A Dissertation

entitled

New Synthesis of Bayesian Network Classifiers and Cardiac SPECT Image Interpretation

by

Jarosław P. Sacha

Submitted as a partial fulfillment of the requirements for

the Doctor of Philosophy degree in

Engineering Science

_____

Advisor: **Dr. Krzysztof J. Cios**

_____

Graduate School

The University of Toledo

December 1999

An abstract of

New Synthesis of Bayesian Network Classifiers and

Cardiac SPECT Image Interpretation

Jarosław P. Sacha

Submitted as a partial fulfillment of the requirements for

the Doctor of Philosophy degree in

Engineering Science

The University of Toledo

December 1999

A new family of Bayesian network classifiers is introduced and demonstrated to outperform existing classifiers. Of particular interest is use of these classifiers for interpretation of cardiac SPECT images. High classification performance on databases from a variety of other domains is also demonstrated.

Cardiac SPECT (Single Photon Emission Computed Tomography) is a diagnostic technique used by physicians for assessing the perfusion of the heart's left ventricle. A physician reaches the diagnosis by comparing SPECT images taken from a patient at rest and at maximum stress. Interpretation of images by strictly visual techniques is burdened with error and inconsistency. Thus, assistance in quantifying and automating the diagnosis is sought.

An important issue in automating the diagnosis is classification of left ventricle perfusion into a number of predetermined categories. The goal of this dissertation is to investi-

gate the use of Bayesian methods for construction of classifiers that would assist in interpretation of cardiac SPECT images. These images and their descriptions are characterized by a significant amount of uncertainty. Bayesian methods build models by approximating the probability distribution of the variables in the problem domain; they are naturally well suited to deal with uncertainty.

This research consisted of three main parts. (1) Data warehousing – assembling cardiac SPECT images and patient records into an easily accessible database and creating software manipulation of SPECT images. (2) Three-dimensional image processing – implementation of custom algorithms for extraction of features from SPECT images. (3) Learning Bayesian network classifiers – research of novel machine learning algorithms that use Bayesian techniques for creation of robust classifiers.

The main contribution of this work is creation of a new family of Bayesian network classifier – their high performance classifying left ventricular perfusion is demonstrated. Additionally, it is shown that they outperform existing Bayesian network classifiers and machine learning algorithm C4.5 using data from University of California at Irvine Repository of Machine Learning Databases. Among other contributions is a method for automated extraction of features from cardiac SPECT images based on the creation of models of normal left ventricles, software for visualization of cardiac SPECT images, automated feature extraction, and creation of Bayesian network classifiers.

To my parents and my grandparents

Acknowledgements

I wish to express my foremost gratitude to my wife Dilchoda for her never ending encouragement and tolerance of me spending long hours working on this dissertation.

I would like to thank my advisor Dr. Krzysztof Cios; my coadvisors: Dr. Lucy Goodenday, Dr. John Miller, Dr. Gursen Serpen, and Dr. Subhash Kwatra; and the advisors with whom I originally started my Ph.D. program in Poland: Prof. Henryk Górecki, Dr. Maciej Szymkat, and Dr. Tomasz Szmuc.

Special thanks to J. Matthew Short for the final proof-reading of this manuscript.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Cardiac SPECT (Single Photon Emission Computed Tomography) is a diagnostic technique used by physicians for assessing the perfusion of the heart's left ventricle. A physician reaches the diagnosis by comparing SPECT images taken from a patient at rest and at maximum stress. It has been shown that interpretation of images by strictly visual techniques is burdened with error and inconsistency. For that reason, assistance in quantifying and automating the diagnosis has been sought. One of the issues in automating the diagnosis is classification of left ventricle perfusion into a number of predetermined categories. The goal of this dissertation is to investigate the use of Bayesian methods for construction of classifiers that would assist in interpretation of cardiac SPECT images. These images, and their descriptions, are characterized by a significant amount of uncertainty. Bayesian methods build models by approximating the probability distribution of the variables in the problem domain; they are naturally well suited to deal with uncertainty.

This dissertation research consists of three main parts:

Data warehousing – assembling cardiac SPECT images and related patient records into an easily accessible database and creating software for reading the SPECT images stored in a proprietary format. Cardiac SPECT images have been collected from Medical College of Ohio and organized together with the relevant patient's record

into a rational database. The objective was to enable easy access to data through use of SQL queries. Collected data were cleaned and preprocessed for further use by image processing and classification.

Three-dimensional image processing – design and implementation of algorithms for extraction of features from SPECT images. The SPECT images cannot be directly used for classification. A typical SPECT image consists of about 133,072 voxels, each having up to 65,536 possible gray level values. Three-dimensional image processing is used to analyze information present in a SPECT image and express it by a small number of features representing information most relevant to the diagnosis of left ventricle perfusion. A model of a normal left ventricle has been created and used for creation of features extraction algorithms.

Learning Bayesian network classifiers – research of novel machine learning algorithms that use Bayesian techniques for creation of robust classifiers. A Bayesian network is a formalism for representing a joint distribution of a set of random variables. A Bayesian network can be used for classification by identifying one of the nodes with the class variable and other nodes with attribute variables. Classification is performed by computing marginal probability distribution of the class variable. Established methods for learning Bayesian networks are concerned with good approximation of the joint probability distribution. Good criteria for constructing a network that accurately represents the probability distribution of the analyzed problem will not necessarily lead to a good classifier. Our research was concerned with the creation of Bayesian network learning methods that are specifically designed for the creation of classifiers. A new family of Bayesian network classifier has been created and used for classification of left ventricular perfusion.

1.1 Contributions

The following are the main contributions of this dissertation:

New family of Bayesian network classifiers A new approach to synthesis of Bayesian network search algorithms specifically designed for creation of classifiers is introduced. We present five new search algorithms created using the new synthesis. We also show that naïve Bayes (Duda and Hart, 1973) and TAN classifiers (Friedman, Geiger, and Goldszmidt, 1997) are special cases of the synthesis. We demonstrate that classifiers based on the new search algorithms outperform existing ones including tree induction algorithm C4.5 (Quinlan, 1993).

Automated extraction of features from cardiac SPECT images A rigid model of a normal left ventricle is created and used for automated registration of SPECT images. Features are extracted from SPECT images mimicking process performed by a physician.

Cardiac SPECT database 4,828 patient records, 8,142 SPECT images and related files, collected for 728 patients, were organized into a coherent database.

Software New software for browsing the cardiac SPECT database, visualization of 3D SPECT images, automated feature extraction, and learning with the new family of Bayesian network classifiers has been created.

1.2 Organization

Chapter 2 briefly describes the function of the human heart, most common heart diseases and a technique for diagnosing left ventricular perfusion – cardiac SPECT imaging. The knowledge discovery process and the creation of the SPECT image database is the content of Chapter 3. Preparation of data for classification – creation of models of normal left

ventricle and the extraction of features from cardiac SPECT images is presented in Chapter 4. Chapter 5 contains an introduction to Bayesian networks. Issues specific to Bayesian network classifiers are introduced in Chapter 6. The main contribution of this work – a new family of Bayesian network classifiers is presented in Chapter 7. Experimental results demonstrating classification of the left ventricular perfusion and benchmarking of the new classifiers against existing ones are described in Chapter 8. Software developed to make this dissertation research possible is described in Appendices A and B.

Chapter 2

Cardiac SPECT Imaging

2.1 Human Heart

The heart is one of the most important organs in the human body, a central part of the circulatory system. The heart is a dual pump circulating blood through two separate systems, each consisting of an *atrium* and a *ventricle* (Fig. 2.1). Blood from the body returns to the right atrium through two large veins, the *superior* and *inferior venae cavae*; in addition the blood that has supplied the heart muscle is drained directly into the right atrium through the coronary sinus. Return of venous blood to the right atrium takes place during the entire heart cycle of contraction and relaxation, and to the right ventricle only during the relaxation part of the cycle, called *diastole*. When both right heart cavities constitute a common chamber; near the end of diastole, contraction of the right atrium completes the filling of the right ventricle with blood (Fig. 2.2). Rhythmic contractions of the right ventricle expel the blood through the pulmonary arteries into the capillaries of the lung where the blood receives oxygen. The lung capillaries then empty into the pulmonary veins, which in turn, empty into the left atrium. Pulmonary venous return to the left atrium and left ventricle proceeds simultaneously in the same manner as the venous return to the right heart cavities. Contraction of the left ventricle rhythmically propels the blood into the aorta and from

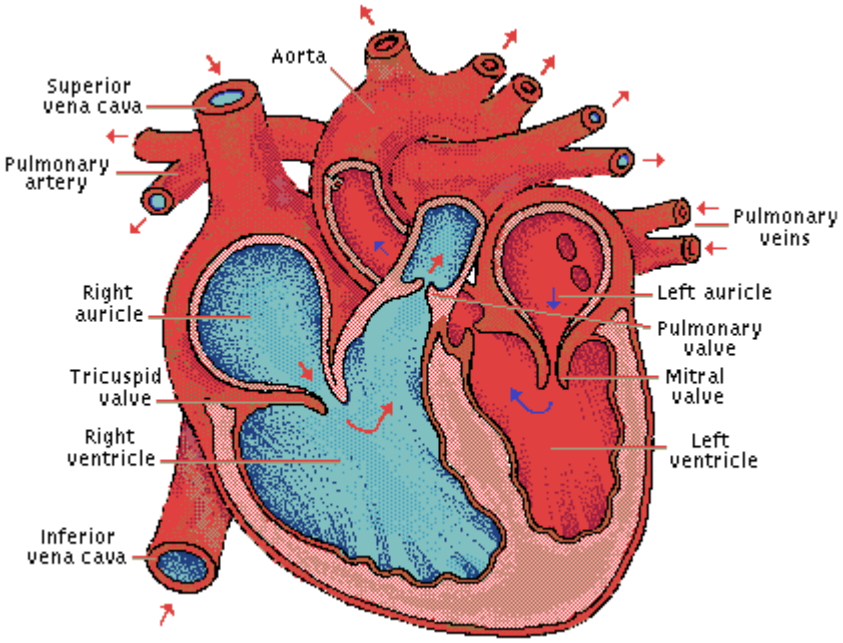
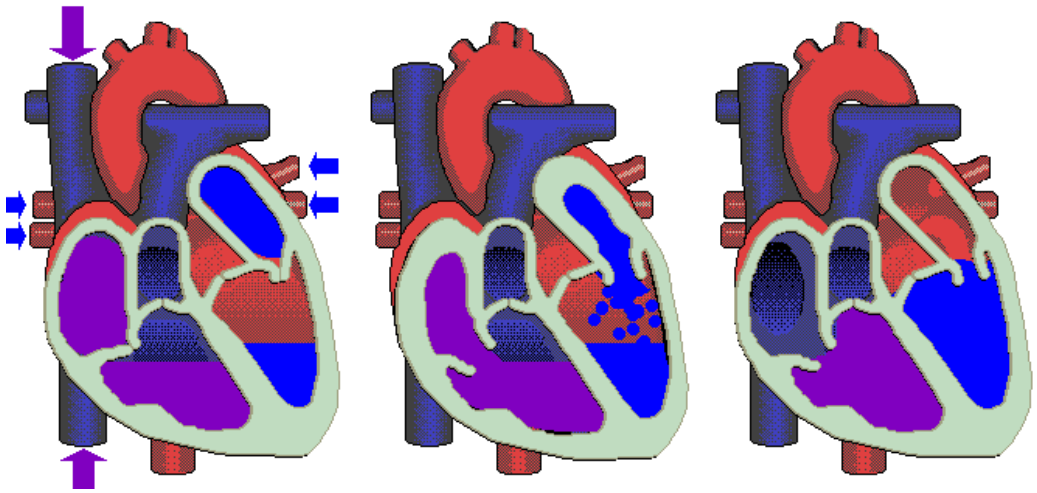


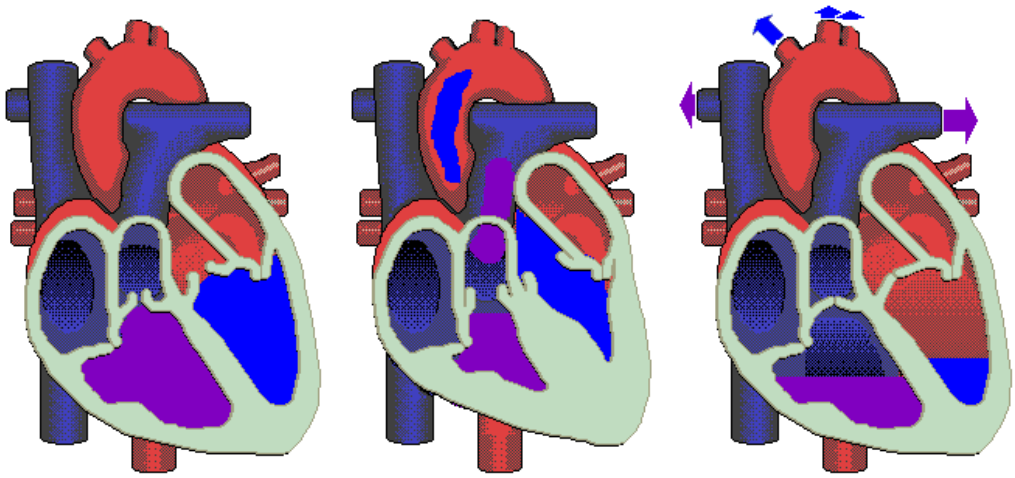
Figure 2.1: Human heart (Encarta, 1999).

there to all arteries of the body, including the coronary arteries which supply the heart muscle. The blood forced from the ventricles during *systole*, or contraction, is prevented from returning during diastole by valves at the openings of the aortic and pulmonary arteries.

Disorders of the heart kill more Americans than any other disease. They can arise from congenital defects, infection, narrowing of the coronary arteries, high blood pressure, or disturbances of heart rhythm. The major form of heart disease in Western countries is *atherosclerosis*. In this condition, fatty deposits called *plaque*, composed of cholesterol and fats, build up on the inner wall of the coronary arteries. Gradual narrowing of the arteries throughout life restricts the blood flow to the heart muscles. Symptoms of this restricted blood flow can include shortness of breath, especially during exercise, and a tightening pain in the chest called *angina pectoris*. The plaque may become large enough to completely obstruct the coronary artery, causing a sudden decrease in oxygen supply to the heart. Obstruction can also occur when part of the plaque breaks away and lodges farther along in the artery. These events are the major causes of *heart attack*, or *myocardial infarction*, which



(a) Relaxation – diastole.



(b) Contraction – systole.

Figure 2.2: Heart's cycle (Encarta, 1999).

is often fatal. Persons who survive a heart attack must undergo extensive rehabilitation and risk a recurrence. Many persons having severe angina because of atherosclerotic disease can be treated with drugs, that enable the heart to work more efficiently. Those who do not obtain relief with pharmacologic means can often be treated by surgery.

The main interest of this narrative is automation of a technique for delineation of areas of reduced blood flow in the heart called *cardiac SPECT imaging*. This technique visualizes the flow of a radioactive isotope of the element *thallium* into heart muscle. A computerized

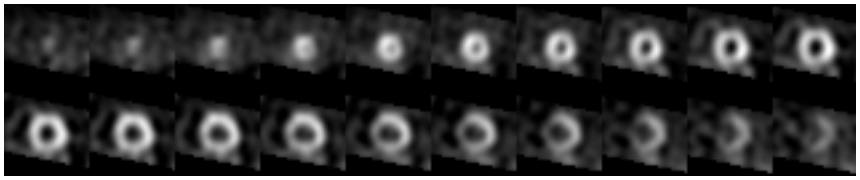
camera records the extent of thallium penetration during the systole-diastole cycle of the heart, showing areas of reduced blood perfusion and tissue damage.

2.2 Cardiac SPECT Imaging

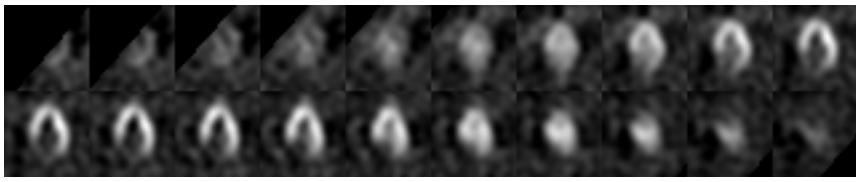
Cardiac single photon emission computed tomography (SPECT) provides a clinician with a set of three-dimensional images to visualize distribution of radioactive counts within the myocardium (the middle layer of the heart wall; heart muscle) and surrounding structures (Cullom, 1994). Images represent radioactive count densities within the heart muscle which are proportional to muscle perfusion, in particular of the left ventricle (LV), which is normally thicker than other cardiac structures. Two studies are performed after a patient is injected with a tracer, one at rest (rest image) and one after injection during maximal stress (stress image). The studies are represented by two, 3-D density images. Clinicians compare the two images in order to detect abnormalities in the distribution of blood flow within the left ventricular myocardium.

Visualization of the SPECT images is complicated by the fact that three-dimensional density images cannot be directly presented using contemporary display devices that produce two-dimensional pictures; some kind of transformation has to be performed. This transformation introduces a reduction of information. There are two practical alternatives: two-dimensional density images or three-dimensional surface renderings (Garcia, Ezquerra, DePuey, et al., 1986). The first preserves most of the intensity information, but three-dimensional relations are only implicit. The second provides explicit three-dimensional information explicit in which density is represented indirectly through the shape of the 3-D surface and/or its color (Faber, Akers, Peshock, and Corbett, 1991; Faber, Cooke, Peifer, et al., 1995).

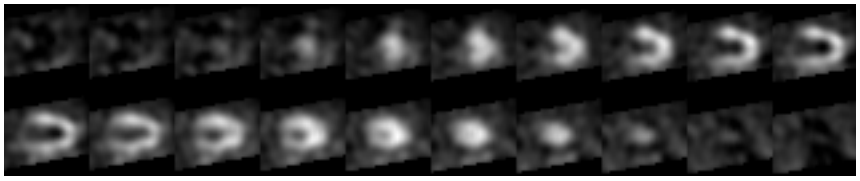
Typically, the LV is visualized as sets of two-dimensional intensity slices. When sliced perpendicular to the long axis of the left ventricle, the view is termed *short axis*. Slices



(a) Short axis view.



(b) Horizontal long axis view.



(c) Vertical long axis view.

Figure 2.3: 2D slices of 3D SPECT images.

parallel to the long axis of the LV are called *vertical long axis*, and *horizontal long axis* views (Fig. 2.3). Three-dimensional relations are only implicit in the views; it is left to the interpreting physician to mentally reconstruct them as a 3-D object.

Another family of 2-D visualization methods is based on projections in non-Cartesian coordinate systems. The three-dimensional left ventricle is *unwrapped* on a two-dimensional plane by radial projection into spherical coordinates (Goris, Boudier, and Briandet, 1987), or combination of spherical and cylindrical coordinates (Van Train, Garcia, Cooke, and Areeda, 1994). They are generally referred to as *bull's-eye* methods since they produce pairs of round images (rest and stress), see Fig. 2.4.

A number of 3-D surface rendering methods exists. They are frequently used in association with *gated blood-pool SPECT* (Corbett, 1994) to produce motion sequences of the left ventricle function. This is a very intensely researched area; however, in this narrative

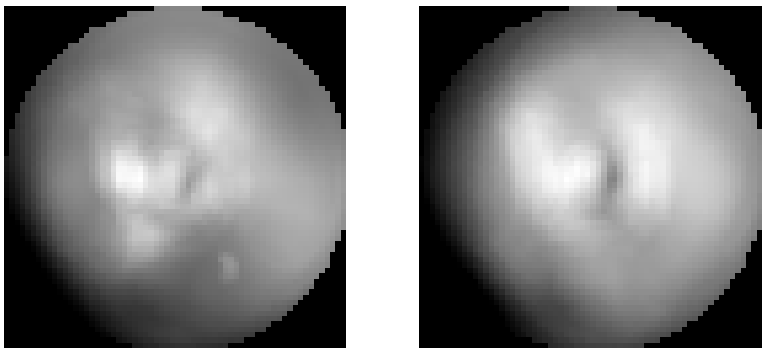


Figure 2.4: Bull's-eye images of the left ventricle.

only static SPECT images are addressed.

A number of techniques have been developed to aid in classification of the images; most of them are concerned with visualization. However, it has been shown that interpretation of images by strictly visual techniques is fraught with error and inconsistency (Cuaron, Acero, Cardenas, et al., 1980). For this reason, assistance in diagnosis has been sought through the use of computer-derived image display and quantitation. Such quantitation has demonstrably decreased the variability in image interpretation (Francisco, Collins, Go, et al., 1982).

One of the few examples of automatic interpretation of SPECT images is the PERFEX expert system (Ezquerra, Mullick, Cooke, Garcia, and Krawczynska, 1992). This system infers the extent and severity of coronary artery disease from the perfusion distribution.

Chapter 3

Knowledge Discovery in the Cardiac SPECT Imaging Database

3.1 Knowledge Discovery in Databases

The overall problem addressed in this narrative, "Bayesian Learning for Cardiac SPECT Image Interpretation", is an example of the process known as *Knowledge Discovery in Databases* or KDD. At an abstract level, the KDD field is concerned with the development of methods and techniques for making sense of data (Fayyad, Piatetsky-Shapiro, and Smyth, 1996). The KDD process consists of the following steps (Cios, Teresinska, Konieczna, Potocka, and Sharma, 2000; Cios, Pedrycz, and Swiniarski, 1998):

1. Understanding the problem domain
    - determination of objectives
    - assessment of the current situation
    - determination of data mining objectives, and
    - preparation of the project plan
2. Understanding the data

- collection of initial data
- description of the data
- initial exploration of the data, and
- verification of the quality of the data

3. Preparation of the data

- data selection
- data cleaning
- constructing and merging the data, and
- reformatting of the data

4. Data mining

- selection of several data mining methods
- building the model, and
- model assessment

5. Evaluation of the discovered knowledge

- assessment of the results versus the objectives
- keeping the approved models
- reviewing the entire knowledge discovery process, and
- determining actions to be taken based on the achieved results

6. Using the discovered knowledge

- implementation and monitoring plans
- generation of a final report, and
- overview of the entire project for future use and improvements

In our previous work, (Sacha, Cios, and Goodenday, 2000), we discussed in detail the relation of these steps to the problem of automating cardiac SPECT diagnosis. The work presented here is concentrated on particular aspects of the KDD process:

- creation of the database of cardiac SPECT data and 3D SPECT images, that can be considered a step partially preceding the KDD process. This discussed later in the chapter.

- data preparation, and in particular extraction of features from 3D cardiac SPECT images. This is discussed in detail in Chapter 4.

- creation of new data mining algorithms that are capable of efficiently dealing with a high level of uncertainty present in the data. This the subject of Chapter 7.

- data mining step: application of newly created and existing algorithms for building models of the cardiac SPECT data including diagnosis of left ventricular perfusion. Data mining step is covered in Chapter 8.

Before we can apply algorithms for learning Bayesian network classifiers, the data mining step, a number of crucial training-data preparation steps need to be completed. Data preparation, data selection, data cleaning, incorporation of additional prior knowledge and proper interpretation of results of data mining are essential to ensure that useful knowledge will be derived from the data. KDD is a highly iterative process, many of the tasks may be performed in a different order, and it is often necessary to repeatedly backtrack to previous tasks and repeat certain actions (Chapman, Clinton, Khobaza, Reinartz, and Wirth, 1999).

A prerequisite to starting a practical KDD process is *data warehousing*. Data warehousing refers to a number of activities involved in collection and cleaning of data to make them available for online analysis and decision support. The objective is to organize the available data into a coherent database system and provide well-defined methods for efficient access to the data. The software we created for this purpose is described in Appendix A. The re-

mainder of this chapter describes the original data obtained from MCO, data warehousing effort, and some of the data selection issues.

3.2 The Original Data

The data collection process was initiated at the Medical College of Ohio (MCO) in 1992. Data were recorded first on paper worksheets then entered manually into an MS Excel spreadsheet. Each row corresponds to a single patient visit – a SPECT procedure. About 184 parameters are recorded. Data include personal patient information such as age, sex, height; information about the procedure, and the nuclear cardiologist's interpretation of the SPECT images (by regions of interest), and perfusion classification (i.e. diagnosis). For the purpose of this work 4,828 records have been obtained from MCO over a period of time.

SPECT images are stored in a proprietary format, without database organization. Archiving of images has not been systematic due to significant storage requirements. Earliest available SPECT images date back only to 1996. We have obtained 8,142 SPECT image files from MCO. Typically there are six three-dimensional images and a number of auxiliary files per case. After cleaning, it corresponded to about 728 cases. Each set was stored in a separate directory, the name of which was a combination of patient's hospital number and the date of study. Images and auxiliary data were stored in a proprietary binary format. Descriptions of image files and their format was not provided. The following information was typically contained in each directory:

- Patient identification number, name, and visit date.
- Row data for rest and stress study
- 3-D images corresponding to short, vertical long, and horizontal long axis views of the heart; for rest and for stress study.

We determined the format of image files by reverse engineering. Image files were

spread between different computer hard drives and archive media. Not all directories contained complete data sets. Combined image data sets occupy currently over 3GB of disk space. Due to cost and licensing issues software for manipulation and visualization of SPECT images was not available; we wrote it from scratch.

3.3 Data Warehousing

The initial effort was data warehousing. Data contained in the spreadsheet have been converted to a relational database. The proprietary SPECT image file format had been reverse-engineered to the level that allowed the most critical information to be extracted - the actual three-dimensional images and patient identification information (hospital number and SPECT date) stored in the header. Software for automatic indexing of images, using patient identification information, was also created. Image indexes were stored in the database table. Images were stored outside of the database within a predetermined directory structure. The database design objective was simplicity of maintenance and ability to add easily new patient records and images as they become available. Software for browsing patient records with simultaneous display of available images in several modes was written. The database also stores data generated by various data mining activities, such as information about generated models of a normal ventricle, and features extracted from the 3D SPECT images. This way an SQL query can be directly used to generate variants of data sets needed for the automation of diagnosis, e.g., learning Bayesian network classifiers.

3.4 Data Selection – Verification of the Data Quality

We have semi-manually inspected the original data to eliminate errors, e.g. typos. The intention was not to modify the data unless the correction was straightforward. Rather, where possible, we constructed SQL queries to filter undesired records.

Table 3.1: Cases with complete sets of SPECT images and complete diagnosis data.

	Male			Female			Total		
	No ART	with ART	*Total*	No ART	with ART	*Total*	No ART	With ART	**Total**
NL	23	19	***42***	36	36	***72***	59	55	**114**
IS	21	6	***27***	17	4	***21***	38	10	**48**
INF	41	4	***45***	18	4	***22***	59	8	**67**
IS-IN	52	2	***54***	9	2	***11***	61	4	**65**
EQ	3	0	***3***	1	0	***1***	4	0	**4**
REV	0	1	***1***	1	1	***2***	1	2	**3**
LVD	0	0	***0***	2	0	***2***	2	0	**2**
	140	*32*	***172***	*84*	*47*	***131***	*224*	*79*	**303**
IS, IS-IN	1	0	***1***	0	0	***0***	1	0	**1**
IS, REV	2	1	***3***	1	0	***1***	3	1	**4**
IS, LVD	1	0	***1***	0	1	***1***	1	1	**2**
IS-IN, LVD	7	0	***7***	0	0	***0***	7	0	**7**
INF, IS	11	0	***11***	1	0	***1***	12	0	**12**
INF, IS-IN	2	0	***2***	0	0	***0***	2	0	**2**
INF, REV	3	1	***4***	0	0	***0***	3	1	**4**
INF, LVD	10	0	***10***	2	0	***2***	12	0	**12**
INF,LVD,REV	1	0	***1***	0	0	***0***	1	0	**1**
LVD, REV	1	0	***1***	0	0	***0***	1	0	**1**
	39	*2*	***41***	*4*	*1*	***5***	*43*	*3*	**46**
total	179	34	***213***	88	48	***136***	267	82	**349**

The numbers of records available were counted to estimate the statistical validity of expected results; e.g., if a sufficient number of examples exists for each learning class. Table 3.1 shows number of cases available in the latest version of the database for each of the left ventricle perfusion classifications (NL – normal, IS – ischemia, INF – infarct, IS-IN – ischemia and infarct, EQ – equivocal, REV – reversible redistribution, LVD – left ventricle dysfunction, ART – artifact). The top of the table corresponds to records with a single classification code, the bottom to records that contain more than one classification code.

By matching image sets to database records some data errors have been found. Most of these errors were resolved as typographical, but some of the images remained without matching patient records. They were eliminated from the analysis. Next, image sets were checked for completeness, e.g. stress study missing, and for quality of the individual images, mostly related to sufficient contrast (photon count).

Chapter 4

Extraction of Features from Cardiac SPECT Images

4.1 Cardiologist's Interpretation Process

Each diagnostic patient study contains two, three-dimensional SPECT cardiac image sets of the left ventricle (one for rest and one for stress study). Comparing the two image sets allows the interpreting physician to decide on diagnoses, such as ischemia, infarct or artifact. Evaluation of the images is a highly subjective process with potential for substantial variability (Cuaron et al., 1980). To analyze the images, we followed a procedure originally described by in (Cios, Goodenday, Shah, and Serpen, 1996). The raw image data taken from multiple planar views are processed by filtered back-projection to create a 3D image. These three-dimensional images are displayed as three sets of two-dimensional images corresponding to the **short axis view**, **horizontal long axis view**, and **vertical long axis view**.

From these two-dimensional views, the interpreting physician may select five slices to represent the final report, see Fig. 4.1. From the short axis view, one slice is taken near the heart's apex, one at the mid-of the ventricle, and one near the heart's base. With this

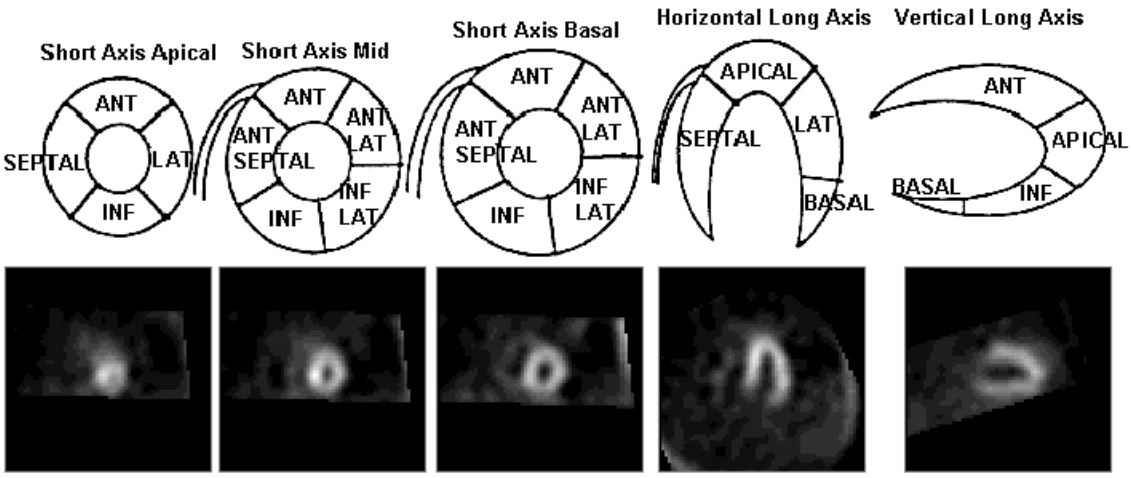


Figure 4.1: Twenty two regions of interest within a left ventricle (22 ROIs). The first three images correspond to a short axis view slices, the last two to horizontal long axis view and vertical long axis view, respectively.

technique, for each of the horizontal and vertical axis views, a single slice is selected corresponding to the center of the LV cavity. Each of these five images is subdivided into a number of regions of interest, from four to five, along the walls of the LV, for a total of 22 regions. The appearance of the LV and maximum count in each of the regions is evaluated. Corresponding region of interest (ROI) locations on the stress and rest images are compared. Perfusion in each of the regions is classified into one of seven defect categories: **normal**, **reversible**, **partially reversible**, **defect**, **defect showing reverse redistribution**, **equivocal**, or **artifact**. The physician's impression of overall LV perfusion, or the final SPECT image analysis result, is concluded from the results of analysis in each of the ROIs. From the analysis, the interpreting physician categorizes a study as showing one or more of eight possible conditions: **normal**, **ischemia**, **infarct and ischemia**, **infarct**, **reverse redistribution**, **equivocal**, **artifact**, or **LV dysfunction**, see Fig. 4.2. Some of the perfusion categories may coexist, for example normal and artifact, reverse redistribution and infarct, etc.

The most fundamental operation performed by the interpreter during analysis of SPECT

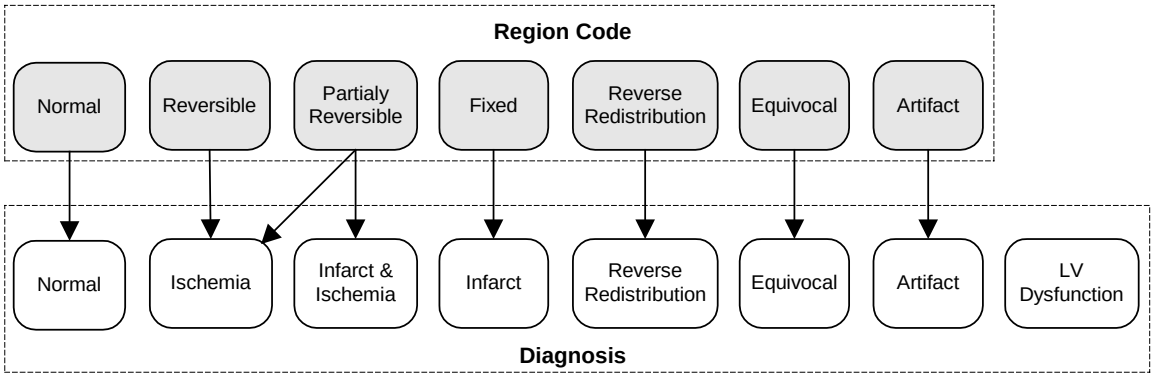


Figure 4.2: Overall classification of left ventricle perfusion based on partial classifications.

images is comparison of the case at hand to a mental image of a normal LV. The first task is to establish the location of the ROIs within the current SPECT image. This process is complicated by two factors that create a major challenge for any algorithmic implementation. Both of these factors modify the apparent shape of the analyzed LV in the SPECT image. They are defined bellow.

Actual LV detects Changes in perfusion of the LV are manifested as changes in the brightness (radioactive counts) of the SPECT image. When perfusion is reduced, the counts are low, and, in effect, parts of the LV may not even be apparent in the image due to extremely poor perfusion. The interpreting physician deals with this loss of counts by mentally "reconstructing" the missing contour of the image based on knowledge of heart anatomy and previous experience with cardiac SPECT imaging. However, this is a major challenge for computer algorithms.

Artifacts The most common artifact for Thallium 201 imaging is decreased count, usually from attenuation by breast tissue in females, or by the diaphragm in males. Artifacts may complicate localization of the 22 ROIs. Also, even after the analysis regions are determined correctly, presence of artifact may lead to false diagnosis since the decreased count may be erroneously taken for real perfusion defects.

Once the predefined ROIs are established, differences between rest and stress images in each location are analyzed, and counts within each region are compared to that of a normal model. The overall impression of myocardial perfusion is directly concluded from the results of this analysis from each of the regions.

In this work, a combination of computer vision and machine learning is used to mimic the diagnostic process performed by an interpreting physician. Before a machine learning algorithm can be used, a set of features needs to be extracted from the three-dimensional images for each case study. The most natural approach is to extract a single feature corresponding to each region of interest in both rest and stress images (see Fig. 4.1), as it was originally done in (Cios et al., 1996). Each feature can be represented by a single number, e.g., maximum count, mean count, median count, etc. Thus we have a set of 44 attributes for each patient's case that can be used to classify LV perfusion. Another approach is to perform local classification first, in each of the 22 regions using information from rest and stress images, and then use these 22 intermediate regional classifications to classify the overall LV perfusion. Results for both of these approaches will be presented in Chapter 8.

It is difficult to automatically perform correct and repeatable determination of the ROIs directly from the 3D images due to artifacts, actual LV defects, and anatomical differences between patients. In order to do that, we use a model of a normal LV. Location of the regions is a part of the model description. The model plays a role analogous to the interpreter's mental image of a normal LV. The first step in the feature extraction process is registration - matching the image at hand to the model using translation, rotation, and scaling operations. The image may be matched with a number of models. The model with highest correlation ratio is selected and used to locate slices and regions of interest in the image. Regional perfusion is determined based on the count/intensity of LV walls within the region. Even when the image and a model are correctly registered, the walls of the model and case under investigation may not completely overlap, thus compromising quality of the feature extraction process. Correct determination of myocardium wall location is

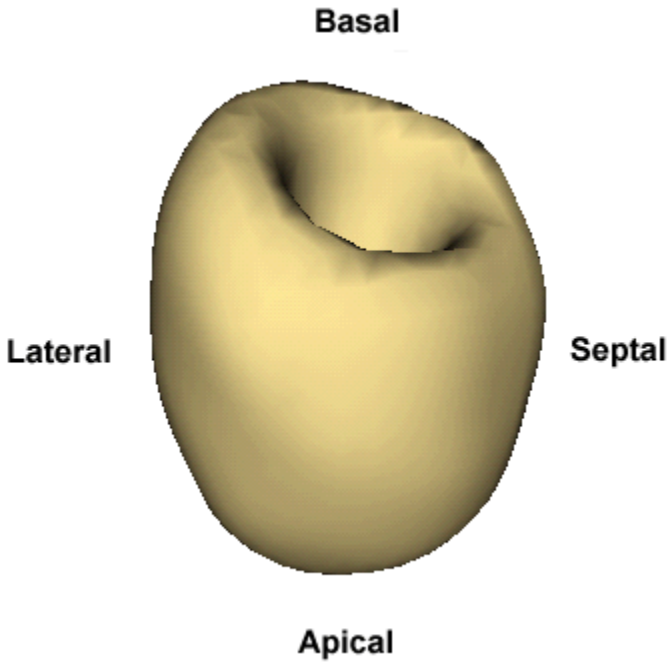


Figure 4.3: Three-dimensional rendering of a normal left ventricle model.

difficult. The approach we use, besides direct reference to the normal LV model, is similar to the radial search as used in SPECT bull's-eye methods (Goris et al., 1987; Van Train et al., 1994). The model is used to determine the center of the left ventricular cavity. A search is performed in a desired direction, starting from the center of the cavity; the maximum intensity value along the search direction is recorded. This is based on the premise that counts within the LV wall are higher than in surrounding areas.

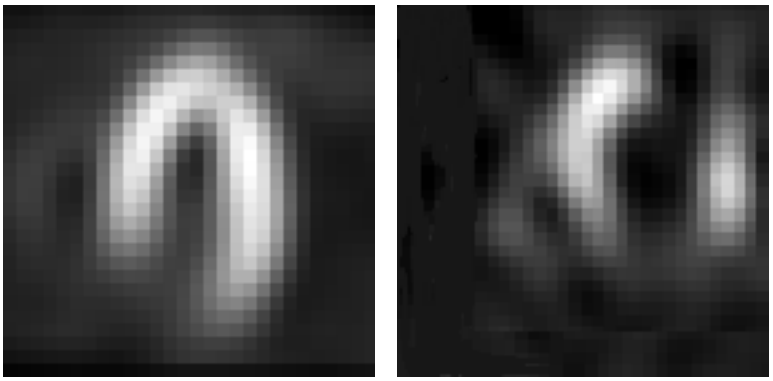
Another critical issue is normalization of the image intensity range. Not only do counts vary significantly between patients, they are also different between the rest and stress images for the same patient. There is no easy way to correct that. Typically, numerical values are normalized as a percentage of the maximum count in a given three-dimensional image.

4.2 Model of a Normal Left Ventricle

There are a number of possible approaches to create a model of the left ventricle. Recently, physics-based deformable models have been gaining in popularity (Declerck, Feldmar, Goris, and Betting, 1997). The premise is that they are well suited to deal with natural anatomical differences, but their drawbacks are their relative complexity, and they are difficult to use in cases when there are large perfusion defects. Physical-based deformable models are particularly useful for tracking a motion of LV, for instance, in gated-pool SPECT imaging. SPECT images used in this research are static. Thus, we decided to build a rigid model of the left ventricle by "averaging" a set of images corresponding to cases diagnosed by the interpreting physician as normal. We also decided to use, for models only, the images that were evaluated by the most experienced physician. Images were additionally screened for excess presence of noise and artifacts. Before averaging, the selected case images are translated, rotated, and scaled to obtain the best match between them. A variant of best-first heuristic was used to make correlation search computationally feasible. Once matched to each other, case images were added, constituting an *averaged model*. Due to anatomical differences between patients, models for females and males were created separately. The format of a model is the same as a three-dimensional SPECT image, so a cardiologist can easily evaluate its quality. Each model was manually inspected, and the locations of slices and regions of interest for this particular model were recorded. An example of three-dimensional rendering of a male rest model is shown in Fig. 4.3. The rendering was created using the Visualization Toolkit library (Schroeder, Martin, and Lorensen, 1998); the library can be freely downloaded from `http://www.kitware.com`.

4.3 3D Image Registration

To ensure repeatability and robustness of the feature extraction, we ideally require that the object of interest on the analyzed images have the same spatial orientation and scale. In



(a) Registration reference (b) Image to be registered

Figure 4.4: Sample 2D image registration problem. Images represent horizontal long view of a left ventricle. Image (a) comes from a model of a normal LV; image (b) from a patient with infarct.

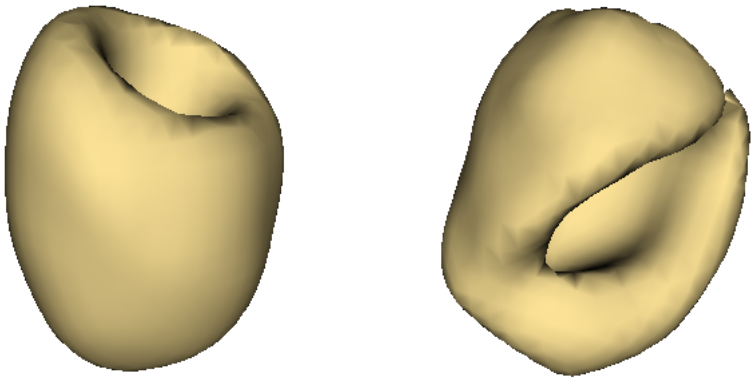
image processing this operation is called *registration* or *image matching*. An example of a 2D image registration problem is presented in Fig. 4.4, 3D image registration problem in Fig. 4.5. Image (a) represents a desired orientation and scale. Image (b) has to be translated, rotated and scaled to match the object model in image (a).

Let I_M denote the three-dimensional reference image – the image containing the model object. Let I_C denote the three-dimensional image that needs to be registered to I_M. Let $I_R = T(I_C)$ denote the image I_C after registration. The problem of image registration is that of finding a transformation T. Transformation T when applied to image I_C matches the considered object in that image, in our case the left ventricle, with the model object in image I_M.

In the approach presented here we search for the optimal registration transformation T by maximizing the *cross-correlation coefficient*

$$r(T) = \frac{\int_W I_M(\mathbf{x})\, T[I_C(\mathbf{x})]\, \mathrm{d}\mathbf{x}}{\sqrt{\int_W I_M^2(\mathbf{x})\, \mathrm{d}\mathbf{x}\ \int_W T^2[I_C(\mathbf{x})]\, \mathrm{d}\mathbf{x}}} \tag{4.1}$$

where $\mathbf{x} = [x_1, x_2, x_3]^T$ is a three-dimensional pixel coordinate, $I(\mathbf{x})$ is the gray level value of pixel $\mathbf{x}$ in image I. W is the integration domain, typically is the whole volume of the



(a) Registration reference (b) Object to be registered

Figure 4.5: Sample 3D image registration problem. Images represent isosurface rendering of 3D cardiac SPECT images. Image (a) comes from a model of a normal LV; image (b) from a patient with a very well visible infarct.

reference image I_M. The cross-correlation coefficient is a useful similarity measure. It is zero for totally dissimilar images and reaches maximum of one for identical images.

Since image coordinates are discrete, we use sums rather than integrals to calculate the correlation coefficients:

$$r(T) = \frac{\sum_W I_M(\mathbf{x})\; T[I_C(\mathbf{x})]}{\sqrt{\sum_W I_M^2(\mathbf{x})\;\; \sum_W T^2[I_C(\mathbf{x})]}}. \tag{4.2}$$

Alternative approaches to left ventricle registration, including these based on physics-based deformable models, can be found in (Declerck et al., 1997) or (Qian, Mitsa, and Hoffman, 1996) among others.

4.3.1 3D Image Transformation

Transformation T used here for registration of images is a superposition of three component transformations:

- translation $T_{\mathbf{t}}$,
- rotation $T_{\boldsymbol{\alpha}}$, and

- scaling T_s.

The superposition of these transformations can be expressed as follows:

$$T[I] = (T_{\mathbf{t}} \oplus T_{\boldsymbol{\alpha}} \oplus T_s)[I] = T_{\mathbf{t}}\left[T_{\boldsymbol{\alpha}}\left[T_s\left[I\right]\right]\right]. \tag{4.3}$$

Transformation $I_R = T[I_C]$ assigns new coordinates to each of the pixels in the original image I_C new coordinates and maintains its intensity. The transformation is continuous. The new coordinates, after transformation, are not discrete. Thus, the transformation is practically computed backwards. We start with discrete coordinates in images I_R and calculate what would be the original coordinates in the image I_C. Since the original image contains only pixels at discrete coordinates a tri-linear interpolation is performed to find the estimation of the intensity in the image I_R.

In the following four sections, we first describe calculation of each of the component transformations $T_{\mathbf{t}}$, $T_{\boldsymbol{\alpha}}$, and T_s followed by the description of the tri-linear interpolation. We will use $\mathbf{x}^{(1)}$ to denote coordinates of a point before a particular transformation, and $\mathbf{x}^{(2)}$ to denote coordinates of the same point after transformation.

Translation

Translation transformation $T_{\mathbf{t}}$ is described by three parameters $\mathbf{t} = [t_1, t_2, t_3]^T$ where t_i is a translation along coordinate axis x_i. Translation is defined by the following formula:

$$T_{\mathbf{t}}\left[I\left(\mathbf{x}^{(1)}\right)\right] \;=\; I\left(\mathbf{x}^{(2)}\right) \;=\; I\left(\mathbf{x}^{(1)} + \mathbf{t}\right) \tag{4.4}$$

Thus

$$\mathbf{x}^{(1)} \;=\; \mathbf{x}^{(2)} - \mathbf{t}. \tag{4.5}$$

Rotation

Rotation transformation $T_{\boldsymbol{\alpha}}$ is described by three angular parameters $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \alpha_3]^T$ where α_i is a rotation around line parallel to the axis x_i, and a center of rotation $\mathbf{x}^{(0)}$. Rota-

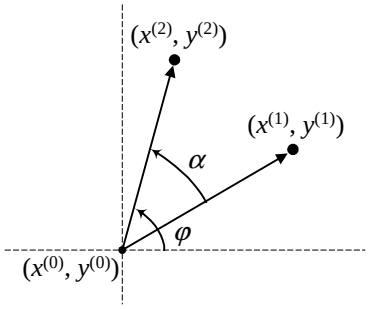


Figure 4.6: Rotation in 2D.

tion $T_{\boldsymbol{\alpha}}$ can be represented by a three simpler rotations performed for each axis separately:

$$T_{\boldsymbol{\alpha}}[I] = (T_{\alpha_1} \oplus T_{\alpha_2} \oplus T_{\alpha_3})[I]. \tag{4.6}$$

Each of the component rotations T_{α_i} can be seen as a two-dimensional transformation in a plane perpendicular to axis x_i.

