Clinical Decision Support System using Real-Time Data Analysis for a Neonatal Intensive Care Unit

by

Jeffrey S. Gilchrist, M.A.Sc., B.C.S.

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Electrical and Computer Engineering

Carleton University
Ottawa, Ontario

©2012
Jeffrey S. Gilchrist
The undersigned hereby recommend to the Faculty of Graduate and Postdoctoral Affairs acceptance of the thesis

**Clinical Decision Support System**

**using Real-Time Data Analysis**

**for a Neonatal Intensive Care Unit**

Submitted by Jeffrey S. Gilchrist, in partial fulfilment of the requirements for the degree of Doctor of Philosophy

--------------------------------------------------

Dr. Howard Schwartz, Chair, Department of Systems and Computer Engineering

--------------------------------------------------

Dr. Evelyn Morin, External Examiner

--------------------------------------------------

Dr. Monique Frize, Thesis Co-supervisor

--------------------------------------------------

Dr. Colleen M. Ennett, Thesis Co-supervisor

Carleton University, April 2012
Abstract

There are two main contributions to knowledge presented in this thesis: (1) an efficient Clinical Data Repository (CDR) with a novel data storage format, and (2) new neonatal mortality risk estimation models that use data analysed in real-time from the CDR.

The CDR collects, stores, and retrieves clinical data in real-time. It uses a novel hybrid Entity-Attribute-Value (EAV) storage format that is faster, uses less complex queries, and allows the data to remain in its original data type while using the same amount of storage space as other popular storage formats. The CDR can collect and store data from patient monitors and laboratory results in real-time using only free open-source tools, and is compatible with industry standard protocols. The CDR design accommodates the fast-changing medical domain, and supports the addition of new attributes automatically. Private patient information is segregated automatically from raw research data before it is stored. This provides the ability to give researchers and physicians access to the real-time research data without further processing, and without violating patient privacy.

We developed novel real-time neonatal mortality risk estimation models using Decision Trees (DTs) and Artificial Neural Networks (ANNs) that met criteria for clinically useful results (>60% sensitivity, >90% specificity) for individual patients. Results showed that mortality models using summary data (data from admission until up to the first 48 hours after Neonatal Intensive Care Unit (NICU) admission) provided, on average, the highest sensitivity and specificity with the least number of false positives, exceeding the performance criteria set by our clinical partners mentioned above. The DT model produced the best results on average (sensitivity=75%, speci-
ficity=96%), while the ANN model produced lower but still clinically useful results (sensitivity=68%, specificity=97%). Three attributes were found to be most important to estimate the risk of mortality during the first 48 hours after NICU admission: lowest serum pH, lowest blood pressure, and lowest heart rate.

In the future, a prototype of the real-time Clinical Decision Support (CDS) system framework should be implemented with the ability to generate intelligent alerts and warnings from medical events as they occur, such as when the risk estimation for a patient changes significantly.
To my amazing children Sébastien and Xavier whose smiles always pushed me to work even harder. I hope one day that you will be able to achieve your dream as I have. To my beautiful wife Johanne who supported me through everything and made it possible for me to become the first doctor in my family. Thank you for going on this journey with me.
Acknowledgements

Thank you to my supervisor Dr. Monique Frize who inspired me to pursue a PhD after learning of her research work. I knew that I could make a difference if we worked together, and we did! Thank you to my co-supervisor Dr. Colleen M. Ennett who provided an invaluable industry perspective on our research work and who had many interesting ideas and improvements for the research we conducted together.

A heartfelt thank you to Dr. Erika Bariciak at the Children’s Hospital of Eastern Ontario (CHEO) who always made herself available to answer any questions that I had. Your help and support of my research project was instrumental in my success.

I would like to thank and acknowledge the extra work that the NICU team at CHEO provided to help collect data for the patients in the unit including nursing manager Patricia O’Flaherty for all of her support, the patient service clerks, the nurses, the biomedical engineering team, especially Kim Greenwood, and the IS dept.

Thank you to my defence committee for all of the useful feedback on my research work to help improve the final result: Dr. Evelyn Morin from Queen’s University, Dr. Tet Yeap from University of Ottawa, Dr. Rafik Goubran from Carleton University, and Dr. Avi Parush from Carleton University.

Thank you to my fellow graduate students who worked with me throughout my PhD: Daphne Townsend, Shauna Mullaly, Helga Ross, Sabine Weyand, Stuart Tozer, Irena Zamboni, Christophe Herry, Megan Holtzman, and Geoff Green.

I would like to thank the office staff in the Systems and Computer Engineering department for all of the help and support they provided to me during my time as a graduate student, especially Jennifer Poll, Coleen Kornelsen, Darlene Hebert, Emily Byron, Lindsay Kristjansson, and Darlene Rupert.
Statement of Originality

This thesis describes the results of the candidate's research conducted at Carleton University and the Children's Hospital of Eastern Ontario (CHEO) during the course of the Ph.D. program. Results presented herein have been published in the journal IEEE Transactions on Instrumentation and Measurement (IEEE - Institute of Electrical and Electronics Engineers), and have been presented at various international conferences. The details of the candidate's contributions to these publications, and the location of the research results in the thesis are summarised below.


The results of this journal paper constitute the first portion of Chapters 4 and 5, and is an expanded version of Gilchrist et al. 2010. The candidate designed and tested a new method to collect, store, and retrieve large amounts of medical data in real-time. The candidate collected the data from two different systems at CHEO during 2009 and 2010, and performed experiments to compare the performance of three storage formats. The resulting new hybrid EAV data storage format combines the benefits of the simple EAV format and the multi-data type EAV format. It uses less complex queries, and provides faster access to the data, while using the same amount of storage space as previous EAV formats. This conclusion was further confirmed using the new 2010 data.

The results of this conference paper are included in the latter portion of Chapters 4 and 5. The candidate developed mortality risk estimation models using DTs and ANNs that are capable of estimating patient-specific outcomes in real-time. Previously, researchers had only used archived data collected within the first 12 hours of birth to assess severity of illness. This new approach used real-time data from patients beyond the first 12 hours of life to produce risk estimations as new information became available. This provides the unique ability to detect clinical events as they happen, and deliver immediate feedback to physicians. The candidate wrote all implementation code, prepared the CHEO data, created the mortality models, and carried out the data analysis.


Portions of the results of this conference paper are included in Chapters 4 and 5. The candidate designed and implemented the Clinical Data Repository (CDR) that facilitates medical research by pulling information together from multiple systems into a single location using free open-source tools and industry standard protocols.
This new approach for CDR accommodates the fast-changing medical domain by supporting the addition of new attributes automatically without the need for any database format changes. The Physician-Parent Decision Support (PPADS) tool relies on the CDR to operate. PPADS uses the data collected and stored in the CDR to provide results from the real-time risk estimation models to physicians and parents. The candidate collaborated with the co-authors on the design and implementation of the PPADS tool.


The results of this conference paper are included in Chapters 4 and 5. The candidate designed, and tested a new method to collect, store, and retrieve large amounts of medical data in real-time. The candidate collected the data from one system at CHEO during 2009, and performed experiments to compare the performance of three storage formats. The resulting new hybrid EAV data storage format combines the benefits of the simple EAV format and the multi-data type EAV format. It uses less complex queries, and provides faster access to the data, while using the same amount of storage space as previous EAV formats.

The results of this conference paper are included in Chapter 3 and Appendix B. The candidate carried out the analysis of the proprietary systems at CHEO to devise a new way to collect and store data using industry standards and open source software. The candidate developed the new CDR and performed the integration work on the CDR to support the existing patient monitoring equipment through a newly-installed clinical central station, and laboratory system. The candidate collaborated with the co-authors to capture the lessons learned from this integration work.


Portions of the results of this conference paper are included in Chapter 2. The candidate collaborated with the co-authors on the analysis and design of the imputation framework. The framework classifies the inconsistencies in a medical database to automatically process the data for research purposes. The goal is to improve the quality of medical databases by identifying the reason that data are missing, and automatically processing outliers and typographical errors to retain more information from a database. This framework provides an alternative to deleting cases with outliers and typographical errors, while also imputing relevant values into cases with missing data based on the reason that the data is missing, the source of the data, and the data type.
Contents

1 Introduction
   1.1 Motivation ................................................................. 1
   1.2 Problem Identification .................................................. 2
   1.3 Thesis Objectives and Problem Statement .............................. 3
   1.4 Significance of Research ............................................... 5
   1.5 Summary of Results .................................................... 7
   1.6 Thesis Outline .......................................................... 8

2 Technical Background ..................................................... 10
   2.1 Clinical Data Repositories (CDRs) .................................... 10
      2.1.1 Electronic Medical Record (EMR) vs CDR ....................... 11
      2.1.2 CDR Design Methodologies ....................................... 11
         2.1.2.1 Traditional Relational Database ............................. 12
         2.1.2.2 Data Warehousing ............................................. 14
         2.1.2.3 Entity-Attribute-Value (EAV) ............................... 15
      2.1.3 Building a CDR using EAV ....................................... 15
         2.1.3.1 Overview of Building a CDR Using EAV Format ............ 18
         2.1.3.2 Multiple Data Types ......................................... 21
2.1.3.3 Enhanced EAV Model ........................................ 23
2.1.3.4 Performance of EAV Format .............................. 25
2.1.3.5 Opportunities for Improvement with EAV Format .... 28

2.2 Pattern Classification ........................................... 28
  2.2.1 Artificial Neural Networks .................................. 29
  2.2.2 Decision Trees ............................................... 33
    2.2.2.1 Creating a Decision Tree ............................... 34
    2.2.2.2 Handling Missing Data .................................. 37
    2.2.2.3 Improvements with C5.0 ................................. 38
    2.2.2.4 Advantages and Limitations of DTs .................. 39

2.3 Neonatal Clinical Decision Support ......................... 40
  2.3.1 Heart Rate Observation System (HeRO) ................... 40
  2.3.2 Artemis Analytic System ................................. 42
  2.3.3 Realtromins .............................................. 44
  2.3.4 NEONATE .................................................. 46
  2.3.5 Cochrane Review of CDS Systems for Neonatal Care .... 47
  2.3.6 Medical Information Technologies Research Group .... 48
  2.3.7 Non-Neonatal Clinical Decision Support Systems ....... 51

3 Clinical Context and Data Collection at Children’s Hospital of Eastern Ontario (CHEO) .... 53
  3.1 NICU Environment at CHEO ................................... 53
    3.1.1 Canadian Neonatal Network (CNN) Database ............. 56
    3.1.2 CHEO NICU Database ..................................... 57
      3.1.2.1 CDR Performance Data Sets .......................... 58
3.1.2.2 Risk Estimation Data Sets ............................................. 59

3.2 Data Collection Systems at CHEO .......................................... 60

4 Methodology ............................................................................. 64

4.1 Introduction .......................................................................... 64

4.2 Hybrid-EAV Storage Format .................................................. 64

4.2.1 Storage Format Design ..................................................... 65

4.2.2 Measures of Performance .................................................. 66

4.2.2.1 Storage Size ............................................................. 66

4.2.2.2 Query Speed ............................................................. 67

4.2.2.3 Query Complexity ...................................................... 70

4.3 Real-Time CDR .................................................................... 71

4.3.1 Data Collection ................................................................. 72

4.3.2 Data Storage ................................................................. 72

4.3.3 Patient Privacy .............................................................. 73

4.3.4 Data Retrieval ............................................................... 74

4.3.4.1 Patient Complete Stays ............................................... 75

4.3.4.2 Time Spans ............................................................... 75

4.3.4.3 Data Frequency ......................................................... 75

4.3.4.4 Meta-Data Descriptions ............................................. 76

4.4 Development of New Neonatal Mortality Risk Models ............... 76

4.4.1 Data Sets ........................................................................ 77

4.4.2 Data Preparation ........................................................... 78

4.4.3 Risk Estimation Models .................................................... 80

4.4.3.1 Baseline Risk Estimation Model with CNN Data ............ 83
4.4.3.2 Summary Data Risk Estimation Models with CHEO Data 84
4.4.3.3 Real-Time Risk Estimation Models with CHEO Data 85
4.4.3.4 CHEO 2010 vs 2011 Data Sets 86
4.4.4 Measures of Performance 87
4.4.5 Imbalanced Data Sets 90
4.4.6 Risk Categories 91
4.4.7 Usage of Neonatal Risk Models with Real-Time Data 92
4.4.8 Updating Neonatal Risk Models 94

