarchical classifier to the testing data set, we achieve an accuracy of 90.9%, as can be seen in Table 2(b).

6 Analysis and Conclusion

From our experimental results, we see that the use of a hierarchical classification scheme reduces the number of misclassifications among the superclasses. While we only concentrated on two subclasses (views) of two different superclasses (windows) in this paper, our approach is easily applicable to the complete hierarchy of fifteen views belonging to four different windows.

Furthermore, our hierarchical classification scheme is flexible enough to allow the use of any classifier at any node in the hierarchy. Indeed, when we use a Support Vector Machine classifier at the root node of the hierarchy and LMT classifiers on the leaves, we achieve a slightly higher accuracy of 92.7% on the testing data set.

References

1. Revell, J., Mirmehdi, M., McNally, D.: Applied review of ultrasound image feature extraction methods. In Houston, A., Zwiggelaar, R., eds.: The 6th Medical Image Understanding and Analysis Conference, BMVA Press (2002) 173–176
2. Chang, R.F., Chen, C.J., Ho, M.F., Chen, D.R., Moon, W.K.: Breast ultrasound image classification using fractal analysis. In: IEEE Symposium on Bioinformatics and Bioengineering. (2004)
3. Loew, M.H., Mia, R., Guo, Z.: An approach to image classification in ultrasound. In: Applied Imagery Recognition Workshop. (2000) 193–199
4. Mojsilovic, A., Popovic, M., Sevic, D.: Classification of the ultrasound liver images with the 2nx1-d wavelet transform. In: Proceedings of the International Conference on Image Processing. (1996) 367–370
5. Yoshida, H., Casalino, D.D., Keserci, B., Coskun, A., Ozturk, O., Savranlar, A.: Wavelet-packet-based texture analysis for differentiation between benign and malignant liver tumours in ultrasound images. Physics in Medicine and Biology **48**(22) (2003) 3735–3753
6. Ebadollahi, S., Chang, S.F., Wu, H.: Automatic view recognition in echocardiogram videos using parts-based representation. In: CVPR (2). (2004) 2–9
7. Zhou, S.K., Park, J.H., Georgescu, B., Simopoulos, C., Otsuki, J., Comaniciu, D.: Image-based multiclass boosting and echocardiographic view classification. In: IEEE Conference on Computer Vision and Pattern Recognition. (2006)
8. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. Second edn. Morgan Kaufmann (2005)
9. Platt, J.C.: Fast training of support vector machines using sequential minimal optimization. (1999) 185–208
10. Keerthi, S.S., Shevade, S.K., Bhattacharyya, C., Murthy, K.R.K.: Improvements to Platt's SMO Algorithm for SVM Classifier Design. Neural Computation **13**(3) (2001) 637–649
11. Landwehr, N., Hall, M., Frank, E.: Logistic model trees. In: ECML. (2003) 241–252
12. Marsolo, K., Parthasarathy, S., Ding, C.: A multi-level approach to scop fold recognition. In: Proceedings of the IEEE Symposium on Bioinformatics and Bioengineering. (2005)