Rotation in 2D Let α be a rotation angle. Let point $(x^{(0)}, y^{(0)})$ be a center of rotation. Let $(x^{(1)}, y^{(1)})$ denote a point coordinates before rotation, let $(x^{(2)}, y^{(2)})$ denote coordinates of the same point after rotation, (see Figure 4.6). Coordinates of the point *before* rotation can be calculated using the following formulas:

$$\begin{aligned} r &= \sqrt{\left(x^{(2)} - x^{(0)}\right)^2 + \left(y^{(2)} - y^{(0)}\right)^2} \\ \varphi &= \arctan \frac{y^{(2)} - y^{(0)}}{x^{(2)} - x^{(0)}} \\ x^{(1)} &= r\cos(\varphi - \alpha) \\ y^{(1)} &= r\sin(\varphi - \alpha) \end{aligned} \tag{4.7}$$

Rotation in 3D Formulas used to compute component rotations in 3D are a straightforward extension of the formulas presented in Eq. (4.7). They are presented here for a convenient reference.

Coordinates of the original point for the rotation T_{α_1}:

$$\begin{aligned} r_1 &= \sqrt{\left(x_2^{(0)} - x_2^{(2)}\right)^2 + \left(x_3^{(0)} - x_3^{(2)}\right)^2} \\ \varphi_1 &= \arctan \frac{x_3^{(2)} - x_3^{(1)}}{x_2^{(2)} - x_2^{(1)}} \\ x_1^{(1)} &= x_1^{(2)} \\ x_2^{(1)} &= r_1 \cos(\varphi_1 - \alpha_1) \\ x_3^{(1)} &= r_1 \sin(\varphi_1 - \alpha_1) \end{aligned} \tag{4.8}$$

Coordinates of the original point for the rotation T_{α_2}:

$$\begin{aligned} r_2 &= \sqrt{\left(x_1^{(2)} - x_1^{(0)}\right)^2 + \left(x_3^{(2)} - x_3^{(0)}\right)^2} \\ \varphi_2 &= \arctan \frac{x_1^{(2)} - x_1^{(0)}}{x_3^{(2)} - x_3^{(0)}} \\ x_1^{(1)} &= r_2 \sin(\varphi_2 - \alpha_2) \\ x_2^{(1)} &= x_2^{(2)} \\ x_3^{(1)} &= r_2 \cos(\varphi_2 - \alpha_2) \end{aligned} \tag{4.9}$$

Coordinates of the original point for the rotation T_{α_3}:

$$\begin{aligned} r_3 &= \sqrt{\left(x_1^{(2)} - x_1^{(0)}\right)^2 + \left(x_2^{(2)} - x_2^{(0)}\right)^2} \\ \varphi_3 &= \arctan \frac{x_2^{(2)} - x_2^{(0)}}{x_1^{(2)} - x_1^{(0)}} \\ x_1^{(1)} &= r_3 \cos(\varphi_3 - \alpha_3) \\ x_2^{(1)} &= r_3 \sin(\varphi_3 - \alpha_3) \\ x_3^{(1)} &= x_3^{(2)} \end{aligned} \tag{4.10}$$

Scaling

We decided to use isotropic scaling; thus, the scaling transformation T_s is described by a parameter s and the center of scaling $\mathbf{x}^{(0)}$, that is the same as the center of rotation.

$$T_s\left[I\left(\mathbf{x}^{(1)}\right)\right] = I\left(\mathbf{x}^{(2)}\right) = I\left(s\left(\mathbf{x}^{(1)} - \mathbf{x}^{(0)}\right) + \mathbf{x}^{(0)}\right) \tag{4.11}$$

Thus

$$\mathbf{x}^{(1)} \;=\; \frac{\mathbf{x}^{(2)} - \mathbf{x}^{(0)}}{s} + \mathbf{x}^{(0)} \tag{4.12}$$

4.3.2 Tri-linear Interpolation

The material presented in this section is an extension of a two-dimensional bi-linear interpolation presented in (Press, Teukolsky, Vetterling, and Flannery, 1992) to three-dimensions. We call it a tri-linear interpolation.

In three dimensions, we imagine that we are given a matrix of image intensity values `Ia[1..m][1..n][1..p]`. We are also given arrays `x1a[1..m]`, `x2a[1..n]`, `x3a[1..p]` that describe coordinates of pixels in the image I in each of the axis x_1, x_2, and x_3, respectively. The relation of these inputs and the underlying image $I(x_1, x_2, x_3)$ is

$$\texttt{Ia[i][j][k]} \;=\; I\left(\texttt{x1a[i]}, \texttt{x2a[j]}, \texttt{x2a[k]}\right) \tag{4.13}$$

We want to estimate, by interpolation, the gray level of image I at some untabulated point (x_1, x_2, x_3).

An important concept is that of the *grid cube* in which the point (x_1, x_2, x_3) falls, that is, the eight tabulated points that surround the desired interior point. For convenience, we will number these points from 1 to 8. More precisely, if inequalities

$$\begin{aligned} \texttt{x1a[i]} \;\le\; x_1 \;\le\; \texttt{x1a[i+1]} \\ \texttt{x2a[j]} \;\le\; x_2 \;\le\; \texttt{x2a[j+1]} \\ \texttt{x3a[k]} \;\le\; x_3 \;\le\; \texttt{x3a[k+1]} \end{aligned} \tag{4.14}$$

define `i`, `j` and `k`, then

$$\begin{aligned} I_1 &\equiv \texttt{ya[i][j][k]} \\ I_2 &\equiv \texttt{ya[i+1][j][k]} \\ I_3 &\equiv \texttt{ya[i+1][j+1][k]} \\ I_4 &\equiv \texttt{ya[i][j+1][k]} \\ I_5 &\equiv \texttt{ya[i][j+1][k+1]} \\ I_6 &\equiv \texttt{ya[i+1][j][k+1]} \\ I_7 &\equiv \texttt{ya[i+1][j+1][k+1]} \\ I_8 &\equiv \texttt{ya[i][j+1][k+1]} \end{aligned} \tag{4.15}$$

The tri-linear interpolation on the grid cube is formulated as follows:

$$\begin{aligned} t &\equiv (x_1 - \texttt{x1a[i]})/(\texttt{x1a[i+1]} - \texttt{x1a[i]}) \\ u &\equiv (x_2 - \texttt{x2a[j]})/(\texttt{x2a[j+1]} - \texttt{x2a[j]}) \\ v &\equiv (x_3 - \texttt{x3a[k]})/(\texttt{x3a[k+1]} - \texttt{x3a[k]}) \end{aligned} \tag{4.16}$$

so that t, u, and v each lie between 0 and 1, and

$$\begin{aligned} I(x_1, x_2, x_3) \;=\; & (1-t)\cdot(1-u)\cdot(1-v)\cdot I_1 \;+\; t\cdot(1-u)\cdot(1-v)\cdot I_2 \\ & +\; t\cdot u\cdot(1-v)\cdot I_3 \;+\; (1-t)\cdot u\cdot(1-v)\cdot I_4 \\ & +\; (1-t)\cdot(1-u)\cdot v\cdot I_5 \;+\; t\cdot(1-u)\cdot v\cdot I_6 \\ & +\; t\cdot u\cdot v\cdot I_7 \;+\; (1-t)\cdot u\cdot v\cdot I_8 \end{aligned} \tag{4.17}$$

4.3.3 Computation of Image Registration Transformation

Computation of the cross-correlation coefficient for three-dimensional images is a computationally intensive process. As presented above, the three-dimensional registration has seven degrees of freedom:

- translation in three dimensions: t_1, t_2, and t_3;

- three rotation angles: α_1, α_2, α_2;
- and a scaling factor s.

To make the registration of cardiac SPECT images based on cross-correlation coefficient practical we first estimate the registration transformation and then refine the transformation parameters by performing a *best-first* heuristic search with decay in the seven-dimensional parameter space. The parameter decay was added to dump oscillations we had experienced when first testing the best-first search.

Estimation of the Image Registration Transform

We estimate two components of the registration transform: translation $T_{\mathbf{t}}$ and scaling T_s. The estimation is based on detecting a three-dimensional blob representing the left ventricle walls in the model and the target image. A SPECT image is first thresholded at 55% of the maximum intensity of that image to create a binary image. Next, a three-dimensional connected component labeling algorithm[1]is applied to the binary image to label all blobs in the image. The largest blob near the center of the image is considered to be the left ventricle. The thresholding level has been experimentally set at its value of 55% to guarantee that it is practically always the case.

Let $\mathcal{B}$ denote a set of pixels constituting a blob. We will assume that $\mathbf{x} \in \mathcal{B}$ means that a pixel with coordinates $\mathbf{x}$ is a member of the blob $\mathcal{B}$. We can also write $\mathcal{B} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$, where n is the number of pixels in the blob.

We define a *center* of a blob $\mathcal{B}$, denoted by $\bar{\mathbf{x}}$ as an average of blob points' coordinates:

$$\bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}^{(i)} \tag{4.18}$$

[1]We generalized the connected component labeling algorithm to three-dimensions. The main difference from the two-dimensional version was how the pixels are selected during GROW/MERGE phase. This is illustrated in Fig. 4.7.

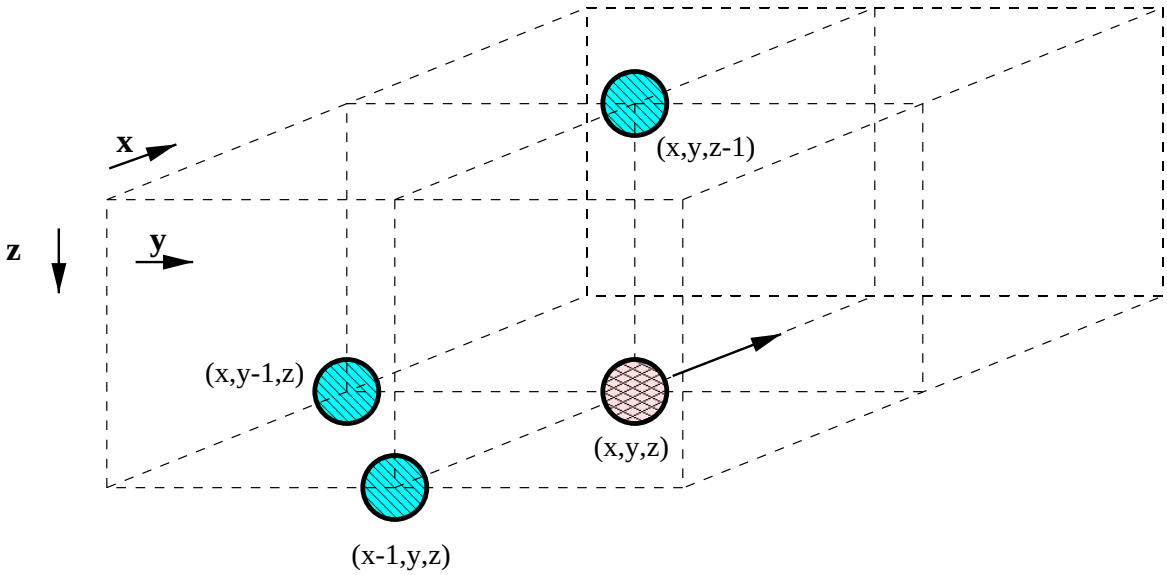


Figure 4.7: Connected component labeling in 3D. Search for GROW/MERGE candidates in the case of 4-neighborhoods.

This definition was giving us better results in estimating translation transformation $T_{\mathbf{t}}$ than other approaches including defining the blob's center as a center of gravity.

Let $\bar{\mathbf{x}}_M$ be a center of the blob $\mathcal{B}_M$ representing left ventricle in the reference image I_M and let $\bar{\mathbf{x}}_C$ be a center of the blob $\mathcal{B}_C$ representing left ventricle in the target image I_C. We estimate the translation transformation $T_{\mathbf{t}}$ as distance between centers of blobs $\mathcal{B}_M$ and $\mathcal{B}_C$:

$$\mathbf{t} \;=\; \bar{\mathbf{x}}_M - \bar{\mathbf{x}}_C \tag{4.19}$$

In order to estimate the scaling transform T_s, we introduce a notion that we call a scale factor. A *scale factor* of a blob $\mathcal{B}$, denoted by ξ is a median distance of pixels belonging to the blob from the blob's center $\bar{\mathbf{x}}$. Let ξ_M be a scale factor of the blob $\mathcal{B}_M$, and ξ_C be a scale factor of the blob $\mathcal{B}_C$. Then, we can estimate the scaling transform T_s as follows

$$s = \frac{\xi_M}{\xi_C} \tag{4.20}$$

As with the definition of blob's center, the approach to estimate the scaling transform based on the scaling factor consistently give better scale estimates than other approaches we

Algorithm 4.1 Cardiac SPECT image registration estimation.

1. **for** the reference image I_M **and** the target image I_C **do**
2. Threshold the image at 55% of the maximum gray level value
3. In the binarized image find the largest blob $\mathcal{B}$ representing the left ventricle wall
4. Calculate: $\bar{\mathbf{x}}$ – center of the blob
5. Calculate: ξ – size factor the blob
6. Estimate translation transformation $\hat{T}_{\mathbf{t}}$: $\mathbf{t} = \bar{\mathbf{x}}_M - \bar{\mathbf{x}}_C$
7. Estimate scaling transformation $\hat{T}_s$: $s = \frac{\xi_M}{\xi_C}$
8. **return** registration transforation estimate $\hat{T} = \hat{T}_{\mathbf{t}} \oplus \hat{T}_s$.

tested.

The algorithm of registration transformation estimation is summarized in Alg. 4.1.

Registration Refinement: Best-First Search with Parameter Decay

The initial estimate of the registration transformation T given by Alg. 4.1 is refined by performing an optimization/search that maximizes the correlation coefficient $r(T)$ given by Eq. (4.2). The search is performed as a series of local neighborhood searches. We start by defining a *search step* Δ

$$\Delta \;=\; [\Delta_{t_1}, \Delta_{t_2}, \Delta_{t_3}, \Delta_{\alpha_1}, \Delta_{\alpha_2}, \Delta_{\alpha_3}, \Delta_s]^T$$

A *search neighborhood* is defined by a search center $T^{(0)}$, and the maximum number of steps γ that can be taken from the center. The search neighborhood consists of $(2\gamma + 1)^7$ points. For $\gamma = 1$ it is $3^7 = 2187$ points, for $\gamma = 2$ it consists of $5^7 = 78125$ points, and so on. Typically $\gamma = 1$ is a good enough tradeoff between accuracy and computational complexity. The transformation T' in the search neighborhood that has the highest correlation coefficient $r(T')$ becomes the center of the next search neighborhood. The local search is repeated as long as the change in the correlation coefficient $\delta \leftarrow \left| r(T') - r(T^{(0)}) \right|$ is greater

Algorithm 4.2 Registration refinement: best-first search with parameter decay

1. $i \leftarrow 0$ $\{$*iteration number*$\}$
2. $\delta \leftarrow \infty$ $\{$*change in the correlation-coefficient*$\}$
3. Estimate the registration transformation $\hat{T}$ using Alg. 4.1
4. $T^{(0)} \leftarrow \hat{T}$ $\{$*search neighborhood center*$\}$
5. **while** $i < i_{MAX}$ **and** $\delta > \delta_{MIN}$ **do**
6. Find best new transformation T' within maximum step range $\gamma \cdot \Delta$ from $T^{(0)}$
7. **if** $r(T') > r(\hat{T})$ **then** $\hat{T} \leftarrow T'$ $\{$*remember the best transformation so far*$\}$
8. $\delta \leftarrow \left| r(T') - r(T^{(0)}) \right|$
9. $T^{(0)} \leftarrow T'$
10. $\Delta \leftarrow \mu \cdot \Delta$ $\{$*reduce search step in next iteration by the decay factor* $\mu\}$
11. $i \leftarrow i + 1$
12. **return** $\hat{T}$.

than the limit value δ_{max} or until the maximum number of iterations has been reached.

During initial tests, we noticed that the best-first search described in the previous paragraph has a tendency to get into a sustained oscillation cycle two or more iterations long. To get rid of this phenomenon we introduced a decay factor μ that decreases the search step in each iteration. This helps to dampen the oscillations whenever they occur.

The final algorithm of registration transformation refinement, including parameter dumping, is summarized in Alg. 4.2.

Image Registration Database

The calculation of the registration transformations for each of the images for over 350 cases (with diagnosis) in the SPECT database was quite time consuming. However this calculation needed to be performed only once. We have added to the SPECT database a table containing the registration transformation parameters and the corresponding value of

the cross-correlation coefficient for each image and a model of normal left ventricle.

4.4 Extraction of Features from Registered Images

Feature extraction consists of two phases. First, objects of interest in the image are detected. Next, for each of the objects, or their parts, some features of interest are calculated. For SPECT images, the objects of interests are walls of the left ventricle. Following the approach described Section 4.1, we are interested in parts of left ventricular walls corresponding to twenty-two regions of interest presented in Fig. 4.1. We understand these twenty-two regions as 3D parts of the left ventricular walls. Once we are able to locate a region within a 3D SPEC image, a set of features is calculated for each of them separately. We can define a feature, for instance, as a median pixel intensity within a region.

4.4.1 Detection of Objects in SPECT Images

We have considered a number of approaches for detection of left ventricle walls. One of these approaches has been presented in Section 4.3.3 – an image is thresholded, and objects are detected by connected-components labeling. This approach is computationally efficient and sufficient for the purpose of registration transformation estimation, but not robust enough for detection of left ventricle walls. It only gives a coarse estimate of their location; a single threshold value is not sufficient.

An approach based on spherical radial gradient search has been proposed by Declerck et al. (1997). The difficulty with this approach is that the SPECT images typically are of low contrast and transitions between pixels in infarcted or ischemic left ventricle wall are small, making gradient detection methods impractical. This is especially true for Thallium-201 SPECT images used in this research.

A method specifically designed for detection of free-shape objects on low contrast images has been presented in (Sacha et al., 1996). This method performs object detection

using region growing approach. We have initially used it for detection of the left ventricle walls, but have encountered problems related to low resolution of SPECT images (small pixel size of objects).

The method that proved most robust in our experiments is based on spherical radial search presented by Goris et al. (1987). We eventually used a variant of this method that performs search in cylindrical coordinates since it is closer to the cardiologist's interpretation process presented in Section 4.1.

Spherical Radial Search

Goris et al. (1987) presented a modified method of *bull's-eye* images creation that uses only spherical search. A typical bull's-eye method uses a combination of cylindrical and spherical radial search; spherical search is performed in the apex area, cylindrical search in the mid and basal section of the left ventricle. The method presented by Goris et al. (1987) uses only spherical radial search with a center positioned near the base of left ventricle. In what follows, we will refer to this method as *GBB* (from name of the authors Goris, Boudier, and Briandet). While describing the GBB method, we will point where our implementation of the spherical search differs.

Background subtraction The initial step in the GBB method is background subtraction. It is done at a fixed level of 33% of the maximum pixel value in the three-dimensional image. Beside a typical maximum search along a radius, the authors also compute integral along the radius. The main reason for the background subtraction is, in our opinion, to improve repeatability of the integration results.

Reorientation and center selection In the GBB method, the LV image is manually reoriented to normalize its position. A center of radial search is selected manually, by the operator before the search is performed. In our approach, a model of a normal left ventricle is utilized for reorientation (registration) and automatic selection of the radial search center.

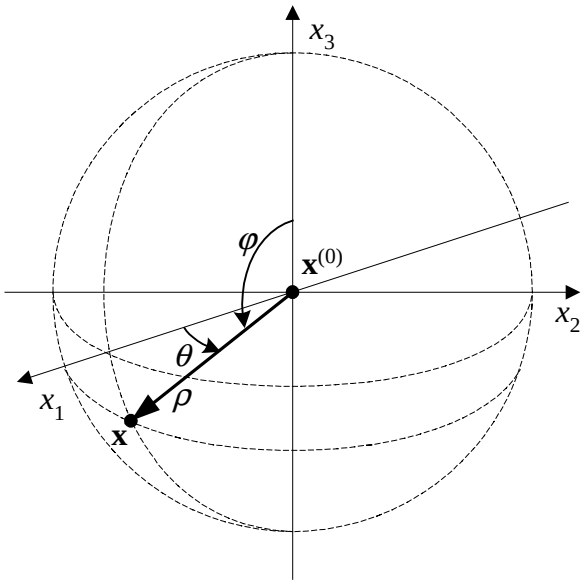


Figure 4.8: Spherical coordinates system.

Radial search The spherical system of coordinates is presented in Fig. 4.8. $\mathbf{x}^{(0)}$ is the center of the spherical coordinates (the same as the center of the radial search). Coordinates of a point $\mathbf{x}$ are represented by a triple (θ, φ, ρ), where θ is an angle between vector $\vec{\mathbf{r}} = \mathbf{x} - \mathbf{x}^{(0)}$ and axis x_1, φ is an angle between vector $\vec{\mathbf{r}}$ and axis x_3, and ρ is the length of vector $\vec{\mathbf{r}}$.

$$\begin{aligned} x_1 &= x_1^{(0)} + \rho\ \sin(\varphi)\ \sin(\theta) \\ x_2 &= x_2^{(0)} + \rho\ \sin(\varphi)\ \cos(\theta) \\ x_3 &= x_3^{(0)} + \rho\ \cos(\varphi) \end{aligned} \tag{4.21}$$

The search is performed along the vector $\vec{\mathbf{r}}$ for ρ ranging from 0 to some maximum value ρ_{max}. Angle θ is changed in the full range form 0^o to 360^o, and angle φ from -135^o to $+135^o$.

Two mappings are created during the search: $I_{MAX}(\varphi, \theta)$ and $I_{TOT}(\varphi, \theta)$. The MAX mapping contains the maximum value found along vector $\vec{\mathbf{r}}$ for fixed φ and θ. The TOT

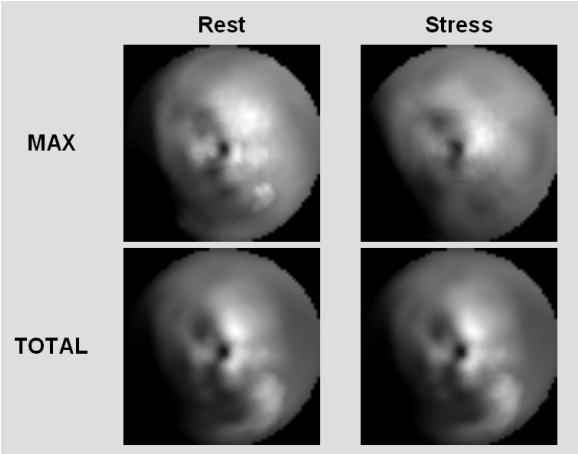


Figure 4.9: Bull's-eye images created by spherical unwrapping of the left ventricle.

mapping contains the integral of values found along vector $\vec{\mathbf{r}}$ for fixed φ and θ.

$$\begin{aligned} I_{MAX}(\varphi,\theta) &= \max_{\rho\in[0,\rho_{max}]} I(\varphi,\theta,\rho) \\ I_{TOT}(\varphi,\theta) &= \int_0^{\rho_{max}} I(\varphi,\theta,\rho)\, d\rho \end{aligned} \tag{4.22}$$

To make the visualization of the radial search results more intuitive, mappings $I_{MAX}(\varphi,\theta)$ and $I_{TOT}(\varphi,\theta)$ are transformed to (x_1,x_2) coordinates forming two round images.

$$\begin{aligned} x_1 &= \varphi\cos(\theta) + x_1^{(0)} \\ x_2 &= \varphi\sin(\theta) + x_2^{(0)} \end{aligned} \tag{4.23}$$

In the $I_{MAX}(x_1,x_2)$ and $I_{TOT}(x_1,x_2)$ the center of an image represents the apex of the left ventricle, edges of an image are near the base of the left ventricle. An example of $I_{MAX}(x_1,x_2)$ and $I_{TOT}(x_1,x_2)$ for REST and STRESS SPECT images created during our experiments is presented in Fig. 4.9.

Cylindrical Radial Search

Cylindrical radial search, or cylindrical unwrapping, is similar to the spherical radial search except it is performed in the cylindrical coordinate system, see Fig. 4.10. The following

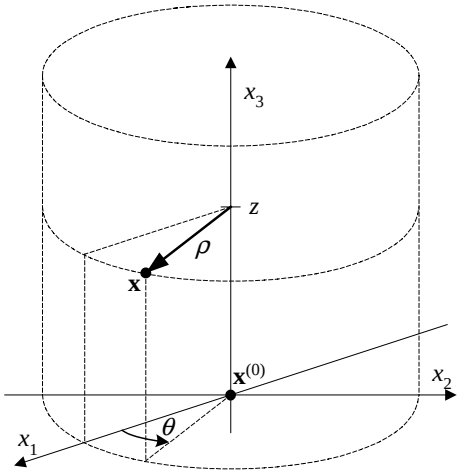


Figure 4.10: Cylindrical coordinate system.

formulas define transformation from cylindrical to cartesian coordinates.

$$\begin{aligned} x_1 &= x_1^{(0)} + \rho\ \sin(\theta) \\ x_2 &= x_2^{(0)} + \rho\ \sin(\theta) \\ x_3 &= x_3^{(0)} + z \end{aligned} \tag{4.24}$$

The MAX and TOT mappings are calculated in a fashion similar to the spherical search.

$$\begin{aligned} I_{MAX}(z,\theta) &= \max_{\rho\in[0,\rho_{max}]} I(z,\theta,\rho) \\ I_{TOT}(z,\theta) &= \int_0^{\rho_{max}} I(z,\theta,\rho)\ d\rho \end{aligned} \tag{4.25}$$

We do not convert these mappings to (x_1, x_2) coordinates, as it is done in bull's eye methods (Van Train et al., 1994). We use them directly for the feature extraction since conversion introduces additional interpolation errors.

4.4.2 Radial Search and Location of 22 3D ROIs

Radial search transforms 3D SPECT images creating 2D maps. The objective is to remove irrelevant information from 3D images and present the relevant information in a simpler 2D form.

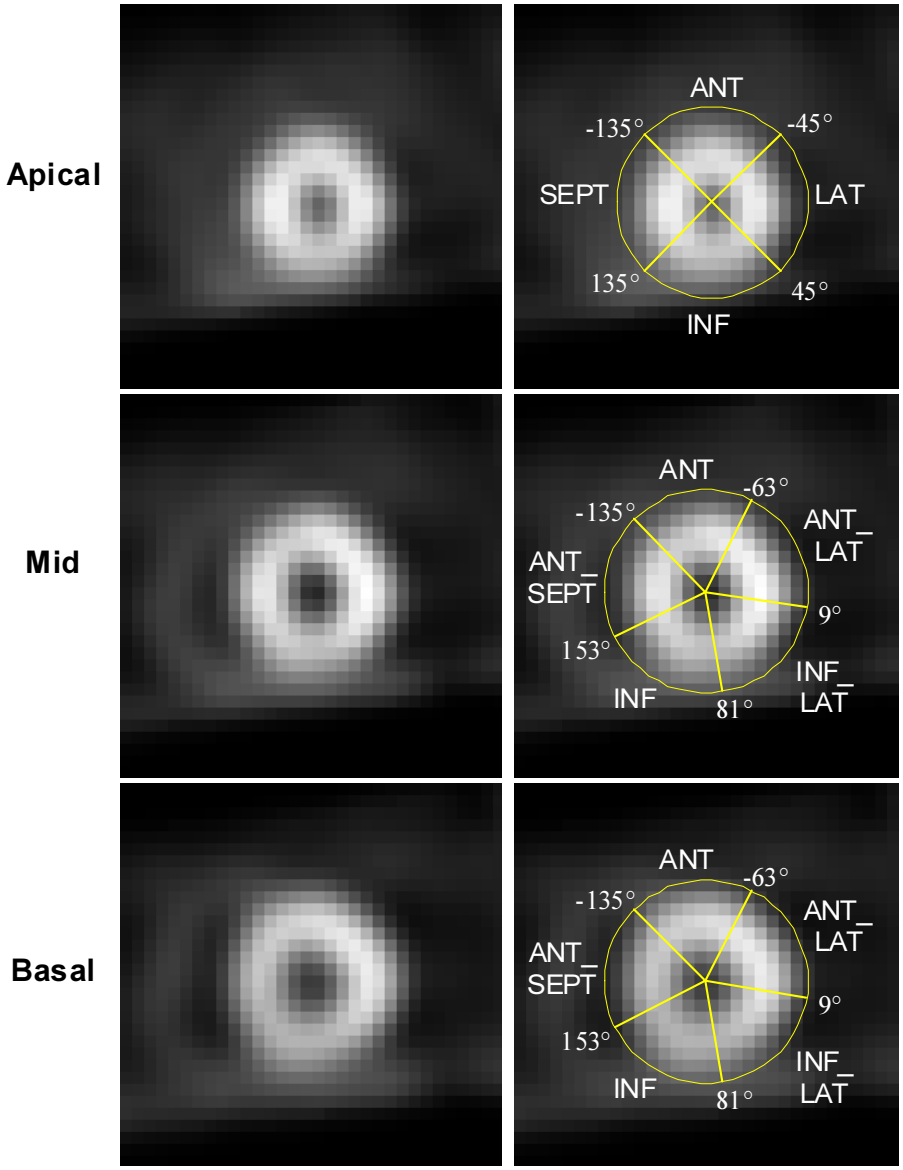
Short Axis Views

The cylindrical search is well suited for detection of left ventricle walls in the short axis views (Fig. 4.1). The walls in these views are roughly cylindrical in shape. We can interpret each of the short axis ROIs, in 3D images, as a wedge. These wedges can be one or more slices in thickness. Fig 4.11 (a) and (b) shows our selection of angles that define the wedges in 3D images. The map created by the spherical radial search is a rectangular image, I_{MAX} or I_{TOT}. Each of the wedges corresponds to a rectangular area in that image, as shown in Fig 4.11 (c). A single image contains data for the apical, mid, and basal views. Angle θ is changing in the full range of 360^o starting at -135^o and finishing at 225^o. The center of search and range for z and ρ depends on the model of the normal left ventricle used. Center of search is located at the center of the left ventricular cavity. z typically spans nine slices, three for each of the views.

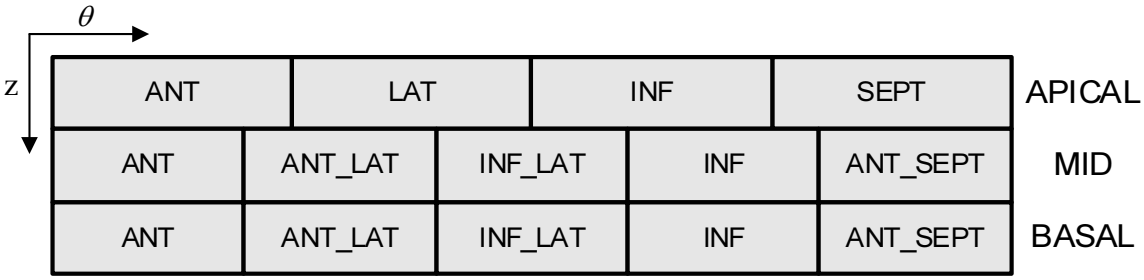
Long Axis Views

We also use radial spherical search for detection of left ventricle walls in long axis views. The left ventricle walls are roughly cylindrical in shape in these views, as shown in Fig. 4.12 and 4.13. Cylindrical search maps are created separately for the horizontal long and horizontal short axis views. Angles θ defining the ROIs and location of the center of search for the horizontal long axis are presented in Fig. 4.12. θ ranges from -235^o to 80^o. Angles θ defining the ROIs and location of the center of search for the vertical long axis are presented in Fig. 4.13. θ ranges from -160^o to 160^o.

Note that we follow the common convention in image processing where pixels coordinates increase from top to bottom and from left to right.

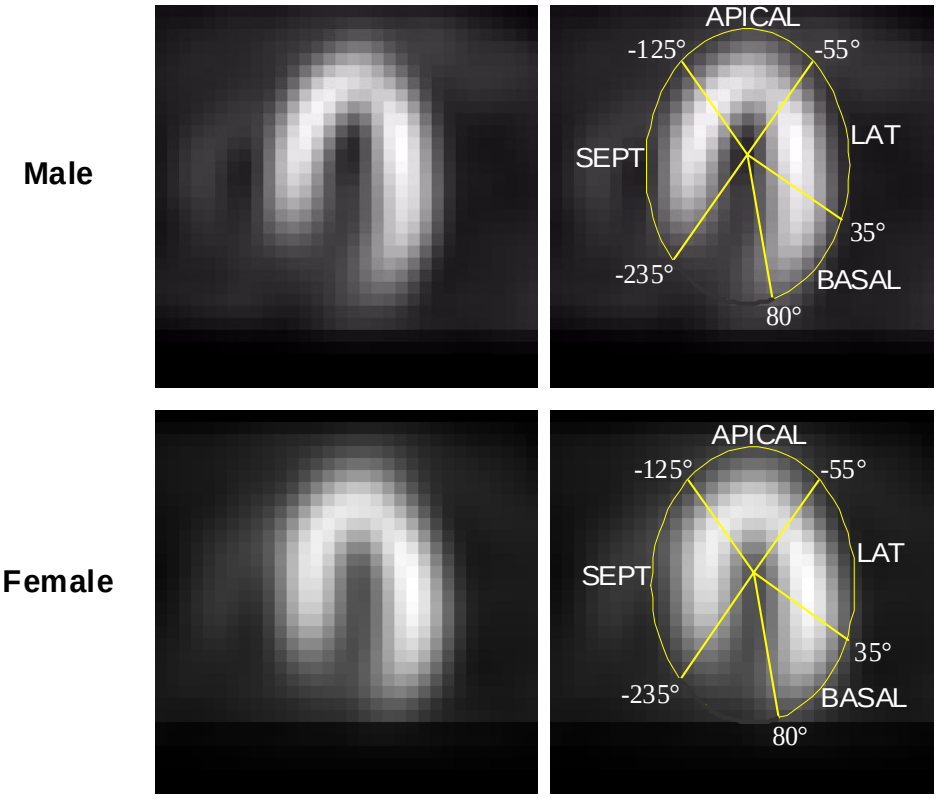


(a) Definitions of ROIs superimposed on male models of normal left ventricle.

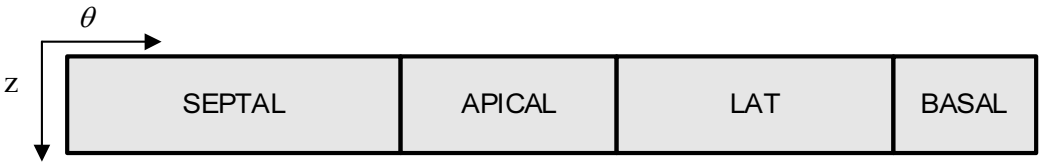


(b) ROIs in the cylindrically unwrapped image are the same for I_{MAX} and I_{TOT}.

Figure 4.11: Regions of interest for cylindrical radial search – short axis views.



(a) Definitions of ROIs superimposed on models of normal left ventricle.

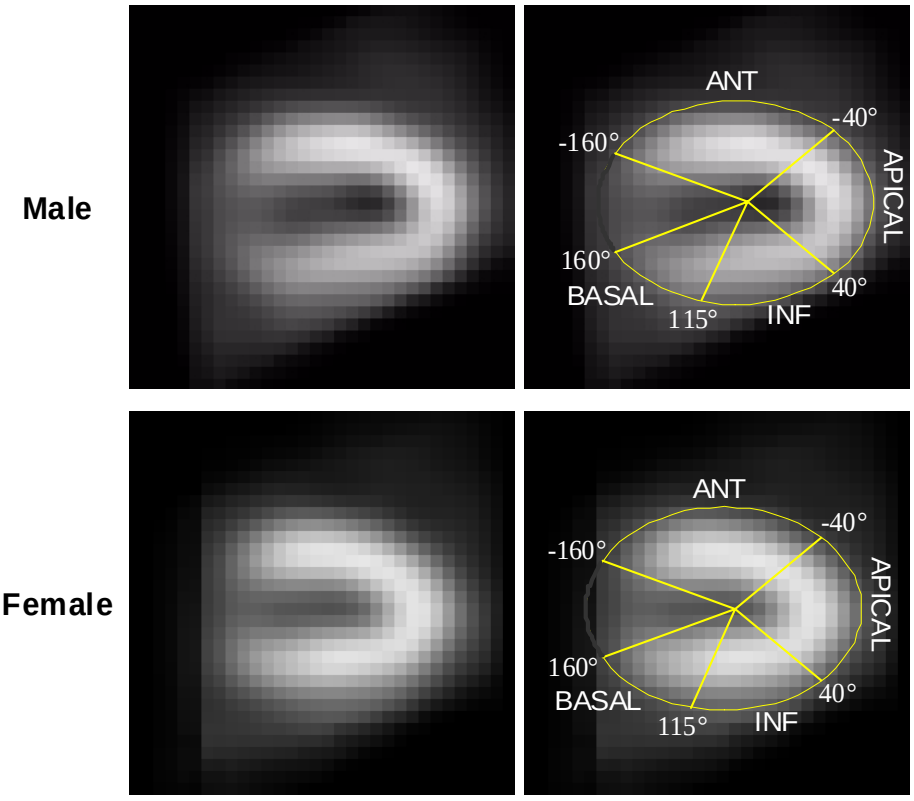


(b) ROIs in the cylindrically unwrapped image are the same for I_{MAX} and I_{TOT}.

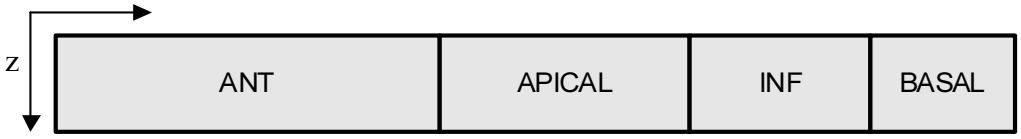
Figure 4.12: Regions of interest for cylindrical radial search – horizontal long axis view.

4.4.3 Feature Extraction from Radial Search Maps

We use the radial search maps I_{MAX} and I_{TOT} created for 3D rest and stress study for feature extraction. Each of the maps is partitioned into regions corresponding to 22 ROIs, as described in the previous section. For each partition, we calculate the maximum, mean, median, and standard deviation of pixel values in that partition. This way we have 16 features extracted in each ROI. Features, after extraction, have been stored in dedicated tables in the SPECT database. Feature records also contain information about normal left



(a) Definitions of ROIs superimposed on models of normal left ventricle.



(b) ROIs in the cylindrically unwrapped image are the same for I_{MAX} and I_{TOT}.

Figure 4.13: Regions of interest for cylindrical radial search – vertical long axis view.

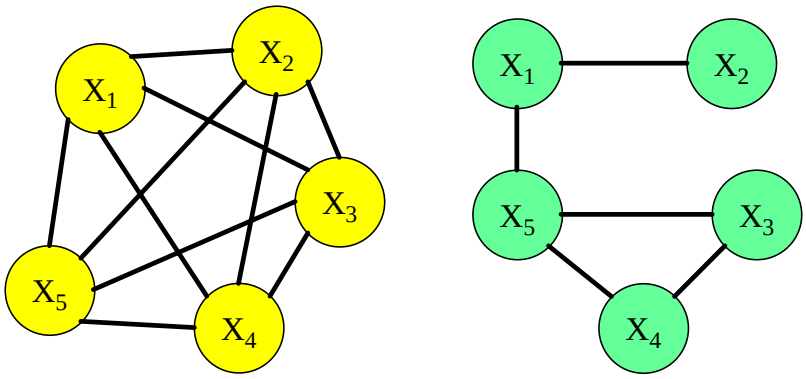
ventricle models used for image registration, registration transformation, and parameters used for creation of I_{MAX} and I_{TOT} images. We use this information to prepare data for perfusion classification experiments described in Chapter 8.

Chapter 5

Bayesian Networks

Almost any practical artificial intelligence application requires dealing with uncertainty. Diagnosis of the left ventricle perfusion is a very good example. The input data, cardiac SPECT images, are intrinsically noisy. The noise is due to technical limitation (low resolution of the detector cameras, quality of the 3D reconstruction algorithms), safety considerations (dosage of the radioactive trace that a patient is injected with cannot be arbitrarily large resulting in reduced image signal-to-noise ratio), and anatomical differences between patients (organ size and shape, distribution of the radioactive trace after injection, artifacts due to an organ diffusing or absorbing diagnostic photons). Our output data, physician's diagnosis, is to a large extent subjective and difficult to quantify.

Until recently, application of a strict mathematical approach to reasoning under uncertainty was considered impractical. This was mostly due to the problem of computing the joint probability distribution of large number of random variables involved in reasoning. The last decade has seen significant theoretical advances and increasing interest in *graphical models*. A graphical model is a way of representing dependency relationships within a set of random variables. Random variables are represented by nodes in a graph. An arc in the graph intuitively corresponds to a dependency relationship between two variables. The lack of an arc can be intuitively interpreted as a lack of dependency between two variables,



(a) Fully connected graphical model. Each variable depends on every other one.

(b) Graphical model demonstrating limited dependency among random variables.

Figure 5.1: Examples of graphs representing graphical models.

see Fig. 5.1. This intuitive graphical interpretation of dependencies between variables is one of the reasons for popularity of graphical models. Other reasons for their popularity are the significant progress in theory and algorithms for inference, and advances in learning structure and parameters of graphical models (Jordan, 1998). Most importantly, there is a considerable number of successfully tested applications of graphical models (Haddawy, 1999).

One of the most popular types of graphical models are *Bayesian networks*. The main characteristic differentiating them from other graphical models is that arcs in the network are directed, representing conditional dependence among variables. The name comes from the fact that most theory relevant to Bayesian networks is bases on Bayesian probability. The remaining material in this section presents overview of topics relevant to the use and learning of Bayesian networks.

5.1 Basic Concepts

Notation Random variables are denoted by capital letters, e.g. X. Values of these variables are denoted by small letters, x. Bold letters denote sets of variables, $\mathbf{X} = \{X_1, \ldots, X_n\}$, or variable values $\mathbf{x} = \{x_1, \ldots, x_n\}$. ξ denotes background knowledge.

Bayes theorem The conditional probability of a random variable a, given a random variable b can be calculated as follows:

$$p(a|b) = \frac{p(b|a)p(a)}{p(b)} \tag{5.1}$$

Chain rule Joint probability of $\mathbf{x}$ can be expressed as a product of conditional probabilities:

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i|x_1, \ldots, x_{i-1}) \tag{5.2}$$

Bayesian Network A Bayesian network for a set of random variables $\mathbf{X} = \{X_1, \ldots, X_n\}$ is a pair $B = \langle S, P \rangle$, where S is a directed acyclic graph (DAG) whose nodes are in one-to-one correspondence with random variables in $\mathbf{X}$. P is a set of local probability distributions associated with each variable. X_i denotes both the variable and its corresponding node in S. We use $\mathbf{Pa}_i$ to denote parents, and $\mathbf{pa}_i$ to denote configuration of parents of node X_i in S as well as variables corresponding to these parents. The joint probability represented by the structure S is given by:

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i|\mathbf{pa}_i) \tag{5.3}$$

The local probability distributions P are the distributions corresponding to the terms in Eq. (5.3). An example of a Bayesian network is presented in Fig.5.2. The probability distribution represented by this network is:

$$p(x_1, x_2, x_3, x_4, x_5) \;=\; p(x_1|x_2, x_5) \cdot p(x_2) \cdot p(x_3|x_5) \cdot p(x_4|x_3, x_5) \cdot p(x_5). \tag{5.4}$$

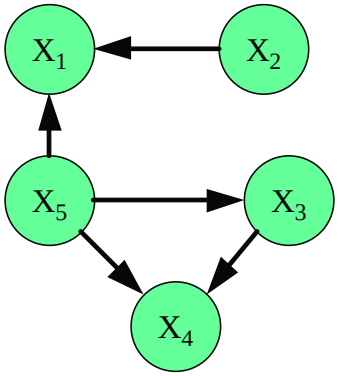


Figure 5.2: Example of a Bayesian network graph.

5.2 Inference

Because a Bayesian network for $\mathbf{X}$ determines a joint probability distribution for X, we can – in principle – use the Bayesian network to compute any probability of interest. For instance:

$$p(x_1|x_2,\ldots,x_n) = \frac{p(x_1,\ldots,x_n)}{\int p(x_1,\ldots,x_n)dx_1} \tag{5.5}$$

For problems with many variables this direct approach is not practical. Some more efficient methods of exact and approximate inference in Bayesian networks have been established. The following algorithms for exact inference in networks with discrete variables exist:

- A Bayesian network is first transformed into a tree where each node corresponds to a subset of variables in $\mathbf{X}$. The algorithm then exploits mathematical properties of this tree to perform probabilistic inference. (Lauritzen and Spiegelhalter, 1988; Jensen, Lauritzen, and Olesen, 1990; Dawid, 1992). A good discussion of practical issues involved in its implementation is presented in (Huang and Darwiche, 1994). This is the most commonly used algorithm.

- The arcs in the network structure are being reverced until the answer to the given probabilistic query can be read directly from the graph. Each arc reversal corresponds

to an application of the Bayes' theorem (Howard and Matheson, 1983; Omstead, 1983; Shachter, 1988).

- A message passing scheme that updates the probability distributions for each node in a Bayesian network in response to observations of one or more variables (Pearl, 1986).

- Symbolic simplification of sums and products (D'Ambrosio, 1991).

Exact inference in networks with continuous distributions have been studied by Shachter and Kenley (1989) for multivariate-Gaussian distributions, and Lauritzen (1992) for Gaussian-mixture distributions.

Exact inference in Bayesian networks is NP-hard [1] (Cooper, 1990; Heckerman, 1996). The problem is due to undirected cycles that may be present in a Bayesian network. Approximate inference in Bayesian networks is a topic of current research. Existing approaches include: pruning the model and performing exact inference on the reduced model (Kjærulff, 1993), cutting loops and bounding the incurred error (Draper and Hanks, 1994), variational methods to bound the node probabilities in sigmoidal belief networks (Jaakkola, 1997; Jordan, Ghahramani, Jaakkola, and Saul, 1998).

[1] *NP*-hard means *non-polynomial*-hard. Generally, algorithms are considered tractable when the time needed to execute the algorithm is a polynomial of the number of parameters, e.g., number of nodes in the network (Even, 1979). In particular, this function can be linear in the number of parameters. When there are no algorithms to solve a given problem so that the time needed to execute that algorithm grows no faster then according to some polynomial function of parameters, the problem is called NP-hard. For instance, a problem for which the most efficient algorithm has the execution time exponential with the number of parameters is NP-hard.

5.3 Learning Parameters

In this section, we assume that the structure of a Bayesian network is known (learning of the network structure will be described in the next section), all data is complete, and there are no hidden variables. Let S^h denote the hypothesis that the joint probability of $\mathbf{X}$ can be factored according to structure S. Let $\boldsymbol{\theta}_S = \{\theta_1, \ldots, \theta_n\}$ be a set where θ_i is the vector of parameters for the local distribution function $p(x_i|\mathbf{pa}_i, \theta_i, S^h)$ [2]. Now we can write the joint probability distribution as

$$p(\mathbf{x}|\boldsymbol{\theta}_S, S^h) = \prod_{i=1}^{n} p(x_i|\mathbf{pa}_i, \theta_i, S^h) \tag{5.6}$$

The problem of learning parameters in a Bayesian network is that of computing the posterior distribution $p(\boldsymbol{\theta}_S|D, S^h)$.