5 Results 96

5.1 Introduction 96
5.2 Hybrid EAV Storage Format Results 97
5.2.1 Storage Format Design 97
5.2.2 Storage Size Results 99
5.2.3 Query Speed Results 100
5.2.3.1 Querying all data for a single patient 100
5.2.3.2 Calculating pulse pressure 100
5.2.3.3 Finding patients with respiratory decompensation 101
5.2.4 Query Complexity Results 103
5.2.4.1 Querying all data for a single patient 103
5.2.4.2 Extracting patients with attribute values within a specified range (OR condition) 104
5.2.4.3 Extracting patients with attribute values within a specified range (AND condition) 105
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.2.5</td>
<td>EAV Performance Summary</td>
<td>107</td>
</tr>
<tr>
<td>5.3</td>
<td>Real-Time CDR Design Results</td>
<td>108</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Data Collection</td>
<td>108</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Data Storage and Patient Privacy</td>
<td>110</td>
</tr>
<tr>
<td>5.3.2.1</td>
<td>Patient Identity Table</td>
<td>114</td>
</tr>
<tr>
<td>5.3.2.2</td>
<td>EAV Attributes Table</td>
<td>115</td>
</tr>
<tr>
<td>5.3.2.3</td>
<td>EAV Data Table</td>
<td>116</td>
</tr>
<tr>
<td>5.3.3</td>
<td>Data Retrieval</td>
<td>118</td>
</tr>
<tr>
<td>5.3.3.1</td>
<td>Patient Complete Stays</td>
<td>118</td>
</tr>
<tr>
<td>5.3.3.2</td>
<td>Time Spans</td>
<td>119</td>
</tr>
<tr>
<td>5.3.3.3</td>
<td>Data Frequency</td>
<td>119</td>
</tr>
<tr>
<td>5.3.3.4</td>
<td>Meta-Data Descriptions</td>
<td>120</td>
</tr>
<tr>
<td>5.4</td>
<td>Neonatal Mortality Risk Model Results</td>
<td>120</td>
</tr>
<tr>
<td>5.4.1</td>
<td>Baseline Risk Estimation Model with CNN Data</td>
<td>121</td>
</tr>
<tr>
<td>5.4.2</td>
<td>Summary Data Risk Estimation Models with CHEO Data</td>
<td>121</td>
</tr>
<tr>
<td>5.4.3</td>
<td>Real-Time Risk Estimation Models with CHEO Data</td>
<td>124</td>
</tr>
<tr>
<td>5.4.4</td>
<td>CHEO 2010 vs 2011 Data Set Results</td>
<td>127</td>
</tr>
<tr>
<td>5.4.5</td>
<td>DT Model Comparison</td>
<td>128</td>
</tr>
<tr>
<td>5.4.6</td>
<td>ANN Model Comparison</td>
<td>131</td>
</tr>
<tr>
<td>5.4.7</td>
<td>Imbalanced Data Set Results</td>
<td>132</td>
</tr>
<tr>
<td>5.4.8</td>
<td>Risk Category Results</td>
<td>134</td>
</tr>
<tr>
<td>5.4.9</td>
<td>Real-Time Data Usage Results</td>
<td>136</td>
</tr>
<tr>
<td>5.4.10</td>
<td>Updating Neonatal Risk Models Results</td>
<td>137</td>
</tr>
<tr>
<td>5.4.11</td>
<td>Intended Use of Models</td>
<td>140</td>
</tr>
</tbody>
</table>
6 Concluding Remarks

6.1 Conclusions ................................................. 142
6.2 Contributions to Knowledge ............................. 147
6.3 Future Research ............................................. 149

A Publications ................................................. 166

B Data Collection at CHEO ...................................... 168

B.1 CNN and CHEO Database Statistics ..................... 168
B.2 CHEO Risk Estimation Data Set Statistics .............. 168
B.3 Evolution of Data Collection Systems at CHEO (2001 - Present) .. 171
  B.3.1 Philips DocVue Data Collection System (2001 - Sep 2008) ....... 171
    B.3.1.1 System Overview .................................. 171
    B.3.1.2 Proprietary Technology of Philips Healthcare .......... 173
    B.3.1.3 Unsupported Components by Microsoft ................. 174
    B.3.1.4 Storage Capacity .................................. 174
  B.3.2 Philips Information Centre (PIC) Data Collection System (Oct
      2008 - Sep 2009) ........................................ 175
    B.3.2.1 System Overview .................................. 175
    B.3.2.2 Industry Standard Technology ....................... 176
    B.3.2.3 Integration Issues ................................ 177
    B.3.2.4 Data Collection Statistics ......................... 177
  B.3.3 Draeger Medical Data Collection System (Sep 2009 - Present) 178
    B.3.3.1 Industry Standard Technology ....................... 179
    B.3.3.2 Updated Data Collection Statistics ................. 180
B.3.4 eClinDoc (June 2010 - Present) ............... 181
B.4 Lessons Learned from CDR Integration at CHEO ............... 182
  B.4.1 Integration Issues ............................................. 182
  B.4.2 User and Staff Buy-In to the Data Collection System .... 183
  B.4.3 User and Staff Buy-In to the Research ................. 185
  B.4.4 IT and Vendor Support ........................................ 185
  B.4.5 Industry Standards .......................................... 186
  B.4.6 Backup Strategies ............................................ 186

C CDR Database Schema .............................................. 188
  C.1 CDR Export Temporary Table Schema ....................... 188

D Useful Data Processing Scripts .................................. 190
  D.1 BASH Script to Create DT 5-by-2 Cross Validation Data Sets .... 190
  D.2 BASH Script to Call C5.0 Tool to Process Data Files ............ 192
  D.3 BASH Script to Create 5-by-2 Cross Validation Data Sets using Matching Patients .......................................... 193
  D.4 SQL Query to Extract Data from CDR for 48-Hour Summary Format 194
**List of Tables**

2.1 UVA 208 column relational CDR table sample (Wisniewski et al., 2003). 13
2.2 Summary Checklist of CDR Designs ............................. 17
2.3 Conventional Database ........................................... 19
2.4 Typical EAV Model ................................................. 19
2.5 Adding attributes to an EAV CDR .............................. 20
2.6 Simple EAV Format ................................................. 21
2.7 Multi-Data Type EAV Format ................................. 22
2.8 EAV/CR Database Design ......................................... 25

3.1 CHEO NICU Admissions Statistics ............................... 54
3.2 List of 18 attributes used in CNN and CHEO Mortality models . 57

4.1 List of 18 attributes used in Mortality models ............... 78
4.2 Fifteen Neonatal Mortality Risk Estimation Models ........... 83

5.1 Hybrid EAV Format ............................................... 97
5.2 Summary of Data Structure Characteristics ................... 98
5.3 SQL queries for querying all data for a single patient ....... 103
5.4 SQL queries for extracting patients with attribute values within a specified range (OR condition) .................................................. 104
5.5 SQL queries for extracting patients with attribute values within a specified range (AND condition) .................................................. 106
5.6 Summary of EAV Format Performance Results .............................................. 107
5.7 Patient Identity Table Sample ................................................................. 115
5.8 EAV Attributes Table Sample ................................................................. 116
5.9 EAV Data Table Sample ................................................................. 118
5.10 Attribute Usage in Best Decision Tree Models Using CHEO Data .......... 123
5.11 Summary of Results for DT Neonatal Mortality Risk Estimation Models 130
5.12 Imbalanced Data Results for Mortality .............................................. 133
5.13 Imbalanced Data Results for IVH ..................................................... 134
5.14 Risk Category Threshold Results ........................................................ 136
5.15 Timing Results of Calculating Mortality Risk Estimations from 3 Patients 137
B.1 CNN and CHEO Database Statistics .............................................. 169
B.2 CHEO Risk Estimation Data Set Statistics ..................................... 170
B.3 eClinDoc NICU Variables ................................................................. 182
List of Figures

2.1 Data Warehouse Designs (Sahama and Croll, 2007) .......................... 16
2.2 EAV/CR schema with classes and relations (Anhoj, 2003) ............... 24
2.3 Artificial Neuron - One node in an ANN (Kriesel, 2007) ................ 30
2.4 Multilayer Perceptron Feedforward Network with 1 hidden layer. ... 31
2.5 Decision tree to classify vehicle type. ....................................... 35

3.1 CHEO NICU Data Collection System .......................................... 62

4.1 5-by-2 Cross Validation Method .................................................. 81
4.2 Sample diagram of Summary and Time Segment data sets ............... 85
4.3 Confusion Matrix ................................................................. 88

5.1 MIRTH HL7 Filtering ............................................................. 109
5.2 CDR System and Network Diagram ............................................. 111
5.3 Extended Entity Relation (EER) Diagram for CDR .......................... 112
5.4 CHEO 48-Hour Summary Data Mortality Risk Estimation Model De-
    cision Tree .............................................................................. 128

B.1 Philips DocVue Data Collection System ...................................... 172
B.2 DocView Client User Interface .................................................. 173
B.3 Philips Information Centre Data Collection System. . . . . . . . . . . . . . 176
B.4 NICU Patient Data Collection Stats . . . . . . . . . . . . . . . . . . . . . . . . 184
# List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADT</td>
<td>Admissions, Discharges, and Transfers</td>
</tr>
<tr>
<td>ANN</td>
<td>Artificial Neural Network</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>BASH</td>
<td>Bourne-Again Shell</td>
</tr>
<tr>
<td>BPD</td>
<td>Bronchopulmonary Dysplasia</td>
</tr>
<tr>
<td>CART</td>
<td>Classification and Regression Trees</td>
</tr>
<tr>
<td>CBR</td>
<td>Case-Based Reasoning</td>
</tr>
<tr>
<td>CCS</td>
<td>Central Collection Station</td>
</tr>
<tr>
<td>CDR</td>
<td>Clinical Data Repository</td>
</tr>
<tr>
<td>CDS</td>
<td>Clinical Decision Support</td>
</tr>
<tr>
<td>CHEO</td>
<td>Children’s Hospital of Eastern Ontario</td>
</tr>
<tr>
<td>CNN</td>
<td>Canadian Neonatal Network</td>
</tr>
<tr>
<td>CO₂</td>
<td>Carbon Dioxide</td>
</tr>
<tr>
<td>CRIB</td>
<td>Clinical Risk Index for Babies</td>
</tr>
<tr>
<td>DBMS</td>
<td>Database Management System</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Full Form</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
</tr>
<tr>
<td>DT</td>
<td>Decision Tree</td>
</tr>
<tr>
<td>EAV</td>
<td>Entity-Attribute-Value</td>
</tr>
<tr>
<td>ECG</td>
<td>Electrocardiogram</td>
</tr>
<tr>
<td>EER</td>
<td>Extended Entity Relation</td>
</tr>
<tr>
<td>EMR</td>
<td>Electronic Medical Record</td>
</tr>
<tr>
<td>FN</td>
<td>False Negative</td>
</tr>
<tr>
<td>FP</td>
<td>False Positive</td>
</tr>
<tr>
<td>HCO$_3$</td>
<td>Bicarbonate</td>
</tr>
<tr>
<td>HeRO</td>
<td>Heart Rate Observation System</td>
</tr>
<tr>
<td>HL7</td>
<td>Health Level 7</td>
</tr>
<tr>
<td>ID3</td>
<td>Iterative Dichotomiser 3</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IQR</td>
<td>Inter Quartile Range</td>
</tr>
<tr>
<td>IS</td>
<td>Information Systems</td>
</tr>
<tr>
<td>IVH</td>
<td>Intraventricular Hemorrhage</td>
</tr>
<tr>
<td>k-NN</td>
<td>k-Nearest Neighbour</td>
</tr>
<tr>
<td>LOINC</td>
<td>Logical Observation Identifiers Names and Codes</td>
</tr>
<tr>
<td>MCC</td>
<td>Matthews Correlation Coefficient</td>
</tr>
</tbody>
</table>
MIRG  Medical Information technologies Research Group
MLP  Multilayer Perceptron
NEC  Necrotising Enterocolitis
NICU  Neonatal Intensive Care Unit
NPV  Negative Predictive Value
NTISS  Neonatal Therapeutic Intervention Scoring System
PCO₂  Partial pressure of carbon dioxide in the blood
PDA  Patent Ductus Arteriosus
PIC  Philips Intellivue Information Centre
PMML  Predictive Modelling Markup Language
PO₂  Partial pressure of oxygen in the blood
PPV  Positive Predictive Value
PSC  Patient Service Clerk
RAID  Redundant Array of Independent Disks
RAID-1  Redundant Array of Independent Disks - Level 1
RDBMS  Relational Database Management System
REB  Research Ethics Board
SpO₂  Oxygen saturation as measured via a pulse oximeter
| **SDN** | Serial Distribution Network |
| **SLP** | Single Layer Perceptron |
| **SNAP** | Score for Neonatal Acute Physiology |
| **SPADE** | Stream Processing Application Declarative Engine |
| **SQL** | Structured Query Language |
| **TCP** | Transmission Control Protocol |
| **TN** | True Negative |
| **TP** | True Positive |
| **UVA** | University of Virginia |
Chapter 1

Introduction

1.1 Motivation

With the increased use of technology in hospitals, large amounts of data are being generated from medical devices and systems. These data are typically stored on multiple systems, located in different departments of the hospital, and frequently use different and incompatible technology. Current proprietary systems are designed for clinical use, and do not make it easy for data to be extracted for research purposes. With this information stored on separate systems, clinicians cannot easily integrate these data into clinically useful knowledge. A new approach is required to facilitate clinical decision making. Data need to be captured and stored in one location to provide researchers and clinicians with access to all the data they require. A Clinical Data Repository (CDR) storage structure would efficiently store all of these clinical data, and allow people to conduct data mining, which they were not able to do before. The CDR would enable research that could be integrated into a Clinical Decision Support (CDS) system. With all of the required data being captured by one
CHAPTER 1. INTRODUCTION

CDR and fed into a CDS system, clinicians could be provided with clinically useful knowledge delivered to them in real-time.