Assume that parameters θ_i are mutually independent, then

$$p(\boldsymbol{\theta}_S|S^h) = \prod_{i=1}^{n} p(\theta_i|S^h) \tag{5.7}$$

Under assumption of complete data and parameter independence, the parameters remain independent given the random sample:

$$p(\boldsymbol{\theta}_S|D, S^h) = \prod_{i=1}^{n} p(\theta_i|D, S^h) \tag{5.8}$$

In other words, each of the parameters θ_i can be computed independently of others.

Learning parameters from complete data is discussed in (Spiegelhalter and Lauritzen, 1990). A more recent discussion can be found in (Buntine, 1994). Computation of parameters θ_i is most convenient for distributions in the exponential family and conjugate priors. Heckerman (1996) discusses parameters for unrestricted multinomial distribution, details of their computation will be presented in Section 7.3. Calculations for linear regression with Gaussian noise are in (Buntine, 1994; Heckerman and Geiger, 1995).

[2] Local distribution function $p(x_i|\mathbf{pa}_i, \theta_i, S^h)$ is a probabilistic classification or regression function. A Bayesian network can be viewed as a collection of probabilistic classification/regression models (Heckerman, 1996).

5.4 Learning Structure

5.4.1 Strict Bayesian Approach

Let us assume that data is complete and there are no hidden nodes; define a discrete variable representing possible hypothesis about network structure S^h and assign probabilities $p(S^h)$. The task of learning the network structure using Bayesian approach is that of computing the posterior distribution $p(S^h|D)$. In principle $p(S^h|D)$ can be computed using Bayes theorem:

$$p(S^h|D) = \frac{p(S^h)p(D|S^h)}{p(D)} \tag{5.9}$$

Parameters of the network are learned as described above by computing posterior distribution $p(\boldsymbol{\theta}_S|D, S^h)$.

When we assume that hypotheses S^h are mutually exclusive we can compute the joint probability distribution of the unobserved case $\mathbf{x}_{N+1}$ given training set D as follows:

$$p(\mathbf{x}_{N+1}|D) = \sum_{S^h} p(S^h|D) \int p(\mathbf{x}_{N+1}|\boldsymbol{\theta}_S, S^h)p(\boldsymbol{\theta}_S|D, S^h)d\boldsymbol{\theta}_S \tag{5.10}$$

Although mathematically correct, the full Bayesian approach is not practical. The problem is with summations like the one over S^h in Eq. (5.10). The number of possible structures increases more than exponentially with the number of variables (network nodes).[3]

In practice, the sum in Eq. (5.10) is approximated in some manner. Typically this approximation is done using only a single model. This model is selected, as representative, from among all of the possible models based on the assumption that posterior distribution $p(S^h|D)$ has a single narrow peak for that selected model. There are two approaches to model selection. In the first, designated here as *search & scoring*, some criterion is used to measure the degree to which a network structure fits the prior knowledge and data; a search

[3]This number can be determined for a given number of nodes n using a function published by (Robinson, 1977). Even for a number of nodes as small as 10, the number of possible nodes is approximately 4.2×10^{18} (Krause, 1998).

method is used to find a model that best fits the data. In the second approach, designated here as *dependency analysis*, the dependency relationships are measured locally using some kind of conditional independence test, such as the χ^2 test or a mutual information test. Generally, the first category of model selection algorithms has less time complexity in the worst case (when the underlying DAG is densely connected), but it may not find the best solution due to its heuristic nature. The second category of algorithms is usually asymptotically correct when the probability distribution of data satisfies certain assumptions, but conditional independence tests with large condition-sets may be unreliable unless the volume of the data is enormous (Cheng, Bell, and Liu, 1997b; Cooper and Herskovits, 1992).

Some researches approximate the sum in Eq. (5.10) by more then a single model using the *selective model averaging*. A manageable number of "good" models is selected and pretended that these models are exhaustive. This approach is much more complex than the model selection. It is advantageous to identify network structures that are significantly different, so that they will represent the whole distribution of models (Krause, 1998). The difficulty is in finding these *diverse* representatives of networks structures. Selective model averaging is discussed in (Buntine, 1991b; Heckerman, Geiger, and Chickering, 1995; Madigan and Raftery, 1994). In the following we will concentrate on the model selection methods.

A number of techniques for learning is presented in (Buntine, 1994). More recently Jordan collected a number of articles related to learning Bayesian networks and graphical models in general (Jordan, 1998).

5.4.2 Model Selection by Search & Scoring Approach

This section presents some examples of quality measures used for scoring network structures, and algorithms that can be used to search through the space of possible network structures.

Bayesian Quality Measures

Bayesian quality measures rely on Bayesian statistics: Bayes' theorem and conjugacy in particular. [4] The basic idea of the Bayesian quality measures is to assign to every network a quality value that is function of the posterior probability distribution $p(S_h|D)$. A frequently used criterion is the log of the posterior probability:

$$\log p(S^h|D,\xi) = \log p(S^h|\xi) + \log p(D|S^h,\xi) \tag{5.11}$$

The logarithm is used for the numerical convenience. This criterion has two components: the log prior and the log marginal likelihood. An equivalent criterion often used is:

$$\log\left(\frac{p(S^h|D,\xi)}{p(S_0^h|D,\xi)}\right) = \log\left(\frac{p(S^h|\xi)}{p(S_0^h|\xi)}\right) + \log\left(\frac{p(D|S^h,\xi)}{p(D|S_0^h,\xi)}\right) \tag{5.12}$$

The ratio $\frac{p(D|S^h,\xi)}{p(D|S_0^h,\xi)}$ is known as *Bayes' factor*. S_0^h is some selected reference hypothesis.

The Bayesian score was originally discussed in (Cooper and Herskovits, 1992) and further developed in (Buntine, 1991b; Heckerman et al., 1995). Two, alternative Bayesian quality measures, derived using two different sets of assumptions, will be discussed in Section 7.5.

Minimum Length Encoding Measures

This concept comes from *theory of coding*, where a string is encoded with as few bits as possible. The score is based on the Minimal Description Length principle of Rissanen (1989); the application of this principle to Bayesian networks was developed by several

[4] Unless the prior family of distributions is carefully selected, the resulting posterior probabilities may not belong to the same families and mathematical treatment gets considerably complicated. If the prior and posterior distributions are in the family of distributions for a given sampling process, we say that they are *natural conjugate* to the given sampling process. That is, if a sample is used to update a prior probability distribution in a conjugate family, the resulting posterior distribution will also belong to the same family. The benefits of conjugacy include mathematical tractability; we obtain closed formulas for important quantities used in Bayesian inference and learning.

authors (Bouckaert, 1994; Lam and Bacchus, 1994; Suzuki, 1993). Proponents of MDL measure argue that using only the posterior probability $p(S^h|D)$ (Bayesian quality measure) results is a criterion that prefers networks with complete graphs, thus a factor penalizing for the size of the network is added to the criterion. However, the use of the MDL score for construction of Bayesian networks has been criticized by Friedman et al. (1997). These authors argue, with support of a theoretical and an empirical study, that the use of MDL may result in networks with a limited number of conditional dependencies leading to a poor approximation of the joint or marginal probability distributions.

Information Theoretic Measures

Another way of measuring the quality of a network is by the information measures. These measures can be seen as a generalization of the MDL measures. The most well know measures are:

MLIC *Maximum likelihood information criterion*. The measure is the log likelihood of a Bayesian network given the training data. Unlike other information theoretic measures, it does not contain a penalty for the size of the network. It is equivalent to Bayesian quality measures.

$$q(B, D) = LL(B|D)$$

AIC *Akaike information criterion* (Akaike, 1974). This measure is not consistent: in the large sample limit, the true model may not be among these receiving maximal scores (Schwarz, 1978).

$$q(B, D) = LL(B|D) - \|B\|$$

BIC *Bayesian information criterion* also known as *Schwarz information criterion* (Schwarz, 1978; Heckerman, 1996). BIC is easy to use and does not require evaluation of prior distributions. Consequently, it can be a practical criterion to use in the appropriate

circumstances (Kass and Raftery, 1995). BIC approximates posterior probabilities in large samples (Glymour, Madigan, Pregibon, and Smyth, 1997). When applied to networks with unrestricted multinomial distributions of variables and Dirichlet priors on parameters, BIC leads to the same criterion as MDL – differing only by a minus sign.

$$q(B, D) = LL(B|D) - \frac{logN}{2} \|B\|$$

Model Search

K2 Algorithm Cooper and Herskovits (1992) describe the use of a greedy search algorithm, K2, for identifying the most probable structure given some of the test data. The algorithm assumes that ordering of nodes is given. It starts with an empty network and iterates through each of X_i, according to their ordering. For each X_i it considers all nodes that could be added to the existing set of parent nodes $\mathbf{pa}_i$. The candidate node that maximizes the local scoring function q_i is selected. If addition of this node to the current set of parents of X_i increases the local scoring function, it is added to $\mathbf{pa}_i$ and the search for the next parent of X_i continues. If the addition of that node does not increase the scoring function, it is assumed that all parents of X_i are found and the algorithm starts to search for parents of X_{i+1}. Pseudocode for the K2 algorithm is shown in Alg. 5.1 (Castillo, Gutiérrez, and Hadi, 1997).

Buntine's Algorithm An algorithm that does not require node ordering has been proposed by Buntine (1991b). It starts with an empty parents set. At each step a new link is added that does not lead to a cycle and maximizes the quality increment. The process is repeated until no more increase of the quality is possible or a complete network is attained. Pseudocode for this algorithm is presented in Fig. 5.2 (Castillo et al., 1997).

Algorithm 5.1 K2 algorithm of Bayesian network construction.

{*Initialization Step*}

1. **for** $i \leftarrow 1$ **to** n **do**
2. $\quad \mathbf{pa}_i \leftarrow \varnothing$

{*Iteration Step*}

3. **for** $i \leftarrow 1$ **to** n **do**
4. **repeat**
5. $\quad$ Select $Y \in (\{X_1, \ldots, X_{i-1}\} \setminus \mathbf{pa}_i)$ that maximizes $g = q_i(\mathbf{pa}_i \cup \{Y\})$
6. $\quad \delta \leftarrow (g - q_i(\mathbf{pa}_i))$
7. $\quad$ **if** $\delta > 0$ **then**
8. $\quad\quad \mathbf{pa}_i \leftarrow (\mathbf{pa}_i \cup \{Y\})$
9. **until** $(\delta \leq 0)$ **or** $(\mathbf{pa}_i = \{X_1, \ldots, X_{i-1}\})$

CB Algorithm Singh and Voltara (1995) proposed an extension to K2 algorithm they called CB. The CB algorithm uses conditional independence tests to generate a "good" node ordering from the data, and then uses K2 algorithm to generate the Bayesian network from set of training samples D using this node ordering. Starting with a complete, undirected graph on all variables, the CB algorithm first deletes the edges between adjacent nodes that are unconditionally independent (conditional independence test of order 0). CB orients the edges in the resulting graph and obtains a total ordering on the variables. It passes this ordering to the K2 algorithm to construct the corresponding network. The algorithm then repeats this process by removing edges (from the undirected graph obtained in the previous iteration) between adjacent edges that are conditionally independent given one node (conditional independence test of order 1). CB keeps constructing the network increasing the order of conditional independence tests as long as the predictive accuracy of the resultant network keeps increasing.

Algorithm 5.2 BUNTINE'S algorithm of Bayesian network construction.

{*Initialization Step*}

1. **for** $i \leftarrow 1$ **to** n **do**
2. $\quad \mathbf{pa}_i \leftarrow \varnothing$
3. **for** ($i \leftarrow 1$ **to** n) and ($j \leftarrow 1$ **to** n) **do**
4. $\quad$ **if** $i \neq j$ **then**
5. $\quad\quad A[i,j] \leftarrow (m_i(X - j) - m(\varnothing))$
6. $\quad$ **else**
7. $\quad\quad A[i,j] \leftarrow -\infty \qquad \{$*Prevent edge* $X_i \rightarrow X_i\}$

{*Iteration Step*}

8. **repeat**
9. $\quad$ select i,j that maximize $A[i,j]$
10. $\quad$ **if** $A[i,j] > 0$ **then**
11. $\quad\quad \mathbf{pa}_i \leftarrow (\mathbf{pa}_i \cup \{X_j\})$
12. $\quad\quad$ **for** $X_a \in Pred_i$, $X_b \in Desc_i$ **do**
13. $\quad\quad\quad A[a,b] \leftarrow -\infty \qquad \{$*Prevent loops*$\}$
14. $\quad\quad$ **for** $k \leftarrow 1$ **to** n **do**
15. $\quad\quad\quad$ **if** $A[i,k] > -\infty$ then
16. $\quad\quad\quad\quad A[i,k] \leftarrow (m_i(\mathbf{pa}_i \cup \{X_k\}) - m_i(\mathbf{pa}_i))$
17. **until** ($A[i,j] \leq 0$) **or** ($A[i,j] = -\infty$), $\forall\, i,j$

Heckerman et al. (1995) discuss and evaluate learning structure of Bayesian networks using hill-climbing and other variants of greedy search. Empirical studies of search algorithms for learning structure on Bayesian networks can be found in (Aliferis and Cooper, 1994; Chickering, 1996; Spites and Meek, 1995). See (Chickering, 1996) for search over equivalence network class.

5.4.3 Model Selection by Dependency Analysis

Learning structure of Bayesian networks by dependency analysis is based on performing conditional independence tests on subsets of edges in the network graph. Two types of tests are typically used: statistical tests and information theoretic tests. Statistical approaches based on χ^2 tests have been used in (Spites, Glymour, and Scheines, 1991; Wermuth and Lauritzen, 1983). The use of information theoretic tests has been investigated by (Cheng, Bell, and Liu, 1997a; Cheng et al., 1997b). The drawback of using conditional independence tests is that they require large data sets when condition-sets are large.

Learning Tree Structures

Computation of conditional independence tests can be quite efficient when condition-sets are small. We can restrict size of the condition-sets by putting constrains on the graph representing the Bayesian network. Chow and Liu (1968) proposed to represent the conditional dependence between random variables $\{X_1, \ldots, X_n\}$ by a tree. A directed acyclic graph on $\{X_1, \ldots, X_n\}$ is a *tree* if each variable X_i has exactly one parent, except for one variable that has no parent (this variable is referred to as a *root*). An example of a Bayesian network that has a tree graph structure is presented in Fig. 5.3.

Chow and Liu (1968) weighted edges in the full graph using mutual information criterion:

$$I(x_i, x_j) = \sum_{x_i, x_j} p(x_i, x_j) \ \log \left(\frac{p(x_i, x_j)}{p(x_i)p(x_j)} \right) \tag{5.13}$$

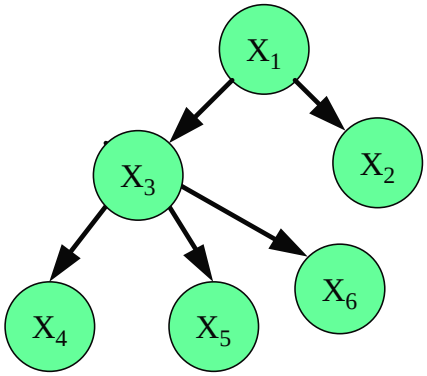


Figure 5.3: Bayesian network that has a tree graph structure.

Then they find a maximum spanning tree (Cormen, Leiserson, and Rivest, 1990) in that full graph. (Chow and Liu, 1968) also demonstrated that the joint probability distribution represented by the tree, constructed using their method, is an optimal approximation of the true probability distribution (under assumption that the true distribution represents a tree dependence among the variables). Despite restrictions imposed on the dependency among random variables, this is an attractive method of Bayesian network construction due to its low computational complexity $(O(n^2 \cdot N))$ and its optimality properties.

Chapter 6

Bayesian Network Classifiers

A Bayesian network can be used for classification in a quite straightforward way. One of the variables is selected as a class variable, and the remaining variables as attribute variables. Inference methods presented in Section 5.2 can be used to calculate marginal distribution of the class variable, see Fig. 6.1.

In general, we could use any of the Bayesian network structure learning methods presented in Chapter 5. Most of these methods are aimed at approximating the joint distribution of the set of random variables $\{X_1, \ldots, X_n\}$. Classification, however, uses only the marginal distribution of the class variable, which suggests the use of more targeted Bayesian network learning methods. This section presents the most important results re-

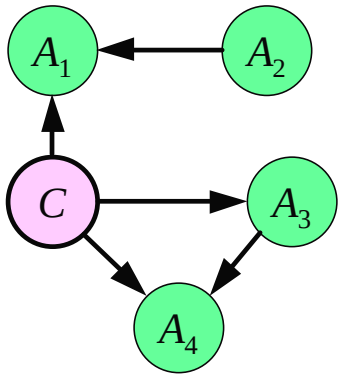


Figure 6.1: Bayesian network classifier.

lated to construction of Bayesian networks for classification.

We will be using the following notation. $\mathcal{D} = \{\mathcal{D}_l\}_{l=1,\ldots,N}$ is a set of N training examples, $F = \{F_k\}_{k=1,\ldots,M}$ is a set of M testing cases. $\mathcal{D}_l = \{a^{(l)}, c^{(l)}\}$ where $a = \{x_1, \ldots, x_{n-1}\}$ is the configuration of attribute variables, and $c = x_n$ is the configuration of the class variable. We also denote by $\boldsymbol{c} = \{c^{(1)}, \ldots, c^{(N)}\}$ and $\boldsymbol{a} = \{a^{(1)}, \ldots, a^{(N)}\}$ sets of class values and attribute values, respectively, corresponding to the training cases $\mathcal{D} = [\boldsymbol{a}, \boldsymbol{c}]$.

6.1 Bayesian Approach to Classification

A classification problem is that of assigning a correct class label to a set of attributes. The *classification error rate*, or *error of prediction*, is the most commonly used measure of classification quality:

$$\varepsilon = \frac{1}{M} \sum_{k=1}^{M} (\hat{c}^{(k)} - c^{(k)}) \tag{6.1}$$

where $\hat{c}^{(k)}$ denotes the correct class and $c^{(k)}$ is the predicted class for the k^{th} case.

If the goal is to minimize error in prediction, the decision theory says we should choose class c' to maximize the posterior class probability $p(c'|a', \boldsymbol{a}, \boldsymbol{c})$. This is the posterior average of the class probabilities predicted for c' from all possible class probability structures S^h:

$$\begin{aligned} p(c'|a', \boldsymbol{a}, \boldsymbol{c}) &= \sum_{S^h} \int_{\boldsymbol{\theta}_S} p(c'|a', S^h, \boldsymbol{\theta}_S)\; p(S^h, \boldsymbol{\theta}_S|\boldsymbol{a}, \boldsymbol{c})\; d\boldsymbol{\theta}_S \\ &= \sum_{S^h} p(S^h|\boldsymbol{a}, \boldsymbol{c})\; \mathbf{E}_{\boldsymbol{\theta}_S|S^h, \boldsymbol{a}, \boldsymbol{c}}(p(c'|a', S^h, \boldsymbol{\theta}_S)) \end{aligned} \tag{6.2}$$

where the summations are over the space of all possible network structures S^h, and

$$p(S^h|\boldsymbol{a}, \boldsymbol{c}) \;\propto\; \int_{\boldsymbol{\theta}_S} p(\boldsymbol{c}|\boldsymbol{a}, S^h, \boldsymbol{\theta}_S)\; p(S^h, \boldsymbol{\theta}_S)\; d\boldsymbol{\theta}_S, \tag{6.3}$$

$$\mathbf{E}_{\boldsymbol{\theta}_S|S^h, \boldsymbol{a}, \boldsymbol{c}}(p(c'|a', S^h, \boldsymbol{\theta}_S)) \;=\; \int_{\boldsymbol{\theta}_S} p(c'|a', S^h, \boldsymbol{\theta}_S)\; p(\boldsymbol{\theta}_S|S^h, \boldsymbol{a}, \boldsymbol{c})\; d\boldsymbol{\theta}_S \tag{6.4}$$

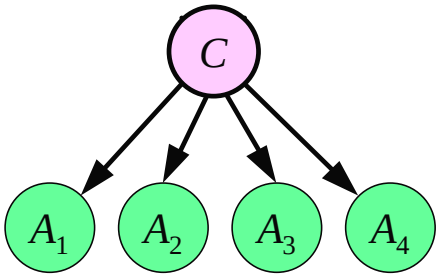


Figure 6.2: Bayesian network representing a naïve Bayes classifier.

Eq. (6.2) simply says to average the class predictions made for each network structure. Where $p(S^h|\boldsymbol{a}, \boldsymbol{c})$, the posterior probability of the network structure S^h, is the weight used in the averaging process. In this formula, $p(S^h, \boldsymbol{\theta}_S)$ is the prior on the space of class probability networks, and $p(\boldsymbol{c}|\boldsymbol{a}, S^h, \boldsymbol{\theta}_S)$ is the likelihood of the training sample.

Note that the classification learning problem given by Eq. (6.2) is similar to the problem of learning network structure given by Eq. (5.10). However, Eq. (5.10) describes the joint probability distribution of a new unobserved case $\mathbf{x}_{n+1} = \{a'_1, \ldots, a'_{n-1}, c'\}$ given the training samples D, while Eq. (6.2) describes marginal distribution of an unknown class c' given known values of attributes $a' = \{a'_1, \ldots, a'_{n-1}\}$ and the training samples D. As with Eq. (5.10), direct use of Eq. (6.2) is not practical due to summation over all possible network structures S^h. Thus, the algorithm design strategies for Bayesian network classifiers are based on designing heuristic procedures to find a single structure or a set of structures that can be used to approximate Eq. (6.2).

6.2 Naïve Bayes Classifier

The naïve Bayes classifier have been popularized by Duda and Hart (1973). Its simplicity, efficiency, and low classification error rate make it one of the most commonly used classifiers. The naïve Bayes has a fixed structure and adjustable parameters. The structure can be represented by a Bayesian network: the class node is a parent to all attribute nodes,

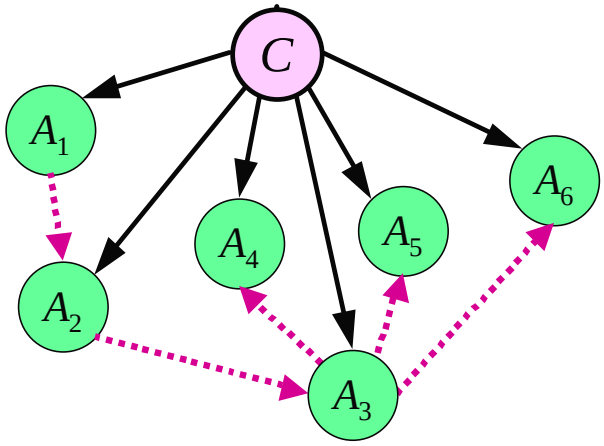


Figure 6.3: Bayesian network representing a tree augmented naïve Bayes classifier.

and there are no edges between the attribute nodes, see Fig. 6.2. In other words, a naïve Bayes classifier assumes that all the attribute variables are conditionally independent given the class variable. Despite that, these assumptions are in most cases unrealistic the naïve Bayes classifier performs in many cases as well as state of the art classifiers. Recently, its properties has been intensively studied, especially by Langley (Kononenko, 1990, 1991; Langley, Iba, and Thompson, 1992; Langley and Sage, 1994; John and Langley, 1995; Langley and Sage, 1999). The naïve Bayes classifier has been an inspiration for a number of other classification approaches. Two of them, that make use of Bayesian networks, are presented below.

6.3 TAN – Tree Augmented Naïve Bayes Classifier

The main drawback of the naïve Bayes classifier is the assumption of conditional independence of attributes. Friedman et al. (1997) proposed a method that introduces dependencies among attributes using the network construction method of Chow and Liu (1968) – it is assumed that dependencies among attribute nodes can be represented by a tree structure, see Fig. 6.3. The TAN algorithm (tree augmented naïve Bayes) has complexity $O(n^3 \cdot N)$ and has been demonstrated by Friedman et al. (1997) to perform as well or better than naïve

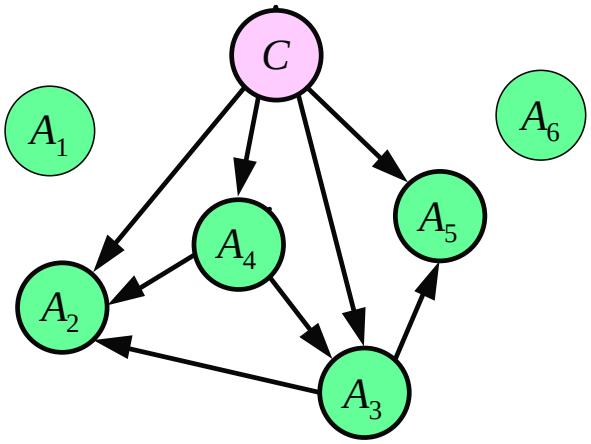


Figure 6.4: Example of a Bayesian network generated by K2-AS algorithm.

Bayes classifier and C4.5 classifier (Quinlan, 1993).

6.4 BAN – Bayesian Network Augmented Naïve Bayes Classifier

This approach is similar to the one in the previous section. TAN augments the naïve Bayes with dependencies among attributes having a tree structure. BAN augments the naïve Bayes with a general Bayesian network of dependencies among the attributes. The network of dependencies among attributes may be constructed using any of the structure learning methods presented in Chapter 5. Unlike TAN, BAN cannot be constructed in a closed form because the problem of construction unrestricted Bayesian networks is NP-hard.

6.5 K2-AS Algorithm

Singh and Provan (1995) combine attribute selection and Bayesian network construction into a single algorithm, called K2-AS. The idea is to remove attributes that may not contribute to classification; to construct a classifier network only from the “best” attributes.

The algorithm consists of two phases:

attribute selection phase In this phase K2-AS chooses the subset of attributes Δ from which the final network is constructed. The algorithm starts with the initial assumption that Δ consists only of the class variable C. It then adds sequentially that attribute whose addition results in the maximum increase in the predictive accuracy, on the set of evaluation cases, of the network constructed from resulting set of attributes. The algorithm stops when the addition of another attribute does not increase predictive accuracy.

network construction phase K2-AS uses the final set of attributes Δ selected in the attribute selection phase to construct a network using training data. This is done by applying the CB algorithm described in Section 5.4.2.

The K2-AS algorithm has a relatively high computational complexity. An example of a network generated by K2-AS is shown in Fig. 6.4.

Chapter 7

Learning Bayesian Network Classifiers: A New Synthesis

Graphical models, and Bayesian networks in particular, provide a powerful mechanism for modeling problem domains that are characterized by a significant amount of noise and uncertainty. Diagnosis of cardiac SPECT images is a perfect example of such a domain. Chapter 5 discussed the general problem of modeling using Bayesian networks. Chapter 6 introduced use of Bayesian networks for classification. This chapter proposes a new approach to learning Bayesian network classifiers; it presents a family of learning algorithms and estimates their complexity. In Chapter 8, we will present results of using these new algorithms for analysis of cardiac SPECT data.

As pointed out in Chapter 6, the problem of learning Bayesian network classifiers is different from the problem of learning Bayesian networks in that the former approximates marginal distribution of a class variable, see Formula (6.2), while the latter approximates joint distribution of all variables, see Formula (5.10). Bayesian network learning algorithms presented in this chapter are specifically designed for creation of classifiers. Our objective is to maximize classification abilities of the constructed networks, and at the same time to minimize the complexity of learning algorithms. The proposed algorithms attempt to strike

balance between these two contradictory objectives.

Our classifier learning algorithms are based on the search-and-scoring approach. The performance of the algorithms is maximized by constraining the structure of searched networks and the use of network metrics well matched to the classification task. We were inspired by a remarkable performance of the simplest Bayesian network classifier: the naïve Bayes. A significant amount of research has been devoted to study its performance. The naïve Bayes classifier makes strict assumptions about the modeled environment: all of the parameter variables are assumed to be mutually independent. This assumption is almost always violated in practice. However the performance of the naïve Bayes is not significantly imparted by this violation (bias). It is believed that the secret of the naïve Bayes is that it has a small number of parameters allowing for their estimation with low variance, even from limited number of training samples (Friedman, 1997). The low variance is able to offset the bias in the estimation of the underlying conditional probability of the class variable introduced by restrictive network structure. Our approach is based on extending the naïve Bayes structure with the intention to minimize the amount of new parameters added to the network.

One of the main factors contributing to steep increases in the number of parameters is the number of parents for each variable in the network. The number of parameters associated with each node in the network is exponential with the number of this node's parents. The first principle of our synthesis is to limit the number of parents for each node. We will assume that each node has no more than two parents.

Our second assumption, constraining the structure of the network, is that any edge between a class node and an attribute node is always directed away from the class variable – a class variable has no parents. This assumption is dictated by the way an information is passed between nodes in a Bayesian network during inference. When the value at a node is known, the node is instantiated, and the node blocks any information passing from its parents to its children. However, the information can be passed from children to parents.

This property of Bayesian networks is called *dependency separation* (Pearl, 1988; Jensen, 1996). Typically, when we perform inference about the class node, all attribute nodes are instantiated. If an attribute node was a parent to the class node, then, since it was instantiated, it would block the information path between its parents and the class variable; it will make the class node conditionally independent from its parents. This is the situation we would like to avoid in classification; we are only interested about the inference in the class node and do not want to have superfluous edges and parameters in the network.

Friedman et al. (1997) introduced a tree-augmented naïve Bayes classifier, TAN, and demonstrated that it benchmarks well on a number of data sets from UCI Machine Learning Repository (Blake, Keogh, and Merz, 1998). Friedman et al. (1997) assume that every attribute depends on the class variable as is the case in the naïve Bayes network structure. Then, they extend the naïve Bayes network structure by adding tree-like dependency among attribute variables. In their approach, there is always an edge from the class node to every attribute node, and there is always an undirected path between any attributes that do not pass through the class variable. This may force dependencies between random variables in the model that do not exist in reality, thus deteriorating classification performance.

We relax both of the constraints posed by Friedman et al. (1997) as well as restrictions imposed by naïve Bayesian classifier. In our approach, not all attributes need to be dependent on the class variable, and there can be no undirected path between two attribute nodes. We introduce a family of network construction and search algorithms. Each of the algorithms in our family differ in the trade-off it makes between computational complexity and richness of the possible network structures it can create.

A naïve Bayes classifier has a fixed structure; only parameters can be learned. TAN classifier builds a classifier network using dependency analysis based on calculation of the mutual information among attribute variables. Our approach uses search and scoring for construction of Bayesian network classifiers. However, we also make use of mutual information to partially limit the domain of search and enhance algorithms' performance.

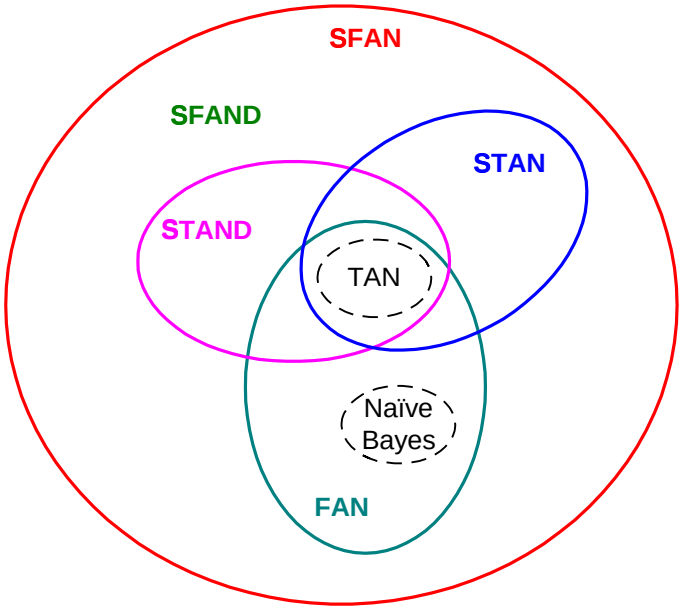


Figure 7.1: Family of the new Bayesian network classifiers with their relation to naïve Bayes and TAN classifiers. The new algorithms are represented in color, the old in black.

Fig. 7.1 depicts richness of structures produced by each of the algorithms in our family and their relationship to naïve Bayes and TAN classifiers. Each algorithm performs a heuristic search through its domain of network structures. The heuristic can be modified by using different structure scoring measures. The search algorithms and quality measures we use are described in the remainder of this chapter.

Each of the search algorithms is created from algorithm primitives that we call *operators*. By combining our operators we can produce any of the Bayesian network learning algorithms presented in Fig. 7.1, including naïve Bayes and TAN. We estimate the complexity of each of the operators and show how to combine them to build network search algorithms.

Presentation of quality measures, that are used for scoring models found by the search algorithms, is proceeded by discussion of learning network parameters and inference optimization. The chapter ends with a summary of complexity of complete algorithms for

learning Bayesian network classifiers and discussion of the discretization of continuous variables. Empirical evaluation of performance of the new family of Bayesian network classifiers is presented in Chapter 8.

Notation

The following notation will be used in this chapter.

B – Bayesian network over random variables $\{X_1, \ldots, X_n\}$, $B = \langle S, \boldsymbol{\theta} \rangle$.

S – directed acyclic graph representing structure of a Bayesian network.

$\boldsymbol{\theta}$ – parameters of a Bayesian network.

X_i – discrete variable having r_i possible configurations $\{x_i^{(1)}, \ldots, x_i^{(r_i)}\}$.

$\mathbf{Pa}_i$ – set of ρ parents of variable X_i, $\{\mathbf{pa}_{i1}, \ldots, \mathbf{pa}_{i\rho}\}$, having q_i possible configurations $\{\text{pa}_i^{(1)}, \ldots, \text{pa}_i^{(q_i)}\}$.

C – the class variable. We use a convention where $C \equiv X_n$.

$\mathbf{A}$ – set of attribute variables $\{A_1, \ldots, A_{n-1}\}$. We use convention where $A_i \equiv X_i$ for $i = 1, \ldots, n-1$.

$\boldsymbol{\gamma}$ – set of attributes dependent on the class variable C.

$\boldsymbol{\lambda}$ – set of attributes independent from the class variable C.

$\|\mathbf{X}\|$ – cardinality, number of elements, of set $\mathbf{X}$.

$\mathcal{D}$ – set of training cases, random samples (no missing values).

N – number of training cases ($\|\mathcal{D}\| = N$).

N_{ijk} – number of cases in $\mathcal{D}$ where the random variable X_i is in configuration k and and its parents, $\mathbf{Pa}_i$, are in configuration j.

Algorithm 7.1 Class dependency creation operator.

CLASS-DEPEND$(S, \boldsymbol{\gamma}, \mathcal{D})$

1. $\hat{S} \leftarrow S$
2. **for** each variable $\gamma_i \in \boldsymbol{\gamma}$ **do**
3. Add directed edge $(C \mapsto \gamma_i)$ to structure network $\hat{S}$
4. **return** $\hat{S}$

N_{ij} – number of cases in $\mathcal{D}$ where parents of the random variable X_i are in configuration j, $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$.

$q(S, \mathcal{D})$ – Bayesian network structure quality function. The higher the value of q the better structure S models training data set $\mathcal{D}$.

7.1 Search Operators

7.1.1 Class Dependency Creation Operator

We start by presenting a simple utility operator that extends network structure S by adding to it dependencies between the class node C and attribute nodes in set $\boldsymbol{\gamma}$. This operator is utilized by some of the search operators, and can be directly used to create a structure of a naïve Bayes classifier. The operators algorithm is presented in Alg. 7.1.

Numerical Complexity

Numerical complexity of the CLASS-DEPEND operator:

$$O_{\text{CLASS-DEPEND}} = \|\boldsymbol{\gamma}\|. \tag{7.1}$$

where $\|\boldsymbol{\gamma}\|$ is the cardinality of set $\boldsymbol{\gamma}$.

Algorithm 7.2 SAN class dependency discovery operator.

SAN$(\mathcal{D}, q,$ AUGMENTER$)$

1. $\boldsymbol{\gamma} \leftarrow \varnothing$ $\{$*set of attributes dependent on the class variable*$\}$
2. $\boldsymbol{\lambda} \leftarrow \boldsymbol{A} = \{A_1, \ldots, A_{n-1}\}$ $\{$*set of attributes independent from the class variable*$\}$
3. $\hat{q} \leftarrow -\infty$ $\{$*highest value of quality measure so far*$\}$
4. **for** $i = 1, \ldots, n-1$ **do**
5. Select an attribute $A \in \boldsymbol{\lambda}$ that maximizes quality measure $q(B, \mathcal{D})$
   where network $B \leftarrow$ AUGMENTER$(\boldsymbol{\gamma} \cup \{A\},\ \boldsymbol{\lambda} \setminus \{A\},\ \mathcal{D},\ q)$
6. **if** $q(B, \mathcal{D}) > \hat{q}$ **then**
7. $\hat{B} \leftarrow B$
8. $\hat{q} \leftarrow q(B, \mathcal{D})$
9. $\boldsymbol{\gamma} \leftarrow \boldsymbol{\gamma} \cup \{A\}$
10. $\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} \setminus \{A\}$
11. **return** $\hat{B}$

7.1.2 SAN Dependency Discovery Operator

The first search operator introduced is called *Selective Augmented Naïve Bayes* (SAN). This operator discovers dependencies between the class variable C and attribute variables A_i. If all of the attributes depended on the class variable, then, the Bayesian network would have the structure of an augmented naïve Bayes classifier (for instance TAN). However, the task of SAN is to determine which of these dependencies are actually needed. Additionally, at each step, SAN augments the discovered structure of dependencies between the class variable and attribute variables by application of the operator AUGMENTER. The SAN algorithm is presented in Alg. 7.2.

The SAN operator performs a greedy search of possible edges from the class variable C to attribute variables $\boldsymbol{A}$. It starts with an empty set of children and, at each step, adds a new child that is optimal according to network structure quality measure q. The children

of node C are elements of set $\boldsymbol{\gamma}$, while set $\boldsymbol{\lambda}$ contains these attribute variables that are not directly dependent on C. For each configuration of children, $\boldsymbol{\gamma}_i$, dependencies among attribute variables $\boldsymbol{A}$ are determined by a suitably chosen operator AUGMENTER. The AUGMENTER operator takes as an input dependencies between the class variable C and attribute variables $\boldsymbol{A}$, represented by set $\boldsymbol{\gamma}_{i-1} \cup \{X\}$, and learns additional dependencies among attribute variables $\boldsymbol{A}$ using the set of training cases D. SAN selects the network that corresponds to configuration of children $\boldsymbol{\lambda}_k$ with the highest score according to the quality measure q.

Numerical Complexity

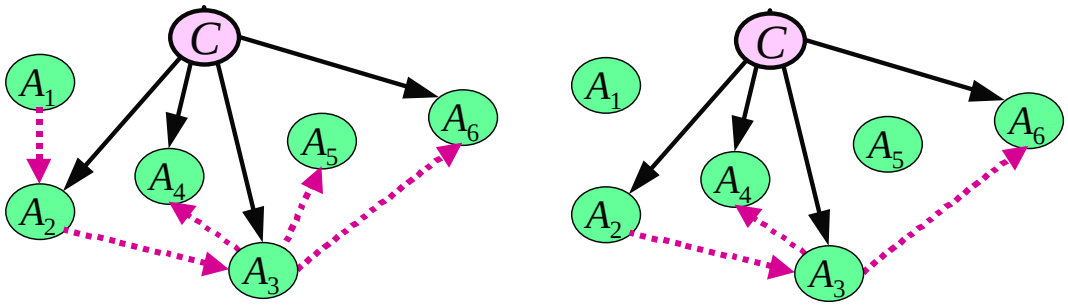
Numerical complexity of the SAN operator depends partially on the numerical complexity of the measure q used to evaluate the quality of created network, O_q, and on the numerical complexity of the AUGMENTER operator, $O_{\text{AUGMENTER}}$.

Complexity of steps 1 to 3 is constant. Complexity of step 5 is $\|\boldsymbol{\lambda}\| \cdot (O_q + O_{\text{AUGMENTER}})$. Remaining steps in the **for** loop have constant complexity. The loop is repeated $n-1$ times. Thus the complexity of the SAN operator is

$$O_{\text{SAN}} \;=\; n^2 \cdot (O_q + O_{\text{AUGMENTER}}). \tag{7.2}$$

7.1.3 SAND Dependency Discovery Operator

Operator *Selective Augmented Naïve Bayes with Discarding* (SAND) is similar to operator SAN. It discovers dependencies between the class variable C and attribute variables A_i. Unlike SAN, however, operator SAND discards attributes that are not determined to be dependent on the class variable before applying the AUGMENTER operator. In effect, operator the SAND performs attribute selection, determines which attributes do not contribute to the classification goal, and discards them from the classification network. The difference between networks produced by SAN and SAND is illustrated in Fig. 7.2. This figure as-



(a) SAN (b) SAND

Figure 7.2: Examples of networks produced by SAN and SAND operators.

sumes that a tree augmentation is used (see the very next section). The pseudo-code for the SAND algorithm is presented in Alg. 7.3.

Numerical Complexity

Numerical complexity of the SAND operator is the same as the numerical complexity of SAN:

$$O_{\text{SAND}} = n^2 \cdot (O_q + O_{\text{AUGMENTER}}). \tag{7.3}$$

7.1.4 Tree-Augmenting Operator

The tree-augmenting operator, TREE-AUGMENTER, is a generalization of the tree-augmented naïve Bayes classifier, TAN, discussed in (Friedman et al., 1997). The difference is that, unlike TAN, we do not require that all of the attribute nodes depend on the class variable C. Operand $\boldsymbol{\gamma}$ specifies attributes that do depend on the class variable, operand $\boldsymbol{\lambda}$ specifies additional attributes for augmentation that do not depend on the class node. The tree-augmenting operator working is presented in Alg. 7.4.

The operator builds the augmenting tree using an extension of algorithm proposed by (Chow and Liu, 1968), see also Section 5.4.3. The difference is in the way the mutual information function is computed. Since some of the nodes depend on the class variable

Algorithm 7.3 SAND class dependency discovery operator.

SAND$(\mathcal{D}, q,$ AUGMENTER$)$

1. $\boldsymbol{\gamma} \leftarrow \varnothing$ $\{$*set of attributes dependent on the class variable*$\}$
2. $\boldsymbol{\lambda} \leftarrow \boldsymbol{A} = \{A_1, \ldots, A_{n-1}\}$ $\{$*set of attributes independent from the class variable*$\}$
3. $\hat{q} \leftarrow -\infty$ $\{$*highest value of quality measure so far*$\}$
4. **for** $i = 1, \ldots, n-1$ **do**
5. Select attribute $A \in \boldsymbol{\lambda}$ that maximizes quality measure $q(B, \mathcal{D})$
   where network $B \leftarrow$ AUGMENTER $(\boldsymbol{\gamma} \cup \{A\},\ \varnothing,\ \mathcal{D})$
6. **if** $q(B, \mathcal{D}) > \hat{q}$ **then**
7. $\hat{B} \leftarrow B$
8. $\hat{q} \leftarrow q(B, \mathcal{D})$
9. $\boldsymbol{\gamma} \leftarrow \boldsymbol{\gamma} \cup \{A\}$
10. $\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} \setminus \{A\}$
11. **return** $\hat{B}$

we use this information while computing conditional mutual information. The following formula is using conditional (on class variable C) or unconditional probability of variables X and Y depending whether they are members of set $\boldsymbol{\gamma}$ or not:

$$I_{\boldsymbol{\gamma}}(X;Y) = \begin{cases} \sum\limits_{x,y} p(x,y|c) \cdot \log \dfrac{p(x,y|c)}{p(x|c)p(y|c)} & \text{if } X \in \boldsymbol{\gamma} \wedge Y \in \boldsymbol{\gamma}, \\ \sum\limits_{x,y} p(x,y|c) \cdot \log \dfrac{p(x,y|c)}{p(x|c)p(y)} & \text{if } X \in \boldsymbol{\gamma} \wedge Y \notin \boldsymbol{\gamma}, \\ \sum\limits_{x,y} p(x,y|c) \cdot \log \dfrac{p(x,y|c)}{p(x)p(y|c)} & \text{if } X \notin \boldsymbol{\gamma} \wedge Y \in \boldsymbol{\gamma}, \\ \sum\limits_{x,y} p(x,y) \cdot \log \dfrac{p(x,y)}{p(x)p(y)} & \text{if } X \notin \boldsymbol{\gamma} \wedge Y \notin \boldsymbol{\gamma}. \end{cases} \tag{7.4}$$

Numerical Complexity

The loop in step 2 is repeated $\frac{1}{2}\|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|(\|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| - 1)$ times. Complexity of computing the mutual information in step 3 depends on the number of states taken by attributes A_i and

Algorithm 7.4 Tree-augmenting operator.

TREE-AUGMENTER($\boldsymbol{\gamma},\ \boldsymbol{\lambda},\ \mathcal{D}$)

1. $G \leftarrow \varnothing$
2. **for** each pair of variables $\{A_i, A_j\} \subset \boldsymbol{\gamma} \cup \boldsymbol{\lambda}$ such that $A_i \neq A_j$ **do**
3. $\quad w_{ij} \leftarrow I_{\boldsymbol{\gamma}}(A_i; A_j)$
4. $\quad$ Add undirected edge $(A_i - A_j)$ to graph G
5. $T \leftarrow$ MAXIMUM-SPANNING-TREE$(G, \boldsymbol{w})$
6. Order edges in undirected tree T by choosing one node as the root and setting the direction of all edges to be outward from it, then convert it to Bayesian network structure S
7. $\hat{B} \leftarrow$ CLASS-DEPEND$(S, \boldsymbol{\gamma}, \mathcal{D})$
8. **return** $\hat{B}$

A_j. Assume that the maximum number of possible states in r_{MAX} then the complexity of step 3 is r^2_{max}, and complexity of the loop in step 2 is

$$O_2 \ = \ \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|^2 \cdot r^2_{max}.$$

We assume here that the maximum spanning tree algorithm is implemented using Prim's algorithm (Cormen et al., 1990). Complexity of the Prim's algorithm implemented using Fibonacci heaps is $E + V \log V$ where E is the number of edges and V is the number of vertices in graph G. In our case $E = \frac{1}{2}\|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|(\|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| - 1)$ and $V = \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|$.

$$\begin{aligned} O_5 &= O\left(\frac{1}{2}\|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|(\|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| - 1) + \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| \log\left(\|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|\right)\right) \ = \ \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|^2 \\ O_6 &= \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|^2 \\ O_7 &= \|\boldsymbol{\gamma}\| \end{aligned}$$

$$O_2 + O_5 + O_6 + O_7 \ = \ O\left(\|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|^2 \cdot r^2_{max}\right)$$

Hence the complexity of the tree-augmenting operator is

$$O_{\text{TREE-AUGMENTER}} = \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|^2 \cdot r_{max}^2. \tag{7.5}$$

7.1.5 Forest-Augmenting Operator

The tree-augmenting operator, TREE-AUGMENTER, creates a path between each of the nodes. This may create dependencies between variables in the model that are not really present. To alleviate this problem we introduce a forest-augmenting operator called FOREST-AUGMENTER. It can create dependencies between variables in a form of a number of disjoint trees. It may also determine that there is no dependency between attributes creating a naïve Bayes classifier; or that all of the nodes are connected by a single tree and create the TAN classifier. An algorithm for learning Bayesian network classifiers based on the FOREST-AUGMENTER operator can be more robust then either naïve Bayes or TAN algorithms. It can create not only naïve Bayes or TAN network structure, but a number of other, intermediate, classifier structures, thus having a better chance of finding optimal approximation of the probability distribution of the class variable. The forest-augmenting operator works as shown in Alg. 7.5.

The forest-augmenting algorithm utilizes a specific property of Kruskal's maximum spanning tree algorithm (Cormen et al., 1990). Kruskal's algorithm builds the spanning tree by adding legal edges in order of their decreasing weights. This way it maintains a graph containing a forest of disjoint trees. The branches of these trees in the forest are clustered by strongest dependency among the node variables. The FOREST-AUGMENTER operator uses the way Kruskal's algorithm was adding new edges as a heuristic for "growing" a forest of dependencies between argument nodes.

Kruskal's maximum spanning tree algorithm

Kruskal's maximum spanning tree algorithm is presented in Alg. 7.6. The algorithm makes use of the following supporting operations for maintenance of disjoint-set data structures:

Algorithm 7.5 Forest-augmenting operator.

FOREST-AUGMENTER($\boldsymbol{\gamma},\ \boldsymbol{\lambda},\ \mathcal{D},\ q$)

1. $G \leftarrow \varnothing$
2. **for** each pair of attributes $\{A_i, A_j\} \subset \boldsymbol{\gamma} \cup \boldsymbol{\lambda}$, such that $A_i \neq A_j$ **do**
3. $\quad w_{ij} \leftarrow I_{\boldsymbol{\gamma}}(A_i; A_j)$
4. $\quad$ Add undirected edge $(A_i - A_j)$ to graph G
5. $T \leftarrow$ MST-KRUSKAL$(G, \boldsymbol{w})$ $\{$*Kruskal's maximum spanning tree*$\}$
6. Direct edges in T
7. $\boldsymbol{E} \leftarrow$ set of edges in T sorted in decreasing order according to weights $\boldsymbol{w}$
8. $B \leftarrow$ CLASS-DEPEND$(\varnothing, \boldsymbol{\gamma}, \mathcal{D})$
9. $\hat{q} \leftarrow -\infty$
10. **for** every $E_i \in \boldsymbol{E}$, in order of decreasing weights **do**
11. $\quad$ Add edge E_i to network B
12. $\quad$ **if** $q(B, \mathcal{D}) > \hat{q}$ **then**
13. $\quad\quad \hat{q} \leftarrow q(B, \mathcal{D})$
14. $\quad\quad \hat{B} \leftarrow B$
15. **return** $\hat{B}$

Algorithm 7.6 Kruskal's algorithm for finding maximum spanning tree in a graph.

MST-KRUSKAL$(G,\ w)$

1. $T \leftarrow \varnothing$
2. **for** each vertex $v \in V[G]$ **do**
3. MAKE-SET(v)
4. Sort the edges of E by non-increasing weight w
5. **for** each edge $(u, v) \in E$, in order by non-increasing weight **do**
6. **if** FIND-SET(u) $\neq$ FIND-SET(v) **then**
7. $T \leftarrow T \cup \{(u, v)\}$
8. UNION(u,v)
9. **return** T

MAKE-SET**(**x**)** creates a new set whose only member (and thus representative) is pointed to by x. Since sets are disjoint, we require that x not already be in a set.

UNION**(**x**,** y**)** unites the dynamic sets that contain x and y, say S_x and S_y, into a new set that is the union of these sets. The two sets are assumed to be disjoint prior to operation. The representative of the resulting set is some member of $S_x \cap S_y$, although many implementations of UNION choose the representative of either S_x or S_y as the new representative. Since we require the sets in the collection to be disjoint, we "destroy" sets S_x and S_y, by removing them from the collection S.

FIND-SET**(**x**)** returns a pointer to the representative of the (unique) set containing x.

MST-KRUSKAL algorithm builds the maximum spanning tree by performing a greedy search, at each step adding to the structure a ***safe*** edge with highest weight. It maintains, at each step, a forest of trees that eventually are joined into a single tree. The complexity of MST-KRUSKAL is $O(E \log E)$, where E is the number of edges (Cormen et al., 1990). The fully connected graph has $\frac{V(V-1)}{2}$ edges, where V is the number of vertices in graph

G. The complexity of Kruskal's algorithm, in our case, is $O(V^2 \log V)$.

Numerical Complexity of Forest-Augmenting Operator

The numerical complexity of the forest-augmenting operator can be estimated as follows.

$$\begin{aligned} O_2 &= \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|^2 \cdot r_{max}^2 \\ O_5 &= \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|^2 \log \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| \\ O_6 &= \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| \\ O_8 &= \|\boldsymbol{\gamma}\| \end{aligned}$$

There are $\|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| - 1$ edges in the maximum spanning tree, thus

$$O_{10} = O\left((\|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| - 1) \cdot O_q\right) = \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| \cdot O_q$$

were O_q is the complexity of calculating quality measure q.

$$\begin{aligned} &O_2 + O_5 + O_6 + O_9 + O_{10} = \\ &\quad O\left(\|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|^2 \cdot r_{max}^2 + \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|^2 \cdot \log \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| + \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| \cdot O_q\right) \end{aligned}$$

Hence the complexity of the forest-augmenting operator is

$$O_{\text{FOREST-AUGMENTER}} = \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|^2 \cdot r_{max}^2 + \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\|^2 \cdot \log \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| + \|\boldsymbol{\gamma} \cup \boldsymbol{\lambda}\| \cdot O_q \quad (7.6)$$

7.2 Search Algorithms

Using the five operators presented above we can create a family of search algorithms for learning Bayesian network classifiers presented in Fig. 7.1. The computational complexity of each of the algorithms in this family is summarized in Table 7.1. Notice that the first two algorithms that are created using our operators are well known naïve Bayes classifier (Duda and Hart, 1973) and tree-augmented naïve Bayes (Friedman et al., 1997). These two algorithms do not perform an actual search, so their only argument is the training dataset $\mathcal{D}$. The remaining algorithms also take as an argument a network quality measure q.

Table 7.1: Family of search algorithms for learning Bayesian network classifiers and their complexity.

Algorithm	Operator Composition	Core Complexity
NAÏVE BAYES($\mathcal{D}$)	CLASS-DEPEND($\varnothing, \boldsymbol{A}, \mathcal{D}$)	n
TAN($\mathcal{D}$)	TREE-AUGMENTER($\boldsymbol{A}, \varnothing, \mathcal{D}$)	$n^2 \cdot r_{max}^2$
FAN($\mathcal{D}, q$)	FOREST-AUGMENTER($\boldsymbol{A}, \varnothing, \mathcal{D}, q$)	$n^2 \cdot r_{max}^2 + n^2 \cdot \log n + n \cdot O_q$
STAN($\mathcal{D}, q$)	SAN($\mathcal{D}, q$, TREE-AUGMENTER)	$n^4 \cdot r_{max}^2 + n^2 \cdot O_q$
STAND($\mathcal{D}, q$)	SAND($\mathcal{D}, q$, TREE-AUGMENTER)	$n^4 \cdot r_{max}^2 + n^2 \cdot O_q$
SFAN($\mathcal{D}, q$)	SAN($\mathcal{D}, q$, FOREST-AUGMENTER)	$\left(n^4 r_{max}^2 + n^4 \log n + n^3 O_q\right) O_q$
SFAND($\mathcal{D}, q$)	SAND($\mathcal{D}, q$, FOREST-AUGMENTER)	$\left(n^4 r_{max}^2 + n^4 \log n + n^3 O_q\right) O_q$

One of the hidden costs of all the algorithms presented here is the estimation of probabilities, or strictly speaking frequencies N_{ijk}, that are used for computation of mutual information and determination of network parameters $\boldsymbol{\theta}$. They are constant for a given training data set $\mathcal{D}$. Thus, when running several different algorithms on the data set we can improve the performance by pre-computing these frequencies and then passing them to each of the algorithms. We will present the complexity of complete algorithms after we introduce issues related to parameter learning, inference, and quality measures.

7.3 Learning Parameters

We have assumed that all variables X_i in the Bayesian network B have unrestricted multinomial distribution (if a dataset contains continuous variables it is discretized, as will be described in Section 7.7). Each local distribution function P_i associated with variable X_i is a collection of multinomial distributions P_{ij}, one distribution for each configuration of parents $\mathbf{Pa}_i$. We assume that

$$p(x_i^k | \mathbf{pa}_i^j, \boldsymbol{\theta}_i, S^h) = \theta_{ijk} > 0 \tag{7.7}$$

In other words, θ_{ijk} denotes the probability that variable X_i is in configuration k and its parents are in configuration j. And that this probability is always greater than zero.

$\boldsymbol{\theta}_i = ((\theta_{ijk})_{k=2}^{r_i})_{j=1}^{q_i}$ denotes all the parameters associated with variable X_i. Index k starts from 2 since parameters θ_{ij1} can be calculated using

$$\theta_{ij1} = 1 - \sum_{k=2}^{r_i} \theta_{ijk}.$$

Vector of parameters associated with each local distribution P_{ij} is denoted by

$$\boldsymbol{\theta}_{ij} = \{\theta_{ij2}, \ldots, \theta_{ijr_i}\}$$

We will denote by r_i number of configurations of variable X_i and by q_i the number of parent configurations of variable X_i:

$$q_i = \prod_{X_p \in \mathbf{Pa}_i} r_p \tag{7.8}$$

7.3.1 Parameter Estimation

The material presented in this subsection is based on (Heckerman, 1996). We present it here to introduce some notions and assumptions about parameter probability distribution that are used in the remainder of this chapter; in particular, parameter priors that are also utilized by quality measures.

As stated in Section 5.3, problem of learning Bayesian network parameters given network structure and training data set $\mathcal{D}$ is that of computing the posterior distribution $p(\boldsymbol{\theta}|\mathcal{D}, S^h)$. The posterior distribution $p(\boldsymbol{\theta}|\mathcal{D}, S^h)$ can be efficiently computed in closed form under two assumptions (Heckerman, 1996). The first assumption is that there are no missing data in the training data set $\mathcal{D}$. The second assumption is that parameter vectors $\boldsymbol{\theta}_{ij}$ are mutually independent:

$$p(\boldsymbol{\theta}|S^h) \;=\; \prod_{i=1}^{n} \prod_{j=1}^{q_i} p(\boldsymbol{\theta}_{ij}|S^h)$$

Under assumption of complete data and parameter independence, the parameters remain independent given the data set $\mathcal{D}$:

$$p(\boldsymbol{\theta}|\mathcal{D}, S^h) \;=\; \prod_{i=1}^{n}\prod_{j=1}^{q_i} p(\boldsymbol{\theta}_{ij}|\mathcal{D}, S^h)$$

This lets us update each vector of parameters independently.

We further assume that each vector $\boldsymbol{\theta}_{ij}$ has a prior Dirichlet distribution

$$p(\boldsymbol{\theta}_{ij}) \;=\; \text{Dir}(\boldsymbol{\theta}_{ij}|\alpha_{ij1}, \ldots, \alpha_{ijr_i}) \;\equiv\; \frac{\Gamma(\alpha_{ij})}{\prod_{k=1}^{r_i}\Gamma(\alpha_{ijk})}\prod_{k=1}^{r_i}\theta_{ijk}^{\alpha_{ijk}-1} \tag{7.9}$$

where $\alpha_{ij} = \sum_{k=1}^{r_i}\alpha_{ijk}$ and $a_{ijk} > 0$. Since Dirichlet distribution is the conjugate prior to multinomial distribution we have

$$p(\boldsymbol{\theta}_{ij}|\mathcal{D}, S^h) \;=\; \text{Dir}(\boldsymbol{\theta}_{ij}|\alpha_{ij1} + N_{ij1}, \ldots, \alpha_{ijr_i} + N_{ijr_i}) \tag{7.10}$$

where N_{ijk} is the number of cases in $\mathcal{D}$ in which variable X_i is in configuration k, $X_i = x_i^k$, and parents of X_i are in configuration j, $\mathbf{Pa}_i = \mathbf{pa}_i^j$.

Given the data set $\mathcal{D}$ and network structure S^h we can estimate parameters $\boldsymbol{\theta}$ by averaging over all possible configurations of $\boldsymbol{\theta}$. Using Eq. (7.7) we have that the probability of a new unobserved case $\mathbf{x}^{(N+1)}$

$$p(\mathbf{x}^{(N+1)}|\mathcal{D}, S^h) \;=\; \text{E}_{p(\boldsymbol{\theta}|\mathcal{D},S^h)}\left(\prod_{i=1}^{n}\theta_{ijk}\right)$$

where E denotes expected value. From the assumption of parameter independence, given data set $\mathcal{D}$

$$p(\mathbf{x}^{(N+1)}|\mathcal{D}, S^h) \;=\; \int\prod_{i=1}^{N}\theta_{ijk}\; p(\boldsymbol{\theta}|\mathcal{D}, S^h)\; d\boldsymbol{\theta} \;=\; \prod_{i=1}^{N}\int\theta_{ijk}\; p(\boldsymbol{\theta}_{ij}|\mathcal{D}, S^h)\; d\boldsymbol{\theta}_{ij}$$

From Eq. (7.10) and properties of Dirichlet distribution we can estimate parameters θ_{ijk} as follows (Heckerman, 1996)

$$p(\mathbf{x}^{(N+1)}|\mathcal{D}, S^h) \;=\; \prod_{i=1}^{n}\bar{\theta}_{ijk} \;=\; \prod_{i=1}^{n}\frac{\alpha_{ijk} + N_{ijk}}{\alpha_{ij} + N_{ij}} \tag{7.11}$$

where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$.

7.3.2 Selection of Priors

Selection of priors, or in our case Dirichlet prior parameters α_{ijk}, is a very important and difficult problem. Priors enable us to encode prior knowledge about the problem domain. They can be used to improve generalization properties of networks with empirically estimated parameters. Sizable research effort is devoted to selection of priors. However, we opt here for a relatively simple solution.

Cooper and Herskovits (1992) in their work on the K2 Bayesian network learning algorithm suggested a simple uniform assignment $\alpha_{ijk} = 1$. Buntine (1991b) suggested the uniform assignment that is dependent on the number of the variable configurations and number of configurations of its parents $\alpha_{ijk} = \frac{\alpha}{r_i \cdot q_i}$. We assume that all parameters α_{ijk} are equal, and make an arbitrary assumption that their values are in range $\{0, \ldots, 9\}$[1] . There is no particular reason for making this second assumption other than it gives a manageable number of values. While creating a network we test each of these values and select one that seems to be optimal for particular data set (based on experiments).

7.3.3 Numerical Complexity

Assuming that learning parameters includes computation of frequencies N_{ijk} from the data set $\mathcal{D}$ its numerical complexity is

$$O_{\text{LEARN-PARAM}} \;=\; n \cdot r_{max}^{\varrho} \cdot N \tag{7.12}$$

where r_{max} is the maximum number of states of each variable in the network, and ϱ is the maximum number of parents of each node in the network.

Notice that we can conclude from Eq. (7.12) that controlling the number of parents of each variable helps prevent combinatoric explosion in the complexity of learning algorithms. If there was no restriction on the number of parents then the complexity of learning

[1] Strictly speaking we do not use value 0 for α_{ijk}, but a small positive number close to zero. So that the assumption given by formula (7.7) is always satisfied.

would be more then exponential with the number of nodes in the network:

$$O_{\text{LEARN-PARAM}} \;=\; n \cdot r_{max}^{n} \cdot N.$$

It is our intention that the algorithms presented in this chapter do not produce networks that have more then two parents for each node so that the numerical complexity of learning parameters is

$$O_{\text{LEARN-PARAM}} \;=\; n \cdot r_{max}^{2} \cdot N \tag{7.13}$$

7.4 Inference

In this section we demonstrate that we can perform very efficient inference in our family of classifiers applying directly formula in Eq. (5.5). This is significant since it allows for a low computational complexity of inference, that is linear with the number of variables n. As stated in Section 5.2: in general, direct application of Eq. (5.5) is not possible, and the inference problem is NP-complete (complexity increases exponentially with number of variables n). Some of the quality measures, presented later in this chapter, use inference in network they are scoring, thus lower complexity of inference means lower complexity of algorithms that use those quality measures.

We can utilize specifics of the structure of networks created by algorithms introduced in the previous section, in particular, that the class node has no parents. When a Bayesian network is used as a classifier we are interested in the probability of the class node state given the values of the attribute nodes states. All attributes are instantiated during the evaluation. The joint probability distribution of a classifier network is

$$p(\boldsymbol{x}) \;=\; p(a_1, \ldots, a_{n-1}, c)$$

where a_i are values of the attribute nodes, and c is the value of the class node.

Using Eq. (5.3) we can express the probability distribution represented by a classifier network as:

$$p(a_1, \ldots, a_{n-1}, c) \;=\; \prod_{i=1}^{n-1} p(a_i|\mathbf{pa}_{a_i}) \cdot p(c|\mathbf{pa}_c) \;=\; p(c) \prod_{i=1}^{n-1} p(a_i|\mathbf{pa}_{a_i})$$

$p(c|\mathbf{pa}_c) \;=\; p(c)$ since, in the networks built by our classifiers, the class node has no parents.

Assume that the goal of our inference is to determine the probability that the class node is in state k given states of attribute nodes $a_1, \ldots, a_{n-1}$. Using Eq. (5.5) and replacing integration by a sum (variables in our networks have multinomial distributions) we have

$$p(c^{(k)}|a_1, \ldots, a_{n-1}) \;=\; \frac{p(c^{(k)}) \prod_{i=1}^{n-1} p(a_i|\mathbf{pa}_{a_i})}{\sum_{j=1}^{r_c} \left[p(c^{(j)}) \prod_{i=1}^{n-1} p(a_i|\mathbf{pa}_{a_i}) \right]}$$

$$p(c^{(k)}|a_1, \ldots, a_{n-1}) \;=\; \frac{1}{\sum_{\substack{j=1 \\ j \neq k}}^{r_c} \left[\frac{p(c^{(j)})}{p(c^{(k)})} \prod_{i=1}^{n-1} \frac{p(a_i|\mathbf{pa}_{a_i})}{p(a_i|\mathbf{pa}_{a_i})} \right]} \tag{7.14}$$

The above formula can be directly used for inference since for each term of the sum in the denominator, all of the variables are instantiated: $C = c^{(j)}$, and $A_i = a_i$ for $i = 1, \ldots, n-1$.

Numerical Complexity

Numerical complexity of performing inference using formula (7.14) is

$$O_{\text{INFERENCE}} \;=\; n \cdot r_{max}. \tag{7.15}$$

7.5 Quality Measures

Choice of a quality measure is crucial for achieving high performance of a Bayesian network classifier learning algorithm. We argue, as stated in Chapter 6, that local measures,

that evaluate network at the class node, rather then global measures, that evaluate the complete network without distinction for any particular node, are better suited for creation of classifiers. We will test it empirically in Chapter 8. We will evaluate the performance of Bayesian network classifier search algorithms using five quality measures presented in this section. The first two are global Bayesian quality measures. The last three are local quality measures.

Computation of a Bayesian network quality measure is associated with a specific numeric cost. It may significantly increase the overall complexity of classifier learning algorithm. For each of the presented measures, we also estimate its numerical complexity.

7.5.1 HGC – Heckerman-Geiger-Chickering Measure

This global Bayesian network quality measure is based on accessing the posterior probability of the network structure given the training data set $\mathcal{D}$. It was proposed by Heckerman et al. (1995) and called *Bayesian Dirichlet* metric. We will refer to it as HGC (from the last names of the authors)

$$\begin{aligned} Q_{HGC} &= \log p(\mathcal{D}, S^h) \\ &= \log p(S) + \sum_{i=1}^{n} \left[\sum_{j=1}^{q_i} \left[\log \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} + \sum_{k=1}^{r_i} \log \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})} \right] \right] \end{aligned} \tag{7.16}$$

where Γ is the gamma function:

$$\Gamma(z) \;=\; \int_0^\infty t^{z-1} e^{-t} dt.$$

In our implementation we use Lonczos' approximation of function $\log \Gamma(z)$ (Press et al., 1992). We use logarithm of $\Gamma(z)$ since Γ is a fast increasing function and can cause numerical overflow even for moderate values of z. The Lonczos' approximation has constant complexity.

Frequently, prior structure probability $p(S)$ is unknown and assumed to be constant

over all possible S. The HGS measure can be rewritten as follows.

$$Q_{HGC} = \sum_{i=1}^{n}\left[\sum_{j=1}^{q_i}\left[\log\Gamma(\alpha_{ij}) - \log\Gamma(\alpha_{ij}+N_{ij}) + \sum_{k=1}^{r_i}\log\Gamma(\alpha_{ijk}+N_{ijk}) - \log\Gamma(\alpha_{ijk})\right]\right]$$

$$\begin{aligned} Q_{HGC} = & \sum_{i=1}^{n}\sum_{j=1}^{q_i}[\log\Gamma(\alpha_{ij}) - \log\Gamma(\alpha_{ij}+N_{ij})] + \\ & \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i}[\log\Gamma(\alpha_{ijk}+N_{ijk}) - \log\Gamma(\alpha_{ijk})] \end{aligned} \tag{7.17}$$

Numerical Complexity

We assume that frequencies N_{ijk} are pre-computed; they do not need to be calculated separately for each evaluation of HGC. Maximum value of q_i in Eq. (7.17) is r_{max}^{ϱ}. Thus, complexity of computing the HGC measure is

$$O_{\text{HGC}} = n \cdot r_{max}^{\varrho+1}. \tag{7.18}$$

7.5.2 SB – Standard Bayesian Measure

HGC measure rewards for good approximation of the joint probability distribution. It may naturally prefer larger networks, since larger number of parameters allows for better fit. However if the number of parameters is too large, overfitting to training data set occurs, and the constructed network may have poor generalization properties.

It may be beneficial to prefer smaller networks. Since that may result in better generalization properties. In a smaller network there are fewer parameters and variance of their estimation from the training data will be smaller. Size of the Bayesian network can be computed using the following definition (Castillo et al., 1997).

Dimension of a Bayesian Network *Let X be a set of random variables and B be a Bayesian network defined over X. The dimension of this network,*

$Dim(B)$, is the number of free parameters required to completely specify the joint probability distribution of X.

$Dim(B)$ can be computed as follows:

$$Dim(B) = \sum_{i=0}^{n}(r_i - 1)\, q_i = \sum_{i=0}^{n}(r_i - 1) \prod_{X_p \in \mathbf{Pa_i}} r_p. \tag{7.19}$$

Castillo et al. (1997) presented a global Bayesian measure that is proportional to the posterior probability distribution $p(S, \boldsymbol{\theta}|\mathcal{D})$ with an added penalty term for the network size. They call it *Standard Bayesian* measure. Below is a simplified derivation. We include it here since it makes some assumptions that are less common among other authors that typically follow approach presented in Section 7.3.1. The main difference is in the way network parameters are estimated.

Posterior probability of the network B given training samples $\mathcal{D}$:

$$p(B|\mathcal{D}) \;=\; p(S, \boldsymbol{\theta}|\mathcal{D}) \;=\; \frac{p(S, \boldsymbol{\theta}, \mathcal{D})}{p(\mathcal{D})} \tag{7.20}$$

Since that data set $\mathcal{D}$ is the same for the networks that are compared (p(D) is constant) then:

$$p(S, \boldsymbol{\theta}|\mathcal{D}) \propto p(S, \boldsymbol{\theta}, \mathcal{D}) = p(S)p(\boldsymbol{\theta}|S)p(\mathcal{D}|S, \boldsymbol{\theta}) \tag{7.21}$$

The Standard Bayesian measure or SB is given by formula:

$$Q_B(B(\boldsymbol{\theta}), S) = \log p(S) + \log p(\hat{\boldsymbol{\theta}}|S) + \log p(\mathcal{D}|S, \hat{\boldsymbol{\theta}}) - \frac{1}{2} Dim(B) \log N \tag{7.22}$$

Castillo et al. (1997, p.494) assume that $\hat{\boldsymbol{\theta}}$ is the following posterior mode of $\boldsymbol{\theta}$

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}|S, \mathcal{D}) \tag{7.23}$$

Assuming that all variables in the network have multinomial distribution we have

$$p(\boldsymbol{\theta}|S) \propto \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \theta_{ijk}^{\alpha_{ijk}-1} \tag{7.24}$$

$$p(\mathcal{D}|S, \boldsymbol{\theta}) \propto \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \theta_{ijk}^{N_{ijk}} \tag{7.25}$$

$$\hat{\theta}_{ijk} = \frac{\alpha_{ijk} + N_{ijk} - 1}{\alpha_{ij} + N_{ij} - r_i} \tag{7.26}$$

If we assume, as before, that $p(S)$ is constant then we have the following formula for the standard Bayesian measure for multinomial distribution:

$$\begin{aligned} Q_{SB}(B, \mathcal{D}) \;&=\; \sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} (N_{ijk} + \alpha_{ijk} - 1) \log \frac{N_{ijk} + \alpha_{ijk} - 1}{N_{ij} + \alpha_{ij} - r_i} \\ &-\; \frac{1}{2}\, Dim(B) \log N. \end{aligned} \tag{7.27}$$

Numerical Complexity

As before we assume that frequencies N_{ijk} are pre-computed. The numerical complexity of computing the Standard Bayesian measure is the same as the complexity of HGS measure:

$$O_{\text{SB}} \;=\; n \cdot r_{max}^{\varrho+1}. \tag{7.28}$$

7.5.3 LC – Local Criterion Measure

Spiegelhalter, David, Lauritzen, and Cowell (1993) suggested the following local criterion measure that could be more adequate for construction of classifiers than global quality measures:

$$\text{LC}(S^h, \mathcal{D}) = \sum_{l=1}^{N} \log p(c^{(l)} | \mathbf{a}^{(l)}, \mathcal{D}^{(l)}, S^h) \tag{7.29}$$

where index l represents the l^{th} training case, $c^{(l)}$ is value of the class variable and $\mathbf{a}^{(l)}$ is the configuration of attributes in the l^{th} case. $\mathcal{D}^{(l)} = \{x^{(1)}, \ldots, x^{(l-1)}\}$.

The model S is trained sequentially with the first $l-1$ cases, then tested with the l^{th} case. This is a form of cross-validation where the training and testing cases are never interchanged. This measure can be interpreted as a local version of the global Bayesian measure HGC.

Numerical Complexity

There are N operations of parameter learning and inference with these new parameters. Frequencies can be calculated cumulatively reducing complexity of learning parameters:

$$O\left(N \cdot n \cdot r_{max} + N \cdot n \cdot r_{max}^{\varrho}\right)$$

Thus numerical complexity of the LC measure is

$$O_{\mathrm{LC}} \;=\; N \cdot n \cdot r_{max}^{\varrho} \tag{7.30}$$

7.5.4 LOO – Leave-One-Out Cross Validation

Let $\mathcal{V}_l$ represent the training data set $\mathcal{D}$ with the l^{th} instance removed:

$$\mathcal{V}_l \;=\; \mathcal{D} \setminus \{\mathbf{x}^{(l)}\} \;=\; \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(l-1)}, \mathbf{x}^{(l+1)}, \ldots, \mathbf{x}^{(n)}\} \tag{7.31}$$

Global Leave-One-Out Cross Validation Measure

The global *leave-one-out* cross validation criterion can be defined as

$$\mathrm{LOO}_{global}(S^h, \mathcal{D}) \;=\; \sum_{l=1}^{N} \log p(\mathbf{x}^{(l)} | \mathcal{V}_l, S^h) \tag{7.32}$$

Heckerman (1996) argues that this criterion is less fit for grading models than the HGC criterion (Eq. 7.16). He follows the argument of David (1894) that LOO_{global} criterion overfits the model to the data. In LOO_{global} training and testing cases are interchanged. It is not the case with the Q_{HGC} criterion:

$$\begin{aligned} Q_{HGC} \;&=\; \log p(\mathcal{D}, S^h) \;=\; \log p(S^h) + \log p(\mathcal{D}|S^h) \\ &=\; \log p(S^h) + \sum_{l=1}^{N} \log p(\mathbf{x}^{(l)} | \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(l-1)}, S^h) \end{aligned}$$

Notice that the last term in the above equation is similar to terms in Eq. (7.32). However the term in the above equation represents incremental predictions $p(\mathbf{x}^{(l)}|\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(l-1)}, S^h)$ without mixing the training set and the testing set (training set is being incrementally enlarged).

Local Leave-One-Out Cross Validation

The LOO_{global} measure given by Eq. (7.32) is not suitable for classification since it tests the joint distribution $p(\mathbf{x})$. We can modify it to test the marginal probability of the class variable C given configuration of attributes $\mathbf{A}$:

$$\mathrm{LOO}(S^h, \mathcal{D}) = \sum_{l=1}^{N} p(c^{(l)} | \mathbf{a}^{(l)}, \mathcal{V}_l, S^h) \tag{7.33}$$

Notice that this measure is very similar to the local criterion measure LC given by Eq. (7.29). The above Heckerman's argument may be also applied to comparing local LOO with LC.

Numerical Complexity

Two components contribute to complexity of evaluating LOO: inference performed to evaluate conditional probability of the class variable and learning of Bayesian network parameters. The training data set V_l is changed N times during the evaluation LOO. This requires that network parameters $\boldsymbol{\theta}$ be estimated N times, for each V_l separately. A straightforward implementation of LOO would then lead to following complexity:

$$O\left(N \cdot (n \cdot r_{max}^{\varrho} \cdot N + n \cdot r_{max})\right) \;=\; n \cdot r_{max}^{\varrho} \cdot N^2$$

Careful implementation through, by temporally modifying the frequencies, may improve the complexity. Instead of recalculating all of the frequencies for every data set V_l we can modify the frequencies for data set $\mathcal{D}$ depending which case is currently removed from it. This may lead to more messy implementation, but can significantly improve the complexity, especially for large data sets:

$$O_{\mathrm{LOO}} \;=\; O\left(N \cdot (n \cdot r_{max}^{\varrho} + n \cdot r_{max})\right) \;=\; n \cdot r_{max}^{\varrho} \cdot N \tag{7.34}$$

7.5.5 $\mathrm{CV}_{\phi,\tau}$ – ϕ-Fold τ-Times Cross Validation

During ϕ-fold cross validation, data set $\mathcal{D}$ is split into ϕ pairs of sets $\mathcal{V}_i$ and $\mathcal{W}_i$. Data set $\mathcal{V}_l$ is used for training and data set $\mathcal{W}_l$ for testing. For each $i = 1, \ldots, \phi$ we have that

$\mathcal{V}_i \cup \mathcal{W}_i = \mathcal{D}$ and the cardinality of the set $\mathcal{W}_i$ is approximately one ϕ^{th} of the cardinality of the set $\mathcal{D}$:

$$\|\mathcal{W}_i\| \approx \frac{1}{\phi}\|\mathcal{D}\|.$$

The ϕ-fold cross validation is repeated τ times and the results are averaged. This defines following quality measure:

$$\text{CV}_{\phi,\tau}(S^h, \mathcal{D}) = \sum_{j=1}^{\tau}\sum_{i=1}^{\phi}\sum_{l=1}^{\|\mathcal{W}_i\|} \log p(c_l|\mathbf{a}^{(l)}, \mathcal{V}_i^{(j)}, S^h) \tag{7.35}$$

Note that if we set $\phi = \|\mathcal{D}\|$ and $\tau = 1$ then measure $\text{CV}_{\phi,\tau}$ is the same as measure LOO. Typically, however, ϕ and τ are set constant and independent from the size of data set $\mathcal{D}$.

Numerical Complexity

As in the case of LOO, the complexity depends on parameter learning and inference. However, now parameter learning can be less frequent than evaluation (for $\phi < N$)

$$O\left(\tau \cdot \phi \cdot \left(n \cdot r_{max}^{\varrho} \cdot N\frac{\phi-1}{\phi}\right) + \tau \cdot \phi \cdot \frac{N}{\phi} \cdot (n \cdot r_{max})\right)$$

Thus, the complexity of the cross validation measure $\text{CV}_{\phi,\tau}$ is

$$O_{\text{CV}_{\phi,\tau}} = \tau \cdot \phi \cdot n \cdot r_{max}^{\varrho} \cdot N. \tag{7.36}$$

For example, for 10-times 10-fold cross validation and the network with no more than two parents for each node the complexity would be

$$O_{\text{CV}_{10,10}} = n \cdot r_{max}^2 \cdot N.$$

7.6 Complexity of Complete Learning Algorithms

Now, we are ready to combine structure search, parameter learning, inference, and quality measures presented in previous sections to create a new family of complete algorithms for

learning Bayesian network classifiers. Table 7.2 presents estimation of complexity for each of the algorithms in the family.

Derivation of these estimates is presented below. To simplify the final formulas, we will assume that the number of variables n is approximately equal to the maximum number of variable configurations r_{max}, $n \approx r_{max}$ and that n is much smaller than the number of cases N in the training data set, $n \ll N$.

Naïve Bayes

Naïve Bayes classifier does not perform a search, so its total complexity is its core complexity (see Table 7.1) plus the complexity of learning parameters given by Eq. (7.12)

$$\begin{aligned} O_{\text{NAÏVE BAYES}} &= O\left(n + n \cdot r_{max} \cdot N\right) = n \cdot r_{max} \cdot N \\ &\approx n^2 \cdot N \end{aligned}$$

Tree Augmented Naïve Bayes

TAN does not perform search thus,

$$\begin{aligned} O_{\text{TAN}} &= O\left(n^2 \cdot r_{max}^2 + n \cdot r_{max}^2 \cdot N\right) = n \cdot r_{max}^2 \cdot (n + N) \\ &\approx n^3 \cdot N \end{aligned}$$

Forest Augmented Naïve Bayes

Complexity of the Heckerman-Geiger-Chickering and the Standard Bayes quality measure are the same:

$$\begin{aligned} O_{\text{FAN-HGC}} = O_{\text{FAN-SB}} &= O\left(n^2 \cdot r_{max}^2 + n^2 \cdot \log n + n \cdot (n \cdot r_{max}^3) + n \cdot r_{max}^2 \cdot N\right) \\ &= n^2 \cdot \log n + n^2 \cdot r_{max}^3 + n \cdot r_{max}^2 \cdot N \\ &\approx n^3 \cdot N \end{aligned}$$

FAN classifier with either leave-one-out cross validation or local criterion quality measure:

$$\begin{aligned} O_{\text{FAN-LOO}} = O_{\text{FAN-LC}} &= O\left(n^2 \cdot r_{max}^2 + n^2 \cdot \log n + n \cdot (n \cdot r_{max}^2 \cdot N) + n \cdot r_{max}^2 \cdot N\right) \\ &= n^2 \cdot \log n \ + \ n^2 \cdot r_{max}^2 \cdot N \ + \ n \cdot r_{max}^2 \cdot N \\ &\approx n^4 \cdot N \end{aligned}$$

FAN classifier with ϕ-folds τ-times cross validation quality measure:

$$\begin{aligned} O_{\text{FAN-CV}_{\phi,\tau}} &= O\left(n^2 \cdot r_{max}^2 + n^2 \cdot \log n + n \cdot (\phi \cdot \tau \cdot n \cdot r_{max}^2 \cdot N) + n \cdot r_{max}^2 \cdot N\right) \\ &= n^2 \cdot \log n \ + \ \phi \cdot \tau \cdot n^2 \cdot r_{max}^2 \cdot N \ + \ n \cdot r_{max}^2 \cdot N \\ &\approx \phi \cdot \tau \cdot n^4 \cdot N \end{aligned}$$

If we assume that ϕ and τ are constant and equal 10 then

$$O_{\text{FAN-CV}_{10,10}} \approx n^4 \cdot N$$

STAN and STAND Classifiers

Numerical complexity of STAN and STAND is the same, so will derive complexity only for the STAN classifier.

$$\begin{aligned} O_{\text{STAN-HGC}} = O_{\text{STAN-SB}} &= O\left(n^4 \cdot r_{max}^2 + n^2 \cdot (n \cdot r_{max}^3) + n \cdot r_{max}^2 \cdot N\right) \\ &= n^4 \cdot r_{max}^2 + n^3 \cdot r_{max}^3 + n \cdot r_{max}^2 \cdot N \\ &\approx n^3 \cdot N \end{aligned}$$

$$\begin{aligned} O_{\text{STAN-LOO}} = O_{\text{STAN-LC}} &= O\left(n^4 \cdot r_{max}^2 + n^2 \cdot (n \cdot r_{max}^2 \cdot N) + n \cdot r_{max}^2 \cdot N\right) \\ &= n^4 \cdot r_{max}^2 + n^3 \cdot r_{max}^2 \cdot N \\ &\approx n^5 \cdot N \end{aligned}$$

$$\begin{aligned} O_{\text{STAN-CV}_{\phi,\tau}} &= O\left(n^4 \cdot r_{max}^2 + n^2 \cdot (\phi \cdot \tau \cdot n \cdot r_{max}^2 \cdot N) + n \cdot r_{max}^2 \cdot N\right) \\ &= n^4 \cdot r_{max}^2 + \phi \cdot \tau \cdot n^3 \cdot r_{max}^2 \cdot N \\ &\approx \phi \cdot \tau \cdot n^5 \cdot N \end{aligned}$$

If we assume that ϕ and τ are constant and equal 10 then

$$O_{\text{STAN-CV}_{10,10}} \approx n^4 \cdot N$$

SFAN and SFAND Classifiers

Numerical complexity of SFAN and SFAND is the same, so will derive complexity only for the SFAN classifier.

$$\begin{aligned} O_{\text{SFAN-HGC}} &= O_{\text{SFAN-SB}} \\ &= O\left(\left(n^4 r_{max}^2 + n^4 \log n + n^3 \cdot (n \cdot r_{max}^3)\right) \cdot \left(n \cdot r_{max}^3\right) + n \cdot r_{max}^2 \cdot N\right) \\ &= n^5 r_{max}^6 + n^5 r_{max}^3 \log n + n \cdot r_{max}^2 \cdot N \\ &\approx n^3 \cdot N \end{aligned}$$

$$\begin{aligned} O_{\text{SFAN-LOO}} &= O_{\text{SFAN-LC}} \\ &= O\left(\left(n^4 r_{max}^2 + n^4 \log n + n^3 (n r_{max}^2 N)\right) \left(n r_{max}^2 N\right) + n r_{max}^2 N\right) \\ &= n^5 r_{max}^4 N + n^5 r_{max}^2 N \log n + n^6 r_{max}^4 N^2 + n \cdot r_{max}^2 \cdot N \\ &\approx n^{10} \cdot N^2 \end{aligned}$$

$$\begin{aligned} O_{\text{SFAN-CV}_{\phi,\tau}} &= O\left(\left(n^4 r_{max}^2 + n^4 \log n + n^3 (\phi\tau n r_{max}^2 N)\right) \left(\phi\tau n r_{max}^2 N\right) + n r_{max}^2 N\right) \\ &= n^5 r_{max}^4 N + n^5 r_{max}^2 N \log n + \phi\tau n^6 r_{max}^4 N^2 + n \cdot r_{max}^2 \cdot N \\ &\approx \phi \cdot \tau \cdot n^{10} \cdot N^2 \end{aligned}$$

If we assume that ϕ and τ are constant and equal 10 then

$$O_{\text{SFAN-CV}_{10,10}} \approx n^{10} \cdot N^2$$

7.7 Discretization

Discretization is a process in which a continuous scalar variable is replaced by a finite number of labels. A discretization algorithm first divides the domain of a continuous vari-

Table 7.2: Estimated complexity of algorithms for learning Bayesian network classifiers: combination of search algorithms and quality measures.

Algorithm	**Without Quality Measure**	**With Quality Measure**			
		HGC, SB	LC, LOO	$\text{CV}_{\phi,\tau}$	$\text{CV}_{10,10}$
NAÏVE BAYES($\mathcal{D}$)	$n^2 \cdot N$	·	·	·	·
TAN($\mathcal{D}$)	$n^3 \cdot N$	·	·	·	·
FAN($\mathcal{D}, q$)	·	$n^3 \cdot N$	$n^4 \cdot N$	$\phi \cdot \tau \cdot n^4 \cdot N$	$n^4 \cdot N$
STAN($\mathcal{D}, q$)	·	$n^3 \cdot N$	$n^5 \cdot N$	$\phi \cdot \tau \cdot n^5 \cdot N$	$n^5 \cdot N$
STAND($\mathcal{D}, q$)	·	$n^3 \cdot N$	$n^5 \cdot N$	$\phi \cdot \tau \cdot n^5 \cdot N$	$n^5 \cdot N$
SFAN($\mathcal{D}, q$)	·	$n^3 \cdot N$	$n^{10} \cdot N^2$	$\phi \cdot \tau \cdot n^{10} \cdot N^2$	$n^{10} \cdot N^2$
SFAND($\mathcal{D}, q$)	·	$n^3 \cdot N$	$n^{10} \cdot N^2$	$\phi \cdot \tau \cdot n^{10} \cdot N^2$	$n^{10} \cdot N^2$

able into a finite number of non-overlapping intervals that cover the whole domain, then replaces each instance of the variable by a label corresponding to an interval that contains the value of that instance.

Some popular inducers have a discretization build-in (Quinlan, 1993; Holte, 1993; Maass, 1994; Auer, Holte, and Maass, 1995). Typically, it is a matter of algorithm implementation rather than an intrinsic feature of the algorithm. A "natural" way of dealing with continuous attributes in Bayesian networks would be to use continuous distribution for variables. However, there is no distribution that can handle a continuous variables as well as a multinomial distribution can handle discrete variables. A naïve Bayes inducer models continuous variables using normal distribution. Many practical variables, and in particular ones considered in this work, cannot be sufficiently well modelled by normal distribution or multivariate-normal distribution. Multivariate-normal distribution is popular primarily due to its nice mathematical properties – it is a member of *exponential family* of distributions; not because it is well suited to model real life probability distributions.

Friedman, Goldszmidt, and Lee (1998) unsuccessfully attempted to extend their tree-augmented naïve Bayes algorithm, TAN (Friedman et al., 1997), to be able to directly

handle continuous attributes. They used normal distributions and mixtures of normal distributions to create a continuous version of the TAN. Results they obtained indicate no improvement in classification of data sets with continuous variables.

Dougherty, Kahavi, and Sahami (1995) reported research that compares a number of discretization methods. They demonstrate, using data sets from UCI Machine Learning repository (Blake et al., 1998), that performance of continuous a naïve Bayes inducer can be significantly improved by discretizing the data sets and using a discrete version of naïve Bayes. The best discretization method reported by Dougherty et al. (1995) is a *minimal entropy* heuristic presented in (Catlett, 1991) and (Fayyad and Irani, 1993).

Bearing the above in mind, we decided to limit our new family of Bayesian network classifiers algorithms to handle only discrete features. Motivated by research of Dougherty et al. (1995), we use the minimal entropy heuristic for discretization of datasets with continuous features (the heuristic is described below in Section 7.7.1). We first discretize the training dataset using the minimal entropy heuristic, then use the discovered discretization scheme to discretize the test data.

7.7.1 Minimal Entropy Heuristic

For the convenience of the reader, this section provides a brief description of the minimal entropy heuristic. In short, the heuristic performs recursive binary cuts of the domain of a continuous attribute into intervals. Cutting of intervals continues till a stop criterion is reached. Each attribute is discretized separately.

Let A denote an attribute for which the discretization is performed. Let ω be an interval. Let $\mathcal{D}$ be the dataset that is being discretized. We will denote by $\mathcal{D}_\omega$ a subset of $\mathcal{D}$ such that values of attribute A for each case in $\mathcal{D}_\omega$ are in interval ω:

$$\mathcal{D}_\omega = \{\mathbf{x}^{(l)} : \mathbf{x}^{(l)} \in \mathcal{D} \text{ and } a^{(l)} \in \omega\}$$

The *class entropy* of dataset $\mathcal{D}_\omega$ is defined as:

$$\text{Ent}(\mathcal{D}_\omega) \;=\; -\sum_{k=1}^{r_c} P(c_k|\mathcal{D}_\omega)\log_2 P(c_k|\mathcal{D}_\omega) \tag{7.37}$$

where r_c is number of configurations of class variable C (number of classes). $P(c_k|\mathcal{D}_\omega)$ is the proportion of the number of cases in $\mathcal{D}_\omega$ when a class variable is in configuration k to the total number of cases in $\mathcal{D}_\omega$. Entropy $\text{Ent}(\mathcal{D}_\omega)$ measures the amount of information, in *bits*, required to specify the classes in $\mathcal{D}_\omega$.

We initially assume that interval ω contains all possible values of attribute A, that is $\omega = (-\infty, \infty)$. Interval ω is then recursively partitioned through binary cuts to establish a set discretization intervals, or discretization bins. We will denote the set of discretization bins by $\mathcal{B}$.

Binary Cut

Let $t \in \omega$ denote a threshold value, a *cat point*, for partitioning interval ω. Partitioning creates two new intervals

$$\begin{aligned}\omega_1 &= \langle a \;:\; a \in \omega \text{ and } a < t\rangle \\ \omega_2 &= \langle a \;:\; a \in \omega \text{ and } a \geqslant t\rangle\end{aligned}$$

This corresponds to partitioning the dataset $\mathcal{D}_\omega$ into two datasets $\mathcal{D}_{\omega_1}$ and $\mathcal{D}_{\omega_2}$. Dataset $\mathcal{D}_{\omega_1}$ contains these cases for which value of attribute A is less then t, Dataset $\mathcal{D}_{\omega_2}$ contains remaining cases in $\mathcal{D}_\omega$:

$$\begin{aligned}\mathcal{D}_{\omega_1} &= \{\mathbf{x}^{(l)} \;:\; \mathbf{x}^{(l)} \in \mathcal{D}_\omega \text{ and } a^{(l)} < t\} \\ \mathcal{D}_{\omega_2} &= \mathcal{D}_\omega \setminus \mathcal{D}_{\omega_1}\end{aligned}$$

Fayyad and Irani (1993) define a *class information entropy of the partition induced by* t as:

$$E(A, t; \mathcal{D}_\omega) \;=\; \frac{\|\mathcal{D}_{\omega_1}\|}{\|\mathcal{D}_\omega\|}\,\text{Ent}(\mathcal{D}_{\omega_1}) \;+\; \frac{\|\mathcal{D}_{\omega_2}\|}{\|\mathcal{D}_\omega\|}\,\text{Ent}(\mathcal{D}_{\omega_2}) \tag{7.38}$$

Optimal cut point $\hat{t}$ for partitioning interval ω is determined by minimizing $E(A, t; \mathcal{D}_\omega)$ over all candidate cut points t.

Note that the above minimization of the class information entropy is similar to heuristics used for induction of decision trees by algorithms like ID3 (Quinlan, 1986), C4.5 (Quinlan, 1993), CART (Breiman, Friedman, Olshen, and Stone, 1984), or CN2 (Clark and Niblett, 1989).

"Cut or not to cut?"