The Neonatal Intensive Care Unit (NICU) takes care of the sickest babies in hospitals, such as the Children's Hospital of Eastern Ontario (CHEO) in Ottawa. Preterm birth is the leading cause of infant mortality in the United States, accounting for approximately one third of infant deaths (Callaghan et al., 2006). Babies who are born preterm (less than 37 weeks gestational age) can develop significant morbidities such as Intraventricular Hemorrhage (IVH) of the brain, Bronchopulmonary Dysplasia (BPD) of the lungs, Necrotising Enterocolitis (NEC) of the gut, and are more likely to require long-term ventilatory support, and a long-term stay in an NICU. More accurate methods of estimating the risk of these and other medical conditions to help with patient care and resource decisions would be of benefit to physicians, to the health care team, and ultimately to the babies and their families. This type of knowledge could be provided by a combined real-time CDR and CDS system.

1.2 Problem Identification

A number of issues exist specific to the NICU clinical environment such as critically ill infants with multi-system organ failure due to underdeveloped organs and few published clinical guidelines to inform therapeutic choices. Many of the clinical guidelines used are centre-specific, based on the experience of the medical professionals who work there. Most of the published risk estimation models are designed to categorise groups of infants and are based on scores from various physiological and therapeutic variables, such as the Score for Neonatal Acute Physiology (SNAP) which uses 37 measurements based on clinical experience (Richardson et al., 1993).
Developing risk estimation models specifically for NICUs in a particular hospital or region that can predict outcomes for individual patients may be more useful than the current scoring systems for making clinical decisions (Frize et al., 2006).

The babies in the NICU are connected to many monitoring devices such as ventilators, pulse oximeters, blood pressure monitors and heart rate monitors, which have the capability to transmit the patient's physiological data to a repository for processing. A CDR is needed that can store patient data that is easy to export and use for data mining and analysis. This CDR can be used as a real-time database that combines various clinical data such as patient physiological data, laboratory results, and admissions data to provide a single location for researchers to access the data of individual patients. The combination of a real-time CDR with a CDS system that is capable of estimating the risk of medical events can help facilitate treatment planning and potentially improve patient care.

1.3 Thesis Objectives and Problem Statement

The two main objectives of this thesis work were (1) to design a method to collect, store, and retrieve clinical data in a real-time environment, and (2) to devise neonatal mortality risk estimation models that use the collected data in real-time to assist clinical decision making.

Obtaining real clinical data for researchers to develop accurate models of significant clinical events for use in CDS systems is invaluable, since this type of data cannot be accurately generated in simulations. Medical data need to be collected, stored, and retrievable for later analysis in an efficient manner. Current clinical cen-

\footnote{Real-time for the purpose of this thesis was defined as completing a task within one minute.}
CHAPTER 1. INTRODUCTION

tral monitoring stations are designed to store physiological data for short periods of time for clinical use to see trends, but are not designed for research purposes. Clinical information is typically stored in multiple locations on different systems, making access to all required clinical data challenging. Fast access to clinical data is necessary to perform real-time risk estimations of medical events. Patient privacy issues must also be taken into consideration when conducting research. The current proprietary systems are meant to work inside the clinical environment, and so do not usually have the ability to de-identify patient information. Maintaining patient privacy is critical for researchers to be able to access medical data.

SNAP (Richardson et al., 1993, 2001) and similar risk estimation models such as the Clinical Risk Index for Babies (CRIB) (Cockburn et al., 1993) and the Neonatal Therapeutic Intervention Scoring System (NTISS) (Gray et al., 1992) provide scoring results to rank patients based on severity of illness. These scoring systems are not designed for specific units or regions, and cannot easily be updated to reflect new practices. They are not accurate enough for clinical use to estimate outcomes for individual patients, and have limited power of estimation for more than one outcome (Frize et al., 2006). Furthermore, these scoring systems only use aggregated data from the first 12 hours of life, when a plethora of data is continuously being collected from patients in the NICU. Physicians and the medical team need more up-to-date and patient-specific information to assist with treatment decisions. Technology in hospitals has advanced to the point where physicians are overloaded with information. Many systems generate high volumes of data for each patient, but most of this information is viewed in isolation and is not being used effectively. Physicians need a way to integrate and interpret all of the available information to transform it into
CHAPTER 1. INTRODUCTION

usable knowledge.

1.4 Significance of Research

To our knowledge, this thesis is the first effort to develop a real-time CDR for an NICU environment, capable of collecting and storing years worth of real-time physiological, admissions, and laboratory data, using only open-source components and industry standard protocols. While a research prototype, Artemis Analytic System (Blount et al., 2010), and a commercial company, Realtromins (Realtromins Inc., 2012), have been attempting to develop real-time neonatal medical systems, they use proprietary hardware and technology, and have not publicly disclosed any performance results. The Heart Rate Observation System (HeRO) project uses proprietary technology, requires a large computing cluster to store and analyse data, and does not update the results for clinicians in real-time (Moorman et al., 2011). While the project has published results on a large clinical trial for mortality reduction (Moorman et al., 2011b), their primary outcome measure was reported as being not statistically significant compared to the control group.

Existing clinical information systems are usually designed for clinical purposes, such as reviewing short-term trends, and contain only a few days worth of data. They are meant to support the clinical staff to treat the patients currently in the NICU, rather than to facilitate retrospective research and predictive alerting of deleterious clinical conditions. A new system is required to store enough data to analyse and create models, and to be able to access the data in real-time to detect medical events as they occur and deliver intelligent alerts. With only clinical trends available, it is not possible to achieve these important research goals.
CHAPTER 1. INTRODUCTION

The CDR provides researchers and physicians access to the data they need while automatically maintaining patient privacy. Data that are accessed or exported are already de-identified, unlike most systems that require data sanitising post-extraction. This automated nature can minimise the chance of private patient information being exposed, and privacy laws being breached. The flexible and open design of the CDR makes it useful for any researcher in the NICU to store new data for a different project, and to access new or current data with their own software.

The real-time data stored in the CDR were used to build new clinically useful models that estimate the risk of medical outcomes more accurately than previous models, which relied on static historical data from the first 12 hours of life. The integration of a real-time CDR with models that generate patient-specific risk estimations on medical outcomes provides physicians with the potential to improve patient outcomes. This new approach will use up-to-date patient data to better evaluate the current state of the patient being monitored. By analysing the data in real-time, the risk estimation models are expected to provide the unique ability to detect clinical events as they happen, and deliver immediate feedback to physicians. These models can be integrated into a CDS system via an intelligent alert and warning system. The resulting integration should greatly improve the performance of a CDS system, providing more relevant risk estimations of medical events, and generating alerts in context for patients in the NICU.

This research also builds an important foundation for future research projects that require access to real-time medical data. The challenges for using real-time data such as accessing and manipulating data, exporting data, and creating risk estimation models from real-time data were explored and discussed. This foundation
led to the development of an early prototype of a CDS system that provides patient-specific decision support for both physicians and parents to improve parent/clinician communication to be developed using the CDR and real-time risk estimation models (Frize et al., 2011). The parents must make medical decisions for their child, and can benefit from the extra support and information made available to them in a decision support tool designed to provide real-time health information about their child.

1.5 Summary of Results

To achieve Objective 1, a novel hybrid Entity-Attribute-Value (EAV) data storage format that combined the advantages of two popular medical storage formats was created and tested. It did not require any more storage space, had less complex queries, and allowed faster access to large amounts of data for risk estimation models to operate in real-time. The hybrid data storage format was capable of remaining efficient even after storing several years' worth of data with tens of millions of data points. This storage format was integrated into a novel CDR that pulls information together from multiple systems into a single location using free open-source tools and industry standard protocols. The design accommodates the fast changing medical domain, and supports the addition of new attributes automatically without the need for any database format changes. The data are organised for efficient use in both research and decision support. The CDR is capable of automatically segregating private patient information from raw data in real-time before the data are stored. This provides the ability to give researchers and physicians access to the real-time raw research data without any further processing and without violating patient privacy.

For Objective 2, to help overcome the limitations of current scoring based systems,
the large amounts of patient data collected in the CDR were used to create neonatal
mortality risk estimation models using Decision Trees (DTs) and Artificial Neural
Networks (ANNs) that can estimate outcomes for a single patient. These novel models
used real-time data from patients beyond the first 12 hours of life to provide risk
estimations as new information became available. The mortality models used the most
recent patient information quickly enough from the CDR to make risk estimations
in real-time. To help physicians with the overload of information they face, the
risk estimation models integrated the relevant available information to provide an
in-context status report of a patient. Multiple parameters were used from different
sources (physiological data, laboratory results) to provide more information than the
simple threshold-based alerts such as “heart rate low” that now occur in the NICU.
Supporting multiple parameters more effectively used all the information that was
available to provide physicians with an overall mortality risk assessment for individual
patients. The mortality models can be integrated into a CDS system to provide
physicians with access to patient-specific risk estimations of clinical outcomes in real-
time. Real-time support facilitates the ability to update the risk estimation models in
a mostly automated fashion when new data are available. This has the added benefit
of reducing the chance of error since the data do not need to be manually extracted
and manipulated to produce a new model.

1.6 Thesis Outline

The thesis is organised as follows. Chapter 2 presents the technical background
information on CDRs, pattern classification using ANNs and DTs, and neonatal clin-
ical decision support. Chapter 3 provides clinical background information about the
CHAPTER 1. INTRODUCTION

NICU, the data collection systems used at CHEO, and the neonatal databases used for this thesis work. Chapter 4 presents the methodology for developing the CDR and mortality risk estimation models, and Chapter 5 presents the results of the design and performance of the real-time CDR and mortality risk estimation models. Finally, Chapter 6 provides concluding remarks, and highlights potential future work.
Chapter 2

Technical Background

This chapter presents a literature review on Clinical Data Repositories, Pattern Classification, and Neonatal Clinical Decision Support.

2.1 Clinical Data Repositories (CDRs)

CDRs are becoming more common as health care systems and medical devices provide data in electronic form (Mullins et al., 2006). They integrate "islands" of information from various departments of health care facilities, as it is time consuming to access all of these data separately for analysis (Sahama and Croll, 2007). Integration of these data into one repository can greatly enhance the ability to provide decision support and a knowledge-base for more sophisticated research (Schubart and Einbinder, 2000). The complexity and fast evolution of clinical information makes development and maintenance of clinical databases challenging (Anhoj, 2003). Clinical databases should ideally be flexible to handle new data types and attributes without needing to change the physical database schema.
2.1.1 Electronic Medical Record (EMR) vs CDR

An EMR contains information about a single patient in digital form to provide a "clinical view" of the patient. They are similar to paper medical records, since they contain all of the information related to treating a patient (Boulus, 2005). EMRs can include both non-clinical data, such as demographic information, and clinical data, such as medications, test results, and diagnoses. A CDR on the other hand provides a multi-patient "research view". It contains data for many patients from different sources, such as laboratory test results, pharmacy information, medical imaging and pathology results, admissions and discharge information, diagnoses, and discharge summaries. Having data for multiple patients in one location makes it easier to perform research than having to access every patient’s EMR individually, and then locate data not included in the EMR.

2.1.2 CDR Design Methodologies

A number of approaches for clinical database design have been used such as network medical databases, hierarchical medical databases, entity-relationship medical databases, and data warehousing (Lee and Lee, 1991). The network data model is based on tables and graphs. The nodes of a graph represent entity types while the connections between nodes represent the relationship types. With a hierarchical database, the relationship types must form an ordered tree. Entity-relationship models are based on tables and graphs. They can be depicted using an entity-relationship diagram for expressing system analysis and design requirements, and are commonly used with traditional relational databases. A data warehouse contains a wide variety of data from multiple data sources, that are used for reporting and analysis. The de-
CHAPTER 2. TECHNICAL BACKGROUND

The design of CDRs varies widely from one institution to another depending on the intended use and needs. Care must be taken when building a CDR to ensure it will meet the requirements of the users.

2.1.2.1 Traditional Relational Database

A traditional relational database is a group of tables, made of columns and rows called tuples. The tables have defined relationships with each other based on constraints that are placed on the data (Date, 2003). Relational databases are queried using Structured Query Language (SQL) which produces result sets. If multiple tables are involved with the query, the data are combined into one, called a join, using the Cartesian product, and then the results are filtered to remove anything not required for the answer.

The University of Virginia (UVA) designed a CDR that contains 667,000 patient records in a relational MySQL database that is 30GB in size (Muffins et al., 2006). Table 2.1 shows how each patient case is stored as a row in the database with the full version containing 208 core columns. The CDR was used to validate a new data mining approach to identify potentially novel clinical disease associations. This approach allows quick access to patient information, since everything for a specific patient is contained within one row of a single table. However, if patients do not need all 208 attributes for their medical record, a lot of space is wasted with empty fields. Also, if any new attributes need to be added for patients over time, the database schema will need to be modified, which can be time consuming and costly.

A CDR for the Chicago Antimicrobial Resistance Project (CARP) was developed to help hospital infection control. It also uses a relational database (Microsoft SQL
Table 2.1: UVA 208 column relational CDR table sample (Wisniewski et al., 2003).

<table>
<thead>
<tr>
<th>PtID</th>
<th>YOB</th>
<th>G</th>
<th>R</th>
<th>S</th>
<th>YOD</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>D5</th>
<th>D6</th>
<th>D7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1994</td>
<td>M</td>
<td>W</td>
<td>A</td>
<td>2000</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>2</td>
<td>1923</td>
<td>F</td>
<td>W</td>
<td>D</td>
<td>1995</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>D8</td>
<td>D9</td>
<td>P1</td>
<td>P2</td>
<td>P3</td>
<td>P4</td>
<td>P5</td>
<td>P6</td>
<td>P7</td>
<td>P8</td>
<td>P9</td>
<td>P10</td>
<td>P11</td>
</tr>
<tr>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>M1</td>
<td>M2</td>
<td>M3</td>
<td>M4</td>
<td>M5</td>
<td>M6</td>
<td>M7</td>
<td>M8</td>
<td>M9</td>
<td>M10</td>
<td>M11</td>
<td>M12</td>
<td>M13</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>9</td>
<td>3</td>
<td>1</td>
<td>10</td>
<td>1</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>26</td>
<td>3</td>
<td>1</td>
<td>10</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

7.0 Server), with data imported from 13 different systems totalling 50GB (Wisniewski et al., 2003). The CDR contained over 130,000 admissions/discharges, 22 million laboratory, 6 million pharmacy, and 2 million radiology records. The largest challenges the researchers faced were not in the CDR design itself, but accommodating the security and confidentiality requirements for health care data, and obtaining approval to access data from all the various clinical departments involved. Some of the database systems, where data for the CDR were imported from, contained hundreds of tables, where only a small portion of the imported data were relevant. Instead of importing the data as it was stored on the different systems, new tables were created for the repository that better suited its needs and uses.

With traditional relational databases, new tables are required to store new concepts (Anhoj, 2003). The overall layout of the database will contain many different tables to capture all the various concepts. Searching for all data on a specific patient will require accessing many different tables in the database using costly "join" functions which can decrease the performance of the system since the multiple tables need to be combined before they can be processed. For systems that will use a number of different concepts with new ones being added over time, performance and
maintainability may not be adequate with a traditional design.

2.1.2.2 Data Warehousing

A data warehouse provides access to data for direct analysis, and allows data to be extracted and transformed into data marts. The data warehouse is a data structure that is optimized for distribution, mass storage, and complex query processing (Sahama and Croll, 2007). The data warehouse approach is useful for high query performance or when data sources are expensive or transitory. Data marts are structures that are optimized for faster access, and usually provide a more focused view of data specifically for particular departments whereas the data warehouse contains data for the entire organisation.

Multiple data warehousing methodologies have been used to implement clinical data warehouses such as Enterprise Warehouse with Operational Data Store Architecture, Distributed Data Warehouse Architecture, and Data Mart Architecture, using SAS Data Warehouse Administrator software (Sahama and Croll, 2007). Figure 2.1 shows two different data warehouse designs. The Extraction-Transformation-Load (ETL) component provides a mechanism to move data from their original sources to the data warehouse and a Relational Database Management System (RDBMS). The Data Mart Architecture provides multiple data marts for different departments to access and manage data, and creates reports related to their interests using both RDBMSs and Multidimensional DataBases (MDB). The Enterprise Warehouse contains a central warehouse with several data analysis and data mart components, but unlike the Data Mart architecture it also contains an operational data store (ODS). This allows reporting and analysis to be performed while business operations are car-
ried out. The ODS stores recent information, and provides simple and fast queries for operations before it is transferred to the central data warehouse for long term storage, where more complex queries can be carried out. The number of components, complexity, and cost of setting up and maintaining a data warehouse is larger than other CDR designs.

2.1.2.3 Entity-Attribute-Value (EAV)

EAV databases provide a flexible and simple schema for biomedical data (Chen et al., 2000). It is optimized for queries that obtain data for a single patient. All data can be stored in one generic table with three columns: entity, attribute, and value (Johnson, 1996). New attribute types can be added without changing the physical database schema. EAV is often used when a large amount of heterogeneous data must be represented consistently since the number of attributes and types do not need to be pre-determined (Nadkarni, 1997). Medical data is frequently heterogeneous, containing a mixture of data types such as digits, characters, character and digits combined, images, signal data, and more. The EAV format can be used with existing commercial relational Database Management System (DBMS) such as Oracle, SQL Server, and MySQL. Outside of the medical domain, EAV has been used for well-known structures such as web cookies, Microsoft Windows Registry, and the ASN.1 encoding format (Nadkarni, 2000).

2.1.3 Building a CDR using EAV

Table 2.2 provides a summary of features for each type of CDR design that was discussed. Traditional relational databases provide easy constraint checking and efficient
use of space for dense data. They provide fast entity-centric and attribute-centric queries, and are compatible with the SQL standard. The data types must be predetermined, and for a large number of tables, performance will be penalised with
CHAPTER 2. TECHNICAL BACKGROUND

Table 2.2: Summary Checklist of CDR Designs

<table>
<thead>
<tr>
<th>Feature</th>
<th>Traditional Relational</th>
<th>Data Warehouse</th>
<th>Entity-Attribute Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complex Queries</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Distribution</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Easy Constraint Checking</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Efficient Space Dense Data</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Efficient Space Sparse Data</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Expensive Data Source</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Fast Entity-Centric Queries</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Fast Attribute-Centric Queries</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Heterogeneous Data</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>High Query Performance</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Mass Storage</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Pre-Determined Data Types</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Simple Design</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>SQL Compatible</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Support New Data Types Easily</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Transitory Data Source</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Unlimited Attributes per Entity</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

many "join" operations. The data warehouse has all the features of a traditional relational database, and additionally allows for complex queries and distribution of data. It is good for mass storage, when data sources are expensive or transitory, and provides high query performance.

The EAV design is simple to understand, allows for complex queries, and efficiently uses space with sparse data. It has fast entity-centric queries, works well with heterogeneous data, and is compatible with the SQL standard. EAV supports new data types easily, and provides the ability to add an unlimited number of attributes per entity. The flexible and simple schema of the EAV database makes it an ideal choice for use in a CDR. Since technology and medical testing is constantly changing, EAV will allow the CDR to use new data types over time without needing to change
the physical database schema. For estimating risk of medical outcomes, a CDR must gather all the data on a single patient, which is exactly what EAV is designed to provide. Using EAV to build a CDR will now be described in more detail.

2.1.3.1 Overview of Building a CDR Using EAV Format

If data are stored using the EAV format of entity, attribute, and value, multiple rows can be used for the data instead of putting all attributes in a single row like the UVA database which contained 208 columns for each patient (Nadkarni, 1997). By converting columns to rows, much less space is used since only the cells that have data in them need to be stored as a row. All information on a single entity can still be retrieved by a single query. While this simplifies the design, it makes it more difficult to query multiple entities that possess shared attributes. Retrieving all data for a single patient is easy while retrieving a specific attribute for all patients who match a particular attribute is costly. For research that is interested in sets of data for specific patients, this design consideration is not a drawback.

Table 2.3 shows the sample data in a conventional database structure. In this format, each column may have a different data type, and not every entity has a value for each attribute. The data structure depends on the contents of each column. Cells in integer and floating point (float) columns use 4 bytes of memory, and cells in text columns use a variable amount of memory depending on the amount of text data they contain, anywhere from 2 to $N$ bytes, where $N$ is the data size of each text cell. Adding a new row of data is simple; since the design of the database does not change, the new data can fit into the existing columns. To add a new attribute such as blood pressure to a conventional database, the physical schema needs to be modified since
Table 2.3: Conventional Database

<table>
<thead>
<tr>
<th>Patient ID</th>
<th>Name</th>
<th>Heart Rate</th>
<th>Temperature</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Integer)</td>
<td>(Text)</td>
<td>(Integer)</td>
<td>(Float)</td>
</tr>
<tr>
<td>4 bytes</td>
<td>2..N bytes</td>
<td>4 bytes</td>
<td>4 bytes</td>
</tr>
<tr>
<td>1</td>
<td>Joe Public</td>
<td>120</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Jane Private</td>
<td></td>
<td>36.5</td>
</tr>
</tbody>
</table>

A new column for the attribute is needed. This changes every row that already exists in the database, and can affect any application that uses the database.

When a large number of attributes needs to be stored in a conventional database, the attributes are usually split into multiple tables. Accessing data in multiple tables requires the use of the “join function, which can reduce querying efficiency considerably (Anhoj, 2003). The empty cells in a conventional database can result in wasted space, if the database software does not support using NULL values. If the data are very sparse, a considerable amount of wasted space may result. With EAV, the columns are converted to rows, and result in a generic table with the entity, attribute, and value columns as seen in Table 2.4. Space is more efficiently used since the CDR only needs to contain the relevant attributes for individual patients instead of reserving room for all possible attributes like the conventional design. All data in the “Value” field must be stored as text, so it can accommodate both characters and digits.

Table 2.4: Typical EAV Model

<table>
<thead>
<tr>
<th>Patient ID</th>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Name</td>
<td>Joe Public</td>
</tr>
<tr>
<td>1</td>
<td>Temperature</td>
<td>38.8</td>
</tr>
<tr>
<td>2</td>
<td>Name</td>
<td>Jane Private</td>
</tr>
<tr>
<td>2</td>
<td>Temperature</td>
<td>36.5</td>
</tr>
</tbody>
</table>
CHAPTER 2. TECHNICAL BACKGROUND

To add new attributes, such as date of birth and phone number, to the EAV database, new codes are added in the attribute column of the database table without any changes to the schema required. Table 2.5 shows an updated database table with date of birth listed for patients 1 and 2. A phone number attribute has also been added for patient 2 which demonstrates the ability to have a different number and different set of attributes for each patient. With the EAV format, there is no limit to the number of attributes that can be used per entity (Anhoj, 2003).

Table 2.5: Adding attributes to an EAV CDR

<table>
<thead>
<tr>
<th>Patient ID</th>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Name</td>
<td>Joe Public</td>
</tr>
<tr>
<td>1</td>
<td>Temperature</td>
<td>38.8</td>
</tr>
<tr>
<td>1</td>
<td>DateOfBirth</td>
<td>1/1/1975</td>
</tr>
<tr>
<td>2</td>
<td>Name</td>
<td>Jane Private</td>
</tr>
<tr>
<td>2</td>
<td>DateOfBirth</td>
<td>3/5/1980</td>
</tr>
<tr>
<td>2</td>
<td>PhoneNumber</td>
<td>555-9876</td>
</tr>
<tr>
<td>2</td>
<td>Temperature</td>
<td>36.5</td>
</tr>
</tbody>
</table>

The first commonly used EAV format is referred to in this thesis as the simple EAV format. Table 2.6 shows the data from the conventional database (Table 2.3) in the simple EAV format (Johnson, 1996). Space is used efficiently, since the CDR only contains the relevant attributes for individual patients. A 4-byte attribute ID is stored instead of the full attribute name. This uses less space than text, and allows for faster searching since the search index can be sorted numerically. A separate table is used to associate the attribute ID and name. For clarity in the tables below, we show the attribute name in parentheses. However, with the simple EAV design, all data must be stored in the same data type. To store both the text characters of the patient name and numerical heart rate and temperature values, the data are stored
Table 2.6: Simple EAV Format

<table>
<thead>
<tr>
<th>Patient ID</th>
<th>Attribute ID</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (Name)</td>
<td>1</td>
<td>Joe Public</td>
</tr>
<tr>
<td>2 (Heart Rate)</td>
<td>2</td>
<td>120</td>
</tr>
</tbody>
</table>
| 1 (Name)    | 3 (Temperature) | Jane Private | 36.5

in a "text" data type. The data structure of a simple EAV database can be large since the value column is text format, where the size of each cell ranges from 2 to \( N \) bytes.

Queries of numerical data stored as text are slower than queries of numerical data stored as integers or floating point values, since you must first convert the data from text to numerical before making a comparison. Therefore, storing numerical data in a text data type reduces the speed with which results can be returned for a query.