To prevent indefinite recursive partitioning of created intervals, we need to decide whether cutting of a particular interval is beneficial or not. It is a binary decision problem that can be solved by Bayesian approach. Let H_ω be hypothesis that partitioning interval ω using a binary cut, described above, would improve error rate of classifier that uses the discretization. Let $\mathcal{N}_\omega$ be a null hypothesis, that is the hypothesis that would result if the partitioning of ω was rejected. For this decision problem, the Bayesian decision strategy is to accept partitioning of ω when:

$$p(H_\omega|\mathcal{D}) \; > \; p(\mathcal{N}_\omega|\mathcal{D})$$

Unfortunately, there is no easy way to compute probabilities $p(H_\omega|\mathcal{D})$ and $p(\mathcal{N}_\omega|\mathcal{D})$ directly. Fayyad and Irani (1993) proposed to approximate these probabilities using Minimum Description Length Principle (Rissanen, 1978). This led them to the following criterion. Partitioning of interval ω is accepted if and only if

$$Gain(A, t; \mathcal{D}_\omega) \; > \; \frac{\log_2(\|\mathcal{D}_\omega\| - 1)}{\|\mathcal{D}_\omega\|} \; + \; \frac{\Delta(A, t; \mathcal{D}_\omega)}{\|\mathcal{D}_\omega\|} \tag{7.39}$$

where $Gain(A, t; \mathcal{D}_\omega)$ is the information gain of the cut point t:

$$Gain(A, t; \mathcal{D}_\omega) \; = \; \text{Ent}(\mathcal{D}_\omega) \; - \; \frac{\|\mathcal{D}_{\omega_1}\|}{\|\mathcal{D}_\omega\|} \, \text{Ent}(\mathcal{D}_{\omega_1}) \; - \; \frac{\|\mathcal{D}_{\omega_2}\|}{\|\mathcal{D}_\omega\|} \, \text{Ent}(\mathcal{D}_{\omega_2}) \tag{7.40}$$

and

$$\Delta(A, t; \mathcal{D}_\omega) \; = \; \log_2(3^{r_{c\omega}} - 2) - [r_{c\omega}\text{Ent}(\mathcal{D}_\omega) - r_{c\omega_1}\text{Ent}(\mathcal{D}_{\omega_1}) - r_{c\omega_2}\text{Ent}(\mathcal{D}_{\omega_2})] \, . \tag{7.41}$$

Algorithm 7.7 Minimal entropy discretization heuristic.

1. **for** every continuous attribute A_i **do**
2. $\quad P_i \leftarrow$ PARTITION-INTERVAL $((-\infty, \infty),\ A_i,\ \mathcal{D})$

Algorithm 7.8 Recursive minimal entropy interval partitioning.

PARTITION-INTERVAL$(\omega, A, \mathcal{D})$

1. $\mathcal{D}_\omega \leftarrow \{D^{(l)} : D^{(l)} \in \mathcal{D}$ and $a^{(l)} \in \omega\}$
2. Find cut point $t \in \omega$ such that $E(A, t; \mathcal{D}_\omega)$ is minimized.
3. **if** $Gain(A, t, \mathcal{D}_\omega) > \dfrac{\log(\|\mathcal{D}_\omega\| - 1)}{\|\mathcal{D}_\omega\|} + \dfrac{\Delta(A, t; \mathcal{D}_\omega)}{\|\mathcal{D}_\omega\|}$ **then**
4. $\quad \omega_1 \leftarrow \langle x : x \in \omega$ and $x < t\rangle$
5. $\quad \omega_2 \leftarrow \langle x : x \in \omega$ and $x \geqslant t\rangle$
6. $\quad \mathcal{B}_1 \leftarrow$ PARTITION-INTERVAL$(\omega_1, A, \mathcal{D})$
7. $\quad \mathcal{B}_2 \leftarrow$ PARTITION-INTERVAL$(\omega_2, A, \mathcal{D})$
8. $\quad \hat{\mathcal{B}} \leftarrow \mathcal{B}_1 \cup \mathcal{B}_2$
9. **else**
10. $\quad \hat{\mathcal{B}} \leftarrow \{\omega\}$
11. **return** $\hat{\mathcal{B}}$

In general, dataset $\mathcal{D}_\omega$ may not contain examples for each possible configuration of class variable C. The number of class variable configurations that have examples in data set $\mathcal{D}_\omega$ is demoted by $r_{c\omega}$. Similarly for datasets $\mathcal{D}_{\omega_1}$ and $\mathcal{D}_{\omega_1}$, it is denoted by $r_{c\omega_1}$ and $r_{c\omega_1}$, respectively.

The algorithm

A discretization algorithm based on the minimal entropy heuristic recursively partitions the domain of each of the continuous attributes. Its working is demonstrated in Alg. 7.7 and Alg. 7.8.

Chapter 8

Experimental Results

This chapter presents the results of four experiments. The first experiment used the categorical data about left ventricular perfusion recorded by cardiologists to estimate the reference error rate for classification based on features extracted from SPECT image. The next two experiments use features extracted from 3D SPECT images to perform classification of the left ventricular perfusion. The last experiment benchmarks our new family of Bayesian network classifiers using datasets from the University of California at Irvine Machine Learning Repository.

Generation of cross validation partition, used during experiments, and discretization of continuous attributes (the minimal entropy heuristic) had been performed using MLC++ package (Kohavi, Sommerfield, and Dougherty, 1997). Note, each of the training partitions was discretized separately, then the discovered scheme was used to discretize a corresponding test partition. MLC++ had been also used as a front end to C4.5 classifier. Implementation of C4.5 classifier was the one provided with (Quinlan, 1993).

8.1 Perfusion Classification Error Reference Level

Figure 4.2 shows a simplified perfusion classification process that is deterministic and entirely based on classification of perfusion in each of the regions of interest. It is reasonable

to stipulate that in practice this classification is more complex, non-deterministic, and involves a number of other factors that are not recorded in the database or are recorded only imprecisely. Cardiologist's interpretation process is, to a large, extent qualitative. The data recorded in the database are, by their nature, quantitative. Intentionally, there is only a limited number of codes that can be used to record perfusion. The intention is that this will standardize the recording process by limiting the number of recorded defect codes, each code having a clear description. This is a *de facto* discretization process. A number of cardiologists contributed to the database. Each of them may have a different bias how to code that do not clearly belong to a single category. Also, interpretation of cardiac SPECT images is complex and not an unambiguous process. There is a significant variability in diagnosing of the same images between cardiologists (Cuaron et al., 1980). Additionally, there is also some possibility of data errors present, eg. typos.

We use the perfusion classification recorded in the database by cardiologists as a *golden standard* while constructing classifier based on features extracted from SPECT images. The main objective of the experiment described in this section is to estimate *goodness* of the golden standard by estimating the perfusion classification error rate. In this experiment the class variable will represent overall perfusion impression code recorded in the database. The attributes will represent partial perfusion code recorded for each of the 22 regions of interest presented in Fig. 4.1 and other information recorded in a study worksheet.

Note, that if classification of myocardial perfusion was completely deterministic, given the classification within ROIs, then the reference classification error would be zero. Since it is not, we wanted to estimate the perfusion classification error rate present in the golden standard. This will help us to set a more realistic performance requirements for classifiers that use only features extracted from SPECT images.

The secondary goal of this experiment is to see if some additional clinical information that is recorded in the database, for instance, patient's gender, weight, etc., may be useful in perfusion classification. We want to determine whether additional patient information

recorded in the database may improve classification error rate.

The experiment described here is an update to similar experiment described by us in (Sacha et al., 2000). The current experiment uses newer database, larger by about 200 cases, and the new family of the Bayesian network classifier described in Chapter 7.

The upper boundary of the classification noise that is present in the database can be estimated by constructing a number of classifiers and measuring their error rate using cross validation. We used a C4.5 classifier (Quinlan, 1993), thirteen of the classifiers described in Chapter 7, and the constant classifier[1]. We used only records that had no missing, incorrect, or multiple values for considered attributes. Three sets of attributes were used to estimate classification error level:

- Diagnosis by a cardiologist for each of the 22 regions of interests (Figure 4.1). We refer to this 22 attribute set as Code22. Each of the attributes can take on seven values which describe defect type, coded as **NL** - normal, **F** - fixed, **R** - reversible, **P** - partially reversible, **E** - equivocal, **X** - reverse redistribution, **ART** - artifact. This set has 1762 cases.

- A new dataset was obtained from Code22 set by counting how many times each of the defect codes were present within the 22 regions for a given case. These data became a set of seven attributes taking on integer values from 0 to 22. Similarly to the Code22 set, this set has 1762 cases. We refer to this data set as Count.

- Still another set was created using criteria such as relevance, completeness, and correctness of attribute values. This resulted in 16 attributes with a reasonably high number of corresponding entries in the Code22 data set. The following variables were

[1]The *constant classifier* predicts a constant class - the majority class in the training set. This is the simplest classifier that can be build. It is used to establish a base performance level. If no classifier can reach performance better then the constant classifier it suggests that a dataset does not contain any information useful for classification; a dataset contains only noise.

Table 8.1: Summary of the estimation of reference classification error level using ten-fold cross validation. The first number represents an error in percent; the second is the standard deviation of the sample mean.

Data set	# attributes	# cases	Constant classifier	C4.5	Discrete naïve Bayes	TAN	Best new BNC
Code22	22	1762	73.38±1.25	21.28 ± 1.04	20.94 ± 1.28	20.37±1.48	**19.30±1.15**
Count	7	1762	73.38±1.25	**16.63±0.72**	18.05 ± 0.98	17.82±0.99	17.59 ± 0.99
Extras	13	1619	73.44±0.74	66.89 ± 0.94	60.72 ± 0.83	60.22±1.29	**59.85±1.01**
Code22 & Extras	35	1619	73.44±0.74	22.54 ± 1.58	21.74 ± 1.09	20.45±0.92	**19.64±0.77**
Count & Extras	20	1619	73.44±0.74	18.78 ± 0.85	17.66 ± 1.04	18.47±0.92	**17.60±0.84**
Code22 & Count	29	1762	73.38±1.25	**17.54±0.90**	19.69 ± 1.23	18.22±1.12	17.70 ± 1.07
Code22 & Count & Extras	42	1619	73.44±0.74	18.53 ± 0.85	20.26 ± 0.98	17.91±0.91	**17.66±0.87**

Table 8.2: Estimation of reference classification error level using ten-fold cross validation and FAN classifier.

Dataset	FAN				
	HGC	SB	LC	LOO	CV1010
Code22	19.81 ± 1.54	19.64 ± 1.24	**19.30 ± 1.15**	19.58 ± 1.23	20.34 ± 1.13
Count	**17.59 ± 0.99**	17.93 ± 1.04	17.76 ± 1.04	17.65 ± 1.01	17.70 ± 1.01
Extras	60.53 ± 1.27	**60.22 ± 0.86**	60.47 ± 0.95	60.41 ± 0.94	60.41 ± 0.94
Code22 & Extras	20.57 ± 0.86	**20.14 ± 0.96**	20.38 ± 1.01	20.94 ± 0.81	21.19 ± 0.88
Count & Extras	18.10 ± 1.06	**17.60 ± 0.84**	17.60 ± 0.98	17.79 ± 0.97	17.66 ± 0.97
Code22 & Count	18.33 ± 1.27	17.93 ± 1.23	17.99 ± 1.31	**17.76 ± 1.27**	17.88 ± 1.30
Code22 & Count & Extras	19.02 ± 1.02	18.40 ± 0.80	18.28 ± 0.99	**17.85 ± 0.85**	18.03 ± 0.84

selected: sex, age, height, weight, body surface area coefficient (BSA), diabetes mellitus, family history, HTN, smoker/non-smoker, chest pain, high cholesterol, prior MI, and left ventricular size. This set has 1619 cases. We refer to it as Extras.

The first two sets contain only information from the interpretation of SPECT images. The last one contains other attributes from patient records that might influence the diagnosis of myocardial perfusion.

We used these sets of attributes and their combinations to build seven datasets: Code22,

Table 8.3: Estimation of reference classification error level using ten-fold cross validation, STAN and STAND classifiers.

Dataset	STAN			STAND		
	HGC	SB	LC	HGC	SB	LC
Code22	22.02 ± 1.15	35.75 ± 1.64	22.64 ± 1.12	**19.97 ± 1.53**	43.02 ± 1.14	20.43 ± 1.18
Count	**17.82 ± 0.99**	**17.82 ± 0.99**	**17.82 ± 0.99**	17.76 ± 1.00	17.76 ± 1.00	17.76 ± 0.94
Extras	**59.85 ± 1.01**	60.96 ± 1.21	59.35 ± 0.90	61.70 ± 0.96	62.87 ± 1.06	59.97 ± 1.13
Code22 & Extras	22.05 ± 1.19	26.13 ± 1.42	23.59 ± 0.98	**19.64 ± 0.77**	24.77 ± 1.01	20.81 ± 0.95
Count & Extras	18.47 ± 0.90	19.09 ± 1.26	18.03 ± 0.95	17.91 ± 1.05	**17.60 ± 1.10**	18.16 ± 1.13
Code22 & Count	21.51 ± 1.02	25.42 ± 0.98	19.69 ± 0.99	18.05 ± 1.08	19.01 ± 0.90	**17.70 ± 1.07**
Code22 & Count & Extras	21.62 ± 1.33	26.19 ± 1.40	19.64 ± 0.91	**17.66 ± 0.87**	19.15 ± 1.09	18.44 ± 0.92

Count, Extra, Code22 & Extra, Count & Extra, Code22 & Count, and Code22 & Count & Extra. Then-fold cross validation has been used. The summary of the classification results for these datasets is presented in Table 8.1. Results for selected new Bayesian network classifiers (BNC) are presented in Tables 8.2 and 8.3. The first number is the mean cross-validation classification error in percent, the second is the standard deviation of the mean[2].

The following conclusions can be drawn from Table 8.1:

- The data seem to have relatively high classification noise. The lowest classification error in the table is above 16%.

- Representation of the regional classification codes, the Count data set, seems to be "easier" for learning.

- Additional attributes (Extras) contain some useful information for the classification. The error rate for the Extras data set is high, however it is over 13% lower than that

[2] *Experimental standard deviation of the mean* of a sample $x_1, \ldots, x_\phi$:

$$\sigma(\bar{x}) = \sqrt{\frac{1}{\phi(1-\phi)} \sum_{i=1}^{\phi} (x - \bar{x})^2}$$

where $\bar{x} = \frac{1}{\phi} \sum_{i=1}^{\phi} x_i$ is the experimental mean of the sample. The standard deviation of the mean is not the standard deviation of the sample. It shows how variable the mean is, which is smaller than the variability of the sample itself by a factor of samples' size, see (Rice, 1988).

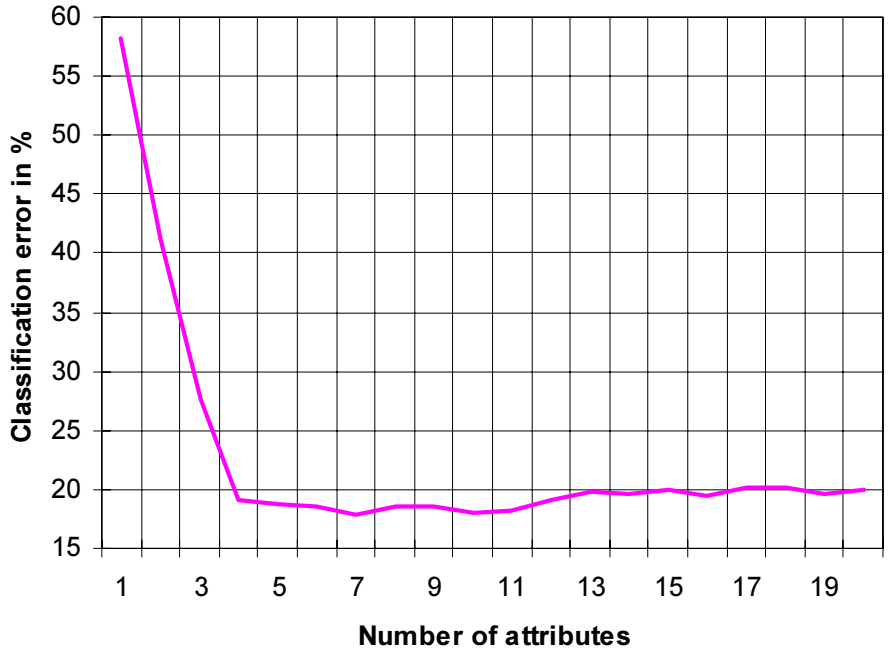


Figure 8.1: Classification error as a function of number of attributes used for classification for dataset Count & Extras.

of the constant classifier, suggesting that some of the attributes in the Extra data set *are* correlated to the classification category.

- Classification using the extended data sets, i.e. regional information with added attributes, suffers from increased classification error.

In our previous work, (Sacha et al., 2000), we studied the relation between number of attributes selected for classification and the classification error. In particular, we studied the Count & Extras dataset created from earlier version of database. That dataset contained 1,433 cases, only 11% less than the current one. The attributes were first ordered according to their contribution to the classification goal based on analysis of classification rules generated by C4.5 classifier. As expected, attributes from Count set ranked highest. We started with a dataset that contained only one attribute, with the highest rank, and performed ten-time cross validation test using C4.5 classifier. Then, we added the second attribute to the

dataset and repeated the cross-validation test; and so on. The results are shown in Fig. 8.1. These results agree with the one presented in Table 8.1: addition of extra attributes from database that are not directly related to myocardial perfusion does not improve classification results, and may actually increase the error. This is despite that Table 8.1 shows that set Extras on its own contain information useful for classification. This is not a contradiction. Larger number of attributes in the dataset means that a classifier needs to be described by a larger number of parameters. As discussed in Chapter 7, if we keep the number of cases in a training dataset constant and increase the number of parameters in a classifier, the variance of estimation of the classifier parameters using that dataset increases. Each of the parameters is estimated with a lower accuracy that may lead to a lower classifier performance. There is usually some optimal tradeoff between the number of attributes in the dataset and the classification error. This is well illustrated in Fig. 8.1. Around seven attributes, for this particular dataset, seems to give the most optimal results. Based on these results, in order to minimize parameter variance, we decided not to include attributes from the Extras set in the remaining tests presented in this Chapter.

8.2 Partial Classification of Left Ventricular Perfusion

This section presents the core experimental results of this work. A partial classification is performed in each of the twenty two 3D regions of interest (Section 4.1), for females and males separately. We tested all of the Bayesian network classifiers presented in Chapter 7. For reference, we also include classification results using the constant classifier, C4.5 classifier, and continuous version of the naïve Bayes classifier.

We had the following goals while conducting this experiment:

1. The main goal of this dissertation: how useful is Bayesian learning for classification of inherently noisy cardiac SPECT images.

2. How do new Bayesian learning algorithms compare to other algorithms.

3. Which new Bayesian network search algorithms perform best on average.

4. Which Bayesian network quality measure performs best on average.

5. Evaluate the quality of the datasets used for this experiment.

This is the largest study presented in this work and from the statistical point of view it is best suited for drawing conclusions about performance of the new Bayesian learning methods. We use 44 datasets having 16 attributes each (one dataset for each of the 22 3D regions of interest, for females and males separately). Then-fold cross validation is performed, and 29 classification algorithms are tested. Total of 12,760 individual classification tests have been performed. The results are summarized in the tables presented on the following pages. We show cross-validation result for each of the algorithms and each of the datasets.

8.2.1 Feature Extraction and Creation of Datasets

The features were extracted from 3D SPECT images using a technique described in Section 4.4.3. We have used more than a single 2D slice for each of the five views (see Figure 4.1). This is to compensate for ambiguity of using exactly the same slice that may have been used by an evaluating physician while recording perfusion codes. For each of the short axis views we used three slices; search radius was set to 9 for females and 10 for males. For the horizontal long axis view we used three slices; search radius was set to 11 for females and 12 for males. For the vertical long axis view we used two slices; search radius was set to 10 for females and 12 for males. Number of slices per view for females and males was coincidentally the same, although it was not intentional. The number of slices and radius of search has been determined by inspecting each of the normal left ventricle models independently.

For each view in 3D **rest** and **stress** images, 2D MAX and TOT images were created using cylindrical radial search, as described in Section 4.4.1. Each of the 2D images has been normalized by dividing each pixel value by the largest pixel intensity found in the

Table 8.4: Case counts for partial LV perfusion classification tests: Females.

ROI Name	Defect Codes						Total
	NL	F	P	R	X	A	
SHORT_AXIS_AP_ANT	67	10	10	8	1	28	124
SHORT_AXIS_AP_LAT	110	4	6	2	0	2	124
SHORT_AXIS_AP_INF	106	12	2	2	0	2	124
SHORT_AXIS_AP_SEPT	114	2	1	5	0	2	124
SHORT_AXIS_MID_ANT	57	13	6	15	2	31	124
SHORT_AXIS_MID_ANT_LAT	99	7	3	8	0	7	124
SHORT_AXIS_MID_INF_LAT	106	9	2	6	0	1	124
SHORT_AXIS_MID_INF	109	9	2	3	0	1	124
SHORT_AXIS_MID_ANT_SEPT	106	4	2	10	0	2	124
SHORT_AXIS_BASAL_ANT	60	14	9	11	1	29	124
SHORT_AXIS_BASAL_ANT_LAT	107	6	2	4	0	4	123
SHORT_AXIS_BASAL_INF_LAT	104	10	3	7	0	0	124
SHORT_AXIS_BASAL_INF	103	11	3	3	1	2	123
SHORT_AXIS_BASAL_ANT_SEPT	102	3	4	9	1	5	124
HORIZ_LONG_SEPT	114	0	0	7	1	2	124
HORIZ_LONG_APICAL	100	8	5	7	1	2	123
HORIZ_LONG_LAT	103	6	4	7	1	3	124
HORIZ_LONG_BASAL	113	5	2	4	0	0	124
VERT_LONG_ANT	61	12	6	14	1	28	122
VERT_LONG_APICAL	103	8	5	5	1	2	124
VERT_LONG_INF	105	11	2	5	0	1	124
VERT_LONG_BASAL	109	10	1	3	1	0	124
Total	2158	174	80	145	12	154	
	79%	6%	3%	5%	0%	6%	

corresponding 3D image. For each of the ROI, we extracted four features from 2D images: maximum intensity in the region, mean intensity, median intensity, and standard deviation of the intensity in the region. This gives a total of 16 features for each ROI. A separate dataset has been created for each ROI; separate for males and females.

Tables 8.4 and 8.5 present case counts for females and males for each of the diagnosis types (**NL** – normal, **F** – fixes defect, **P** – partially reversible defect, **R** – reversible defect, **X** – defect showing reverse redistribution, **A** – artifact). We used cases that had complete cardiologist's evaluation record for each of the 22 ROI. For males we used cases that had the normal left ventricle model correlation match 0.95 or better (see Section 4.3); this resulted in 170 patient cases. For females we used cases that had the normal left ventricle model correlation match 0.93 or better; this resulted in 124 cases.

Table 8.5: Case counts for partial LV perfusion classification tests: Males.

ROI Name	Defect Codes						Total
	NL	F	P	R	X	A	
SHORT_AXIS_AP_ANT	137	10	8	11	1	3	170
SHORT_AXIS_AP_LAT	147	8	7	7	1	0	170
SHORT_AXIS_AP_INF	64	53	22	16	4	9	168
SHORT_AXIS_AP_SEPT	154	6	4	5	0	1	170
SHORT_AXIS_MID_ANT	141	5	9	13	0	2	170
SHORT_AXIS_MID_ANT_LAT	157	2	5	5	1	0	170
SHORT_AXIS_MID_INF_LAT	104	23	20	12	2	8	169
SHORT_AXIS_MID_INF	67	53	26	10	2	11	169
SHORT_AXIS_MID_ANT_SEPT	145	6	8	9	1	0	169
SHORT_AXIS_BASAL_ANT	148	5	6	8	1	2	170
SHORT_AXIS_BASAL_ANT_LAT	164	1	1	2	1	1	170
SHORT_AXIS_BASAL_INF_LAT	101	28	21	9	0	10	169
SHORT_AXIS_BASAL_INF	62	50	27	13	2	15	169
SHORT_AXIS_BASAL_ANT_SEPT	146	3	9	11	0	1	170
HORIZ_LONG_SEPT	139	5	9	15	1	0	169
HORIZ_LONG_APICAL	111	31	7	19	1	1	170
HORIZ_LONG_LAT	143	11	10	4	2	0	170
HORIZ_LONG_BASAL	151	11	10	4	2	0	178
VERT_LONG_ANT	137	6	11	13	1	2	170
VERT_LONG_APICAL	95	32	11	26	2	3	169
VERT_LONG_INF	54	53	29	19	3	10	168
VERT_LONG_BASAL	141	17	4	3	0	5	170
Total	2708	419	264	234	28	84	
	72%	11%	7%	6%	1%	2%	

The last column in Tables 8.4 and 8.5 shows the total number of cases in each dataset. This not always sums to 124 and 170, respectively, since there were some coding errors in the databases that we were not able to resolve: there was more than a single code recorded for particular ROI. We did not drop these cases completely since they had correct diagnosis for other ROI and could be used for creation of other datasets.

The last two rows in Tables 8.4 and 8.5 show total number of times each defect was present in 22 ROIs. Due to low number of cases and low relative count for defect types, other than normal, we combined all the defect codes into one **abnormal**. Thus, each of the datasets used for experiments had two classes, **normal** and **abnormal**, and 16 attributes.

8.2.2 Experiments

Experimental results are presented in Tables 8.6 to 8.11. We abbreviated names to narrow the ROI Name column and fit more columns per page. Tables 8.6 presents results for the reference classifiers and overall best result for the new Bayesian network classifiers. Tables 8.7 to 8.11 present results for FAN, STAN, STAND, SFAN, and SFAND classifier, respectively. The first number is the ten-fold cross validation error. The number after $\pm$ is the standard deviation of the mean. Numbers in bold indicate the lowest error rate in a given table for a given dataset. Numbers with gray background indicate the smallest error for a given dataset among all of the tested classifiers.

Bottom of each table shows two performance indicators for each of the classifiers: an average error rate (lower is better) and an average advantage ratio (higher is better). The advantage ratio compares performance of a classifier to the constant classifier; it will be described in detail in the next section on page 122.

Table 8.6: Ten-fold cross-validation error of the partial LV perfusion classification for reference classifiers and summary for new Bayesian network classifiers.

	ROI Name	Constant	C4.5	Naïve Bayes	TAN	Best New BNC
Female	SHORT_AP_ANT	46.09 ± 3.86	37.69 ± 3.06	33.65 ± 4.05	31.99 ± 4.03	**27.18 ± 3.32**
	SHORT_AP_LAT	11.45 ± 2.84	9.68 ± 2.57	14.36 ± 3.40	11.99 ± 3.56	**8.08 ± 2.39**
	SHORT_AP_INF	14.42 ± 3.05	20.71 ± 3.43	14.29 ± 3.18	15.06 ± 3.91	**13.53 ± 3.31**
	SHORT_AP_SEPT	8.01 ± 2.35	**7.18 ± 2.20**	9.68 ± 2.30	8.01 ± 2.35	8.01 ± 2.35
	SHORT_MID_ANT	46.03 ± 4.03	36.35 ± 3.96	31.35 ± 2.88	32.24 ± 3.55	**27.44 ± 2.77**
	SHORT_MID_ANT_LAT	20.13 ± 3.42	17.69 ± 5.07	20.19 ± 3.83	16.79 ± 4.48	**15.19 ± 4.89**
	SHORT_MID_INF_LAT	14.55 ± 1.11	13.78 ± 2.42	15.96 ± 2.55	15.32 ± 2.73	**12.12 ± 1.78**
	SHORT_MID_INF	11.79 ± 3.85	11.09 ± 3.48	8.72 ± 2.95	7.88 ± 2.30	**6.28 ± 1.93**
	SHORT_MID_ANT_SEPT	14.42 ± 3.05	12.76 ± 3.11	14.62 ± 2.45	16.09 ± 2.96	**11.35 ± 1.85**
	SHORT_BASAL_ANT	59.10 ± 4.04	25.83 ± 3.58	31.92 ± 5.97	28.59 ± 6.00	**22.44 ± 3.61**
	SHORT_BASAL_ANT_LAT	13.01 ± 2.53	21.09 ± 3.69	17.05 ± 3.42	16.92 ± 4.45	**10.51 ± 2.11**
	SHORT_BASAL_INF_LAT	**16.22 ± 1.80**	21.03 ± 3.40	21.15 ± 3.32	18.65 ± 2.52	**16.22 ± 1.80**
	SHORT_BASAL_INF	**16.28 ± 2.11**	21.92 ± 3.82	22.05 ± 3.74	17.95 ± 3.23	17.12 ± 2.88
	SHORT_BASAL_ANT_SEPT	**17.76 ± 2.36**	26.79 ± 5.54	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**
	HORIZ_LONG_SEPT	7.95 ± 2.70	**5.58 ± 2.08**	7.12 ± 2.47	5.58 ± 1.74	**5.58 ± 1.74**
	HORIZ_LONG_APICAL	18.46 ± 3.86	21.92 ± 3.23	25.13 ± 2.70	18.59 ± 3.76	**15.45 ± 2.57**
	HORIZ_LONG_LAT	16.86 ± 3.24	16.73 ± 3.98	16.03 ± 2.00	16.03 ± 3.42	**12.76 ± 2.40**
	HORIZ_LONG_BASAL	8.85 ± 2.59	9.68 ± 2.40	12.12 ± 2.75	10.45 ± 1.71	**7.95 ± 2.39**
	VERT_LONG_ANT	58.14 ± 2.02	39.36 ± 4.44	31.86 ± 4.07	**30.19 ± 4.15**	**30.19 ± 4.15**
	VERT_LONG_APICAL	**16.86 ± 2.81**	20.90 ± 3.19	21.73 ± 3.60	17.63 ± 2.87	**16.86 ± 2.81**
	VERT_LONG_INF	**15.13 ± 3.08**	21.54 ± 3.82	29.81 ± 4.19	15.90 ± 3.18	**15.13 ± 3.08**
	VERT_LONG_BASAL	12.05 ± 2.99	19.29 ± 4.58	25.71 ± 3.49	12.05 ± 2.99	**11.99 ± 2.43**
Male	SHORT_AP_ANT	19.41 ± 1.53	26.47 ± 2.36	19.41 ± 2.33	17.65 ± 2.32	**16.47 ± 2.29**
	SHORT_AP_LAT	13.53 ± 2.33	12.94 ± 2.45	12.94 ± 2.11	11.18 ± 2.23	**10.00 ± 1.97**
	SHORT_AP_INF	38.09 ± 3.30	39.96 ± 3.51	32.28 ± 2.44	32.90 ± 3.08	**28.68 ± 3.32**
	SHORT_AP_SEPT	**9.41 ± 2.35**	12.94 ± 1.92	13.53 ± 2.64	10.59 ± 1.47	**9.41 ± 2.18**
	SHORT_MID_ANT	**17.06 ± 2.05**	21.18 ± 2.35	20.59 ± 2.67	20.59 ± 2.67	**17.06 ± 2.05**
	SHORT_MID_ANT_LAT	**7.56 ± 1.76**	**7.65 ± 1.26**	8.24 ± 1.57	**7.65 ± 1.76**	**7.65 ± 1.76**
	SHORT_MID_INF_LAT	38.38 ± 4.26	38.38 ± 2.58	33.09 ± 3.26	33.05 ± 3.25	**29.52 ± 2.99**
	SHORT_MID_INF	39.74 ± 4.51	41.32 ± 4.32	27.17 ± 4.46	25.99 ± 4.38	**25.92 ± 5.04**
	SHORT_MID_ANT_SEPT	**14.23 ± 2.53**	17.76 ± 3.04	21.88 ± 3.03	14.82 ± 2.54	**14.23 ± 2.53**
	SHORT_BASAL_ANT	12.94 ± 2.75	14.12 ± 2.66	17.06 ± 2.97	12.94 ± 2.75	**12.35 ± 2.55**
	SHORT_BASAL_ANT_LAT	**3.53 ± 0.96**	4.12 ± 1.26	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**
	SHORT_BASAL_INF_LAT	40.18 ± 4.75	33.68 ± 4.17	33.09 ± 2.89	31.91 ± 3.02	**30.74 ± 3.56**
	SHORT_BASAL_INF	36.65 ± 4.16	37.83 ± 4.00	31.36 ± 3.16	28.90 ± 4.39	**27.72 ± 3.00**
	SHORT_BASAL_ANT_SEPT	**14.12 ± 2.51**	17.65 ± 2.63	18.82 ± 3.70	**14.12 ± 2.51**	**14.12 ± 2.51**
	HORIZ_LONG_SEPT	17.79 ± 2.81	13.57 ± 3.03	17.13 ± 2.82	12.46 ± 2.26	**12.39 ± 2.83**
	HORIZ_LONG_APICAL	34.71 ± 3.87	28.24 ± 3.49	25.88 ± 3.19	23.53 ± 3.82	**21.18 ± 2.66**
	HORIZ_LONG_LAT	15.88 ± 3.17	18.82 ± 2.88	14.12 ± 3.53	14.12 ± 3.19	**11.76 ± 2.63**
	HORIZ_LONG_BASAL	11.18 ± 2.05	13.53 ± 2.49	9.41 ± 2.51	8.82 ± 2.19	**8.24 ± 2.18**
	VERT_LONG_ANT	**19.41 ± 2.33**	22.35 ± 2.29	20.00 ± 2.51	20.00 ± 2.51	**19.41 ± 2.33**
	VERT_LONG_APICAL	43.75 ± 3.59	44.93 ± 1.89	39.60 ± 3.78	41.40 ± 2.27	**36.07 ± 3.52**
	VERT_LONG_INF	**32.10 ± 3.26**	35.74 ± 3.19	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**
	VERT_LONG_BASAL	17.06 ± 3.09	16.47 ± 3.49	15.88 ± 3.17	15.88 ± 3.17	**14.71 ± 3.07**
	Average error	21.82	21.78	20.67	18.54	***17.55***
	Average advantage	0.00	-7.85	-6.82	7.51	***11.98***

Table 8.7: Ten-fold cross-validation error rate of the partial left ventricle perfusion classification using FAN classifier.

	ROI Name	FAN				
		HGC	SB	LC	LOO	CV1010
Female	SHORT_AP_ANT	**31.99 ± 4.03**	**31.99 ± 4.03**	32.82 ± 4.15	**31.99 ± 4.03**	**31.99 ± 4.03**
	SHORT_AP_LAT	12.76 ± 3.90	**11.99 ± 3.56**	12.82 ± 3.34	**11.99 ± 3.56**	**11.99 ± 3.56**
	SHORT_AP_INF	14.36 ± 3.03	15.96 ± 3.06	15.96 ± 3.06	**14.29 ± 3.45**	**14.29 ± 3.45**
	SHORT_AP_SEPT	**8.01 ± 2.35**	9.68 ± 2.30	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**
	SHORT_MID_ANT	32.24 ± 3.55	**31.41 ± 2.67**	32.24 ± 3.55	32.24 ± 3.55	32.24 ± 3.55
	SHORT_MID_ANT_LAT	16.86 ± 3.63	16.79 ± 4.48	**16.03 ± 3.73**	16.86 ± 3.63	16.86 ± 3.63
	SHORT_MID_INF_LAT	15.32 ± 2.73	**14.49 ± 2.77**	**14.49 ± 2.81**	15.26 ± 3.36	**14.49 ± 2.81**
	SHORT_MID_INF	**7.95 ± 2.34**	8.72 ± 2.71	8.65 ± 3.16	8.59 ± 3.70	8.59 ± 3.70
	SHORT_MID_ANT_SEPT	15.32 ± 3.07	13.72 ± 2.43	14.42 ± 3.05	15.32 ± 3.07	**12.88 ± 2.13**
	SHORT_BASAL_ANT	28.59 ± 6.00	28.59 ± 6.00	**28.65 ± 4.87**	29.42 ± 5.74	**28.65 ± 4.87**
	SHORT_BASAL_ANT_LAT	16.92 ± 4.45	**14.55 ± 4.07**	16.09 ± 4.54	16.09 ± 4.54	16.09 ± 4.54
	SHORT_BASAL_INF_LAT	**18.65 ± 2.52**	**18.65 ± 2.52**	21.15 ± 2.90	**18.65 ± 2.52**	**18.65 ± 2.52**
	SHORT_BASAL_INF	**17.95 ± 3.23**	**17.95 ± 3.23**	**17.95 ± 3.23**	**17.95 ± 3.23**	**17.95 ± 3.23**
	SHORT_BASAL_ANT_SEPT	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**
	HORIZ_LONG_SEPT	**6.35 ± 2.00**	7.12 ± 2.47	7.12 ± 2.47	**6.35 ± 2.00**	**6.35 ± 2.00**
	HORIZ_LONG_APICAL	**17.82 ± 3.56**	19.55 ± 2.51	19.55 ± 2.80	17.88 ± 3.42	18.72 ± 3.26
	HORIZ_LONG_LAT	16.79 ± 2.98	**15.90 ± 3.25**	15.96 ± 3.50	16.86 ± 3.31	16.86 ± 3.31
	HORIZ_LONG_BASAL	10.45 ± 1.71	10.51 ± 2.45	11.28 ± 1.80	**9.68 ± 2.05**	**9.68 ± 1.64**
	VERT_LONG_ANT	**30.19 ± 4.15**	31.03 ± 4.30	**30.19 ± 4.15**	**30.19 ± 4.15**	**30.19 ± 4.15**
	VERT_LONG_APICAL	**17.63 ± 2.87**	19.17 ± 3.53	21.67 ± 3.56	19.17 ± 3.53	19.17 ± 3.53
	VERT_LONG_INF	15.90 ± 3.18	15.90 ± 3.18	16.67 ± 3.45	**15.13 ± 3.08**	**15.13 ± 3.08**
	VERT_LONG_BASAL	**12.05 ± 2.99**	13.72 ± 3.21	14.55 ± 2.91	13.72 ± 3.21	13.72 ± 3.21
Male	SHORT_AP_ANT	**17.06 ± 2.05**	17.65 ± 2.32	17.65 ± 2.32	18.24 ± 2.39	18.24 ± 2.39
	SHORT_AP_LAT	**11.18 ± 2.23**	11.76 ± 1.52	**11.18 ± 2.23**	**11.18 ± 2.23**	**11.18 ± 2.23**
	SHORT_AP_INF	33.49 ± 2.80	**30.48 ± 2.59**	33.49 ± 2.51	33.49 ± 3.30	33.49 ± 3.30
	SHORT_AP_SEPT	**10.59 ± 1.47**	**10.59 ± 1.47**	**10.59 ± 1.92**	**10.59 ± 1.47**	**10.59 ± 1.47**
	SHORT_MID_ANT	20.59 ± 2.67	20.00 ± 2.35	**18.82 ± 2.11**	20.00 ± 2.51	20.00 ± 2.51
	SHORT_MID_ANT_LAT	**7.65 ± 1.76**	8.24 ± 1.57	**7.65 ± 1.76**	**7.65 ± 1.76**	**7.65 ± 1.76**
	SHORT_MID_INF_LAT	33.05 ± 3.25	**32.46 ± 3.49**	33.05 ± 3.25	33.05 ± 3.25	33.05 ± 3.25
	SHORT_MID_INF	25.99 ± 4.38	**25.92 ± 5.04**	25.99 ± 4.38	25.99 ± 4.38	25.99 ± 4.38
	SHORT_MID_ANT_SEPT	**14.82 ± 2.54**	**14.82 ± 2.54**	15.37 ± 2.65	**14.82 ± 2.54**	**14.82 ± 2.54**
	SHORT_BASAL_ANT	**14.12 ± 2.66**	**14.12 ± 2.66**	**14.12 ± 2.66**	**14.12 ± 2.66**	**14.12 ± 2.66**
	SHORT_BASAL_ANT_LAT	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**
	SHORT_BASAL_INF_LAT	**31.32 ± 3.25**	31.91 ± 3.27	31.91 ± 2.89	31.91 ± 3.02	31.91 ± 3.02
	SHORT_BASAL_INF	29.49 ± 4.17	27.83 ± 3.06	30.66 ± 3.33	**27.72 ± 3.00**	**27.72 ± 3.00**
	SHORT_BASAL_ANT_SEPT	**16.47 ± 3.26**	17.65 ± 3.51	17.65 ± 3.51	**16.47 ± 3.26**	**16.47 ± 3.26**
	HORIZ_LONG_SEPT	**12.46 ± 2.26**	13.01 ± 2.45	13.01 ± 2.28	13.01 ± 2.28	13.01 ± 2.28
	HORIZ_LONG_APICAL	**23.53 ± 3.82**	**23.53 ± 3.82**	24.12 ± 4.15	24.12 ± 4.15	24.12 ± 4.15
	HORIZ_LONG_LAT	12.94 ± 3.14	**12.35 ± 3.09**	13.53 ± 3.29	13.53 ± 3.04	13.53 ± 3.04
	HORIZ_LONG_BASAL	**8.82 ± 2.19**	**8.82 ± 2.19**	**8.82 ± 2.19**	**8.82 ± 2.19**	**8.82 ± 2.19**
	VERT_LONG_ANT	**20.00 ± 2.51**	**20.00 ± 2.51**	**20.00 ± 2.51**	**20.00 ± 2.51**	**20.00 ± 2.51**
	VERT_LONG_APICAL	**39.60 ± 3.23**	**39.60 ± 3.23**	**39.60 ± 3.23**	**39.60 ± 3.23**	**39.60 ± 3.67**
	VERT_LONG_INF	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**
	VERT_LONG_BASAL	**15.88 ± 3.17**	**15.88 ± 3.17**	**15.88 ± 3.17**	**15.88 ± 3.17**	**15.88 ± 3.17**
	Average error	18.56	18.58	18.93	18.62	**18.55**
	Average advantage	7.05	6.14	4.69	6.81	**7.24**

Table 8.8: Ten-fold cross-validation error rate of the partial left ventricle perfusion classification using STAN classifier.

	ROI Name	STAN				
		HGC	SB	LC	LOO	CV1010
Female	SHORT_AP_ANT	31.22 ± 5.26	34.36 ± 5.32	**28.01 ± 3.59**	31.22 ± 3.52	29.55 ± 3.97
	SHORT_AP_LAT	11.41 ± 3.09	**10.45 ± 2.45**	12.05 ± 2.13	10.58 ± 2.10	12.12 ± 2.75
	SHORT_AP_INF	**13.65 ± 3.32**	14.42 ± 3.05	14.49 ± 2.61	15.26 ± 2.75	15.19 ± 2.96
	SHORT_AP_SEPT	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**
	SHORT_MID_ANT	30.58 ± 3.48	**27.44 ± 2.77**	31.41 ± 3.71	31.35 ± 4.03	31.35 ± 4.03
	SHORT_MID_ANT_LAT	17.69 ± 3.89	20.90 ± 4.04	**15.96 ± 4.82**	**15.96 ± 5.08**	**15.96 ± 5.08**
	SHORT_MID_INF_LAT	12.95 ± 1.35	13.72 ± 1.69	13.78 ± 2.42	**12.88 ± 2.07**	**12.88 ± 2.07**
	SHORT_MID_INF	9.36 ± 3.76	8.59 ± 3.70	**7.05 ± 2.43**	7.82 ± 2.57	7.82 ± 2.57
	SHORT_MID_ANT_SEPT	**11.35 ± 1.85**	**11.35 ± 2.23**	**11.35 ± 2.23**	13.01 ± 3.12	**11.35 ± 2.23**
	SHORT_BASAL_ANT	27.88 ± 4.24	31.28 ± 4.55	**22.56 ± 3.71**	23.33 ± 3.44	23.33 ± 3.44
	SHORT_BASAL_ANT_LAT	13.85 ± 1.77	**11.41 ± 1.84**	14.55 ± 3.46	13.85 ± 2.79	14.55 ± 2.83
	SHORT_BASAL_INF_LAT	**16.22 ± 1.80**	**16.22 ± 1.80**	18.65 ± 2.52	18.65 ± 2.52	17.88 ± 2.49
	SHORT_BASAL_INF	19.55 ± 2.18	18.78 ± 2.80	17.18 ± 3.39	**17.12 ± 2.88**	**17.12 ± 2.88**
	SHORT_BASAL_ANT_SEPT	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**
	HORIZ_LONG_SEPT	6.35 ± 2.00	6.35 ± 2.00	**5.58 ± 1.74**	**5.58 ± 1.74**	**5.58 ± 1.74**
	HORIZ_LONG_APICAL	**17.76 ± 3.08**	19.23 ± 3.86	18.59 ± 3.33	**17.76 ± 3.45**	17.82 ± 3.10
	HORIZ_LONG_LAT	15.19 ± 2.14	**14.36 ± 2.56**	15.19 ± 3.03	15.96 ± 3.34	16.79 ± 2.98
	HORIZ_LONG_BASAL	9.62 ± 2.67	8.78 ± 1.85	8.78 ± 2.83	**7.95 ± 2.39**	**7.95 ± 2.39**
	VERT_LONG_ANT	**31.79 ± 3.97**	**31.79 ± 3.71**	36.92 ± 3.58	36.03 ± 3.78	32.69 ± 3.56
	VERT_LONG_APICAL	**16.86 ± 2.81**	17.63 ± 2.87	**16.86 ± 2.81**	17.63 ± 2.87	17.63 ± 2.87
	VERT_LONG_INF	**15.13 ± 3.08**	**15.13 ± 3.08**	**15.13 ± 3.08**	**15.13 ± 3.08**	**15.13 ± 3.08**
	VERT_LONG_BASAL	12.05 ± 2.99	12.05 ± 2.99	12.82 ± 2.09	**11.99 ± 2.43**	**11.99 ± 2.43**
Male	SHORT_AP_ANT	**17.06 ± 2.05**	**17.06 ± 2.05**	17.65 ± 1.96	18.24 ± 2.23	18.24 ± 2.23
	SHORT_AP_LAT	11.18 ± 2.83	**10.59 ± 1.92**	11.18 ± 1.63	11.76 ± 1.52	11.76 ± 1.52
	SHORT_AP_INF	33.97 ± 1.39	33.42 ± 2.21	**31.62 ± 2.29**	35.29 ± 3.15	34.12 ± 3.47
	SHORT_AP_SEPT	11.18 ± 2.55	10.00 ± 2.16	10.00 ± 2.33	**9.41 ± 2.51**	10.00 ± 2.49
	SHORT_MID_ANT	17.65 ± 1.75	**17.06 ± 2.05**	20.59 ± 2.94	21.18 ± 2.66	21.18 ± 2.66
	SHORT_MID_ANT_LAT	**7.65 ± 1.76**	**7.65 ± 1.76**	**7.65 ± 1.76**	**7.65 ± 1.53**	**7.65 ± 1.53**
	SHORT_MID_INF_LAT	34.23 ± 4.14	**31.84 ± 3.37**	36.65 ± 2.84	35.44 ± 2.24	34.96 ± 2.78
	SHORT_MID_INF	27.87 ± 3.68	28.42 ± 3.27	27.24 ± 4.59	27.79 ± 3.61	**27.17 ± 3.71**
	SHORT_MID_ANT_SEPT	**14.23 ± 2.53**	**14.23 ± 2.53**	**14.23 ± 2.53**	**14.23 ± 2.53**	**14.23 ± 2.53**
	SHORT_BASAL_ANT	**12.35 ± 2.55**	12.94 ± 2.75	12.94 ± 2.75	14.12 ± 2.66	14.12 ± 2.66
	SHORT_BASAL_ANT_LAT	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**
	SHORT_BASAL_INF_LAT	32.50 ± 3.39	33.68 ± 2.99	**31.91 ± 3.38**	33.09 ± 3.70	33.68 ± 3.58
	SHORT_BASAL_INF	**28.90 ± 3.93**	33.09 ± 3.26	30.66 ± 4.15	29.49 ± 3.89	30.66 ± 4.51
	SHORT_BASAL_ANT_SEPT	**14.12 ± 2.51**	**14.12 ± 2.51**	**14.12 ± 2.51**	15.29 ± 2.93	15.29 ± 2.93
	HORIZ_LONG_SEPT	13.60 ± 2.63	15.37 ± 2.65	**12.39 ± 2.83**	14.19 ± 3.30	14.15 ± 3.18
	HORIZ_LONG_APICAL	24.71 ± 3.59	24.12 ± 3.33	**23.53 ± 3.28**	24.12 ± 3.22	25.29 ± 3.40
	HORIZ_LONG_LAT	15.29 ± 3.19	**12.35 ± 2.70**	14.12 ± 2.18	**12.35 ± 1.85**	13.53 ± 2.78
	HORIZ_LONG_BASAL	**8.24 ± 2.18**	9.41 ± 1.80	9.41 ± 1.80	8.82 ± 2.52	8.82 ± 2.52
	VERT_LONG_ANT	**19.41 ± 2.33**	**19.41 ± 2.33**	**19.41 ± 2.33**	**19.41 ± 2.33**	**19.41 ± 2.33**
	VERT_LONG_APICAL	**36.07 ± 3.52**	38.42 ± 4.00	39.56 ± 3.21	40.15 ± 3.05	40.74 ± 3.13
	VERT_LONG_INF	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**
	VERT_LONG_BASAL	15.29 ± 3.19	15.29 ± 3.19	**14.71 ± 3.07**	15.29 ± 3.19	15.88 ± 3.51
	Average error	***18.12***	*18.27*	*18.13*	*18.36*	*18.34*
	Average advantage	*10.14*	***10.37***	*10.30*	*9.77*	*9.35*

Table 8.9: Ten-fold cross-validation error rate of the partial left ventricle perfusion classification using STAND classifier.