2.1.3.2 Multiple Data Types

The second common EAV format used in this thesis is referred to as the multi-data type EAV format. To overcome the data type limitation of the simple EAV format, the multi-data type EAV format requires multiple data types in addition to the basic text values used in Table 2.6. Support for multiple data types is achieved by using a separate table for each type of data such as dates, integers, text, real numbers, binary data for images or signals, and enumerated values (Nadkarni, 2002a). Table 2.7 shows the EAV database using multiple data types with the same example as Table 2.6. Since three different data types are used, we have separate EAV tables for text (2 to \( N \) bytes), integers (4 bytes), and floating point values (4 bytes). The attribute ID, name,
### Table 2.7: Multi-Data Type EAV Format

<table>
<thead>
<tr>
<th>Attribute ID (Integer)</th>
<th>Description (Text)</th>
<th>Type (Text)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Name</td>
<td>Text</td>
</tr>
<tr>
<td>2</td>
<td>Heart Rate</td>
<td>Integer</td>
</tr>
<tr>
<td>3</td>
<td>Temperature</td>
<td>Float</td>
</tr>
</tbody>
</table>

#### Text Data

<table>
<thead>
<tr>
<th>Patient ID (Integer)</th>
<th>Attribute ID (Integer)</th>
<th>Value (Text)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Joe Public</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>Jane Private</td>
</tr>
</tbody>
</table>

#### Integer Data

<table>
<thead>
<tr>
<th>Patient ID (Integer)</th>
<th>Attribute ID (Integer)</th>
<th>Value (Integer)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>120</td>
</tr>
</tbody>
</table>

#### Floating Point Data

<table>
<thead>
<tr>
<th>Patient ID (Integer)</th>
<th>Attribute ID (Integer)</th>
<th>Value (Float)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>36.5</td>
</tr>
</tbody>
</table>

and type are stored in an Attributes table. When making a generic EAV database, tables for other data types such as binary and integer would also be included, so that the database schema would not need to be changed in the future when the first integer value must be stored in the CDR. More details and descriptions of an attribute can be stored in the Attributes table, if required (Nadkarni, 2002b).

The multi-data type EAV format design allows the use of any data type.
ing numerical values stored as integer or floating point values allows mathematical operations to be performed on the values without first having to convert them. The efficiency of querying a single data type comes with the cost of more complex queries to extract data. Instead of searching one table, the queries must search one table for each data type, which can be more time-consuming.

2.1.3.3 Enhanced EAV Model

The EAV model can be enhanced with classes and relations (EAV/CR) to provide an object-oriented framework on top of an EAV physical structure (Anhoj, 2003). This was created specifically for use with scientific data. EAV/CR allows modelling of inter-class relationships, classes to contain other classes, and inheritance of attributes between classes (Nadkarni, 2000). Figure 2.2 shows an EAV/CR schema using classes and relations. The Class and Attribute tables hold definitions of classes and their data fields (Anhoj, 2003). ClassHierarchy shows relations between classes. Instances of classes are stored in the Object table and instances of the data fields are stored in the Data table.

With only these 5 tables, any real-world object can be recorded. Objects can also be part of other objects, and through inheritance, objects can also be related (Anhoj, 2003). Table 2.8 illustrates the EAV/CR model using inheritance (is-a relationship) and composition (has-a) relationship. Two main classes are used, one for patient and one for doctor. These are sub-classes of Person. A patient is a person and doctor is a person given by inheritance. A patient has a doctor given by composition. Name is an attribute of Person, but can be used by Patient and Doctor through inheritance.

EAV provides a simple physical schema for a CDR (Nadkarni, 2000). The users
of the system, however, expect the data to be in a form that is easily understood. A logical schema can be created to provide an easier interface to the system. With a conventional database, the physical and logical schemas are very similar. The logical schema with an EAV system, however, is quite different from the simple three column physical format used to store the data. A user-interface to the system should match the logical schema as much as possible for the benefit of the users. The logical schema is created through the use of metadata to describe the system. With our example in Table 2.8, the Class, ClassHierarchy and Attribute tables provide the metadata for the database. This metadata preserves the relationships between data, so users of the system can more easily see that Joe Public is a male and has a doctor Al Sharp who is his surgeon, rather than just seeing Patient01 is a male in isolation.
CHAPTER 2. TECHNICAL BACKGROUND

Table 2.8: EAV/CR Database Design

<table>
<thead>
<tr>
<th>Class</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>className</td>
<td>attributeName</td>
</tr>
<tr>
<td>Person</td>
<td>Name</td>
</tr>
<tr>
<td>Patient</td>
<td>Doctor</td>
</tr>
<tr>
<td>Doctor</td>
<td>Gender</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ClassHierarchy</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>superClassID</td>
<td>objectID</td>
</tr>
<tr>
<td>Person</td>
<td>Patient01</td>
</tr>
<tr>
<td>Person</td>
<td>Patient01</td>
</tr>
<tr>
<td>Person</td>
<td>Patient01</td>
</tr>
<tr>
<td>Person</td>
<td>Doctor01</td>
</tr>
<tr>
<td>Person</td>
<td>Doctor01</td>
</tr>
</tbody>
</table>

The EAV/CR model is more complex than EAV, and requires more work to create a suitable user-interface for the system (Anhoj, 2003). This extra work is during the initial design, while the users will benefit for the lifetime of the system. A system administrator will also require an understanding of object-oriented principles to take advantage of the framework.

2.1.3.4 Performance of EAV Format

Several EAV query implementations were explored and compared to a conventional schema. While data are stored in a simpler manner, queries can be more complex and data retrieval can take longer to execute (Nadkarni, 1997). The performance was found to be similar for entity-centered queries, while the EAV schemas were three to five times slower for attribute-centered queries (Chen et al., 2000). With a conventional database, attribute-centered queries can be performed with fast Boolean
operations (AND, OR, NOT), while in an EAV database, much slower set-based operations (Union, Intersection, Difference) must be used. The entity-centred queries being tested with SQL Server 7.0 took 1 second using EAV and 0.9 seconds using the conventional format. The attribute-centric tests took 28.9 seconds with EAV and only 8.2 seconds using the conventional model. The performance of attribute-centred queries is not important when looking at data for individual patients (Anhoj, 2003). CDRs that are designed to analyse data for specific patients use entity-centred queries.

There are many benefits to using EAV for a CDR. EAV allows the flexibility to add new attribute types without needing to change the physical database schema (Anhoj, 2003). There is also no limit to the number of attributes that can be used per entity. With sparse data, EAV has more efficient use of space. Only the attributes related to specific patients need to be stored. The physical database schema is simple with entity, attribute, and value columns being used (Nadkarni, 2000). The data are partially self-describing and can be easily mapped to formats such as XML for use in web applications. Entity-centred queries such as those looking for information about a single patient are efficient with EAV (Anhoj, 2003). In the medical domain where data and technology are quickly evolving, EAV can more easily accommodate change than a conventional database (Nadkarni, 2000). EAV design can be used with popular commercial database software such as Oracle, SQL Server, and MySQL. Database administrators should already be familiar with these commercial software packages requiring less training for maintenance and support of the CDR. Commercial support is also available for the database software.

One drawback to the EAV format for a CDR is the data formatting. Using only
the three columns is difficult for users to visually process compared to a relational database with all of a patient’s data neatly placed as columns (Nadkarni, 1997). EAV systems that have human users will require a more complex interface design that modifies the logical schema into a form that is easier to understand and use. Certain information systems and data mining tools also require data to be in a specific format, and may not be directly compatible with the physical EAV schema (Nadkarni, 2002a). The output of queries using EAV may need to be reformatted for compatibility with the software that will be using the data. Constraint checking is more difficult with EAV because each entity may have a different set of attributes which could lead to inconsistent data (Anhoj, 2003). Adding constraints to a conventional database is a simple matter of checking columns for valid or missing data. For example, if a patient’s name is required for each patient record, the data entry interface must be programmed to ensure that a name has been provided before the record is inserted into the EAV database.

The overhead of an EAV design with simple or static database schemas will outweigh its advantages (Nadkarni, 2000). If the database schema never changes, the flexibility of a system that can easily handle new attributes is not needed. If the database is very dense so that every entity has all the attributes recorded, the EAV design will use more space than a conventional design (Chen et al., 2000). However, clinical data are typically a mixture of dense and sparse data, like dense vital signs such as heart rate, respiratory rate, and blood pressure, and sparse data such as ventilation parameters, temperature, and laboratory results. As mentioned previously, the performance of attribute-centered queries will be slower with EAV. The queries will also be more complex to write than an equivalent query using a conventional
CHAPTER 2. TECHNICAL BACKGROUND

2.1.3.5 Opportunities for Improvement with EAV Format

The two commonly used EAV formats are the simple EAV and the multi-data type EAV formats. The simple EAV allows all data to be stored in one table for simple access, but is limited to a single data type which makes performing numerical comparisons or mathematical operations on the data inefficient, since the data must first be converted to the required data type. The multi-data type EAV stores data in their original types, so comparisons and operations can be performed without converting first, but since the data are stored in multiple tables, accessing the data becomes more complex and incurs a performance penalty. What is missing is an EAV format that combines the best of both common types for simple and efficient processing of multiple data types. Therefore, this thesis work explores, designs, and tests a new approach to CDR design with the objective of improving performance, flexibility, and usefulness for medical research using real-time and archived data.

2.2 Pattern Classification

Pattern classification attempts to assign an input of data to one of a set of classes, for example, trying to assign an input of patient physiological data to a class of either "survives" or "dies". Two different pattern classification methods, ANNs and DTs are used in this thesis work.
2.2.1 Artificial Neural Networks

An ANN is a type of network that is inspired by biology, whose nodes mimic biological neurons (Penny and Frost, 1996; Gershenson, 2003). In a biological neural network, incoming signals are received by neurons through synapses. When the signal is strong enough, the neuron is activated, and may send an outgoing signal to another neuron through an axon. With an the artificial neuron, inputs of data instead of synapses are multiplied by weights to mimic signal strength. Signals can also have negative values, thus reducing the strength of the signal. An activation function then calculates the output of the artificial neuron. Figure 2.3 show an artificial neuron with one or more data inputs leading to a propagation function which uses the connection weights to transform the data to be used by the activation function, frequently with a weighted sum (Penny and Frost, 1996; Kriesel, 2007). The activation function will then use the input from the propagation function along with a threshold to determine if or how the artificial neuron will activate. A common activation function used in ANNs is the hyperbolic tangent which maps values to [-1,1]. Finally, the output function calculates the data that are sent to other artificial neurons that the neuron is connected to, using the result from the activation function.

Multiple artificial neurons are assembled to create a network that can process information. A feedforward network is one that consists of an input layer, one or more hidden layers, and an output layer (Penny and Frost, 1996; Duda et al., 2001a). The hidden layers are called “hidden”, because their activations are not visible externally from the network. Every artificial neuron in one layer has only directed connections to artificial neurons of the next layer. If every neuron is connected to every other neuron in the next layer, they are called completely linked (Penny and Frost, 1996;
A perceptron is a feedforward network with an input layer, trainable weight layers, and an output layer. If the perceptron has only one weight layer, it is called a Single Layer Perceptron (SLP) while a perceptron with two or more trainable weight layers is called a Multilayer Perceptron (MLP). While a SLP can separate the input space using a hyperplane, a MLP is capable of classifying sets of any form using multiple convex polygons. Figure 2.4 shows a MLP feedforward network with one hidden layer.

An algorithm can be used to adjust the weights of the ANN to produce the desired output from the network. A common algorithm used in multilayer networks for this purpose is the backpropagation algorithm (Rumelhart et al., 1986; Duda et al.,
Figure 2.4: Multilayer Perceptron Feedforward Network with 1 hidden layer.

While the signals in the network are sent forward, the errors are propagated backwards. Learning in the ANN occurs when training data are input into the network and an output value is calculated. This output value is compared to the known result and an error value is calculated if there are any differences. The learning rule for backpropagation is based on gradient descent, starting with random weights, and then adjusting the weights in the network to try and minimise the error.

The Medical Information technologies Research Group (MIRG) has developed an ANN research framework (Ennett et al., 2004; Rybachynski, 2005) that uses a MLP feedforward network with backpropagation. It exhaustively modifies the training parameters in an automated fashion to find the optimal ANN structure in networks.
ranging from zero to $2n + 1$ hidden nodes where $n$ is the number of inputs. The ANN uses train, test, and verification sets. Since the test set is used to determine when to stop training, a separate verification set which has never been previously seen by the ANN is used to evaluate the performance. The MLP can approximate any non-linear function, which makes it well suited for dealing with complex data that have non-linear relationships (Frize et al., 1995; Kriesel, 2007). Medical data are thought to possess non-linear relationships between input attributes and outcomes (Catley et al., 2006). ANNs can generalise, even with noisy input data, and do not require any prior knowledge of the domain, therefore they require less statistical training to develop. They have the ability to implicitly detect complex relationships between both dependent and independent variables, and can detect all potential interactions between predictor variables (Tu, 1996). The connection weights in an ANN will be automatically adjusted to reflect any significant amount of non-linearity between the outcome and corresponding predictor variables. The hidden layer of the network allows interactions and associations between all input variables to be detected. ANNs are considered parallel networks since the functions in each neuron for a given layer can be calculated in parallel. This feature allows the processing speed to be increased significantly when using a computer system with multiple processors, something that is not possible with all types of pattern classification methods. ANNs have the added benefit that different training algorithms can be used to develop models depending on the type of data being analysed or the type of output required.