	ROI Name	STAND				
		HGC	SB	LC	LOO	CV1010
Female	SHORT_AP_ANT	31.99 ± 4.03	**27.18 ± 3.93**	32.12 ± 3.11	28.01 ± 3.37	29.68 ± 3.38
	SHORT_AP_LAT	11.99 ± 3.56	11.99 ± 3.56	9.62 ± 2.00	**8.08 ± 2.39**	**8.08 ± 2.39**
	SHORT_AP_INF	**13.53 ± 3.31**	14.42 ± 3.05	15.96 ± 2.55	16.79 ± 3.36	16.79 ± 3.16
	SHORT_AP_SEPT	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**
	SHORT_MID_ANT	31.41 ± 3.40	**29.68 ± 4.07**	33.78 ± 2.99	31.47 ± 2.53	32.95 ± 3.12
	SHORT_MID_ANT_LAT	16.03 ± 3.73	16.03 ± 3.73	**15.96 ± 5.08**	19.29 ± 5.31	17.63 ± 4.67
	SHORT_MID_INF_LAT	15.32 ± 2.73	15.26 ± 3.36	13.78 ± 2.42	13.72 ± 2.39	**12.95 ± 2.45**
	SHORT_MID_INF	9.42 ± 3.23	8.59 ± 3.13	7.12 ± 2.47	**7.05 ± 2.92**	**7.05 ± 2.92**
	SHORT_MID_ANT_SEPT	15.32 ± 3.07	**11.35 ± 2.23**	**11.35 ± 2.23**	14.49 ± 3.35	12.05 ± 2.68
	SHORT_BASAL_ANT	27.82 ± 5.80	**22.50 ± 3.71**	23.27 ± 3.56	24.04 ± 3.99	23.27 ± 3.56
	SHORT_BASAL_ANT_LAT	16.92 ± 5.25	**13.01 ± 2.53**	14.68 ± 2.43	15.45 ± 2.57	16.22 ± 2.92
	SHORT_BASAL_INF_LAT	17.82 ± 2.08	**16.22 ± 1.80**	18.65 ± 2.52	18.65 ± 2.52	18.65 ± 2.52
	SHORT_BASAL_INF	17.95 ± 3.23	17.82 ± 3.02	**17.12 ± 2.88**	**17.12 ± 2.88**	**17.12 ± 2.88**
	SHORT_BASAL_ANT_SEPT	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**
	HORIZ_LONG_SEPT	**5.58 ± 1.74**	6.35 ± 2.00	**5.58 ± 1.74**	**5.58 ± 1.74**	**5.58 ± 1.74**
	HORIZ_LONG_APICAL	18.72 ± 2.70	18.72 ± 2.70	**15.45 ± 2.57**	18.65 ± 2.92	19.42 ± 4.01
	HORIZ_LONG_LAT	16.03 ± 3.42	**14.29 ± 2.73**	15.26 ± 2.54	15.19 ± 2.43	15.19 ± 2.43
	HORIZ_LONG_BASAL	10.45 ± 2.12	8.85 ± 2.59	**8.78 ± 2.83**	9.62 ± 2.67	9.62 ± 2.67
	VERT_LONG_ANT	31.03 ± 4.12	30.32 ± 3.89	31.03 ± 3.70	31.09 ± 4.00	**30.26 ± 4.22**
	VERT_LONG_APICAL	17.63 ± 2.87	**16.86 ± 2.81**	**16.86 ± 2.81**	17.63 ± 2.87	17.63 ± 2.87
	VERT_LONG_INF	15.90 ± 3.18	**15.13 ± 3.08**	**15.13 ± 3.08**	**15.13 ± 3.08**	**15.13 ± 3.08**
	VERT_LONG_BASAL	12.82 ± 2.73	**12.05 ± 2.99**	**12.05 ± 2.99**	**12.05 ± 2.99**	**12.05 ± 2.99**
Male	SHORT_AP_ANT	**17.65 ± 2.32**	**17.65 ± 2.32**	18.24 ± 2.05	**17.65 ± 2.32**	**17.65 ± 2.32**
	SHORT_AP_LAT	**11.18 ± 2.23**	**11.18 ± 2.23**	11.76 ± 1.75	12.94 ± 2.11	12.35 ± 2.23
	SHORT_AP_INF	31.73 ± 3.20	**28.68 ± 3.32**	34.67 ± 2.72	37.68 ± 3.24	37.06 ± 2.90
	SHORT_AP_SEPT	10.59 ± 1.47	**9.41 ± 2.35**	10.59 ± 2.45	11.18 ± 2.97	11.18 ± 2.97
	SHORT_MID_ANT	20.00 ± 2.93	**17.06 ± 2.05**	20.59 ± 2.94	20.00 ± 2.66	20.59 ± 2.36
	SHORT_MID_ANT_LAT	8.24 ± 1.57	**7.65 ± 1.76**	8.24 ± 1.57	**7.65 ± 1.53**	**7.65 ± 1.53**
	SHORT_MID_INF_LAT	33.05 ± 3.25	**29.52 ± 2.99**	32.46 ± 3.49	34.26 ± 2.38	34.26 ± 2.96
	SHORT_MID_INF	**25.99 ± 4.11**	**25.99 ± 4.11**	28.97 ± 3.85	28.35 ± 3.87	28.38 ± 4.26
	SHORT_MID_ANT_SEPT	14.82 ± 2.54	**14.23 ± 2.53**	**14.23 ± 2.53**	**14.23 ± 2.53**	**14.23 ± 2.53**
	SHORT_BASAL_ANT	**12.94 ± 2.75**	**12.94 ± 2.75**	**12.94 ± 2.75**	**12.94 ± 2.75**	**12.94 ± 2.75**
	SHORT_BASAL_ANT_LAT	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**
	SHORT_BASAL_INF_LAT	31.91 ± 3.02	31.91 ± 2.89	**30.74 ± 3.56**	31.91 ± 3.71	32.50 ± 3.39
	SHORT_BASAL_INF	**28.31 ± 4.84**	31.88 ± 3.48	30.66 ± 4.51	30.07 ± 3.29	30.66 ± 3.09
	SHORT_BASAL_ANT_SEPT	**14.12 ± 2.51**	**14.12 ± 2.51**	**14.12 ± 2.51**	**14.12 ± 2.51**	**14.12 ± 2.51**
	HORIZ_LONG_SEPT	**12.98 ± 2.59**	**12.98 ± 2.27**	**12.98 ± 3.25**	**12.39 ± 2.96**	**12.98 ± 2.87**
	HORIZ_LONG_APICAL	23.53 ± 3.82	22.94 ± 3.45	22.94 ± 4.25	22.94 ± 3.33	**22.35 ± 3.80**
	HORIZ_LONG_LAT	14.12 ± 2.35	14.12 ± 3.19	**13.53 ± 2.78**	14.12 ± 3.06	**13.53 ± 3.29**
	HORIZ_LONG_BASAL	**8.82 ± 2.19**	**8.82 ± 2.19**	**8.82 ± 2.01**	**8.82 ± 2.01**	**8.82 ± 2.01**
	VERT_LONG_ANT	**19.41 ± 2.33**	**19.41 ± 2.33**	**19.41 ± 2.33**	**19.41 ± 2.33**	**19.41 ± 2.33**
	VERT_LONG_APICAL	40.18 ± 3.75	**38.42 ± 4.10**	39.56 ± 3.33	40.74 ± 3.37	40.74 ± 3.37
	VERT_LONG_INF	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**
	VERT_LONG_BASAL	15.88 ± 3.17	**15.29 ± 3.19**	**15.29 ± 3.19**	**15.29 ± 3.19**	15.88 ± 3.51
	Average error	*18.42*	***17.55***	*18.08*	*18.35*	*18.32*
	Average advantage	*7.73*	***11.98***	*10.57*	*8.97*	*9.30*

Table 8.10: Ten-fold cross-validation error rate of the partial left ventricle perfusion classification using SFAN classifier.

	ROI Name	SFAN				
		HGC	SB	LC	LOO	CV1010
Female	SHORT_AP_ANT	32.88 ± 4.68	29.49 ± 4.85	**28.78 ± 3.70**	28.85 ± 3.15	28.85 ± 3.15
	SHORT_AP_LAT	11.41 ± 3.09	**8.91 ± 2.54**	11.28 ± 3.40	10.51 ± 3.19	10.51 ± 2.93
	SHORT_AP_INF	17.56 ± 3.21	15.13 ± 3.11	16.79 ± 2.67	**14.36 ± 3.03**	15.13 ± 3.54
	SHORT_AP_SEPT	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**
	SHORT_MID_ANT	30.58 ± 3.48	**29.04 ± 2.80**	32.24 ± 2.66	33.08 ± 3.67	33.27 ± 3.69
	SHORT_MID_ANT_LAT	17.69 ± 3.89	20.90 ± 4.04	16.73 ± 5.00	15.96 ± 5.08	**15.19 ± 4.89**
	SHORT_MID_INF_LAT	**12.12 ± 1.78**	15.26 ± 2.94	16.15 ± 1.26	14.49 ± 2.56	14.49 ± 2.56
	SHORT_MID_INF	8.59 ± 3.70	**7.82 ± 3.04**	8.72 ± 2.95	8.59 ± 3.70	9.42 ± 3.79
	SHORT_MID_ANT_SEPT	**11.35 ± 1.85**	12.18 ± 2.26	**11.35 ± 2.23**	13.85 ± 2.83	**11.35 ± 2.23**
	SHORT_BASAL_ANT	27.88 ± 4.24	31.22 ± 5.11	24.04 ± 4.36	23.27 ± 3.97	**22.44 ± 3.61**
	SHORT_BASAL_ANT_LAT	14.68 ± 1.68	**10.51 ± 2.11**	13.72 ± 4.11	13.78 ± 2.91	15.45 ± 3.04
	SHORT_BASAL_INF_LAT	17.88 ± 2.49	**16.22 ± 1.80**	18.65 ± 2.52	18.65 ± 2.52	17.88 ± 2.49
	SHORT_BASAL_INF	19.62 ± 2.56	18.78 ± 3.30	17.88 ± 2.70	**17.12 ± 2.88**	**17.12 ± 2.88**
	SHORT_BASAL_ANT_SEPT	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**
	HORIZ_LONG_SEPT	6.35 ± 2.00	6.35 ± 2.00	7.12 ± 2.47	**5.58 ± 1.74**	**5.58 ± 1.74**
	HORIZ_LONG_APICAL	17.95 ± 2.41	17.88 ± 3.18	17.05 ± 3.08	**16.15 ± 3.12**	17.82 ± 3.10
	HORIZ_LONG_LAT	15.19 ± 2.14	12.82 ± 2.43	**12.76 ± 2.40**	15.96 ± 2.55	17.63 ± 3.26
	HORIZ_LONG_BASAL	9.62 ± 2.67	**8.78 ± 1.85**	**8.78 ± 2.22**	9.55 ± 1.95	9.55 ± 1.95
	VERT_LONG_ANT	**31.79 ± 3.97**	31.86 ± 3.22	31.86 ± 3.64	33.65 ± 3.82	33.46 ± 4.32
	VERT_LONG_APICAL	**17.63 ± 2.87**	19.36 ± 3.07	**17.63 ± 2.87**	21.67 ± 3.56	21.67 ± 3.56
	VERT_LONG_INF	**15.13 ± 3.08**	**15.13 ± 3.08**	**15.13 ± 3.08**	**15.13 ± 3.08**	**15.13 ± 3.08**
	VERT_LONG_BASAL	13.72 ± 3.21	**12.05 ± 2.99**	14.29 ± 3.41	13.72 ± 3.21	13.72 ± 3.21
Male	SHORT_AP_ANT	**16.47 ± 2.29**	17.65 ± 2.15	18.24 ± 2.05	17.65 ± 2.32	17.65 ± 2.32
	SHORT_AP_LAT	11.18 ± 2.83	**10.00 ± 1.97**	11.76 ± 1.52	12.35 ± 2.23	11.18 ± 1.85
	SHORT_AP_INF	33.97 ± 1.39	**31.65 ± 2.70**	33.49 ± 3.18	34.71 ± 3.06	34.08 ± 2.77
	SHORT_AP_SEPT	11.76 ± 2.32	11.18 ± 1.63	**9.41 ± 2.18**	10.59 ± 2.29	10.59 ± 2.29
	SHORT_MID_ANT	**17.65 ± 1.75**	**17.65 ± 1.75**	20.59 ± 2.67	21.18 ± 2.18	21.76 ± 2.33
	SHORT_MID_ANT_LAT	**7.65 ± 1.76**	**7.65 ± 1.76**	**7.65 ± 1.76**	**7.65 ± 1.53**	**7.65 ± 1.53**
	SHORT_MID_INF_LAT	34.23 ± 4.14	**31.84 ± 3.37**	33.64 ± 3.35	34.82 ± 3.03	33.64 ± 2.71
	SHORT_MID_INF	27.87 ± 3.68	27.83 ± 3.94	29.01 ± 4.43	**27.72 ± 3.99**	**27.72 ± 3.99**
	SHORT_MID_ANT_SEPT	**14.23 ± 2.53**	14.82 ± 2.54	**14.23 ± 2.53**	**14.23 ± 2.53**	**14.23 ± 2.53**
	SHORT_BASAL_ANT	**12.35 ± 2.55**	12.94 ± 2.75	12.94 ± 2.75	14.12 ± 2.66	14.12 ± 2.66
	SHORT_BASAL_ANT_LAT	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**
	SHORT_BASAL_INF_LAT	**31.91 ± 3.60**	32.50 ± 3.27	32.50 ± 2.90	33.68 ± 3.47	33.75 ± 4.04
	SHORT_BASAL_INF	**29.49 ± 3.89**	30.66 ± 4.06	31.88 ± 3.13	30.66 ± 3.21	30.66 ± 3.44
	SHORT_BASAL_ANT_SEPT	**14.12 ± 2.51**	**14.12 ± 2.51**	15.29 ± 2.93	16.47 ± 3.26	16.47 ± 3.26
	HORIZ_LONG_SEPT	13.60 ± 2.63	15.33 ± 2.92	**12.39 ± 3.21**	14.78 ± 3.19	14.74 ± 3.18
	HORIZ_LONG_APICAL	24.12 ± 3.22	24.12 ± 2.83	**23.53 ± 3.40**	24.12 ± 3.66	24.12 ± 3.22
	HORIZ_LONG_LAT	14.12 ± 3.06	**13.53 ± 2.49**	**13.53 ± 3.17**	14.12 ± 2.80	14.12 ± 3.06
	HORIZ_LONG_BASAL	**8.24 ± 2.18**	9.41 ± 1.80	9.41 ± 1.80	8.82 ± 2.52	8.82 ± 2.52
	VERT_LONG_ANT	**19.41 ± 2.33**	**19.41 ± 2.33**	**19.41 ± 2.33**	**19.41 ± 2.33**	**19.41 ± 2.33**
	VERT_LONG_APICAL	**36.07 ± 3.52**	38.42 ± 4.00	38.97 ± 3.88	38.38 ± 3.68	**36.07 ± 3.52**
	VERT_LONG_INF	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**
	VERT_LONG_BASAL	**15.29 ± 3.19**	**15.29 ± 3.19**	**15.29 ± 3.19**	**15.29 ± 3.19**	15.88 ± 3.51
	Average error	*18.29*	***18.12***	*18.31*	18.55	*18.49*
	Average advantage	*9.02*	***10.56***	*8.80*	*7.68*	*7.57*

Table 8.11: Ten-fold cross-validation error rate of the partial left ventricle perfusion classification using SFAND classifier.

	ROI Name	SFAND				
		HGC	SB	LC	LOO	CV1010
Female	SHORT_AP_ANT	31.99 ± 4.03	29.49 ± 4.98	28.85 ± 3.15	**27.18 ± 3.32**	29.62 ± 3.49
	SHORT_AP_LAT	12.76 ± 3.90	11.99 ± 3.56	12.12 ± 2.75	**8.08 ± 2.39**	**8.08 ± 2.39**
	SHORT_AP_INF	**14.36 ± 3.03**	14.42 ± 3.05	16.03 ± 2.86	17.50 ± 3.76	16.73 ± 3.53
	SHORT_AP_SEPT	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**	**8.01 ± 2.35**
	SHORT_MID_ANT	31.41 ± 3.40	**30.77 ± 3.28**	34.62 ± 3.06	31.47 ± 2.53	33.08 ± 2.83
	SHORT_MID_ANT_LAT	16.03 ± 3.73	16.03 ± 3.73	**15.96 ± 5.08**	18.53 ± 4.87	17.63 ± 4.67
	SHORT_MID_INF_LAT	14.49 ± 2.77	**14.42 ± 3.19**	16.03 ± 3.04	14.49 ± 3.03	14.49 ± 3.03
	SHORT_MID_INF	8.59 ± 3.70	8.72 ± 2.71	7.88 ± 3.28	**6.28 ± 1.93**	8.59 ± 3.70
	SHORT_MID_ANT_SEPT	15.32 ± 3.07	**12.18 ± 2.26**	12.88 ± 2.95	14.55 ± 3.59	12.88 ± 2.95
	SHORT_BASAL_ANT	27.82 ± 5.80	24.17 ± 3.36	**24.10 ± 3.55**	24.81 ± 3.87	24.81 ± 3.87
	SHORT_BASAL_ANT_LAT	16.09 ± 4.36	**13.01 ± 2.53**	14.68 ± 2.43	15.45 ± 3.35	15.38 ± 3.71
	SHORT_BASAL_INF_LAT	18.65 ± 2.52	**16.22 ± 1.80**	18.65 ± 2.52	18.65 ± 2.52	18.65 ± 2.52
	SHORT_BASAL_INF	17.95 ± 3.23	17.82 ± 3.02	17.88 ± 2.93	**17.12 ± 2.88**	**17.12 ± 2.88**
	SHORT_BASAL_ANT_SEPT	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**	**17.76 ± 2.36**
	HORIZ_LONG_SEPT	**5.58 ± 1.74**	6.35 ± 2.00	6.35 ± 2.00	**5.58 ± 1.74**	**5.58 ± 1.74**
	HORIZ_LONG_APICAL	18.72 ± 2.70	18.78 ± 2.50	20.13 ± 3.51	**17.88 ± 2.65**	17.95 ± 2.41
	HORIZ_LONG_LAT	16.79 ± 3.67	**14.29 ± 2.73**	14.42 ± 2.91	14.36 ± 2.80	14.36 ± 2.80
	HORIZ_LONG_BASAL	10.45 ± 2.12	8.85 ± 2.59	**8.01 ± 2.10**	10.38 ± 1.65	10.45 ± 1.71
	VERT_LONG_ANT	31.03 ± 4.12	31.86 ± 3.67	**31.09 ± 4.00**	33.53 ± 4.07	31.92 ± 3.95
	VERT_LONG_APICAL	17.63 ± 2.87	**16.86 ± 2.81**	17.63 ± 2.87	20.90 ± 3.23	20.90 ± 3.23
	VERT_LONG_INF	15.90 ± 3.18	**15.13 ± 3.08**	**15.13 ± 3.08**	**15.13 ± 3.08**	**15.13 ± 3.08**
	VERT_LONG_BASAL	12.82 ± 2.73	**12.05 ± 2.99**	12.76 ± 3.32	14.49 ± 3.54	15.38 ± 2.83
Male	SHORT_AP_ANT	**17.06 ± 1.85**	17.65 ± 2.32	18.24 ± 2.05	17.65 ± 2.32	17.65 ± 2.32
	SHORT_AP_LAT	**11.18 ± 2.23**	11.76 ± 1.52	**11.18 ± 2.23**	11.76 ± 2.15	**11.18 ± 2.23**
	SHORT_AP_INF	32.90 ± 3.20	**31.07 ± 2.52**	32.94 ± 3.73	34.74 ± 3.39	34.71 ± 2.95
	SHORT_AP_SEPT	10.59 ± 1.47	**9.41 ± 2.35**	10.59 ± 2.45	11.76 ± 2.91	11.76 ± 2.32
	SHORT_MID_ANT	20.00 ± 2.80	**17.06 ± 2.05**	20.59 ± 2.67	21.18 ± 2.18	21.76 ± 2.33
	SHORT_MID_ANT_LAT	8.24 ± 1.57	**7.65 ± 1.76**	8.24 ± 1.57	**7.65 ± 1.53**	**7.65 ± 1.53**
	SHORT_MID_INF_LAT	32.46 ± 3.49	**29.52 ± 2.99**	32.46 ± 3.49	34.85 ± 2.77	34.26 ± 2.96
	SHORT_MID_INF	**25.99 ± 4.11**	**25.99 ± 4.11**	29.01 ± 4.92	27.79 ± 3.81	28.38 ± 4.44
	SHORT_MID_ANT_SEPT	14.82 ± 2.54	**14.23 ± 2.53**	**14.23 ± 2.53**	**14.23 ± 2.53**	**14.23 ± 2.53**
	SHORT_BASAL_ANT	14.12 ± 2.66	**12.94 ± 2.75**	**12.94 ± 2.75**	14.12 ± 2.66	14.12 ± 2.66
	SHORT_BASAL_ANT_LAT	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**	**3.53 ± 0.96**
	SHORT_BASAL_INF_LAT	**31.32 ± 3.25**	**31.32 ± 3.25**	33.09 ± 3.49	33.71 ± 3.38	33.68 ± 3.47
	SHORT_BASAL_INF	**28.90 ± 4.56**	31.91 ± 3.91	29.49 ± 3.89	30.07 ± 4.03	30.66 ± 3.86
	SHORT_BASAL_ANT_SEPT	**14.12 ± 2.51**	**14.12 ± 2.51**	15.29 ± 2.93	16.47 ± 3.26	16.47 ± 3.26
	HORIZ_LONG_SEPT	12.98 ± 2.59	**12.39 ± 3.21**	**12.39 ± 3.21**	**12.39 ± 2.96**	12.98 ± 2.87
	HORIZ_LONG_APICAL	23.53 ± 3.82	**21.18 ± 2.66**	22.94 ± 4.06	24.71 ± 3.37	24.71 ± 3.01
	HORIZ_LONG_LAT	12.94 ± 3.14	**11.76 ± 2.63**	13.53 ± 3.04	13.53 ± 3.29	12.94 ± 3.14
	HORIZ_LONG_BASAL	**8.82 ± 2.19**	**8.82 ± 2.19**	**8.82 ± 2.01**	**8.82 ± 2.01**	**8.82 ± 2.01**
	VERT_LONG_ANT	**19.41 ± 2.33**	**19.41 ± 2.33**	**19.41 ± 2.33**	**19.41 ± 2.33**	**19.41 ± 2.33**
	VERT_LONG_APICAL	39.60 ± 3.11	**38.42 ± 4.10**	38.38 ± 3.68	38.97 ± 3.78	38.97 ± 3.78
	VERT_LONG_INF	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**	**32.10 ± 3.26**
	VERT_LONG_BASAL	15.88 ± 3.17	**15.29 ± 3.19**	**15.29 ± 3.19**	**15.29 ± 3.19**	15.88 ± 3.51
	Average error	*18.42*	***17.65***	*18.31*	18.56	18.64
	Average advantage	*7.69*	***11.97***	*8.66*	7.44	7.08

8.2.3 Discussion of Results

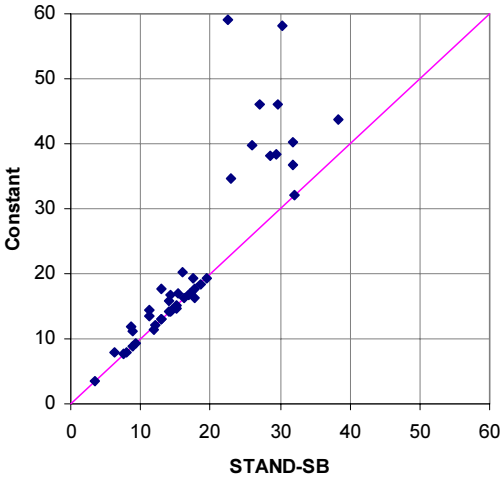
Usefulness of new Bayesian Classifiers for Interpretation of SPECT Data

As demonstrated in Table 8.6, 42 out of 44 times the new Bayesian classifiers were better than or equal to any of the reference classifiers. This has not been intended to be an exhaustive study of existing classification approaches, since only three other nontrivial classifiers were tested (C4.5, naïve Bayes, and TAN). However, it shows that the new Bayesian classifiers perform well on the SPECT data. Especially, that they outperformed well known C4.5 classifier.

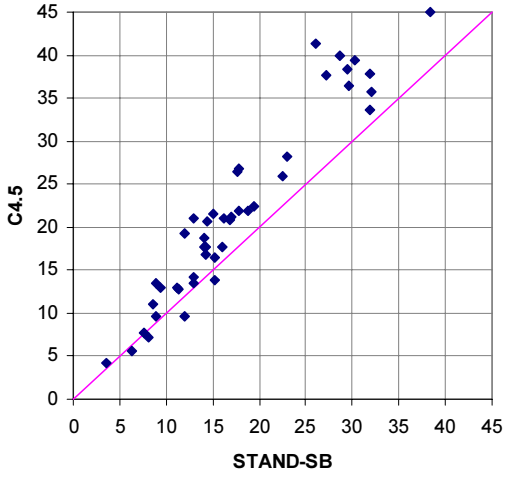
We have selected the STAND-SB classifier, as a representative of the new Bayesian network classifiers, and compared it to the four reference classifiers: constant, C4.5, naïve Bayes, and TAN. The diagrams showing the comparison are presented in Fig. 8.2. Axis in each diagram represent percentage error rate. The pink diagonal line represents equal of error for the two compared classifiers. A blue mark represents a dataset. When a blue mark is above the line it means that a classifier compared to STAND-SB had a larger error for that dataset. When a mark is below the diagonal pink line it means that STAND-SB had a larger error. Diagrams in Fig. 8.2 demonstrate that STAND-SB outperforms all of the reference classifiers.

Performance of Network Search Algorithms

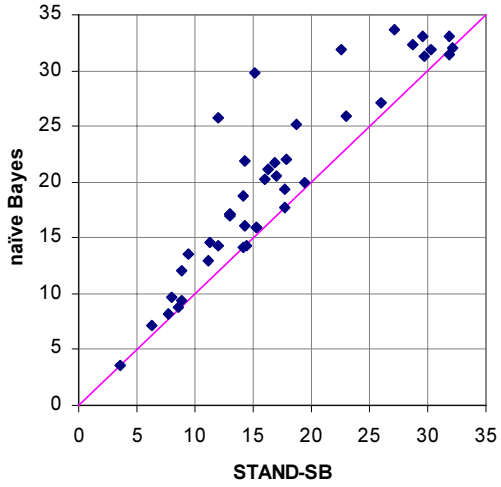
We ranked performance of the new Bayesian network search algorithms by counting how many times each algorithm had the lowest error among all classifiers (the number of cells with gray background in Tables 8.7 through 8.11). The ranking is presented in Table 8.12. STAN algorithm received the highest rating, FAN the lowest by a significant margin. Other three algorithms performed close to each other and not much worse than STAN.



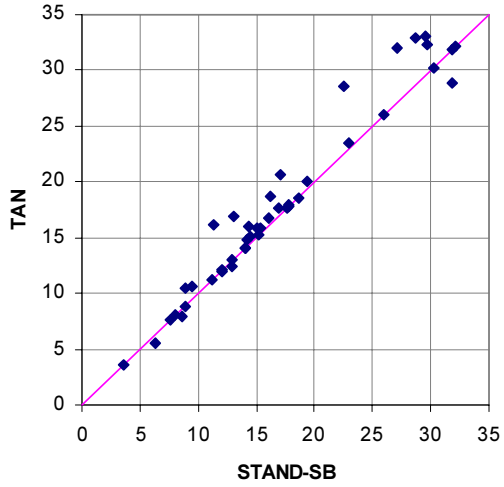
(a) STAND-SB versus constant classifier.



(b) STAND-SB versus C4.5 classifier.



(c) STAND-SB versus naïve Bayes classifier.



(d) STAND-SB versus TAN classifier.

Figure 8.2: Comparison of STAND-SB inducer versus constant classifier, C4.5, naïve Bayes, and TAN inducers on partial LV perfusion classification data.

Table 8.12: Partial classification: ranking of Bayesian network classifier search algorithms.

	FAN	STAN	STAND	SFAN	SFAND
HGC	5	14	7	14	6
SB	4	12	16	10	16
LC	5	13	12	11	7
LOO	7	11	10	8	12
CV10101	7	11	10	12	9
total	28	**61**	55	55	50

Table 8.13: Partial classification: ranking of Bayesian network quality measures.

	HSC	SB	LC	LOO	CV1010
FAN	27	23	18	24	27
STAN	20	21	21	17	16
STAND	15	30	22	19	22
SFAN	21	21	18	14	15
SFAND	14	30	16	17	13
total	97	**125**	92	91	93

Performance of Network Quality Measures

We ranked performance of the network structure quality measures by counting how many times each measure produced lowest error for a given algorithm and a given dataset. The ranking is presented in Table 8.13. The SB measure ranked highest. The remaining measures ranked similar to each other, with the HGC slightly better than others.

It is a bit surprising the SB ranked highest since it is a global measure. However, this is the only measure that includes penalty for the network size, thus limiting number of parameters in the network.

Overall Performance of the New Bayesian Network Classifiers

We have used the following three indicators to compare the new family of Bayesian network classifiers to the reference classifiers (the constant classifier, C4.5, naïve Bayes, and TAN):

Dataset error rate indicates for how many datasets the error rate of the best new Bayesian

Table 8.14: Comparison of the new family of Bayesian network classifiers to reference classifier (the constant classifier, C4.5, naïve Bayes, and TAN) using the partial left ventricle perfusion datasets datasets. A number indicates how many times a best of the new classifiers was better, equal, or worse than a best of reference classifiers.

Indicator	Better		Equal		Worse	
Dataset error rate	28	64%	14	32%	2	4%
Average error rate	17	68%	0	0%	8	32%
Average advantage ratio	18	75%	0	0%	7	25%

network classifier was better (lower), equal to, or worse (higher) than that of the best reference classifier for a given dataset.

Average error rate indicates how many of the new Bayesian network classifiers had average error rate that is better (lower), equal to, or worse (higher) than that of the best (lowest) average error rate of the reference classifiers.

Average advantage ratio indicates how many of the new Bayesian network classifiers had average advantage ratio, Eq.(8.1), that is better (higher), equal to, or worse (lower) than that of the best (highest) average advantage ratio of the reference classifiers.

The indicators are shown in Table 8.14. This table clearly demonstrates that the new family of Bayesian network algorithms produces classifiers that are performing better, on the partial left ventricle perfusion datasets, than the reference classifiers.

Quality of the Datasets with Features Extracted from SPECT Images

By a *quality of a dataset* we understand here the amount of information in the dataset that can be utilized by a classifier to perform classification with low error. In an extreme case a dataset may contain only the class variable (no attribute variables). The only reasonable way to make a decision in this case is to create a classifier that will always predict the class that is the most frequent in the training dataset. This is the same as constructing the constant classifier – it always predicts the majority class, regardless of values of attributes.

Table 8.15: Partial classification: ranking of dataset quality.

Rank	Dataset	Advantage ratio
1	F_SHORT_AXIS_BASAL_ANT	62%
2	F_VERT_LONG_ANT	48%
3	F_SHORT_AXIS_MID_INF	47%
4	F_SHORT_AXIS_AP_ANT	41%
5	F_SHORT_AXIS_MID_ANT	40%
6	M_HORIZ_LONG_APICAL	39%
7	M_SHORT_AXIS_MID_INF	35%
8	M_HORIZ_LONG_SEPT	30%
9	F_HORIZ_LONG_SEPT	30%
10	F_SHORT_AXIS_AP_LAT	29%
11	M_HORIZ_LONG_BASAL	26%
12	M_SHORT_AXIS_AP_LAT	26%
12	M_HORIZ_LONG_LAT	26%
14	M_SHORT_AXIS_AP_INF	25%
14	F_SHORT_AXIS_MID_ANT_LAT	25%
16	M_SHORT_AXIS_BASAL_INF	24%
16	F_HORIZ_LONG_LAT	24%
16	M_SHORT_AXIS_BASAL_INF_LAT	24%
19	M_SHORT_AXIS_MID_INF_LAT	23%
20	F_SHORT_AXIS_MID_ANT_SEPT	21%
21	F_SHORT_AXIS_BASAL_ANT_LAT	19%
22	M_VERT_LONG_APICAL	18%
23	F_SHORT_AXIS_MID_INF_LAT	17%
24	F_HORIZ_LONG_APICAL	16%
25	M_SHORT_AXIS_AP_ANT	15%
26	M_VERT_LONG_BASAL	14%
27	F_SHORT_AXIS_AP_SEPT	10%
27	F_HORIZ_LONG_BASAL	10%
29	F_SHORT_AXIS_AP_INF	6%
30	M_SHORT_AXIS_BASAL_ANT	5%
31	F_VERT_LONG_BASAL	1%
32	F_SHORT_AXIS_BASAL_ANT_SEPT	0%
32	M_SHORT_AXIS_BASAL_ANT_LAT	0%
32	M_SHORT_AXIS_BASAL_ANT_SEPT	0%
32	M_SHORT_AXIS_MID_ANT_SEPT	0%
32	M_VERT_LONG_INF	0%
32	F_SHORT_AXIS_BASAL_INF_LAT	0%
32	F_VERT_LONG_INF	0%
32	M_SHORT_AXIS_MID_ANT	0%
32	F_VERT_LONG_APICAL	0%
32	M_VERT_LONG_ANT	0%
32	M_SHORT_AXIS_AP_SEPT	0%
43	M_SHORT_AXIS_MID_ANT_LAT	-1%
44	F_SHORT_AXIS_BASAL_INF	-5%

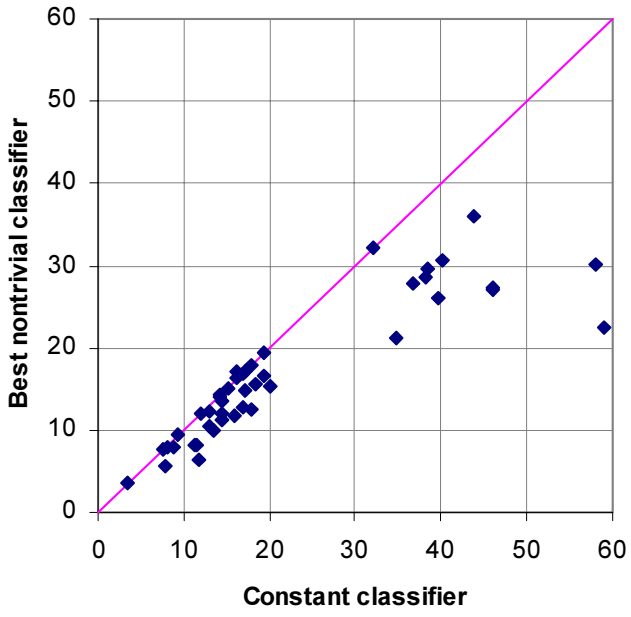


Figure 8.3: Quality of the partial left ventricle classification datasets: comparison of the the constant classifier to a best nontrivial classifier tested.

The constant classifier behaves as if there is no useful information in the attribute variables – dataset contains only noise. Thus we use the constant classifier error rate as a reference error rate for other classifiers.

If we can build a classifier that produces error rate significantly lower for a given dataset than the constant classifier, we can judge that this dataset has good quality. For a perfect dataset we should be able to build a classifier that has error rate close to zero, regardless of the error rate of the constant classifier. We are proposing the following *advantage ratio* to rank datasets:

$$\text{advantage ratio} \;=\; \frac{\varepsilon_{const} - \varepsilon_{best}}{\varepsilon_{const}} \cdot 100\% \tag{8.1}$$

where ε_{const} is the error rate of the constant classifier for a given dataset, and ε_{best} is the error rate of a best classifier we were able to build for that dataset. The advantage ratio defined by Eq. (8.1) reaches value of 100% if we can build a classifier with error rate equal to zero; it equals 0% if the error rate of our best classifier is the same as the constant classifier; and it is negative if our best classifier produces error rate higher than the constant classifier.

Table 8.15 shows ranking of datasets sorted in order of decreasing advantage ratio. We calculated the advantage ratio using ten-fold cross-validation results presented in Table 8.6. We can say that datasets with advantage ratio close or less than zero are of low quality. A direct comparison of the classification error of the constant classifier versus a best classifier for given dataset is presented in Fig. 8.3.

We can see from Table 8.15 that there is a significant variability in quality of the datasets. Dataset F-SHORT-AXIS-BASAL-ANT has the highest advantage ratio indicating that it contains most useful information contributing to the classification goal, but its best error rate of 22.44% is still quite high. On the other hand, the dataset M-SHORT-AXIS-BASAL-ANT-LAT had the lowest error rate of 3.53% among all datasets, but it had the advantage ratio equal zero indicating that it does not contain information useful for classification. Table 8.5 shows that this dataset contains no more than two examples for each

of the defect types. From statistical point of view, this is not sufficient for adequate representation of these defects in the dataset, even if these defects were combined into a single category.

There are three main factors determining the quality of a dataset. First, the number of cases in the dataset – if it is sufficient to represent the information about the underlying phenomenon (perfusion of the left ventricle in our case). Second, noise in coding the values of a class variable, which we estimated in Section 8.1. Third, the information coding scheme for attributes; in our case, the feature extraction process described in Chapter 4. The advantage ratio is only able to judge these factors cumulatively. But we can combine it with information presented in Tables 8.4, 8.5, and 8.6. Our overall conclusion is that the results of partial classification look very promising, but a database containing significantly larger number of cases is needed before better results can be achieved.

8.3 Overall Classification of the Left Ventricle Perfusion

In the experiment described in the previous section partial classification of the left ventricular perfusion was performed, classifying each of the 22 3D ROIs separately. The goal of the experiment described in this section is to determine feasibility of performing an overall classification of the perfusion of the left ventricle. We use the same feature extraction process as described in Section 8.2; however, we create different datasets. We selected cases that had a single overall perfusion code recorded in the database, see Table 3.1. Additionally, we used only defect types that have at least five cases representing them. And, as before, we used only these cases that have sufficiently high left ventricle model fit correlation ratio (0.95 for males, and 0.93 for females). Count for the cases satisfying these criteria is presented in Table 8.16.

The difficulty we faced is that we had a low number of cases and a large number of attributes. As described in Section 8.2, 16 features were extracted for each of the 22 3D

Table 8.16: Case counts for overall LV perfusion classification tests.

Sex	# attributes	# cases	Classes			
			NL	IS	INF	IS-IN
Female	44	74	36 49%	16 22%	14 19%	8 11%
Male	44	118	23 19%	20 17%	30 25%	45 38%

ROI resulting in 352 attributes available for the overall classification. This is larger than number of available cases. If we were to use all of the features we could expect very high variance in estimation of classifier parameters. We decided to use only one feature for each ROI in rest and stress images resulting in 44 attributes. We decided to use features based on maximum, mean, and median of pixel values in a 3D region. We used either MAX or TOT radial search image. Thus we had six feature types: MAX-Max, MAX-Mean, MAX-Median, TOT-Max, TOT-Mean, and TOT-Median. We created one dataset for each six feature types, for females and males separately, resulting in 12 datasets. Each dataset had four classes: NL – normal, IS – ischemia, INF – infarct, and IS-IN – ischemia-infarct.

Ten-fold cross validation has been performed using reference classifiers (the constant classifier, C4.5, naïve Bayes, and TAN), and all of the 25 new Bayesian network classifiers. The results are presented in Tables 8.17 through 8.22. As before, a number in bold indicates the lowest error for a given dataset in a particular table. Numbers with gray background indicate lowest error rate among all classifiers tested for a given dataset.

The summary of the results is presented in Table 8.17. The best feature type, for both females and males, is the median of pixel intensity taken from TOT images. Two surprising observations can be concluded from Table 8.17:

1. The error rate and the advantage ratio is better for females than for males, despite the female sample is smaller and we would expect that given large number of attributes the variance of classifier parameters will be high. However, the male sample is also small compared to number of attributes. Distribution of classes in the male sample,

see Table 8.16, is more uniform. In the female sample class **NL** appears in almost half of the cases making it an easier concept to learn.

2. The error rate for females is surprisingly low – we will explain shortly. At first the absolute value of the error may look large, however, the absolute value is deceiving. The error rate in the golden standard, estimated in Section 8.1, is about 17%. The average error rate of partial classification of the left ventricle perfusion is about 18% (see Table 8.6). If we build a hierarchical classifier that first performs partial classification, then uses these results to perform overall classification[3] then estimated error rate of such a classifier will be the sum of the partial classification error rate and the error level of the golden standard (the error rate of the best classifier that performs overall classification using partial classification codes). This is about 35%. The error rate for direct overall left ventricle perfusion classification using TOT-Max approach is $38.75\% \pm 6.06\%$. Within the margin of error, these two approaches, hierarchical and direct using TOT-Max, produce the same error rate.

To summarize above, we can say that we are positively surprised by the error rate shown in Table 8.17. The available sample of patient cases was very small. And we expect that direct classification approach could give good error rate given larger sample of patient cases.

[3]Such a hierarchical classifier can be build by using in the lower level the 22 classifiers constructed in Section 8.6 One classifier for each of the 22 ROIs. And using in the upper level the best classifier constructed in Section 8.1, that issues the overall diagnosis of the left ventricle perfusion based on partial diagnosis in each of the 22 ROIs.

Table 8.17: Ten-fold cross-validation error rate of the overall left ventricle perfusion classification using reference classifiers and the best results for the new Bayesian network classifiers. The last column shows value of the advantage ratio of the best classifier for a given dataset (higher is better).

Sex	Image Type	Feature Type	Constant Classifier	C4.5	Naïve Bayes	TAN	Best New BNC	Adv. Ratio
Female	MAX	Max	**51.07 ± 4.39**	59.46 ±2.78	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**	0.00
		Mean	**51.07 ± 4.39**	53.93 ±5.07	57.86 ± 5.01	57.86 ± 5.01	57.86 ± 5.01	0.00
		Median	**51.07 ± 4.39**	61.96 ±5.41	59.11 ± 5.29	59.11 ± 5.29	59.11 ± 5.29	0.00
	TOT	Max	51.07 ± 4.39	55.71 ±4.51	53.75 ± 4.43	53.75 ± 3.24	**49.82 ± 3.63**	0.02
		Mean	51.07 ± 4.39	43.57 ±3.71	43.04 ± 4.51	48.57 ± 5.44	**41.79 ± 4.46**	0.18
		Median	51.07 ± 4.39	48.21 ±4.24	41.61 ± 4.73	40.36 ± 4.65	**38.75 ± 6.06**	**0.24**
Male	MAX	Max	61.82 ± 3.15	69.62 ±4.84	62.80 ± 5.09	62.80 ± 5.38	**59.47 ± 5.57**	0.04
		Mean	61.82 ± 3.15	64.55 ±3.14	61.06 ± 3.26	60.23 ± 2.68	**55.15 ± 5.14**	0.11
		Median	61.82 ± 3.15	66.21 ±4.39	58.56 ± 3.95	56.74 ± 3.17	**54.17 ± 3.63**	0.12
	TOT	Max	**61.82 ± 3.15**	74.47 ±3.69	66.29 ± 4.61	63.64 ± 3.21	62.73 ± 3.05	0.00
		Mean	61.82 ± 3.15	57.58 ±3.13	61.06 ± 3.43	**55.30 ± 4.31**	55.98 ± 4.20	0.11
		Median	61.82 ± 3.15	71.89 ±2.96	62.80 ± 3.46	61.97 ± 3.23	**57.73 ± 3.60**	0.07

Table 8.18: Ten-fold cross-validation error rate of the overall left ventricle perfusion classification using FAN classifier.

Sex	Image Type	Feature Type	FAN				
			HGC	SB	LC	LOO	CV1010
Female	MAX	Max	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**
		Mean	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**
		Median	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**
	TOT	Max	**52.32 ± 4.00**	53.75 ± 4.43	53.75 ± 4.43	53.75 ± 4.43	53.75 ± 4.43
		Mean	43.04 ± 4.10	43.04 ± 4.51	**41.79 ± 4.46**	**41.79 ± 4.46**	43.04 ± 4.51
		Median	41.61 ± 5.19	43.04 ± 4.98	41.61 ± 4.90	**40.18 ± 4.90**	**40.18 ± 4.90**
Male	MAX	Max	62.88 ± 5.92	62.80 ± 5.09	62.80 ± 5.92	**62.05 ± 5.92**	**62.05 ± 5.92**
		Mean	60.30 ± 3.60	60.38 ± 4.37	60.30 ± 2.68	**60.23 ± 2.68**	**60.23 ± 2.68**
		Median	55.98 ± 4.20	58.41 ± 2.69	56.74 ± 3.63	**54.17 ± 3.63**	55.98 ± 4.20
	TOT	Max	63.64 ± 3.21	**62.73 ± 3.05**	65.45 ± 3.83	64.47 ± 3.83	66.21 ± 4.76
		Mean	**56.14 ± 4.28**	61.89 ± 3.68	59.32 ± 3.44	58.64 ± 3.44	58.48 ± 3.12
		Median	61.97 ± 3.67	62.80 ± 3.46	**61.14 ± 3.67**	61.97 ± 3.67	61.97 ± 3.67

Table 8.19: Ten-fold cross-validation error rate of the overall left ventricle perfusion classification using STAN classifier.

Sex	Image Type	Feature Type	STAN				
			HGC	SB	LC	LOO	CV1010
Female	MAX	Max	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**
		Mean	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**
		Median	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**
	TOT	Max	53.75 ± 4.01	56.61 ± 4.56	**49.82 ± 3.63**	50.89 ± 4.19	50.89 ± 4.19
		Mean	**44.46 ± 5.50**	45.71 ± 4.68	45.71 ± 4.79	46.07 ± 5.07	46.25 ± 5.97
		Median	**38.75 ± 6.06**	47.14 ± 4.80	44.46 ± 4.97	40.54 ± 4.26	41.79 ± 3.92
Male	MAX	Max	**59.47 ± 5.57**	60.30 ± 5.33	61.14 ± 5.64	60.30 ± 3.15	61.97 ± 5.52
		Mean	60.83 ± 3.13	58.48 ± 3.12	58.64 ± 4.25	**57.58 ± 2.82**	60.15 ± 2.80
		Median	67.73 ± 3.30	63.48 ± 2.89	57.65 ± 3.46	**57.58 ± 3.33**	**57.58 ± 3.33**
	TOT	Max	68.79 ± 4.12	**65.38 ± 4.50**	67.05 ± 4.16	**65.38 ± 2.81**	**65.38 ± 2.81**
		Mean	**55.98 ± 4.20**	57.05 ± 3.63	56.74 ± 3.83	60.45 ± 4.33	59.39 ± 3.17
		Median	**58.64 ± 4.25**	**58.64 ± 3.87**	60.23 ± 3.86	60.23 ± 3.20	60.23 ± 4.05

Table 8.20: Ten-fold cross-validation error rate of the overall left ventricle perfusion classification using STAND classifier.

Sex	Image Type	Feature Type	STAND				
			HGC	SB	LC	LOO	CV1010
Female	MAX	Max	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**
		Mean	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**
		Median	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**
	TOT	Max	51.07 ± 3.36	53.75 ± 5.35	51.07 ± 3.36	**50.89 ± 4.19**	52.14 ± 4.73
		Mean	50.00 ± 5.46	47.32 ± 4.88	**43.21 ± 5.18**	45.89 ± 6.12	47.32 ± 5.83
		Median	**38.93 ± 4.28**	41.79 ± 5.38	44.29 ± 4.99	44.29 ± 4.99	44.29 ± 4.99
Male	MAX	Max	61.14 ± 5.64	61.97 ± 5.52	61.14 ± 5.64	60.30 ± 4.88	**59.47 ± 4.98**
		Mean	60.30 ± 4.38	60.98 ± 2.86	58.64 ± 3.21	59.39 ± 2.58	**58.56 ± 5.14**
		Median	59.32 ± 3.67	60.15 ± 2.18	56.82 ± 4.50	56.89 ± 3.03	**55.98 ± 3.24**
	TOT	Max	67.05 ± 4.16	**64.47 ± 3.16**	67.05 ± 4.16	67.05 ± 4.68	66.21 ± 3.67
		Mean	**57.80 ± 3.76**	58.64 ± 4.06	58.41 ± 3.45	58.56 ± 3.31	58.48 ± 2.86
		Median	**57.73 ± 3.60**	**57.73 ± 4.00**	60.23 ± 2.95	60.23 ± 3.86	59.39 ± 3.36

Table 8.21: Ten-fold cross-validation error rate of the overall left ventricle perfusion classification using SFAN classifier.

Sex	Image Type	Feature Type	SFAN				
			HGC	SB	LC	LOO	CV1010
Female	MAX	Max	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**
		Mean	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**
		Median	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**
	TOT	Max	53.57 ± 4.12	53.93 ± 4.33	**51.07 ± 3.36**	53.75 ± 4.43	53.75 ± 4.43
		Mean	**43.04 ± 5.09**	44.46 ± 5.50	43.21 ± 5.18	45.71 ± 5.47	45.54 ± 4.95
		Median	44.46 ± 4.97	45.71 ± 5.14	42.86 ± 4.66	43.21 ± 4.22	**41.79 ± 4.46**
Male	MAX	Max	**59.47 ± 5.57**	60.30 ± 5.33	61.82 ± 3.15	61.21 ± 3.42	62.80 ± 4.61
		Mean	57.58 ± 2.87	56.67 ± 3.31	**55.15 ± 1.56**	59.47 ± 3.31	59.47 ± 3.31
		Median	65.30 ± 3.84	63.48 ± 2.89	59.32 ± 3.31	**56.74 ± 3.63**	**56.74 ± 3.63**
	TOT	Max	67.80 ± 3.46	65.23 ± 3.40	65.38 ± 3.31	**64.55 ± 2.89**	**64.55 ± 2.89**
		Mean	57.80 ± 4.51	**56.14 ± 4.79**	58.71 ± 3.82	57.80 ± 3.33	57.80 ± 3.33
		Median	**58.64 ± 4.25**	**58.64 ± 3.87**	59.39 ± 3.79	62.73 ± 4.12	63.56 ± 4.47

Table 8.22: Ten-fold cross-validation error rate of the overall left ventricle perfusion classification using SFAND classifier.

Sex	Image Type	Feature Type	SFAND				
			HGC	SB	LC	LOO	CV1010
Female	MAX	Max	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**	**51.07 ± 4.39**
		Mean	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**	**57.86 ± 5.01**
		Median	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**	**59.11 ± 5.29**
	TOT	Max	53.57 ± 4.12	53.93 ± 4.33	**51.07 ± 3.36**	53.75 ± 4.43	53.75 ± 4.43
		Mean	46.07 ± 5.30	49.82 ± 5.30	**43.21 ± 5.18**	44.64 ± 5.58	45.71 ± 5.47
		Median	44.46 ± 5.18	44.29 ± 4.62	**43.04 ± 3.97**	43.21 ± 5.18	43.21 ± 4.72
Male	MAX	Max	**60.98 ± 2.86**	61.14 ± 5.50	61.82 ± 3.15	62.05 ± 3.21	62.05 ± 3.21
		Mean	60.30 ± 4.38	60.38 ± 4.37	**56.89 ± 2.06**	60.45 ± 4.15	59.62 ± 5.00
		Median	58.41 ± 2.69	58.41 ± 2.69	57.65 ± 4.61	56.67 ± 4.50	**55.08 ± 3.27**
	TOT	Max	66.14 ± 4.45	**62.73 ± 3.05**	65.38 ± 3.31	63.64 ± 2.38	63.64 ± 2.38
		Mean	58.64 ± 3.87	**58.41 ± 4.06**	59.39 ± 1.89	58.64 ± 2.96	58.71 ± 3.62
		Median	**57.73 ± 3.60**	**57.73 ± 4.00**	61.82 ± 2.62	61.89 ± 3.73	61.89 ± 3.73

8.4 Benchmarking on Datasets from UCI Machine Learning Repository

The goal of this experiment is to benchmark the new Bayesian network classifiers against the results published by Friedman et al. (1997) in their work on TAN and other Bayesian network classifiers. We already established that new Bayesian network classifiers are well suited for analysis of cardiac SPECT data. In this experiment, we want to determine how our new Bayesian network classifier algorithms perform on datasets from variety of domains and how they compare to previous research. We use here the same datasets and

Table 8.23: UCI Machine Learning Repository datasets used for testing.

Dataset	# Attributes	# Classes	# Cases	
			Train	Test
australian	14	2	690	CV-5
breast	10	2	683	CV-5
chess	36	2	2,130	1,066
cleve	13	2	296	CV-5
corral	6	2	128	CV-5
crx	15	2	653	CV-5
diabetes	8	2	768	CV-5
flare	10	2	1,066	CV-5
german	20	2	1,000	CV-5
glass	9	7	214	CV-5
glass2	9	2	163	CV-5
heart	13	2	270	CV-5
hepatitis	19	2	80	CV-5
iris	4	3	150	CV-5
letter	16	26	15,000	5,000
lymphography	18	4	148	CV-5
mofn-3-7-10	10	2	300	1,024
pima	8	2	768	CV-5
satimage	36	6	4,435	2,000
segment	19	7	1,540	770
shuttle-small	9	7	3,866	1,934
soybean-large	35	19	562	CV-5
vehicle	18	4	846	CV-5
vote	16	2	435	CV-5
waveform-21	21	3	300	4,700

the same testing methods as published by Friedman et al. (1997). Table 8.23 presents the datasets used for testing, number of attributes, number of classes, and number of cases in each dataset. Friedman et al. (1997) used five-times cross validation (CV-5) for smaller datasets and a single test for larger datasets. To ensure objectivity of comparison, we used classification error data published by Friedman et al. (1997) rather than those that could be produced by our version of the TAN classifier.

8.4.1 The Datasets

Here is a brief description of the datasets used for benchmarking; more information can be found in (Blake et al., 1998).

australian Australian credit approval data.

breast Breast cancer databases from the University of Wisconsin Hospitals, Madison.

chess Chess End-Game – King+Rook versus King+Pawn on a7 (usually abbreviated KRKPA7). The pawn on a7 means it is one square away from queening. It is the King+Rook's side (white) to move.

cleve Cleveland heart disease database. Eight attributes are symbolic, six numeric. There are two classes: healthy (buff) or with heart-disease (sick). The attributes are: age, sex, chest pain type (angina, abnang, notang, asympt), resting blood pressure, serum cholesterol in mg/dl, fasting blood sugar $<$ 120 mg/dl (true or false), resting ECG (norm, abnormal, hyper), max heart rate, exercise induced angina (true or false), oldpeak = ST depression induced by exercise relative to rest, the slope of the peak exercise ST segment (up, flat, down), number of major vessels (0-3) colored by fluoroscopy, thallium (normal, fixed defect, reversible defect).

corral An artificial dataset designed to show that decision trees might pick a really bad attribute for the root. The target concept is $(a_1$ **xor** $a_2)$ **or** $(a_3$ **xor** $a_4)$, attribute A_5 is

correlated to the class variable and attribute A_6 is irrelevant.

crx Credit card applications data.

diabetes Pima Indians diabetes database.

flare Classification of solar flares.

german German credit database.

glass Glass identification database.

glass2 Variant of the glass identification database with two classes and corresponding cases removed.

heart Another heart disease database. It has the structure similar to cleve database, the same classes and attributes.

hepatitis Survival of hepatitis patients.

iris This is perhaps the best known database to be found in the pattern recognition literature. The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other.

letter The objective is to identify each of a large number of black-and-white rectangular pixel displays as one of the 26 capital letters in the English alphabet. The character images were based on 20 different fonts and each letter within these 20 fonts was randomly distorted to produce a file of 20,000 unique stimuli.

lymphography Classification of lymphography data.

mofn-3-7-10 Artificial dataset: 10 bits; 3 out of 7 should be on; remaining three are irrelevant (A_1, A_2,A_{10}).

pima Pima Indians diabetes database from the National Institute of Diabetes and Digestive and Kidney Diseases.

satimage Landsat Satellite data: multi-spectral values of pixels in 3x3 neighborhoods in a satellite image, and the classification associated with the central pixel in each neighborhood.

segment Image segmentation database. The instances were drawn randomly from a database of 7 outdoor images. The images were hand-segmented to create a classification for every pixel.

shuttle-small The shuttle dataset contains 9 attributes all of which are numerical. Approximately 80% of the data belongs to class 1.

soybean-large Soybean disease databases.

vehicle Vehicle silhouettes: 3D objects within a 2D image by application of an ensemble of shape feature extractors to the 2D silhouettes of the objects.

vote Voting records drawn from the Congressional Quarterly Almanac, 98th Congress, 2nd session 1984, Volume XL: Congressional Quarterly Inc. Washington, D.C., 1985.

waveform-21 Artificial dataset from waveform generator. Three classes of waveforms. Each class is generated from a combination of 2 or 3 “base” waves. All 21 attributes include noise.

8.4.2 Experiments

The results of experiments are presented in Tables 8.24 to 8.29. As before, a number in bold indicates the lowest error for a given dataset in a particular table. Numbers with gray background indicate lowest error rate among all classifiers tested for a given dataset. Bottom of each table shows two performance indicators for each of the classifiers: an

Table 8.24: Benchmarks on UCI datasets: the reference classifiers and the best results for the new Bayesian network classifiers.

Dataset	Constant	C4.5	Naïve Bayes	TAN	TAN-S	Best New BNC
australian	44.49 ± 2.99	15.07 ± 0.81	22.90 ± 1.58	18.70 ± 1.10	15.80 ± 1.24	**11.74 ± 1.15**
breast	35.00 ± 2.95	5.13 ± 0.96	3.96 ± 0.55	4.25 ± 1.25	3.08 ± 0.67	**2.34 ± 0.43**
chess	46.81 ± 1.53	**0.47 ± 0.21**	12.85 ± 1.03	7.60 ± 0.81	7.69 ± 0.82	4.03 ± 0.60
cleve	45.93 ± 2.63	28.38 ± 1.83	18.57 ± 2.55	20.94 ± 0.65	18.24 ± 0.33	**16.20 ± 1.33**
corral	43.63 ± 4.07	2.31 ± 2.31	14.12 ± 3.25	4.68 ± 2.26	3.94 ± 2.51	**0.00 ± 0.00**
crx	45.34 ± 2.11	14.08 ± 1.21	23.12 ± 1.73	16.23 ± 1.34	14.24 ± 1.16	**11.94 ± 0.69**
diabetes	34.88 ± 2.35	27.34 ± 1.08	25.00 ± 1.77	24.87 ± 0.98	24.48 ± 1.16	**22.66 ± 0.68**
flare	17.07 ± 2.05	17.45 ± 1.75	19.04 ± 1.44	17.26 ± 1.60	17.73 ± 1.86	**16.32 ± 1.52**
german	30.00 ± 1.14	28.70 ± 0.93	25.00 ± 1.57	27.80 ± 1.54	26.90 ± 1.54	**24.90 ± 0.90**
glass	64.50 ± 1.05	34.13 ± 3.54	52.81 ± 0.71	30.82 ± 2.64	32.22 ± 3.43	**20.42 ± 1.09**
glass2	46.69 ± 2.72	20.87 ± 3.60	40.55 ± 2.83	20.83 ± 1.71	22.08 ± 1.11	**17.80 ± 2.26**
heart	44.44 ± 3.88	21.48 ± 3.13	15.93 ± 2.24	17.04 ± 2.51	16.67 ± 2.48	**14.81 ± 2.11**
hepatitis	16.25 ± 3.19	18.75 ± 1.98	8.75 ± 2.50	15.00 ± 2.50	8.75 ± 2.50	**6.25 ± 2.80**
iris	74.67 ± 1.33	6.00 ± 1.25	**4.67 ± 1.33**	6.67 ± 1.05	6.00 ± 1.25	**4.67 ± 1.33**
letter	96.30 ± 0.27	**12.16 ± 0.46**	35.96 ± 0.68	16.56 ± 0.53	14.14 ± 0.49	12.78 ± 0.47
lymphography	45.20 ± 6.61	22.97 ± 0.59	19.56 ± 1.57	33.13 ± 3.37	14.97 ± 3.09	**14.18 ± 1.64**
mofn-3-7-10	22.66 ± 1.31	16.02 ± 1.15	13.57 ± 1.07	8.30 ± 0.86	8.89 ± 0.89	**6.25 ± 0.76**
pima	34.90 ± 1.88	26.18 ± 2.05	23.97 ± 1.61	24.87 ± 1.36	24.48 ± 1.27	**22.92 ± 1.02**
satimage	76.95 ± 0.94	14.35 ± 0.78	20.35 ± 0.90	22.45 ± 0.93	12.80 ± 0.75	**12.60 ± 0.74**
segment	86.88 ± 1.22	5.84 ± 0.85	20.65 ± 1.46	14.68 ± 1.63	4.42 ± 0.74	**3.90 ± 0.70**
shuttle-small	21.10 ± 0.93	0.57 ± 0.17	8.74 ± 0.64	1.14 ± 0.24	0.47 ± 0.15	**0.36 ± 0.14**
soybean-large	87.10 ± 1.03	7.82 ± 1.15	8.54 ± 0.91	41.83 ± 1.43	7.83 ± 1.02	**6.22 ± 1.28**
vehicle	77.19 ± 0.39	**26.71 ± 0.85**	55.79 ± 1.58	32.14 ± 2.92	30.37 ± 2.11	28.95 ± 2.29
vote	38.62 ± 2.64	4.14 ± 0.46	9.66 ± 0.86	10.80 ± 1.61	6.44 ± 0.28	**3.22 ± 0.76**
waveform-21	66.26 ± 0.69	29.30 ± 0.66	**19.32 ± 0.58**	24.62 ± 0.63	21.62 ± 0.60	21.04 ± 0.59
Average error	49.71	16.25	20.94	18.53	14.57	**13.51**
Av. advantage	0.00	58.84	52.71	57.43	64.80	**67.70**

Table 8.25: Benchmarks on UCI datasets: five-fold cross-validation error rate for FAN classifier.

Dataset	FAN				
	HGC	SB	LC	LOO	CV1010
australian	12.46 ± 0.84	**12.32 ± 0.76**	12.46 ± 0.42	12.61 ± 0.54	12.61 ± 0.54
breast	2.49 ± 0.18	**2.34 ± 0.43**	2.78 ± 0.43	2.78 ± 0.43	2.78 ± 0.43
chess	7.32 ± 0.80	7.41 ± 0.80	**6.75 ± 0.77**	**6.75 ± 0.77**	**6.75 ± 0.77**
cleve	**16.55 ± 1.64**	18.58 ± 1.92	16.89 ± 2.20	18.24 ± 1.12	17.90 ± 1.14
corral	1.60 ± 1.60	**0.00 ± 0.00**	1.60 ± 1.60	1.60 ± 1.60	1.60 ± 1.60
crx	13.17 ± 0.92	**13.01 ± 0.62**	13.48 ± 0.70	13.48 ± 0.70	13.48 ± 0.70
diabetes	23.18 ± 0.59	24.09 ± 0.61	**22.91 ± 0.77**	23.05 ± 0.52	23.31 ± 0.64
flare	**17.16 ± 1.68**	17.26 ± 1.84	**17.16 ± 1.68**	**17.16 ± 1.68**	**16.78 ± 1.58**
german	25.40 ± 1.10	25.40 ± 1.26	**24.90 ± 0.90**	25.00 ± 1.05	25.00 ± 1.05
glass	27.55 ± 1.27	28.49 ± 0.78	**27.08 ± 1.65**	27.55 ± 1.27	27.55 ± 1.27
glass2	19.66 ± 1.89	19.64 ± 1.57	18.45 ± 2.03	**17.82 ± 1.85**	**17.82 ± 1.85**
heart	**16.67 ± 2.48**	17.04 ± 2.51	**16.67 ± 2.48**	**16.67 ± 3.04**	**16.67 ± 3.04**
hepatitis	8.75 ± 1.53	10.00 ± 1.53	8.75 ± 2.50	**7.50 ± 2.34**	**7.50 ± 2.34**
iris	**5.33 ± 1.33**	**5.33 ± 1.33**	**5.33 ± 1.33**	**5.33 ± 1.33**	**5.33 ± 1.33**
letter	13.30 ± 0.48	23.06 ± 0.60	16.42 ± 0.52	**13.22 ± 0.48**	13.30 ± 0.48
lymphography	14.21 ± 1.98	14.90 ± 2.33	14.87 ± 1.36	**14.18 ± 1.64**	**14.18 ± 1.64**
mofn-3-7-10	9.47 ± 0.92	12.50 ± 1.03	8.50 ± 0.87	**8.11 ± 0.85**	**8.11 ± 0.85**
pima	**24.22 ± 1.40**	24.62 ± 1.60	25.01 ± 1.65	**24.22 ± 1.40**	24.48 ± 1.69
satimage	**12.60 ± 0.74**	13.10 ± 0.76	**12.60 ± 0.74**	**12.60 ± 0.74**	**12.60 ± 0.74**
segment	**5.84 ± 0.85**	7.40 ± 0.94	6.23 ± 0.87	5.97 ± 0.85	5.97 ± 0.86
shuttle-small	0.41 ± 0.15	**0.36 ± 0.14**	**0.36 ± 0.14**	**0.36 ± 0.14**	**0.36 ± 0.14**
soybean-large	**6.76 ± 0.99**	**6.76 ± 0.99**	8.18 ± 0.51	8.72 ± 1.10	8.36 ± 0.36
vehicle	30.14 ± 2.79	**29.54 ± 3.13**	30.14 ± 2.79	30.61 ± 3.19	31.08 ± 3.31
vote	5.75 ± 0.51	5.75 ± 0.36	**5.52 ± 0.43**	5.75 ± 0.51	5.75 ± 0.51
waveform-21	**21.04 ± 0.59**	21.26 ± 0.60	21.13 ± 0.60	**21.04 ± 0.59**	21.23 ± 0.60
Average error	13.64	14.41	13.77	**13.61**	13.62
Av. advantage	66.82	65.19	66.85	67.32	**67.36**

Table 8.26: Benchmarks on UCI datasets: five-fold cross-validation error rate for STAN classifier.

Dataset	STAN				
	HGC	SB	LC	LOO	CV1010
australian	13.48 ± 1.37	14.49 ± 1.39	**12.75 ± 0.43**	**12.75 ± 0.78**	**12.75 ± 0.78**
breast	**2.78 ± 0.43**	3.37 ± 0.86	3.08 ± 0.71	3.08 ± 0.71	3.08 ± 0.71
chess	6.47 ± 0.75	7.04 ± 0.78	**4.32 ± 0.62**	**4.32 ± 0.62**	**4.32 ± 0.62**
cleve	17.56 ± 1.54	18.58 ± 1.05	18.23 ± 1.31	**17.23 ± 1.65**	17.57 ± 0.44
corral	3.96 ± 1.79	**3.91 ± 2.46**	4.71 ± 2.88	4.71 ± 2.88	4.71 ± 2.88
crx	13.32 ± 0.70	**12.71 ± 0.84**	13.02 ± 0.80	13.32 ± 1.18	13.01 ± 0.75
diabetes	**22.92 ± 0.68**	25.13 ± 0.97	23.18 ± 0.51	**22.92 ± 0.68**	**22.92 ± 0.68**
flare	17.07 ± 2.05	16.70 ± 1.83	**16.60 ± 1.96**	17.07 ± 2.05	17.07 ± 1.80
german	26.20 ± 0.86	26.70 ± 0.58	25.70 ± 1.28	25.60 ± 1.07	**25.40 ± 1.33**
glass	27.55 ± 1.27	43.43 ± 2.62	**27.09 ± 1.51**	**27.09 ± 1.51**	**27.09 ± 1.51**
glass2	**20.27 ± 2.10**	21.50 ± 2.40	**20.27 ± 2.10**	**20.27 ± 2.10**	**20.27 ± 2.10**
heart	17.04 ± 2.51	17.04 ± 2.51	**16.30 ± 1.98**	18.15 ± 3.23	17.04 ± 2.51
hepatitis	7.50 ± 1.25	12.50 ± 0.00	**6.25 ± 0.00**	**6.25 ± 2.80**	**6.25 ± 2.80**
iris	6.00 ± 1.25	6.00 ± 1.25	6.00 ± 1.25	**5.33 ± 1.33**	6.00 ± 1.25
letter	13.24 ± 0.48	46.98 ± 0.71	**13.22 ± 0.48**	13.70 ± 0.47	13.70 ± 0.47
lymphography	**14.92 ± 2.37**	31.17 ± 4.07	16.94 ± 1.65	16.21 ± 1.97	16.21 ± 1.97
mofn-3-7-10	7.03 ± 0.80	**6.25 ± 0.76**	7.03 ± 0.80	**6.25 ± 0.76**	**6.25 ± 0.76**
pima	24.61 ± 2.12	26.17 ± 0.66	**24.22 ± 0.87**	24.35 ± 1.34	24.35 ± 1.34
satimage	12.60 ± 0.74	16.75 ± 0.84	15.50 ± 0.81	14.90 ± 0.80	**14.85 ± 0.80**
segment	4.68 ± 0.76	20.78 ± 1.46	**4.42 ± 0.74**	4.81 ± 0.77	4.81 ± 0.77
shuttle-small	**0.41 ± 0.15**	5.23 ± 0.52	0.47 ± 0.15	**0.41 ± 0.15**	**0.41 ± 0.15**
soybean-large	9.25 ± 1.07	36.29 ± 2.45	8.00 ± 1.19	7.11 ± 0.73	**6.94 ± 0.75**
vehicle	31.67 ± 2.06	34.51 ± 1.33	**28.95 ± 2.29**	29.66 ± 2.53	29.90 ± 2.45
vote	4.60 ± 0.51	5.52 ± 0.67	4.37 ± 0.43	**3.68 ± 0.76**	**3.68 ± 0.76**
waveform-21	**21.15 ± 0.60**	**21.15 ± 0.60**	**21.15 ± 0.60**	**21.15 ± 0.60**	**21.15 ± 0.60**
Average error	13.85	19.20	13.67	13.61	**13.59**
Av. advantage	66.88	57.86	67.49	67.62	**67.70**

Table 8.27: Benchmarks on UCI datasets: five-fold cross-validation error rate for STAND classifier.

Dataset	STAND				
	HGC	SB	LC	LOO	CV1010
australian	13.33 ± 1.56	14.20 ± 1.04	12.61 ± 1.31	**11.74 ± 1.15**	**11.74 ± 1.15**
breast	2.93 ± 0.33	**2.64 ± 0.38**	**2.64 ± 0.38**	3.66 ± 0.61	3.22 ± 0.60
chess	7.41 ± 0.80	7.41 ± 0.80	4.13 ± 0.61	**4.03 ± 0.60**	4.13 ± 0.61
cleve	**16.89 ± 0.49**	19.25 ± 1.34	17.90 ± 0.82	16.90 ± 1.22	17.92 ± 1.40
corral	**3.91 ± 2.46**	**3.91 ± 2.46**	**3.91 ± 2.46**	**3.91 ± 2.46**	**3.91 ± 2.46**
crx	12.86 ± 1.00	12.86 ± 0.60	12.56 ± 0.62	**12.40 ± 0.93**	**12.40 ± 0.93**
diabetes	**22.92 ± 0.68**	23.31 ± 0.71	23.83 ± 0.77	23.83 ± 0.77	23.83 ± 0.77
flare	**16.32 ± 1.52**	17.07 ± 2.05	16.70 ± 1.80	17.07 ± 2.05	17.07 ± 2.05
german	25.90 ± 0.75	27.90 ± 1.35	26.20 ± 0.82	**25.70 ± 1.31**	26.00 ± 1.21
glass	**27.09 ± 1.51**	34.10 ± 3.61	28.95 ± 1.48	28.48 ± 1.25	28.02 ± 1.17
glass2	**20.23 ± 2.79**	20.27 ± 2.10	20.27 ± 2.10	20.27 ± 2.10	20.27 ± 1.29
heart	16.67 ± 2.62	17.04 ± 2.51	16.67 ± 3.15	15.93 ± 2.16	**15.56 ± 2.08**
hepatitis	**7.50 ± 3.64**	10.00 ± 2.50	10.00 ± 3.75	8.75 ± 4.68	8.75 ± 4.68
iris	6.00 ± 1.25	6.00 ± 1.25	6.00 ± 1.25	6.00 ± 1.25	**5.33 ± 0.82**
letter	13.50 ± 0.48	25.96 ± 0.62	16.50 ± 0.52	**12.86 ± 0.47**	**12.86 ± 0.47**
lymphography	**16.23 ± 1.32**	20.94 ± 0.62	17.58 ± 1.31	**16.23 ± 1.29**	16.94 ± 3.95
mofn-3-7-10	7.03 ± 0.80	**6.25 ± 0.76**	7.03 ± 0.80	**6.25 ± 0.76**	**6.25 ± 0.76**
pima	24.35 ± 1.34	23.70 ± 1.33	23.96 ± 1.09	**22.92 ± 1.02**	23.44 ± 0.87
satimage	**12.60 ± 0.74**	**12.60 ± 0.74**	13.30 ± 0.76	12.90 ± 0.75	12.90 ± 0.75
segment	5.45 ± 0.82	7.40 ± 0.94	4.55 ± 0.75	**3.90 ± 0.70**	**3.90 ± 0.70**
shuttle-small	**0.41 ± 0.15**	0.47 ± 0.15	0.47 ± 0.15	0.47 ± 0.15	0.47 ± 0.15
soybean-large	9.25 ± 1.18	19.02 ± 3.13	8.18 ± 1.17	**6.93 ± 1.40**	**6.93 ± 0.85**
vehicle	30.73 ± 2.47	29.90 ± 3.08	30.25 ± 1.60	29.43 ± 1.95	**29.19 ± 1.91**
vote	5.52 ± 0.23	4.37 ± 0.43	4.14 ± 0.59	4.83 ± 0.43	**3.45 ± 0.51**
waveform-21	**21.15 ± 0.60**	**21.15 ± 0.60**	21.96 ± 0.60	22.30 ± 0.61	22.30 ± 0.61
Average error	13.85	15.51	14.01	**13.51**	13.47
Av. advantage	67.01	64.07	66.32	67.36	**67.40**

Table 8.28: Benchmarks on UCI datasets: five-fold cross-validation error rate for SFAN classifier.

Dataset	SFAN				
	HGC	SB	LC	LOO	CV1010
australian	13.48 ± 1.56	13.91 ± 1.06	12.90 ± 0.42	12.90 ± 0.42	**12.75 ± 0.49**
breast	**2.34 ± 0.43**	**2.34 ± 0.43**	3.37 ± 0.55	3.37 ± 0.68	3.37 ± 0.68
chess	6.47 ± 0.75	7.04 ± 0.78	4.50 ± 0.64	**4.32 ± 0.62**	4.58 ± 0.64
cleve	**17.21 ± 2.31**	18.24 ± 2.30	17.21 ± 1.59	18.24 ± 0.96	17.90 ± 0.82
corral	2.40 ± 2.40	**0.00 ± 0.00**	1.60 ± 1.60	2.40 ± 2.40	2.40 ± 2.40
crx	13.02 ± 0.68	**12.86 ± 0.82**	13.01 ± 0.62	13.63 ± 0.81	13.63 ± 0.81
diabetes	23.18 ± 0.59	23.30 ± 0.49	**22.92 ± 0.75**	23.31 ± 0.70	23.31 ± 0.70
flare	17.70 ± 2.05	**16.98 ± 2.14**	**16.98 ± 1.72**	**16.98 ± 1.80**	**16.98 ± 1.80**
german	**25.30 ± 1.55**	26.40 ± 0.97	**25.30 ± 0.25**	26.10 ± 1.16	25.70 ± 1.37
glass	27.55 ± 1.27	**20.42 ± 1.09**	28.02 ± 1.38	28.02 ± 1.38	28.02 ± 1.38
glass2	20.27 ± 2.10	**18.43 ± 2.19**	**18.43 ± 2.19**	**18.43 ± 2.19**	**18.43 ± 2.19**
heart	15.93 ± 2.39	**14.81 ± 2.11**	17.41 ± 2.39	18.15 ± 3.01	17.04 ± 3.43
hepatitis	8.75 ± 1.53	10.00 ± 1.53	8.75 ± 1.53	**6.25 ± 2.80**	**6.25 ± 2.80**
iris	5.33 ± 1.33	**4.67 ± 1.33**	5.33 ± 1.33	5.33 ± 1.33	5.33 ± 1.33
letter	**13.24 ± 0.48**	24.76 ± 0.61	16.50 ± 0.53	13.70 ± 0.49	13.70 ± 0.49
lymphography	17.59 ± 1.99	20.92 ± 1.14	18.28 ± 0.99	**15.59 ± 2.39**	16.21 ± 1.97
mofn-3-7-10	7.81 ± 0.84	11.72 ± 1.01	**7.03 ± 0.80**	7.81 ± 0.84	7.81 ± 0.84
pima	24.87 ± 1.13	24.61 ± 1.08	24.48 ± 1.32	**24.35 ± 1.31**	**24.35 ± 1.31**
satimage	**12.60 ± 0.74**	16.75 ± 0.84	15.15 ± 0.80	16.30 ± 0.82	16.75 ± 0.84
segment	**4.68 ± 0.76**	7.47 ± 0.94	5.19 ± 0.80	4.94 ± 0.78	7.14 ± 0.93
shuttle-small	0.41 ± 0.15	5.43 ± 0.52	**0.36 ± 0.14**	0.47 ± 0.15	0.47 ± 0.15
soybean-large	9.07 ± 0.90	9.60 ± 1.76	9.60 ± 1.76	**6.22 ± 1.28**	9.60 ± 1.76
vehicle	32.85 ± 2.55	34.04 ± 1.26	**29.30 ± 2.63**	30.13 ± 2.43	29.90 ± 2.79
vote	4.69 ± 0.63	5.06 ± 0.46	4.14 ± 0.86	**3.22 ± 0.76**	**3.22 ± 0.76**
waveform-21	21.15 ± 0.60	21.13 ± 0.60	**21.09 ± 0.60**	21.15 ± 0.60	21.15 ± 0.60
Average error	13.92	14.84	13.87	**13.65**	13.84
Av. advantage	66.44	64.08	66.87	**67.43**	67.28

Table 8.29: Benchmarks on UCI datasets: five-fold cross-validation error rate for SFAND classifier.

Dataset	SFAND				
	HGC	SB	LC	LOO	CV1010
australian	13.04 ± 1.73	13.77 ± 1.15	**11.74 ± 0.62**	11.88 ± 0.99	11.88 ± 0.99
breast	2.49 ± 0.37	**2.34 ± 0.43**	3.22 ± 0.59	3.66 ± 0.61	3.81 ± 0.78
chess	7.41 ± 0.80	7.41 ± 0.80	4.13 ± 0.61	**4.03 ± 0.60**	**4.03 ± 0.60**
cleve	**16.20 ± 1.33**	18.58 ± 1.84	16.54 ± 1.42	16.90 ± 1.22	17.92 ± 1.40
corral	2.40 ± 2.40	**0.00 ± 0.00**	1.60 ± 1.60	1.60 ± 1.60	2.40 ± 2.40
crx	13.01 ± 0.86	13.17 ± 0.98	**11.94 ± 0.69**	12.70 ± 1.06	12.86 ± 1.05
diabetes	23.18 ± 0.51	24.09 ± 0.88	**22.66 ± 0.68**	23.57 ± 0.48	23.57 ± 0.48
flare	**16.32 ± 1.52**	17.07 ± 2.05	16.98 ± 1.66	17.07 ± 2.05	17.07 ± 2.05
german	**25.50 ± 1.44**	25.90 ± 1.22	25.70 ± 1.15	26.60 ± 1.64	26.60 ± 1.03
glass	**27.09 ± 1.51**	28.49 ± 0.78	28.48 ± 1.25	28.95 ± 1.48	28.95 ± 1.48
glass2	20.23 ± 2.02	**17.80 ± 2.26**	18.43 ± 2.19	18.43 ± 2.19	18.43 ± 2.19
heart	17.04 ± 2.95	17.04 ± 2.51	16.67 ± 2.27	**15.56 ± 2.24**	**15.56 ± 2.24**
hepatitis	**7.50 ± 1.25**	8.75 ± 1.53	8.75 ± 3.19	10.00 ± 2.50	8.75 ± 3.19
iris	5.33 ± 1.33	**4.67 ± 1.33**	5.33 ± 1.33	5.33 ± 1.33	5.33 ± 1.33
letter	13.30 ± 0.48	22.92 ± 0.59	16.62 ± 0.53	12.86 ± 0.47	**12.78 ± 0.47**
lymphography	16.23 ± 1.32	17.61 ± 1.40	16.25 ± 1.38	**15.59 ± 1.48**	16.21 ± 1.93
mofn-3-7-10	7.81 ± 0.84	11.72 ± 1.01	**7.03 ± 0.80**	7.81 ± 0.84	7.81 ± 0.84
pima	24.36 ± 1.31	23.96 ± 1.43	24.48 ± 1.40	**23.57 ± 0.96**	**23.57 ± 0.96**
satimage	**12.60 ± 0.74**	13.10 ± 0.76	13.30 ± 0.76	12.90 ± 0.75	16.30 ± 0.83
segment	5.71 ± 0.84	7.40 ± 0.94	**5.32 ± 0.81**	6.36 ± 0.88	6.36 ± 0.88
shuttle-small	0.41 ± 0.15	0.41 ± 0.15	0.52 ± 0.16	**0.36 ± 0.14**	**0.36 ± 0.14**
soybean-large	8.89 ± 0.88	9.78 ± 1.81	7.65 ± 0.82	**6.40 ± 0.94**	**6.40 ± 0.94**
vehicle	30.14 ± 2.79	**29.54 ± 3.13**	29.90 ± 1.67	30.02 ± 1.46	**29.54 ± 1.59**
vote	5.52 ± 0.43	5.98 ± 0.43	**3.68 ± 0.76**	4.37 ± 0.43	4.37 ± 0.43
waveform-21	**21.15 ± 0.60**	21.32 ± 0.60	21.34 ± 0.60	22.17 ± 0.61	22.17 ± 0.61
Average error	13.71	14.51	**13.53**	13.55	13.72
Av. advantage	67.20	65.32	**67.51**	66.99	66.90

average error rate (lower is better) and an average advantage ratio (higher is better). The advantage ratio compares performance of a classifier to the constant classifier and is given by Eq. (8.1).

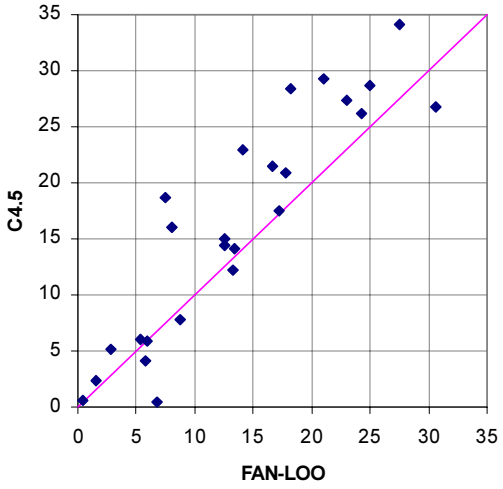
Table 8.24 contains summary of the classification results. Column TAN-FGG contains results published by Friedman et al. (1997) for a TAN classifier that does not use Dirichlet priors. Acronym FGG stands for the first names of the authors of the TAN classifier: Friedman, Geiger, and Goldszmidt. Column TAN-FGG-S contains results published by Friedman et al. (1997) for a TAN classifier that uses Dirichlet priors (smoothing); all of the α_{ijk} parameters were set to be the same and equal 5.

8.4.3 Discussion of the Results

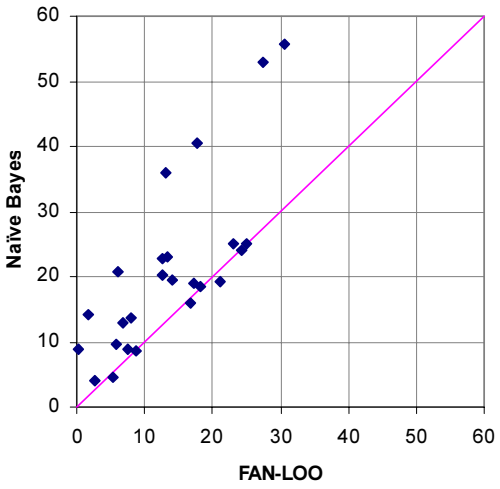
We have selected three new classifiers, each using different search algorithm, and compared them graphically to the reference classifiers. The reference classifier used for comparison are C4.5, naïve Bayes, TAN-FGG, and TAN-FGG-S classifier. The new classifiers used for graphical comparison are FAN-LOO, STAND-LOO, and SFAND-LC, presented in figures 8.4, 8.5, and 8.6, respectively. As before, axis in each diagram represent percentage error rate. The pink diagonal line represents equality of error for the two compared classifiers. A blue mark represents a dataset. When a blue mark is above the line it means that a reference classifier compared to a new classifier had larger error for that dataset. When a mark is below the diagonal pink line it means that the new classifier had larger error.

Overall Performance of the New Bayesian Network Classifiers

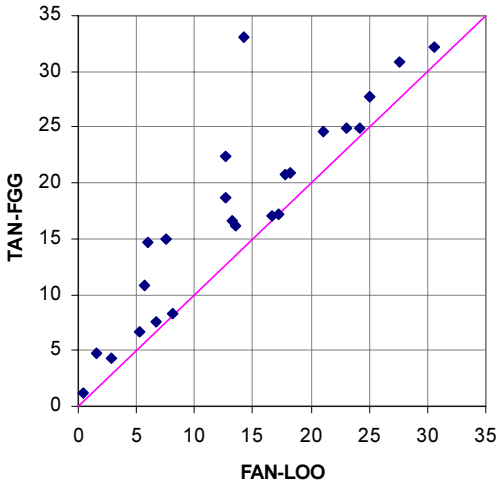
Fig. 8.7 shows average error rate and average advantage ratio of the new classifiers compared to the two best reference classifiers, TAN-FGG-S and C4.5. Notice, that only in some cases when new algorithms are combined with Standard Bayesian measure (SB) they perform equal or worse than the reference classifiers. For all other quality measures and all search algorithms the performance is better than that of the reference classifiers.



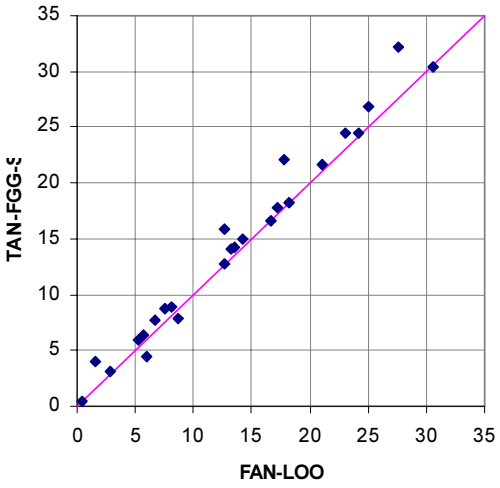
(a) FAN-LOO versus C4.5 classifier.



(b) FAN-LOO versus naïve Bayes classifier.

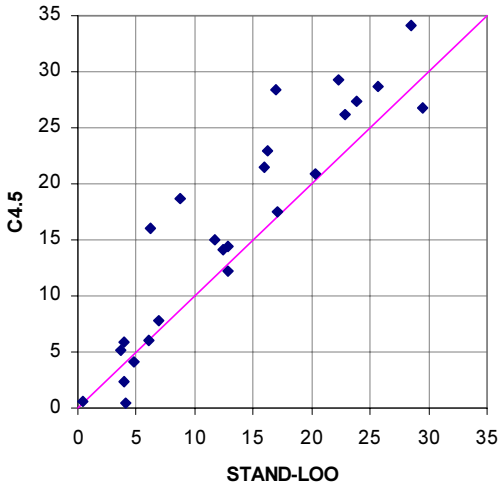


(c) FAN-LOO versus TAN-FGG classifier.

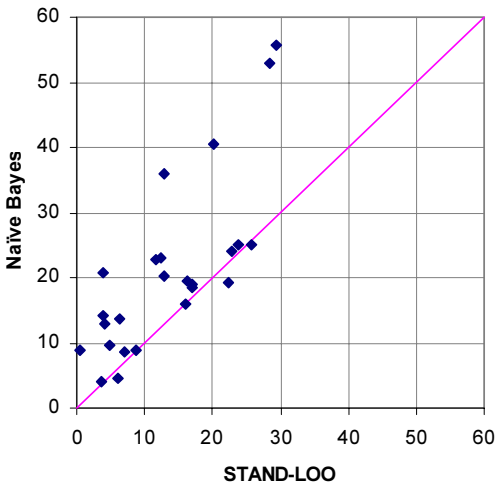


(d) FAN-LOO versus TAN-FGG-S classifier.

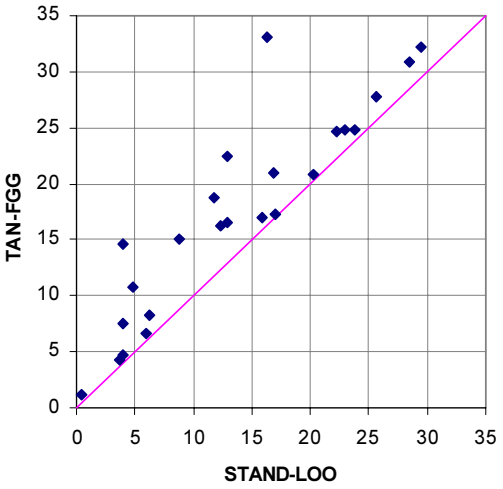
Figure 8.4: Comparison of FAN-LOO inducer versus C4.5, naïve Bayes, TAN-FGG, and TAN-FGG-S inducers on datasets from UCI Machine Learning Repository.



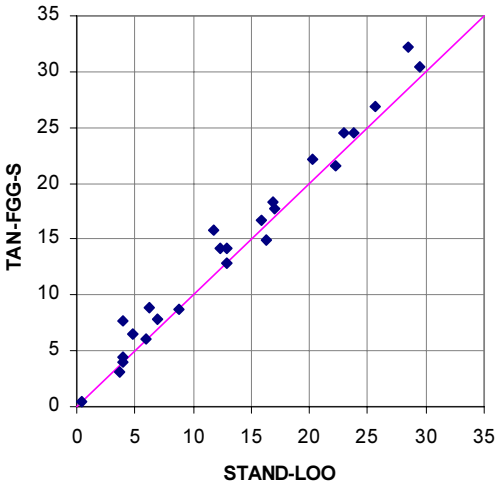
(a) STAND-LOO versus C4.5 classifier.



(b) STAND-LOO versus naïve Bayes classifier.

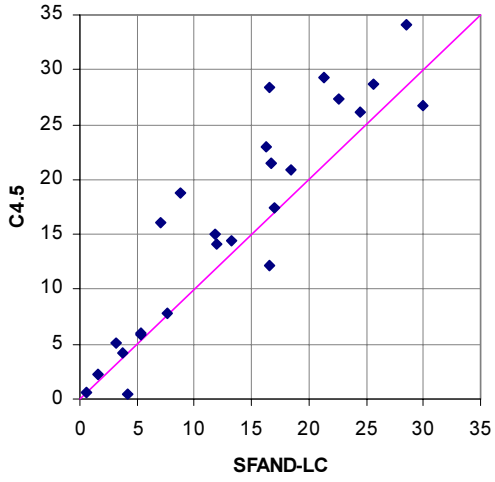


(c) STAND-LOO versus TAN-FGG classifier.

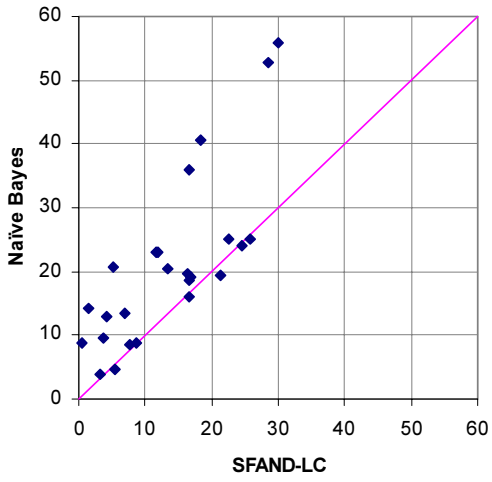


(d) STAND-LOO versus TAN-FGG-S classifier.

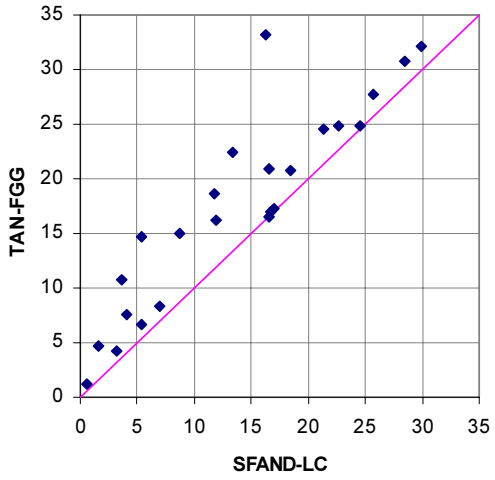
Figure 8.5: Comparison of STAND-LOO inducer versus C4.5, naïve Bayes, TAN-FGG, and TAN-FGG-S inducers on datasets from UCI Machine Learning Repository.