ANNs require input data to be normalised, which means the data must first be analysed and processed before they can be used in the ANN model. This can be more challenging when working in a real-time environment, where a new data value
CHAPTER 2. TECHNICAL BACKGROUND

is not known in advance and will need to be calculated as it arrives. Maximum and minimum values will need to be estimated to use the correct normalisation calculation for all data. The networks do not work well with missing input values. A missing data value needs to be replaced with some imputed value before being input to the network as this is not handled automatically. Missing data also provide challenges in a real-time environment where access to a potentially large number of previous data inputs are required to impute an accurate missing value either by using mean values of all previous values of that specific input or with more advanced approaches using the \( n \) closest matching sets of inputs (Ennett, 2003; Ennett and Frize, 2003; Ennett et al., 2008). It is difficult to interpret how an ANN provides the output that it does, since most of the calculations occur in the hidden layers which are not visible to the user of the network. While it is possible to see which inputs contributed most to the final output (Zhou and Frize, 2006), and thus infer which inputs are the most important, it is not easy to explain how the ANN arrived at that conclusion. This method currently requires the Matlab software tool which is proprietary and requires a paid license to use.

2.2.2 Decision Trees

A DT is a pattern classification technique that is very easy to use and understand. A number of DT algorithms exist such as Classification and Regression Trees (CART), Iterative Dichotomiser 3 (ID3), C4.5, and C5.0 (Rulequest Research, 2009a). The C5.0 variant is an improvement over C4.5 which itself is an improvement over ID3. Most decision trees use the same basic principles with variations on pruning methods, handling of missing data, and stopping rules (Duda et al., 2001b). This section focuses
on the C4.5 and C5.0 algorithms as C5.0 was used for the DTs in this thesis.

The DT mimics the shape of an inverted tree with a root node at the top of the tree, and branches that are connected to the root node (Duda et al., 2001b). Each branch can connect to nodes further down on the tree, and continue until a leaf (terminal) node is reached. Classification of data is completed by starting at the root node. Every node corresponds to a particular attribute or variable, and each branch from that node represents a possible value. Based on the input values, the appropriate branch is followed to the descendant node, and the process repeats until a terminal node is reached. Each terminal node represents a category, and the one that is reached descending the tree is the classification of the input. Figure 2.5 shows a simple decision tree to classify the type of vehicle by its weight and number of tires. For example if the input data was 4 tires and weight of 3,000 pounds, at the root node, the branch labelled 4 would be followed to the descendent node which then asks if weight was less than 5,000 pounds. Since the weight is 3,000 pounds, the “yes” branch is followed to the terminal node which is labelled “car”. The DT would, therefore, classify the input data as a car.

2.2.2.1 Creating a Decision Tree

To classify data using a DT, it first needs to be created, or “grown” using training data. The DT will progressively split the data into smaller subsets. With the C4.5 algorithm, trees are created top-down and then a pruning process is applied (Song et al., 2008). A divide and conquer algorithm is used to construct a tree from training data \( D \) such that if all data in \( D \) belong to an identical class \( (C_j) \), then the node is a terminal node labelled \( C_j \) (Kohavi and Quinlan, 2002). Otherwise let \( T \) be a test
that partitions $D$ into $D_i$ which will then continue until all nodes become terminal nodes. C4.5 uses a greedy search algorithm, choosing a test that maximizes a splitting criterion heuristic. The two criteria used are information gain and gain ratio.

Information gain is used to help measure the relevance on an attribute, which is the amount of entropy reduction achieved for an attribute by learning the state of another random attribute (Diaz-Martinez et al., 2004). To calculate information gain, the information content of a set of input data must first be measured. The relative frequency of data in $D$ that belong to class $C_j$ is denoted as $RF(C_j, D)$ and the information content of a set of input data that identifies the class of data in $D$ is (Kohavi and Quinlan, 2002):

$$I(D) = -\sum_{j=1}^{x} RF(C_j, D) \log(RF(C_j, D))$$ (2.1)
Information gain is then measured using the partitions created by test $T$ of $D$ into subsets $D_1, D_2, ..., D_y$ with the following equation (Kohavi and Quinlan, 2002):

$$G(D, T) = I(D) - \sum_{i=1}^{y} \frac{|D_i|}{|D|} I(D_i)$$  \hspace{1cm} (2.2)

The gain criterion chooses the test $T$ that maximises $G(D, T)$ (Kohavi and Quinlan, 2002).

Since information gain on its own will more likely select attributes with a large number of distinct values, C4.5 uses a gain ratio to help bias the tree against this limitation (Song et al., 2008). Gain ratio takes into account the information from the partition itself. This partition information can be calculated using (Kohavi and Quinlan, 2002):

$$P(D, T) = -\sum_{i=1}^{y} \frac{|D_i|}{|D|} \log\left(\frac{|D_i|}{|D|}\right)$$  \hspace{1cm} (2.3)

The gain ratio criterion chooses the test $T$ from among the tests with at least average gain that maximises (Kohavi and Quinlan, 2002):

$$GainRatio = \frac{G(D, T)}{P(D, T)}$$  \hspace{1cm} (2.4)

The root node is selected by finding the highest information gain attribute test to be applied at the root of the tree. The data are then split according to the test that is chosen and the sub-problems are then recursively solved until the entire tree is built (Song et al., 2008). After the divide and conquer algorithm is complete, it is possible that the decision tree can correctly classify all training data. This overfitting may lead to a loss of generalisation for the tree on future data, since it is so specific to the training data (Kohavi and Quinlan, 2002). To help alleviate this issue, the C4.5
algorithm uses a post-pruning process where some of the decision tree is removed after it has been produced. The pruning occurs in a single pass from the bottom of the tree to the top. A branch of the tree is replaced by a terminal node based on a predicted error rate (Diaz-Martinez et al., 2004). When the error rate for the last terminal node is lower than the rate for the branch, the terminal node will replace the branch.

2.2.2.2 Handling Missing Data

C4.5 uses a probabilistic algorithm to handle missing data when constructing the decision tree, so missing values do not need to be imputed and are handled automatically. The DT uses a two phase approach: (1) to penalise gain ratios for attributes that have missing values, and (2) to send a fraction of cases with missing values down each branch of the tree based on the observed frequencies at a given node (Song et al., 2008). For example, consider an attribute $A$ that has two possible values (‘true’ and ‘false’) where the ‘true’ value is seen 80% of the time in the training data, and the ‘false’ value is seen 20% of time. For input data that is missing the $A$ attribute, a fraction (80%) of the missing values are sent down the ‘true’ branch, while a fraction (20%) of the values are sent down the ‘false’ branch of the tree to help distribute the missing values. Information gain is computed from these fractional cases and can be further divided if a second missing attribute must be tested at a later branch of the tree (Song et al., 2008).
2.2.2.3 Improvements with C5.0

The C5.0 algorithm (Rulequest Research, 2009a) is an improvement over C4.5. The DTs created with C5.0 have comparable predictive accuracies, but are smaller, use less memory, and are considerably faster (Rulequest Research, 2009b). In a comparison done by Rulequest Research using multiple data sets, C4.5 produced the trees in 37 seconds on average, while C5.0 produced the trees in 2.2 seconds on average. The C4.5 trees had an average of 4,660 leaves while the C5.0 trees had an average of 3825 leaves. C5.0 uses an order of magnitude less memory than C4.5 during tree construction. One data set required 3GB of memory with C4.5 while only 200MB was required for C5.0 (Rulequest Research, 2009b).

Some other improvements with the C5.0 algorithm are support for boosting, variable misclassification costs, and winnowing (Rulequest Research, 2009b). Boosting is the process of generating multiple DTs and combining them to improve accuracy. Each tree votes for its predicted class and a final class is determined by the vote count. The first DT is created using the normal process, but subsequent trees try to focus on correctly predicting cases that had wrong predictions in the previously generated trees. By default, C5.0 will generate 10 trees when using the boosting feature. While all errors in C4.5 are treated equally, it is possible to use variable misclassification costs in C5.0. When trying to classify data, some classification errors can be more grave than others. Separate costs can be configured for class pairs, and C5.0 will then try to minimise expected misclassification costs instead of error rates (Rulequest Research, 2009b). C5.0 has the ability to automatically winnow attributes from data that have very high dimensionality before the tree is constructed. It does this by removing the attributes that it predicts have minimal relevance to create smaller trees
with higher accuracy (Rulequest Research, 2009b). C5.0 also supports additional data types, such as case labels, dates, times, and ordered discrete attributes.

### 2.2.2.4 Advantages and Limitations of DTs

DTs work well with both metric (numerical, real values) and non-metric (categorical, descriptions) data. They have been shown to have comparable accuracy to other methods such as ANNs and nearest-neighbour classifiers in a wide variety of applications (Duda et al., 2001b). DTs are especially useful when no prior knowledge of an appropriate form of classifier is known. Only minimal data preparation is required for creating DTs since the data do not need to be normalised nor missing data imputed. The output of the DT is easy to understand and interpret. A user can look at the tree and see exactly how the classification was completed by following the branches of the tree based on the input data. Large amounts of data can be processed by DTs quickly (Duda et al., 2001b). The C5.0 algorithm has open source code and libraries available for both constructing DTs, and also integrating the DT model interpreter into third party software and applications (Rulequest Research, 2009a). This makes C5.0 DTs created during research, very easy to use in prototypes and systems applying the research without the need for licenses or using proprietary software.

DTs can have difficulty inferring simple concepts, such as if less than 50% of discrete attributes in the data set have a specific value (Duda et al., 2001b). They can create excessively complex trees that while perfectly classifying the training data, result in overfitting and do not generalise well. DTs algorithms such as C5.0 use heuristic algorithms that make local optimisations, which cannot guarantee a globally optimal tree. The divide and conquer algorithms in DTs perform well if a small
number of highly relevant attributes exist, but decrease if many attributes have complex interactions (Rokach and Maimon, 2008). Other classification techniques can represent a classifier that is more compact and difficult to represent using a DT since several duplicates of the same subtree could be found in a DT to represent several concepts. Due to the greedy search algorithm used in DTs such as C5.0, resulting trees can be unstable. A small change in one split could change the entire subtree below. Data sets that have small variations in the training data could cause the algorithm to choose an attribute which is not truly the best and lead to an entirely different DT result. Every branch on a DT comes from an exact split, for example, a node may split heart rate at 120 beats per minute, one branch is for values less than 120, and one for greater or equal to 120. With medical data, such an exact split likely does not accurately characterise the biological system, and may limit the degree of accuracy that DTs can achieve using medical data.

2.3 Neonatal Clinical Decision Support

There are very few research groups in the world working on neonatal-specific medical research, and even fewer researching neonatal CDS. This section highlights the research projects related to CDS focusing on the neonatal population.

2.3.1 Heart Rate Observation System (HeRO)

HeRO is a project led by J. Randall Moorman at UVA that looks at heart rate variability with the goal of early detection of neonatal sepsis in very low birthweight (<1500g) infants. Their research is based on the observation that heart rate variability is found in healthy patients while a reduction in variability could signify illness. In the
case of neonatal sepsis, early detection could allow for early treatment with antibiotics and improve the outcome for patients. The group collected data from 3003 patients in nine NICUs from April 2004 to May 2010 (Moorman et al., 2011b). The data was collected daily including vital signs from patient monitors, respiratory impedance waveforms, O₂ saturation, and three Electrocardiogram (ECG) leads. To store and process all of the data, a grid computing cluster consisting of 80 processing cores, 40GB of RAM, and 100TB of storage was required (Moorman et al., 2011).

HeRO uses multiple mathematical and statistical techniques to analyse heart rate variability such as the mean and variance of time-domain parameters from the heart rate signal, band specific variances in the frequency-domain parameters, the heart rate impact of breaths using the Hilbert transformation in the phase domain, and deceleration or acceleration detection using wavelet transformations to try to detect neonatal sepsis (Moorman et al., 2011). The results from these techniques are combined using logistic regression, k-nearest neighbour analysis, and ANNs. The output of the model provides the probability of an event in the next 24 hours, which is divided by the average probability of the event, and presented to clinicians as a HeRO score, the fold-increase in risk of an upcoming event. For example, a patient with a HeRO score of 1 represents the average risk, while a HeRO score of 4 represents a 4-fold risk of sepsis.

A large randomised trial of the HeRO system was conducted to test the hypothesis that heart rate characteristics monitoring improves neonatal mortality outcomes (Moorman et al., 2011b). The 3003 patients (described previously) were divided into two groups, one where the HeRO system would be displayed for patients, and one where the display was covered. The HeRO system displayed the current HeRO score
for patients, updated every hour, and included a graph that showed the trend for the last 5 days. The primary outcome measure was the number of days alive and off the ventilator in the 120 days after randomisation. The trial also looked at in-hospital mortality, number of days on antibiotics, length of NICU stay, and number of ventilator days. They found that a 2.3 day increase of the primary outcome measure was not statistically significant ($P = 0.08$), but did find a 22% clinically significant reduction in mortality rate ($P = 0.04$) for patients that were monitored with the HeRO system. They found no significant differences in the secondary outcome measures. While the mechanism to determine how monitoring patients with HeRO reduced mortality was not studied, the trial did find some interesting indirect evidence. The patients who were monitored by the HeRO system had a 10% increase in blood cultures to test for sepsis, 5% more days on antibiotics, and greater than 30% fewer deaths in the 30 days following a confirmed sepsis diagnosis.

The HeRO system updates its data every hour, but requires the use of a grid computing cluster with large amounts of processing power, RAM, and storage. It also uses proprietary technology. The primary use of the single HeRO score that is generated is to warn physicians when the risk of neonatal sepsis is increasing. The results of a large clinical trial found that providing a HeRO score to clinicians did not have a significant difference in the number of days patients used antibiotics or days in the NICU.

2.3.2 Artemis Analytic System

The Artemis Analytic System (Blount et al., 2010) is a pilot project collaboration between researchers at the University of Ontario Institute of Technology (UOIT),
IBM, and the Hospital for Sick Children (SickKids) in Toronto, Canada. The first goal of the project is to perform temporal data mining on physiological data from infants in a NICU. The second goal is to run clinical rules online and in real-time to detect early indicators of medical conditions, and provide the information to clinicians. The conditions the system is trying to detect are IVH, nosocomial infection, periventricular leukomalacia, and pneumothorax.

To capture data in the NICU, the system uses the Capsule Tech Inc. DataCaptor which supports cables and drivers for more than 450 different devices with up to 8 devices per unit. The project is collecting the full waveforms of ECG, heart rate, respiratory rate, blood Oxygen saturation as measured via a pulse oximeter (SpO$_2$), and blood pressure. It is important to note that SickKids does not allow non-clinical access to clinical information system or laboratory results which are stored in an Oracle database. Instead of direct access to the data, they create a shadow database, and export data for specific enrolled patients every 30 minutes for researchers. This means the Artemis system is not actually getting the data in real-time, but rather with a 30 minute time delay. Data processing is completed by the IBM InfoSphere Stream Computing system which provides processing of multiple high-volume data streams. The system computes data and stores it in the database at the same time. The Stream Processing Application Declarative Engine (SPADE) programming language is used for programming and controlling the stream computing system. Knowledge extraction is supported by the use of multidimensional, temporal, and data stream data mining frameworks developed by the Artemis team. Prediction rules are represented using the Predictive Modelling Markup Language (PMML). The stream computing system can read PMML models, and then score them as data arrive.
Artemis attempts to predict medical events as data arrive from multiple data sources, and then reports conditions that meet the clinical rules. An example of one such clinical rule is: “If mean arterial BP (MBP) is less than the neonatal patient’s current gestational age (eg: 24 mmHg for 24 weeks gestation) for 20 seconds or more AND if SpO2 is less than 85% for the same period of 20 seconds or more, then a reportable condition is present” (Blount et al., 2010). To protect the privacy of patients, data are de-identified before being stored in the Artemis system. A unique Artemis identifier is created for each patient, and used in place of the hospital patient identifier. The researchers at UOIT periodically download the de-identified data for their research via an encrypted Internet link to the SickKids hospital. The IBM team then does the same thing using an encrypted Internet link to UOIT. While the Artemis Analytic System research prototype team have been attempting to develop a real-time neonatal medical system, they use proprietary hardware and technology, and have not publicly disclosed any performance results.

2.3.3 Realtromins

Realtromins (Real Time Risk of Mortality & Instability) is a medical system that provides predictive analytics to alert healthcare providers that critically ill patients are deteriorating. It attempts to identify high risk patients to improve patient care and outcomes (Realtromins Inc., 2012). The system uses continuous physiological signals such as ECG, patient monitor data, laboratory values, and clinical parameters using the Health Level 7 (HL7) messaging standard. These attributes are used to assess any changes in the severity of illness of patients in the NICU and help clinicians assess the effectiveness of medical interventions. Realtromins uses a mixture of heart rate
variability analysis, signal processing to remove artifacts from clinical signals, and the neural networks available in the SAS Enterprise Miner Software.

Physiological data from the patient are collected on admission and at least once after admission (Kocis and Kocis, 2007). The data are analysed with a statistical model and the risk of mortality is presented to a clinician using their user-interface. The model is created using multivariate physiological data to identify the variables affecting mortality risk for the patient, and assigning weights to the variables accordingly. Signal processing is applied to detect information that is not easily found by clinicians looking at the raw signal data. Neural networks are used to find patterns in the physiological data, statistical analysis, and signal processing of the data. Furthermore, bispectral methods are used to detect nonlinear trends in patients, and linear signal processing techniques are used measure heart rate variability.

Realtromins uses spectral analysis of heart rate data, spectral analysis of arterial blood pressure, patient demographic data, temperature, carbon dioxide concentration, oxygen saturation, and respiratory rate to predict mortality risk. Their system consists of a computer system with a data collection interface to receive the physiological data, a temporary data store to analyse the physiological data, and a mortality risk function that uses data from the data store, calculates a mortality risk, and presents the risk for a patient to the user interface. It is not clear how they present the information, but a simple number alone is not meaningful to physicians. While Realtromins has been attempting to develop a real-time neonatal medical system, they use proprietary hardware and technology, and have not publicly disclosed any performance results.
2.3.4 NEONATE

The NEONATE research project’s aim was to investigate decision support in the NICU, and implement decision support tools that can draw a clinician’s attention to events where specific actions can be taken or avoided (Hunter et al., 2003). The project was a partnership between the University of Aberdeen (UK), and the NICU in the Simpson Centre for Reproductive Health in Edinburgh (UK) (Ewing et al., 2003).

Data collection included 400 patient hours of data (32,000 data records) from 31 patients (October 2001 to February 2002) with physiological data sampled every second and observations made by nurses of actions performed on the patients. After interviewing clinical staff, a lexicon of 51 terms were generated for observations (eg: Crying, Movement) of patients, and 191 terms for actions (eg: Communication, Feeding) that can be taken with patients (Hunter et al., 2003). At the patient bed, a research nurse was employed for four months to record activity of the patient, actions taken, alarm limits, equipment used, and drugs administered, into a laptop. At the same time, an automated system collected physiological data such as heart rate, oxygen saturation, temperature, and blood pressure from patient monitors every second. The data were stored in a Microsoft Access database with an anonymous internal identifier for the patient.

The NEONATE project focused on trying to predict when manual ventilation of a patient might be required. A clinical expert derived a protocol for when manual ventilation should occur, when transcutaneous oxygen is less than 3 mmHg or oxygen saturation is less than 60% for at least 10 seconds and heart rate is less than 100 beats per minute (Hunter et al., 2003). The group then looked at the data to see how many
false positives and false negatives were present to determine how many instances of manual ventilation should have been done, and how many could have been avoided to deliver more “optimal” treatment. The researchers found that the clinical experts were reluctant to rate specific manual ventilation actions without first looking at a more detailed patient history, and the actions that occurred for some time preceding the action. They concluded that the prediction protocol should take into account respiratory management as a whole, instead of just individual actions which makes the protocol much more complex to develop.

2.3.5 Cochrane Review of CDS Systems for Neonatal Care

A Cochrane systematic review of CDS systems for neonatal care was published in 2009 (Tan et al., 2009). While decision support has been used for some time in adult intensive care, this was the first systematic review on CDS systems used in neonatal medicine. The review focused on systems which compared the effects of CDS versus no CDS in the care of neonatal patients. Only three studies were found that met the study criteria, two were for computer-aided prescribing and only one looked at physiological monitoring.

The objective of the physiological monitoring study was to assess if physiological trend data presented to clinical staff could improve the outcome of patients in a NICU with 12 beds (Cunningham et al., 1998). This was a randomized controlled trial with four groups being monitored for the first week of life. The first group had no display of trend data (230 patients), the second group had a continuous display of trend data (215 patients), the third group had display of trend data only during even days of the study (76 patients), and the fourth group had display of
trend data only during odd days of the study (79 patients). It is important to note that no type of statistical or other processing was done on the physiological data, but simply the trends displayed to the clinical staff. The short term effects were measured by the number of actions such as number of blood gases taken, volume of colloid administered, and cranial ultrasound findings (Cunningham et al., 1998). Medium term effects were measured by time ventilated, time to death or discharge, amount of supplemental oxygen required, and cranial ultrasound findings at discharge. Long term effects were measured by neurodevelopmental status at ages 1 to 4.

The study concluded that based on the short, medium, and long term measurements, no benefit to patients could be demonstrated from using simple physiological trend monitoring (Cunningham et al., 1998). There was no significant difference in mortality, short-term physiological stability, risk of IVH, or long term neurodevelopment between the four groups. The Cochrane review as a whole concluded that there was insufficient data to determine the benefit or harm of using CDS in neonatal care (Tan et al., 2009).

2.3.6 Medical Information Technologies Research Group

The MIRG at Carleton University and the University of Ottawa in Ottawa, Canada has been actively involved in neonatal medical research. Our research group has developed risk estimation models for a number of clinical events and complications, such as mortality in adult and neonatal intensive care units, length of stay, duration of ventilation, and pre-term birth (Frize et al., 1995, 2001; Ennett et al., 2004; Zhou and Frize, 2006; Catley et al., 2006; Townsend and Frize, 2008; Frize et al., 2011).

Missing values in medical databases can significantly impact the ability to perform
biomedical research for identifying risk factors and causes of clinical events and disease progression. Some pattern classification techniques, such as ANNs cannot operate with missing values and require missing data to be replaced with some form of data, or cases with missing values need to be removed (Ennett et al., 2001). MIRG developed a method to impute such missing values in neonatal data by integrating neural networks and Case-Based Reasoning (CBR) (Ennett, 2003; Ennett and Frize, 2003; Ennett et al., 2008). This hybrid imputation approach used a linear ANN to estimate patient mortality, and the resulting connection weights generated by the ANN were used as the relative match weights for the CBR to find the ten closest matching patient cases. These ten patient cases were used to calculate the mean of the missing attribute, which was then used to replace the missing value. MIRG has also presented a framework that classifies the inconsistencies in a medical database to automatically process the data for research purposes. The goal is to improve the quality of medical databases by identifying the reason that data are missing, and automatically processing outliers and typographical errors to retain more information from a database. This framework provides an alternative to deleting cases with outliers and typographical errors, while also imputing relevant values into cases with missing data based on the reason that the data value is missing, the source of the data, and the data type (Gilchrist et al., 2008).

The most recent neonatal mortality model developed by MIRG (Townsend and Frize, 2008) used the summary data for 27 out of 37 SNAP attributes collected over the first 12 hours following NICU admission by the Canadian Neonatal Network (CNN) between January 1996 and October 1997 (Lee et al., 2000; Sankaran et al., 2002). Other important outcomes identified by MIRGs medical partners such as length of
stay, duration of ventilation, BPD, and IVH were also modelled using ANNs, and the estimated risk stratification was obtained (Townsend, 2007). Risk stratification classified the risk of the outcome into three categories: low (0% - 20%), moderate (21% - 74%), and high (75% - 100%) which the medical partners found easier to understand and relay to parents than a percentage value. It was determined that imputing missing values using a k-Nearest Neighbour (k-NN) approach with uniform weights produced a better performing mortality model than the SNAP and SNAPPE-II models. Answers from questionnaires revealed that physicians are willing to use an accurate CDS system to help increase the speed of decisions, improve the quality of care delivered, and improve the accuracy of predictions.

Hybrid DT-ANN models have been developed by MIRG for estimating high risk pre-term birth in an obstetrical environment for both women who have had previous children (parous) and those who have not (nulliparous) (Frize et al., 2011). DTs were used to select features for an ANN by eliminating variables which had little impact on estimating the risk for outcome of interest. The ANN then processed the remaining variables to create the model. The hybrid DT-ANN was found to perform better than either method alone (Frize and Yu, 2010). If the DT step were to find that all variables had some impact on estimating the risk, then the ANN would also use all variables and the result would be no different than using the DT and ANN on their own.

These models were all developed with several quality databases, but none contained real-time data.
2.3.7 Non-Neonatal Clinical Decision Support Systems

While there are few research groups looking at neonatal clinical decision support, many groups are investigating CDS for adults. A systematic review was conducted in 2005 to look at the effectiveness of CDS in improving clinical practice. After examining 70 different studies, the review found that 68% significantly improved clinical practice (Kawamoto et al., 2005). In particular, four features of CDS were identified as independent predictors of improved clinical practice.

1. Integrated and automated as part of clinician workflow
2. Provided recommendations instead of just assessments
3. Available at time and location of decision making
4. Computer based decision support

The CDS systems in the review that possessed all four features had a 94% success rate of significantly improving clinical practice. During the review, experimental justification was found for sharing recommendations with patients, providing performance feedback, and documenting reasons for not following recommendations.