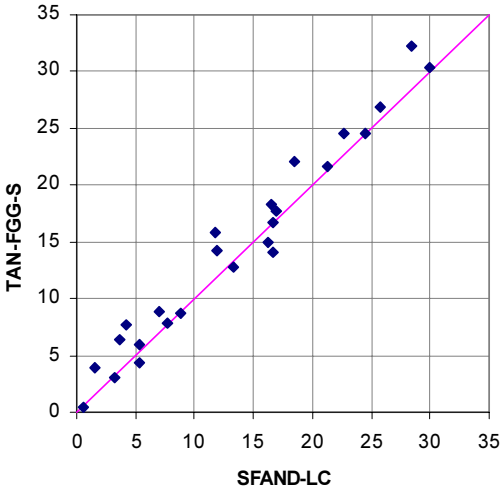
(a) SFAND-LC versus C4.5 classifier.



(b) SFAND-LC versus naïve Bayes classifier.



(c) SFAND-LC versus TAN-FGG classifier.



(d) SFAND-LC versus TAN-FGG-S classifier.

Figure 8.6: Comparison of SFAND-LC inducer versus C4.5, naïve Bayes, TAN-FGG, and TAN-FGG-S inducers on datasets from UCI Machine Learning Repository.

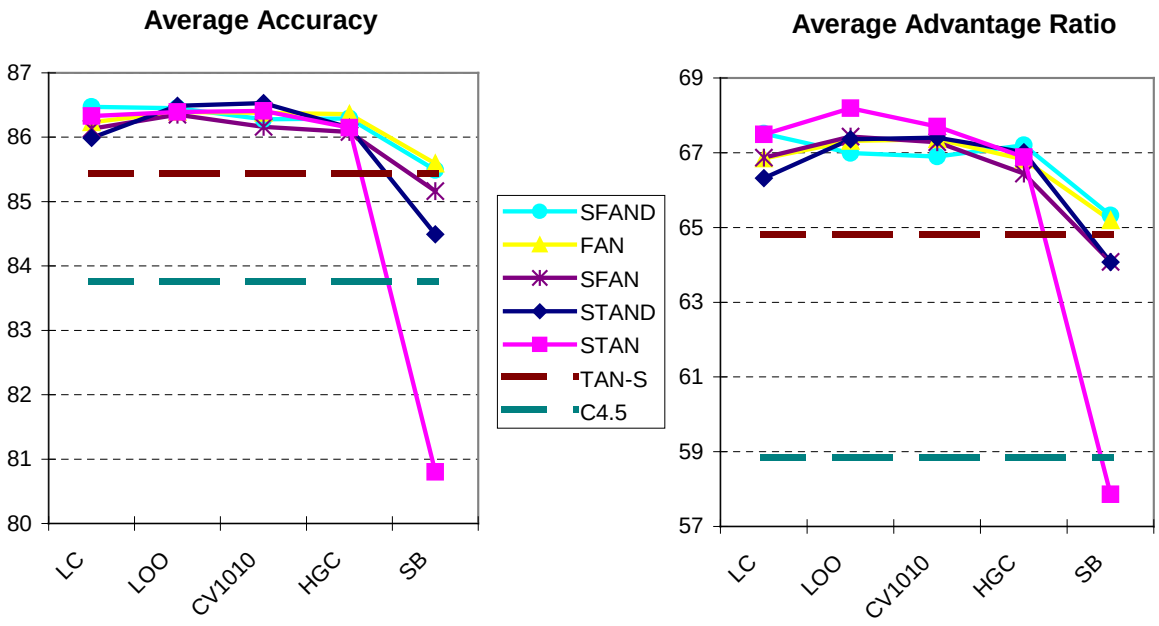


Figure 8.7: Benchmarks on UCI datasets: Average error rate and average advantage ratio for the new classifiers and for best reference classifiers: TAN-FGG-S and C4.5.

To quantitatively compare the performance of the new family of Bayesian network classifiers to the reference classifiers (the constant classifier, C4.5, naïve Bayes, TAN-FGG, and TAN-FGG-S) we used approach similar to the one presented in Section 8.2. We used the same three indicators:

Dataset error rate indicates for how many datasets the error rate of the best new Bayesian network classifier was better (lower), equal to, or worse (higher) than that of the best reference classifier for a given dataset.

Average error rate indicates how many of the new Bayesian network classifiers had average error rate that is better (lower), equal to, or worse (higher) than that of the best (lowest) average error rate of the reference classifiers.

Average advantage ratio indicates how many of the new Bayesian network classifiers had average advantage ratio, Eq.(8.1), that is better (higher), equal to, or worse (lower) than that of the best (highest) average advantage ratio of the reference classifiers.

Table 8.30: Comparison of the new family of Bayesian network classifiers to reference classifier (the constant classifier, C4.5, naïve Bayes, TAN-FGG, and TAN-FGG-S) using datasets from UCI Machine Learning Repository. A number indicates how many times the best of the new classifiers was better, equal, or worse than the best of reference classifiers.

Indicator	Better		Equal		Worse	
Dataset error rate	20	80%	1	4%	4	16%
Average error rate	21	84%	0	0%	4	16%
Average advantage ratio	22	88%	0	0%	3	12%

The indicators are shown in Table 8.30. The table demonstrates that the new family of Bayesian network algorithms produces classifiers that are performing significantly better than the reference classifiers in a variety of domains. Notice, that if we excluded classifier using Standard Bayesian quality measure (SB) the average error rate and average advantage ratio for new classifiers would always be better (100%) than the reference classifiers.

Chapter 9

Conclusions and Suggestions for Future Research

In this dissertation, we dealt with the issues of automating cardiac SPECT image interpretation: creation of a database of training cases, processing of 3D SPECT images and feature extraction, and use of a new family of Bayesian network classifiers for learning diagnosis of left ventricular perfusion.

9.1 Summary of the Results

In chapter 3 we discussed the process of knowledge discovery in databases. Creation of the new cardiac SPECT database has been described there. It discussed database organization and initial cleaning of data.

In chapter 4 we discussed process of inspecting cardiac SPECT images performed by a cardiologist. Then we used it as an inspiration for feature extraction process based on building a model of normal left ventricle.

In chapter 7 we introduced a new synthesis of Bayesian network classifiers and presented five new search algorithms created that were using this synthesis. It also demon-

strated that naïve Bayes and TAN classifier are special cases of the synthesis.

In chapter 8 we presented empirical evaluation of the new family of algorithms. It was demonstrated that the new classifiers are able to deal better than existing algorithms with the high level of noise present in the cardiac SPECT data and features extracted from 3D SPECT images. It was also shown that the new algorithms outperform existing ones on datasets from UCI Machine Learning Repository.

9.2 Suggestions for Future Research

- Creation of a larger database that would contain statistically significant number of examples for each diagnosed left ventricular perfusion defect.
- Use of information about motion of the left ventricle (gated-pool cardiac SPECT) and use of physics-based deformable models.
- Consider improved system for recording the information at a hospital about diagnosed cases.
- New quality measures for scoring the Bayesian networks for classifiers, for instance, a variant of the LC measure with a penalty for the network size.
- Heuristics for adjustments of network parameter priors based on information about a training dataset.

9.3 Concluding Remarks

The most significant contribution of this research was the introduction of the new family of Bayesian network classifiers. High performance of these classifiers was demonstrated, not only on cardiac SPECT data, but also on data from a variety of domains using UCI Machine Learning datasets.

Unfortunate conclusion of this research was that the number of available cardiac SPECT imaging cases was not sufficiently large to reliably classify left ventricular perfusion, and that a significantly larger number needs to be collected before attempting practical applications.

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, *19*, 716-723.

Aliferis, C. F., and Cooper, G. F. (1994). An evaluation of an algorithm for an inductive learning of Bayesian belief networks using simulated data sets. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence* (p. 8-14). Seattle, WA: Morgan Kaufmann.

Auer, P., Holte, R., and Maass, W. (1995). Theory and application of agnostic PAC-learning with small decision trees. In A. Preditis and S. Russell (Eds.), *Proceedings of the Twelfth International Conference on Machine Learning.* Morgan Kaufmann.

Bishop, C. M. (1995). *Neural networks for pattern recognition.* New York: Oxford University Press.

Blake, C., Keogh, E., and Merz, C. (1998). *UCI repository of machine learning databases.* (`http://www.ics.uci.edu/~mlearn/MLRepository.html`)

Bouckaert, R. R. (1994). Properties of bayesian network learning algorithms. In R. L. de Mantarás and D. Poole (Eds.), *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence* (p. 102-109). San Francisco, CA.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and regression trees.* Monterey, CA: Wadsworth & Brooks.

Buntine, W. (1991a). Classifiers: A theoretical and empirical study. In *International Joint Conference on Artificial Intelligence.* Sydney.

Buntine, W. (1991b). Theory refinement on Bayesian networks. In B. D. D'Ambrosio, P. Smets, and P. P. Bonissone (Eds.), *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence* (p. 52-60). Los Angeles, CA: Morgan Kaufmann.

Buntine, W. L. (1994). Operations for learning with graphical models. *Journal of Artificial Intelligence Research*, *2*, 159-225.

Castillo, E., Gutiérrez, J. M., and Hadi, A. S. (1997). *Expert systems and probabilistic network models.* New York: Springer-Verlag.

Catlett, J. (1991). On changing continuous attributes into ordered discrete attributes. In Y. Kodratoff (Ed.), *Proceedings of the European Working Session on Learning* (p. 164-178). Berlin, Germany: Springer-Verlag.

Chapman, P., Clinton, J., Khobaza, T., Reinartz, T., and Wirth, R. (1999, March). *The CRISP-DM process model.* Discussion paper, Available from `http://www.crisp-dm.org/`, CRISP-DM Consortium.

Cheng, J., Bell, D. A., and Liu, W. (1997a). An algorithm for Bayesian belief network construction from data. In *Proceedings of the Sixth International Workshop on Artificial Intelligence and Statistics, AI&STAT'97* (p. 83-90).

Cheng, J., Bell, D. A., and Liu, W. (1997b). Learning belief networks: An information theory based approach. In *Proceedings of the Sixth ACM International Conference on Information and Knowledge Management, CIKM'97.*

Cheng, J., and Greiner, R. (1999). Comparing Bayesian network classifiers. In K. Laskey and H. Prade (Eds.), *Proceedings of the Fifteenth International Conference on Uncertainty in Artificial Intelligence.* Morgan Kaufmann.

Chickering, D. M. (1996). Learning equivalence classes of Bayesian-network structure. In E. Horovitz and F. Jensen (Eds.), *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence.* San Francisco, CA: Morgan Kaufmann.

Chow, C. K., and Liu, C. N. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, *14*(462-467).

Ciesielski, K., Sacha, J. P., and Cios, K. J. (1999). *Synthesis of neural networks in supremum error bound.* Submitted to *IEEE Transactions on Neural Networks*.

Cios, K. J., Goodenday, L. S., Shah, K. K., and Serpen, G. (1996). A novel algorithm for classification of SPECT images of a human heart. In *Proceedings of the CBMS'96.* Ann Arbor, MI.

Cios, K. J., Pedrycz, W., and Swiniarski, R. W. (1998). *Data mining methods for knowledge discovery.* Kluwer.

Cios, K. J., Teresinska, A., Konieczna, S., Potocka, J., and Sharma, S. (2000). Diagnosing myocardial perfusion SPECT bull's-eye map – a knowledge discovery approach. *IEEE Engineering in Medicine and Biology Magazine*. (in print)

Clark, P., and Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning*, *3*, 261-284.

Cooper, G. (1990). Computational complexity of probabilistic inference using Bayesian belief networks (Research note). *Artificial Intelligence*, *42*, 393-405.

Cooper, G. F., and Herskovits, E. (1992). Bayesian method for induction of probabilistic networks from data. *Machine Learning*, *9*, 309-347.

Corbett, J. R. (1994). Gated blood-pool SPECT. In E. G. DePuey, D. S. Berman, and E. V. Garcia (Eds.), *Cardiac SPECT imaging.* New York: Raven Press.

Cormen, T. H., Leiserson, C. E., and Rivest, R. L. (1990). *Introduction to algorithms.* The MIT Press.

Cuaron, A., Acero, A. P., Cardenas, M., et al.. (1980). Interobserver variability in the interpretation of myocardial images with Tc-99m-labeled diphosphonate. *Journal of Nuclear Medicine, 21*(1).

Cullom, S. J. (1994). Principles of cardiac SPECT. In E. G. DePuey, D. S. Berman, and E. V. Garcia (Eds.), *Cardiac SPECT imaging.* New York: Raven Press.

D'Ambrosio, B. (1991). Local expression languages for probabilistic dependence. In *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence* (p. 95-102). Los Angeles, CA: Morgan Kaufmann.

David, P. (1894). Present position and potential developments: Some personal views, statistical theory, the prequential approach (with discussion). *Journal of the Royal Statistical Society A, 147*, 178-292.

Dawid, P. (1992). Application of general propagation algorithm for probabilistic expert systems. *Statistics and Computing, 2*, 25-36.

Declerck, J., Feldmar, J., Goris, M. L., and Betting, F. (1997). Automatic registration and alignment on a template of a cardiac stress and rest reoriented SPECT images. *IEEE Transactions on Medical Imaging, 16*(6), 727-737.

DePuey, E. G., Berman, D. S., and Garcia, E. V. (Eds.). (1994). *Cardiac SPECT imaging.* New York: Raven Press.

Dougherty, J., Kahavi, R., and Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. In A. Prieditis and S. Russell (Eds.), *Proceedings of the Twelfth International Conference on Machine Learning.* San Francisco, CA: Morgan Kaufmann.

Draper, D. L., and Hanks, S. (1994). Localized partial evaluation of belief networks. In *Proceedings of the Thenth Conference on Uncertainty in Artificial Intelligence.* Morgan Kaufmann.

Duda, R. O., and Hart, P. E. (1973). *Pattern classification and scene analysis.* New York: John Wiley & Sons.

Encarta 99 Encyclopedia. (1999). Microsoft Corporation.

Even, S. (1979). *Graph algorithms.* Computer Science Press.

Ezquerra, N., Mullick, R., Cooke, D., Garcia, E., and Krawczynska, E. (1992). *PER-FEX: An expert system for interpreting 3D myocardial perfusion* (Tech. Rep. No. GIT-GVU-92-06). Atlanta, GA 30332-0280: Graphics, Visualization and Usability Center, Georgia Institute of Technology.

Faber, T. L., Akers, M. S., Peshock, R. M., and Corbett, J. R. (1991). Three-dimensional motion and perfusion quantification in gated single-photon emission computed tomograms. *Journal of Nuclear Medicine, 32*, 2311.

Faber, T. L., Cooke, C. D., Peifer, J. W., et al.. (1995). Three-dimensional displays of left ventricular epicardial surface from standard cardiac SPECT perfusion quantification techniques. *Journal of Nuclear Medicine, 36*, 697-703.

Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine, 17*(3), 37-54.

Fayyad, U. M., and Irani, K. B. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence* (p. 1022-1027). Morgan Kaufmann.

Francisco, D. A., Collins, S. M., Go, R. T., et al.. (1982). Tomographic thallium-201 myocardial perfusion scintigrams after maximal coronary vasodilation with intravenous

dipyridamole: Comparison of qualitative and quantitative approaches. *Circulation*, *66*(2), 370.

Friedman, J. H. (1997). On bias, variance, 0/1 - loss, and the curse-of-dimensionality. *Data Mining and Knowledge Discovery*, *1*(1), 55-77.

Friedman, N., Geiger, D., and Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, *29*(2/3), 131-163.

Friedman, N., Goldszmidt, M., and Lee, T. J. (1998). Bayesian network classification with continuous attributes: Getting the best of both discretization and parametric fitting. In *Proceedings of the Fifteenth International Conference on Machine Learning.*

Garcia, E. V., Ezquerra, N. F., DePuey, E. G., et al.. (1986). An artificial intelligence approach to interpreting thallium-201 3-dimensional myocardial distributions. *Journal of Nuclear Medicine*, *27*(1005). (Abstract)

Geiger, D. (1992). An entropy-based learning algorithm of Bayesian conditional trees. In D. Dubois, M. P. Wellman, B. D. D'Ambrosio, and P. Smets (Eds.), *Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence* (p. 92-97). San Francisco, CA: Morgan Kaufmann.

Geiger, D., and Heckerman, D. (1994). *A characterization of the Dirichlet distributions through global and local parameter independence* (Tech. Rep. No. MSR-TR-94-16). Redmond, WA: Microsoft Research.

Glymour, C., Madigan, D., Pregibon, D., and Smyth, P. (1997). Statistical themes and lessons for data mining. *Data Mining and Knowledge Discovery*, *1*(1), 11-28.

Goris, M. L., Boudier, S., and Briandet, P. A. (1987). Two-dimensional mapping of three-dimensional SPECT data: A preliminary step to the quantification of thallium my-

ocardial perfusion single photon emission tomography. *American Journal of Physiologic Imaging*, *2*, 176-180.

Haddawy, P. (1999). An overview of some recent developments in Bayesian problem-solving techniques. *AI Magazine*, *20*(2), 11-19.

Heckerman, D. (1996). *A tutorial on learning with Bayesian networks* (Tech. Rep. No. MSR-TR-95-06). Redmond, WA: Microsoft Research. (Also reprinted in (Jordan, 1998))

Heckerman, D., and Geiger, D. (1995). *Likelihood and parameter priors for Bayesian networks* (Tech. Rep. No. MSR-TR-95-54). Redmond, WA: Microsoft Research.

Heckerman, D., Geiger, D., and Chickering, D. M. (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, *20*, 197-243.

Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, *11*, 63-90.

Howard, R. A., and Matheson, J. E. (1983). Influence diagrams. In R. A. Howard and J. E. Matheson (Eds.), *Readings on the principles and applications of decision theory* (Vol. II, p. 719-762). Strategic Decision Group.

Huang, C., and Darwiche, A. (1994). Inference in belief networks: A procedural guide. *International Journal of Approximate Reasoning*, *11*.

Jaakkola, T. S. (1997). *Variational methods for inference and estimation in graphical models.* Ph.D. Dissertation, Massachusetts Institute of Technology.

Jensen, F., Lauritzen, S., and Olesen, K. (1990). Bayesian updating in recursive graphical models by local computations. *Computational Statistics Quarterly*, *4*, 269-282.

Jensen, F. V. (1996). *An introduction to Bayesian networks.* Springer.

John, G. H., and Langley, P. (1995). Estimating continuous distributions in Bayesian classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence.* San Mateo, CA: Morgan Kaufmann.

Jordan, M. I. (Ed.). (1998). *Learning in graphical models.* Kluwer.

Jordan, M. I., and Bishop, C. M. (1997). Neural networks. In A. B. Tucker (Ed.), *Computer science & engineering handbook.* CRC Press.

Jordan, M. I., Ghahramani, Z., Jaakkola, T., and Saul, L. K. (1998). An introduction to variational methods for graphical models. In M. I. Jordan (Ed.), *Learning in graphical models.* Kluwer.

Kass, R., and Raftery, A. (1995). Bayes factors. *Journal of American Statistical Association, 90,* 773-795.

Kjærulff, U. (1993). *Approximation of Bayesian networks through edge removals* (Tech. Rep. No. 94-2007). Denmark: Aarlborg University, Department of Mathematics and Computer Science.

Kohavi, R., Sommerfield, D., and Dougherty, J. (1997). Data mining using MLC++, a machine learning library in C++. *International Journal of Artificial Intelligence Tools*, *6*(4), 537-566.

Kononenko, I. (1990). Comparison of inductive and naive Bayesian learning approaches to automatic knowledge acquisition. In B. Welinga et al. (Eds.), *Current trends in knowledge acquisition.* Amsterdam: IOS Press.

Kononenko, I. (1991). Semi-naive Bayesian classifier. In *Proceedings of the Sixth European Working Session on Learning* (p. 206-219). Porto, Portugal: Pittman.

Kontkanen, P., Myllymäki, P., Silander, T., and Tirri, H. (1999). On supervised selection of Bayesian networks. In K. Laskey and H. Prade (Eds.), *Proceedings of the Fifteenth*

International Conference on Uncertainty in Artificial Intelligence (p. 334-342). Morgan Kaufmann.

Krause, P. J. (1998). *Learning probabilistic networks.* Available from `http://www.auai.org/bayesUS_krause.ps.gz`, or by request from the author `krause@prl.research.philips.com`.

Lam, W., and Bacchus, F. (1994). Learning Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence, 10*, 269-193.

Langley, P., Iba, W., and Thompson, K. (1992). An analysis of Bayesian classifiers. In *Proceedings of the Tenth National Conference on Artificial Intelligence.* San Jose: AAAI Press.

Langley, P., and Sage, S. (1994). Induction of selective Bayesian classifiers. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence.* Seattle, WA: Morgan Kaufmann.

Langley, P., and Sage, S. (1999). Tractable average-case analysis of naive Bayesian classifier. In *Proceedings of the Sixteenth International Conference on Machine Learning.* Bled, Slovenia: Morgan Kaufman.

Lauritzen, S. L. (1992). Propagation of probabilities, means, and variances in mixed graphical association models. *Journal of American Statistical Association, 87,* 1098-1108.

Lauritzen, S. L., and Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society B, 50*(2), 240-265.

Maass, W. (1994). Efficient agnostic PAC-learning with simple hypothesis. In *Proceedings*

of the Seventh Annual ACM Conference on Computational Learning Theory (p. 67-75).

Madigan, D., and Raftery, A. (1994). Model selection and accounting for model uncertainty in graphical models using Occam's window. *Journal of American Statistical Association, 89,* 1535-1546.

Monti, S., and Cooper, G. F. (1999). A Bayesian networks classifier that combines a finite mixture model and a naïve Bayes model. In K. Laskey and H. Prade (Eds.), *Proceedings of the Fifteenth International Conference on Uncertainty in Artificial Intelligence.* Morgan Kaufmann.

Neal, R. (1996). *Bayesian learning for neural networks.* New York: Springer-Verlag.

Omstead, S. (1983). *On representing and solving decision problems.* Ph.D. Dissertation, Stanford University, Department of Engineering-Economic Systems.

Pearl, J. (1986). Fusion, propagation, and structuring in belief networks. *Artificial Intelligence, 29,* 241-288.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks for plausible inference.* Morgan Kaufmann.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical recipes in c: The art of scientific computing* (Second ed.). Cambridge University Press.

Qian, J., Mitsa, T., and Hoffman, E. (1996). Contour/surface registration using a physically deformable model. In *IEEE Workshop on Mathematical Methods in Biomedical Image Analysis.* San Francisco.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning, 1,* 81-106.

Quinlan, J. R. (1993). *C4.5: Programming for machine learning.* Morgan Kaufmann.

Rice, J. A. (1988). *Mathematical statistics and data analysis.* Wadswordth & Brooks/Cole.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica, 14,* 465-471.

Rissanen, J. (1989). *Stochastic complexity in statistical inquiry.* River Edge, NJ: World Scientific.

Robinson, R. W. (1977). Computing unlabeled acyclic digraphs. In A. Dold and B. Eckmann (Eds.), *Lecture notes in mathematics 622: Combinatorial mathematics V.* Berlin: Springer-Verlag.

Sacha, J. P., and Cios, K. J. (1994). On the synthesis and complexity of feedforward networks. In *The 1994 International Joined Conference on Neural Networks* (Vol. IV, p. 2185-2190). Orlando, Florida.

Sacha, J. P., Cios, K. J., and Goodenday, L. (2000). Issues in automating cardiac SPECT diagnosis. *IEEE Engineering in Medicine and Biology Magazine*. (in print)

Sacha, J. P., Cios, K. J., Roth, D. J., and Vary, A. (1995a). Application of fuzzy reasoning for filtering and enhancement of ultrasonic images. In *OAI Neural Network Symposium and Workshop.* Athens, Ohio.

Sacha, J. P., Cios, K. J., Roth, D. J., and Vary, A. (1995b). USPVA: Package for processing, visualization, and analysis of ultrasonic data. In *1995 MATLAB Conference.* Boston, Massachusetts.

Sacha, J. P., Cios, K. J., Roth, D. J., and Vary, A. (1996). *USPVA: Package for processing, visualization, and analysis of ultrasonic data* (Tech. Rep. Nos. NASA-TM-110451, NAS 1.15:110451, NIPS-96-35578). Cleveland, Ohio: NASA Lewis Research Center.

Sacha, J. P., Cios, K. J., Roth, D. J., and Vary, A. (1999). Application of fuzzy reasoning for filtering and enhancement of ultrasonic images. *Journal of Mathematical Modeling and Scientific Computing*. (in print)

Sacha, J. P., Shabestari, B. N., and Cios, K. J. (1996). Structured region growing and recognition algorithm for nondestructive evaluation. In *Applications of Digital Image Processing XIX, SPIE's Annual Meeting* (p. 87-94). Denver, Colorado.

Sacha, J. P., Shabestari, B. N., and Kohler, T. A. (1997). On the design of a flexible image processing library in C++. In *Machine Vision Applications, Architectures, and Systems Integration VI* (p. 82-89). Pittsburgh, Pennsylvania.

Schroeder, W., Martin, K., and Lorensen, B. (1998). *The visualization toolkit* (Second ed.). Prentice Hall.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics, 6*, 461-464.

Shachter, R., and Kenley, C. (1989). Gaussian influence diagrams. *Management Science*, *35*, 527-550.

Shachter, R. D. (1988). Probabilistic inference in influance diagrams. *Operations Research*, *36*, 589-604.

Singh, M., and Provan, G. (1995). A comparison of induction algorithms for selective and non-selective Bayesian classifiers. In *Proceedings of the Twelfth International Conference on Machine Learning* (p. 497-505).

Singh, M., and Voltara, M. (1995). Construction of Bayesian network structures from data: A brief survey and an efficient algorithm. *International Journal of Approximate Reasoning*, *12*, 111-131.

Spiegelhalter, D. J., David, A. P., Lauritzen, S. L., and Cowell, R. G. (1993). Bayesian analysis in expert systems. *Statistical Science*, *8*(3), 219-283.

Spiegelhalter, D. J., and Lauritzen, S. L. (1990). Sequential updating of conditional probabilities on directed graph structures. *Networks*, *20*, 579-605.

Spites, P., Glymour, C., and Scheines, R. (1991). An algorithm for fast recovery of sparse casual graphs. *Social Science Computer Review*, *9*, 62-72.

Spites, P., and Meek, C. (1995). Learning Bayesian networks with discrete variables from data. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining.* Montreal Que.: Morgan Kaufmann.

Suzuki, J. (1993). A construction of Bayesian networks from databases based on an MDL scheme. In D. Heckerman and A. Mamdani (Eds.), *Proceedings of the Nine Conference on Uncertainty in Artificial Intelligence* (p. 266-273). San Francisco, CA: Morgan Kaufmann.

Van Train, K. F., Garcia, E. V., Cooke, C. D., and Areeda, J. (1994). Quantitative analysis of SPECT myocardial perfusion. In E. G. DePuey, D. S. Berman, and E. V. Garcia (Eds.), *Cardiac SPECT imaging.* New York: Raven Press.

Wermuth, N., and Lauritzen, S. (1983). Graphical and recursive models for contingency tables. *Biometrika*, *72*, 537-552.

Appendix A

Database, Visualization, and Feature Extraction Software

A.1 Database

The original data, containing patient information and cardiologist's evaluation of SPECT images, was recorded at Medical College of Ohio (MCO) in an MS Excel spreadsheet. Thus natural choice for the database software was MS Access. The spreadsheet was converted into a single table with the same format to allow easy addition of new patient records as they became available. The records in the table were indexed using combination: the SPECT study date and the patient hospital number.

The MS Access database, however, was used only as the main repository and for manual operation on data. Automated operations on data from non-Microsoft software proven to be very unreliable, despite that, over time, we went through a number of updates of Access drivers from Microsoft. Thus, for automated operation the required portion of data were converted to Paradox database format, maintaining the same structure of the data.

A.1.1 Cardiac SPECT Images

The cardiac SPECT images obtained from MCO were stored in a proprietary format. We reverse-engineered the format to the extent that allow as reading on image data and patient identification data stored with images. A C++ class library for manipulation of cardiac SPECT images has been created.

The images were stored at MCO one patient case per directory. The directory typically contained preprocessed 3D SPECT images, additional patient information related to image creation, and in some cases the original unprocessed projection data (data form SPECT camera before 3D reconstruction through back-projections). There were typically six 3D SPECT images per directory: short axis view, horizontal long axis view, and vertical long axis view for rest and stress study.

We maintained the case-per-directory organization. We decided to keep the actual image data outside of the database (did not included them into tables). We renamed each image directory using combination of SPECT data and patients' hospital number. The directories were grouped by year and month. Having this organization, it was easy to automatically locate images knowing SPECT date and hospital number. Additionally, to optimize SQL queries, we added to the database a table containing list of all available images. The table had two fields SPECT date and patients' hospital number. The image data were quite large, over 3GB. To conserve space, each image file was individually compressed using, freely available, gzip utility. We decided to use gzip format since it provides good compression and since C++ libraries for compression/uncompression in gzip format are freely available. After compression images unoccupied less then 0.7GB.

There were more than 8,000 image files. We have created, using Borland C++Builder, a number of fully automated software utilities for maintenance of the images. These included:

- Verification whether files in an image directory correspond to a single case.

- Renaming of the directories, based on information read from images, to comply with convention described above (SPECT date, hospital number)
- Automated creation of image indexes in the database or verification that information in existing indexes is correct.

A.1.2 Other Structures in the Database

The database contained almost any information generated during model building, feature extraction, and classification stored. Each information type was stored in dedicated tables:

- Information about created models of normal left ventricle.
- Transformation needed to register each of the SPECT images to corresponding model of normal left ventricle and the quality of correlation.
- A number of tables corresponding to various feature extraction experiments. The final experiments were reported in Chapter 8.
- Datasets generated from SPECT data used for experiments described in Chapter 8.
- Results of experiments described in Chapter 8.

A.2 Visualization

A.2.1 SPECT Image Browser

As described in Chapter 3, the first two steps of a knowledge discovery process is understanding the problem domain and understanding the data. One of the first software applications we created was for visualization of the 3D SPECT images. This tool is called SPECT Image Browser. It is shown in Fig. A.1. The tool displays, side by side, a slice of a SPECT image and a 3D rendering of image iso-surface. SPECT date and patient number contained

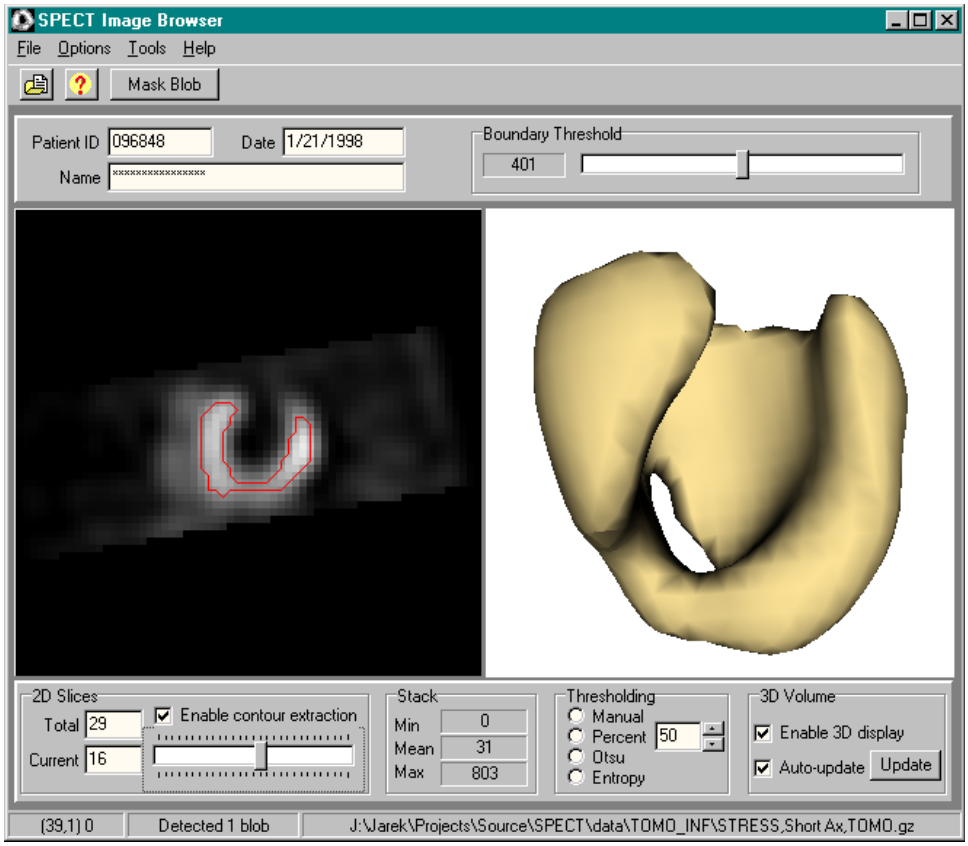


Figure A.1: SPECT Image Browser – a tool for visualization of 3D SPECT images.

in the image is also displayed. User can specify iso-surface level (threshold level) or it can be determined automatically. User has an option to display the threshold level overlayed on the slice image. It is shown in Fig. A.1 with a red line. The tool can automatically convert between axis views (short axis, horizontal long axis, vertical long axis). Image slices can be extracted to individual files. The 3D image can be converted to TIFF stack format. The 3D rendering can be saved in VRML (Virtual Reality Markup Language) and can be then displayed with a number of VRML viewers or Internet browsers. We have used the Visualization Toolkit (VTK) library for 3D rendering (Schroeder et al., 1998). The SPECT Image Browser was designed for inspection of individual 3D SPECT images.

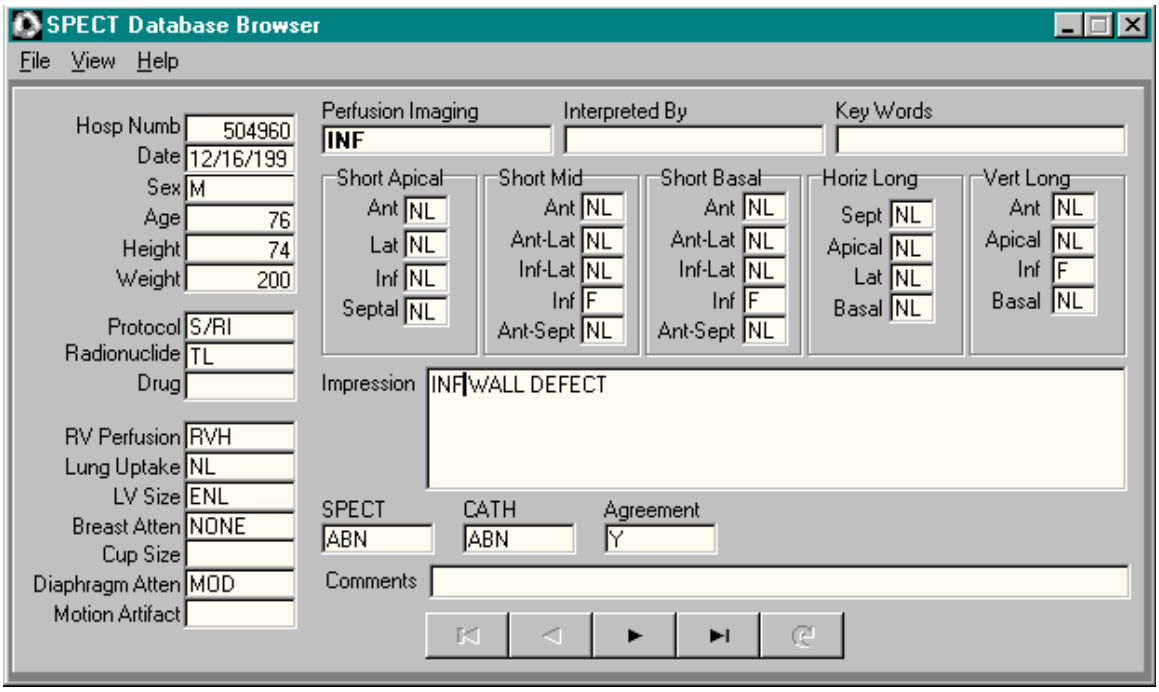


Figure A.2: Patient data display window of the SPECT database browser.

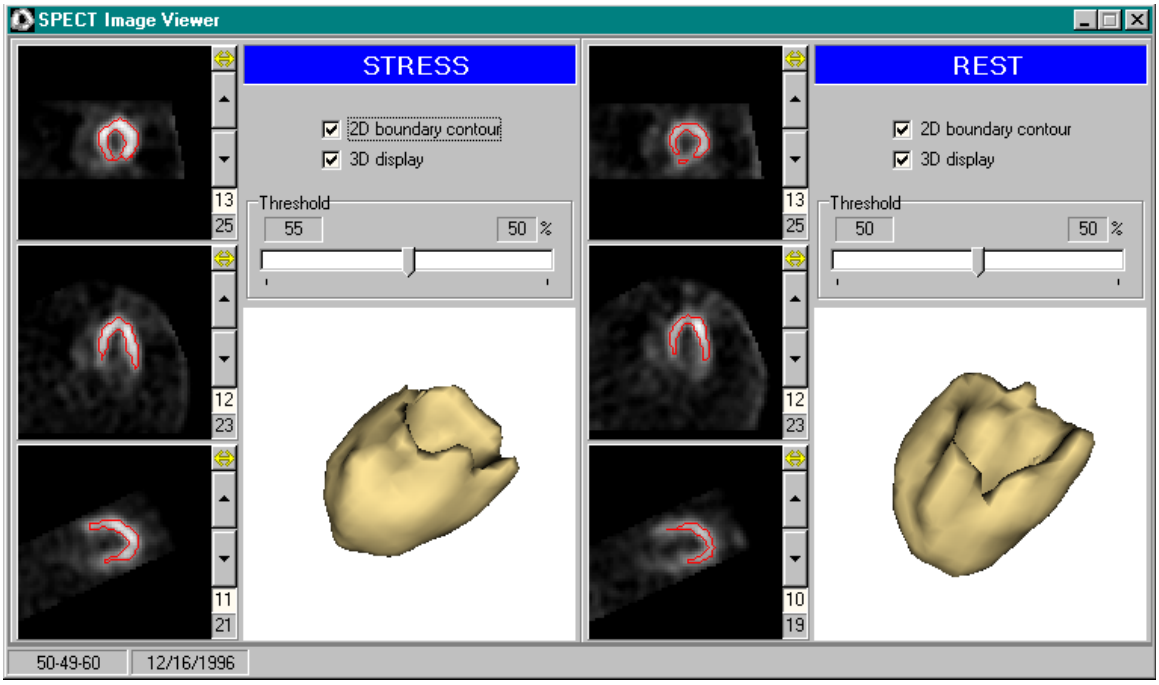


Figure A.3: Main image display window of the SPECT database browser.

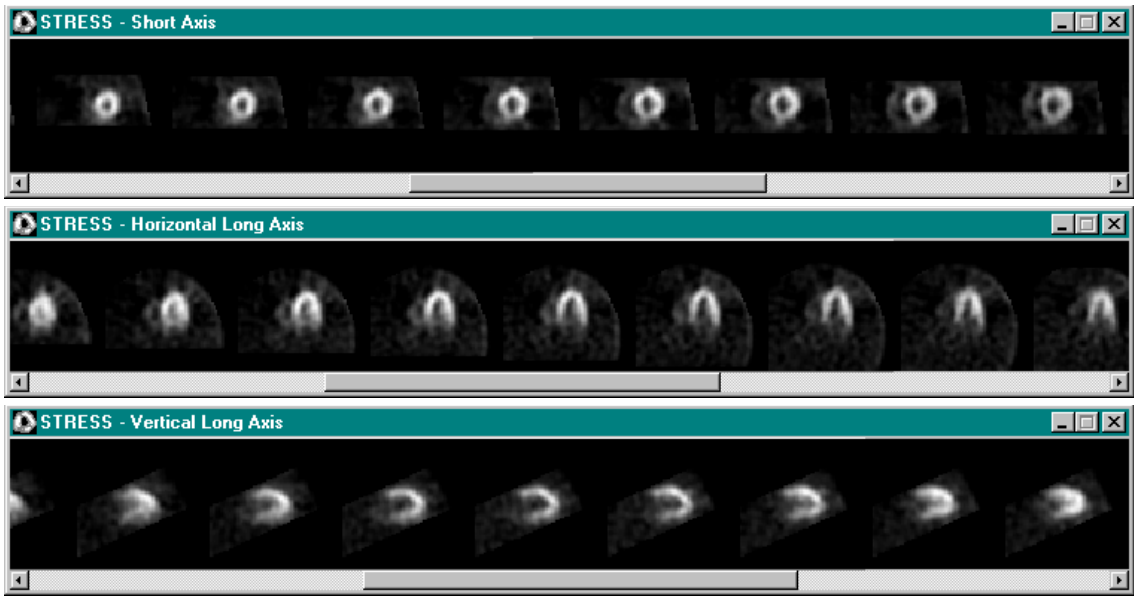


Figure A.4: Slice display windows of the SPECT database browser.

A.2.2 SPECT Database Browser

We have created another tool called SPECT Database Browser to enable visualization of most important patient information together with corresponding 3D SPECT images. The information is presented to the user in a number of windows that can be opened or closed as needed. The main window displays textual patient information, see Fig. A.2. The image display window shows SPECT images corresponding to case displayed in the main window. Both, rest and stress, images are shown together. The window displays all available axis views and 3D renderings based on short axis views, see Fig. A.3. For each of axis views, the user can open a window showing slices is row format, see Fig. A.4.

A.3 Model Building and Feature Extraction

We have created a C++ class library that implements 3D image registration, model creation, and feature extraction procedures described in Chapter 4. The library also includes all necessary database communication classes. A simple user interface, created in Borland

C++Builder, allows interaction with the library. The interface allows for automatic processing of a number of selected images by customizing appropriate SQL queries executed by the software.

A.4 Creation of Datasets for Classifier Learning

The datasets based on feature extraction were automatically created by software described in previous section. Datasets for the first experiment described in Chapter 8 were created by manual execution of SQL queries in MS Access; since all the data needed were already present in the database.

Appendix B

BNC: Bayesian Network Classifiers Toolbox

The software for learning Bayesian network classifier (BNC) has been implemented independently for the SPECT database, visualization, and feature extraction software described in Appendix A. The intention was to make it convenient to use BNC for classification in other domains.

We decided to implement BNC in Java programming language to ensure easy portability to various operating systems. We used Java 2 and tested the software under MS Windows NT, MS Windows 95/98, Linux, and Solaris operating systems. Although, not tested, the software should run on any other operating systems capable of running Java 2.

The main part of BNC is a class library implementing all of the new Bayesian network classifier learning algorithms described in Chapter 7, naïve Bayes (Duda and Hart, 1973), and TAN (Friedman et al., 1997).

BNC provides command line interface to the library. Command line interface allows for easy running of multiple learning algorithms on number of datasets. An example of Bourne Shell script that was used to test UCI datasets using five-fold cross validation, see Section 8.4, is presented in Fig. B.1. This single script executed the BNC CrossVal utility

```
#!/bin/sh

DATASETS="australian breast cleve crx diabetes german glass\
          glass2 heart iris pima soybean-large vehicle vote"

ALGOR="fan stan stand sfan sfand"
QM="HGC SB LC LOO CV1010"

for d in $DATASETS; do
  for a in $ALGOR; do
    for q in $QM; do
      java CrossVal -f cv5/$d -a $a -q $q -s 9 -l UCI;
    done;
  done;
done
```

Figure B.1: Bourne Shell script that was used to test all 25 new Bayesian network classifiers on 14 UCI datasets using five-fold cross validation. The script executes 350 cross validation tests. The results are automatically logged to a database table named UCI.

350 times.

B.1 BNC Utilities

Command line interface to BNC class library consists of three utilities: Classifier, CrossVal, and DatasetInfo. They are described in following sections.

B.1.1 Classifier and CrossVal Utilities

Classifier utility is used to learn and test a Bayesian network classifier. CrossVal utility performs a cross-validation test of a Bayesian network classifier, it assumes that cross validation datasets are already generated and attempts to read them using given file stem. The file name format is *filestem-repetition-fold*. For instance for file stem `vote` the file names could be `vote-0-0.*`, `vote-0-1.*`, etc. The GenCVFiles utility from MLC++ li-

brary (Kohavi et al., 1997) can be used for generation of cross validation files. It generates cross validation file names in the format described above. Both, Classifier and CrossVal utilities accept the same command line options:

-a *algor_name* Bayesian network classifier algorithm choices: naive (for naïve Bayes), TAN, FAN, STAN, STAND, SFAN, SFAND.

-c *class_name* Name of the class variable. The default value is class.

-d Print debugging information.

-f *filestem* Load test data in C4.5 format (`.names` + `.data` + `.test`). For Classifier the files will be: *filestem*`.names` – file with specification of attributes, *filestem*`.data` – file with training cases, and *filestem*`.test` – file with test cases. For CrossVal the file names will be *filestem*`-?-?.names`, *filestem*`-?-?.data`, and *filestem*`-?-?.test`.

-l *table-name* Log result to database table *table-name*. It is assumed that results are logged to a database with ODBC name jtest.

-n *filename* Save constructed network(s) to *filename*`.bif`. File is saved in BIF 0.15 format.

-q *quality-measure* Bayesian network quality measure choices: HGC – Heckerman-Geiger-Chickering, SB –Standard Bayesian, LC – Local criterion, LOO – leave-one-out cross validation, CV10 – ten-fold cross validation, CV1010 – ten-fold ten-times cross validation.

-s *number* Number of smoothing priors to test; has to be an integer greater or equal to zero.

-t Print execution time in milliseconds.

```
Classifier tester

File stem: vote
Algotithm: FAN
Quality measure: Leave-one-out cross validation

Error = 5.926% +- 2.04% (94.074%) [127/8/135]
```

Figure B.2: Sample output of Classifier utility.

For example, to test FAN classifier using LOO quality measure on dataset vote following command line can be used:

```
java Classifier -a FAN -q LOO -f vote
```

An example of Classifier output is shown in Fig. B.2. Number after `-+` is an estimate of the standard deviation according to binomial model. The number in parenthesis is accuracy (100%-error), the numbers in square brackets are: number of correct classifications, number of false classifications, and total number of cases in test set, respectively.

B.1.2 DatasetInfo Utility

The DatasetInfo utility is used to print information about a dataset. It takes a single attribute: the file name stem. It assumes that the file is saved in C4.5 format. DatasetInfo prints information about number of classes and number of attributes in the dataset (defined in file *filestem*`.names`). It also prints frequency of each of the classes in train and test datasets. A sample output of DatasetInfo utility is shown in Fig. B.3.

```
DatasetInfo

Filestem: ../db/vote
Number of classes    = 2
Number of attributes = 16

File '../db/vote.all'  has 435 cases.
  democrat : 267 [61.38%]
  republican : 168 [38.62%]

File '../db/vote.data'  has 300 cases.
  democrat : 184 [61.33%]
  republican : 116 [38.67%]

File '../db/vote.test'  has 135 cases.
  democrat : 83 [61.48%]
  republican : 52 [38.52%]
```

Figure B.3: Sample output of DatasetInfo utility.