A systemic review of the effect of CDS systems was completed in 2012. The review examined studies completed from 1976 to 2011, finding 148 unique studies to investigate (Bright et al., 2012). The review found that the majority of studies (86%) assessed health care process measures, while few assessed clinical outcomes (20%), and measured costs (15%). Some examples from the studies that assessed clinical outcomes include a CDS system that presented a quantitative pretest probability of acute coronary syndrome to see if it would safely reduce unnecessary resource use in
low-risk emergency department chest pain patients (Kline et al., 2009), and a system that provided alerts to physicians for hospitalised patients who were at high risk for deep-vein thrombosis and then allow them to order or withhold treatment for those patients (Kucher et al., 2005). Another example is a CDS system designed to help physicians choose the appropriate antibiotic treatment in an effort to improve the rate of appropriate treatment while reducing antibiotic costs and antibiotic resistance (Paul et al., 2006). CDS systems have also been developed to recommend more precise drug doses of erythropoietin to improve anemia management in renal disease patients to avoid adverse effects (Brier et al., 2010), and as a way to provide systematic patient and physician reminders to increase the screening rates for colorectal cancer of patients who were overdue for screening (Sequist et al., 2009). Many other CDS systems exist that cover a wide range of topics such as diagnosis, pharmacotherapy, chronic disease management, laboratory test ordering, immunisations, preventative care, and discharge planning (Bright et al., 2012).
Chapter 3

Clinical Context and Data Collection at CHEO

This chapter provides background information on the NICU environment, the neonatal databases used at CHEO, and the evolution of the data collection systems.

3.1 NICU Environment at CHEO

Data are being collected from the NICU at CHEO in Ottawa for this thesis work. CHEO is an academic teaching hospital with 167 beds and is affiliated with the University of Ottawa. The NICU has 20 beds servicing Eastern Ontario, Western Quebec, and Nunavut. It is a medical and surgical unit. CHEO is an outborn-only NICU, meaning that sick newborns who were born at other locations are transported to this NICU. These patients are often several hours or days old by the time they are admitted to the unit. There are approximately 420 admissions per year with 40% being low birth weight or preterm births. Table 3.1 show the admission statistics for
Table 3.1: CHEO NICU Admissions Statistics

<table>
<thead>
<tr>
<th></th>
<th>2006/07</th>
<th>2007/08</th>
<th>2008/09</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beds Available</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>% Occupancy</td>
<td>73.0%</td>
<td>73.0%</td>
<td>80.1%</td>
</tr>
<tr>
<td><strong>Inpatient Discharges:</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total Discharges</td>
<td>345</td>
<td>420</td>
<td>416</td>
</tr>
<tr>
<td>Total NICU Days</td>
<td>5,715</td>
<td>5,208</td>
<td>5,083</td>
</tr>
<tr>
<td>Average Length of Stay in NICU (days)</td>
<td>16.6</td>
<td>12.4</td>
<td>12.2</td>
</tr>
<tr>
<td><strong>Admit Weight:</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;1000 grams</td>
<td>77</td>
<td>62</td>
<td>71</td>
</tr>
<tr>
<td>1000-1499 grams</td>
<td>27</td>
<td>29</td>
<td>24</td>
</tr>
<tr>
<td>1500-2499 grams</td>
<td>47</td>
<td>63</td>
<td>81</td>
</tr>
<tr>
<td>&gt;2500 grams</td>
<td>194</td>
<td>266</td>
<td>240</td>
</tr>
</tbody>
</table>

the NICU during the past few years. The patients are connected to several monitoring devices such as ventilators, pulse oximeters, blood pressure monitors and heart rate monitors, that can transmit the data for storage, retrieval, and processing.

Our research clinical partner at CHEO on this project is Dr. Erika Bariciak, MD, FRCPC, a Neonatologist who works in the NICUs at both CHEO and the Ottawa Hospital’s General Campus; she is also an Assistant Professor of Pediatrics at the University of Ottawa, and an Adjunct Professor in Systems and Computer Engineering at Carleton University. Working directly with clinicians for biomedical research is important. It allows direct feedback from experts in the field, better acceptance of the research by the clinical staff and patients, and affords them the ability to help shape the research to provide something useful to the medical community.

We used two different data sets for this thesis work: (1) summary data from the first 12 hours following NICU admission collected by the Canadian Neonatal Network (CNN) from multiple NICUs, and (2) a data set of real-time medical measurements
collected from a single, outborn-only NICU. The second data set was collected by our research group at CHEO. CHEO is also a member of the CNN partnership. These data points were stored in a CDR structure that we developed, the details of which are discussed in Chapter 4. This real-time CDR is capable of acquiring and storing patient vital signs, Admissions, Discharges, and Transfers (ADT) information, and laboratory results in real-time as the data arrive from the various medical systems (Gilchrist et al., 2008). The CDR was designed to store patient data efficiently, and provides the ability to access data for individual patients almost instantaneously (Gilchrist et al., 2011b). Quick access to patient data is critical for CDS applications to make risk estimations in real-time using the most recent medical measurements for each patient.

Table B.1 in Appendix B provides an overview of the median values recorded in the CNN and CHEO databases for the patient populations used for the mortality risk estimation model development. The attributes, taken from the SNAP scoring system, that we considered for inclusion in the mortality risk estimation models included: vital signs (mean blood pressure, heart rate, respiratory rate) and blood-based laboratory values (white blood cell count, serum glucose, absolute neutrophil count, immature/total neutrophil ratio, blood hematocrit, platelet count, serum pH, Partial pressure of carbon dioxide in the blood (PCO₂), Partial pressure of oxygen in the blood (PO₂), serum sodium). Recall that SNAP is a risk estimation model based on scores from various physiological and therapeutic variables chosen based on clinical experience (Richardson et al., 1993).
CHAPTER 3. CLINICAL CONTEXT AND DATA COLLECTION AT CHEO  56

3.1.1 Canadian Neonatal Network (CNN) Database

The CNN is a group of researchers from multiple disciplines interested in perinatal and neonatal care that currently collects data from seventeen Level 3 NICUs across Canada with inborn and outborn patients (Lee et al., 2000; Sankaran et al., 2002). Level 3 NICUs offer advanced life support for critically ill newborns including invasive ventilation therapy, which is not usually provided in a Level 2 or Level 1 NICU. Inborn NICUs will include infant data from the first 12 hours of life, when the neonate is still transitioning from fetal life to the outside world. Outborn infants are often greater than 12 hours old before NICU admission.

This data set only contains a single data point for each SNAP attribute collected during the first 12 hours after NICU admission for each patient. SNAP's strategy is to extract only the "worst" measurements for each clinical parameter to indicate the greatest deviation from normal during the first 12 hours in the NICU, and use this snapshot to estimate the patient's severity of illness (Richardson et al., 1993). Vital sign data in this database are typically manually abstracted from charting flowsheets, where information is periodically recorded for each patient. Recorded values are representative of the patient's condition during the most recent time period, and would not include data artifacts resulting from manipulating the baby for routine care.

The CNN database contains many clinical values including the 37 SNAP illness severity attributes, plus patient outcome information like mortality. For comparative purposes with the CHEO data, we only used the 18 attributes listed in Table 3.2 that were available electronically in the real-time database being collected in the NICU at CHEO. Other SNAP attributes such as birth weight, presence of apnea, presence of
Table 3.2: List of 18 attributes used in CNN and CHEO Mortality models

<table>
<thead>
<tr>
<th>Attribute List</th>
</tr>
</thead>
<tbody>
<tr>
<td>lowest mean blood pressure</td>
</tr>
<tr>
<td>highest mean blood pressure</td>
</tr>
<tr>
<td>lowest heart rate</td>
</tr>
<tr>
<td>highest heart rate</td>
</tr>
<tr>
<td>lowest serum glucose</td>
</tr>
<tr>
<td>highest serum glucose</td>
</tr>
<tr>
<td>lowest serum sodium</td>
</tr>
<tr>
<td>highest serum sodium</td>
</tr>
<tr>
<td>lowest pO$_2$ level</td>
</tr>
<tr>
<td>highest pCO$_2$ level</td>
</tr>
<tr>
<td>lowest serum pH</td>
</tr>
<tr>
<td>highest respiratory rate</td>
</tr>
<tr>
<td>lowest blood hematocrit</td>
</tr>
<tr>
<td>highest blood hematocrit</td>
</tr>
<tr>
<td>lowest absolute neutrophil count</td>
</tr>
<tr>
<td>highest immature/total neutrophil ratio</td>
</tr>
<tr>
<td>lowest platelet count</td>
</tr>
<tr>
<td>lowest white blood cell count</td>
</tr>
</tbody>
</table>

seizures, temperature, and urine output were recorded by the nursing staff manually, and thus were not available in an automated manner in the CHEO database. The entire CNN database contains 20,488 patients. When we excluded patients who had measurements recorded for less than 50% of the 18 attributes, the final CNN data set contained 17,364 patients with a 4% mortality rate (687 patients) and 41.8% of patients were female.

3.1.2 CHEO NICU Database

A major difference between the CHEO and CNN databases is that the attributes recorded in the CHEO database are coming directly from the patient monitoring devices, and so may contain data artifacts that occur during routine care procedures (such as examining the baby or doing a procedure). Since CHEO is an outborn-only
hospital, the patients may even have a higher severity of illness beyond the first 12 hours of life compared to the CNN population. For the work in this thesis, four different CHEO data sets were used. The first two are referred to as the CHEO CDR Performance data sets, which were used to measure the performance of accessing the data stored in the CDR, and compare how the performance changed as the size of the CDR increased. The third and fourth are referred to as the CHEO Risk Estimation data sets, which were used to research and evaluate the mortality risk estimation models. The four data sets were taken from the same CDR, but at different periods of time as research progressed and more data were available.

3.1.2.1 CDR Performance Data Sets

The first CDR performance data set contained 5,447,737 data points (5,316,630 integer, 95,507 text, and 35,600 floating point) for 157 patients (Gilchrist et al., 2010). For one year, laboratory results and vital signs from patient monitors from 16 neonatal beds were stored in our CDR from an older system, which required manual connection of some devices to the data capture system. As a result, we did not capture all available data. Following the initial data collection period, new equipment was installed in the NICU, generating much more data for capture than the older system; all data were captured automatically. The second data set contained 16,690,455 data points (16,608,990 integer, 58,822 text, 22,643 floating point) with laboratory results and vital signs from patient monitors from 20 neonatal beds for 170 patients collected over a five month period (Gilchrist et al., 2011b).
3.1.2.2 Risk Estimation Data Sets

The first CHEO risk estimation data set is a snapshot of the CDR that was taken for the calendar year 2010. This data set contains 43 million data points from 357 patients with a 7.6% mortality rate, and 37.8% of patients were female. It was collected prospectively and automatically (Gilchrist et al., 2011a). Due to the differences between the CNN data (summary data only), and the CHEO data (real-time data collected over the entire patient stay), Table B.1 in Appendix B presents the median values for each of the 18 attributes during the first 12 hours after NICU admission, and also over the patient's entire NICU stay for the CHEO data set. Although all patients are continuously monitored throughout their NICU stay, the data set representing only the first 12 hours are missing patient monitoring data for 24 patients. This occurred because certain patient monitors were not yet configured to a specific patient, and thus the data from the monitors were marked as “unknown”, and not stored in the CDR.

The length of stay for the 357 patients in the NICU for 2010 was a minimum of 1 day, maximum of 193 days, with a median stay of 7 days ±14.0 Inter Quartile Range (IQR). While patients who survived and those who didn’t both had minimum stays of 1 day, patients who survived had a maximum stay of 193 days with a median of 7.0 ±13.3 IQR. Patients who died had a maximum stay of 176 days with a median of 8.0 ±16.5 IQR. Table B.2 in Appendix B shows the number of patients who had data for each attribute of interest, the percentage of patients who were missing those attributes, minimum, maximum, and median values for individual attributes. Since not all tests are ordered for every patient, that amount of data for each patient varies. The attributes with the lowest amount of missing data are vital signs reported
Automatically by the patient monitors, where fewer than 4% of patients are missing those attributes. Common neonatal blood tests such as serum glucose and sodium tests were not performed on approximately 10% of patients. The attribute with the most missing data is the white blood cell count laboratory test with 31.7% of patients missing the result.

The second data set adds the first three months of 2011 and contains 62 million data points from 568 patients with a 7% mortality rate. This new data from 2011 was used to compare the mortality models created with the 2010 data to see how well the risk estimation models performed with data from new patients.

3.2 Data Collection Systems at CHEO

Several data collection systems have been used in the NICU at CHEO since 2001. From 2001 to 2008, a Philips DocVue system that provided short-term clinical trends was installed. It stored ADT, laboratory results, and vital signs, but vital sign trends could only be stored for a maximum of 4 days, which is not sufficient for conducting research. The DocVue system used proprietary Serial Distribution Network (SDN) network cables and network switches. Due to the proprietary nature of the DocVue database, the data could not be directly retrieved for research analysis or for processing queries that were not built into the existing client software. The system was limited to applications provided by the original designers, since no new functionality could be added.

In 2008, the DocVue system was replaced by a Philips Intellivue Information Centre (PIC) system, capable of exporting patient data to an external system, which was used from 2008 to 2009. While the proprietary SDN network and patient mon-
itors remained due to high replacement costs, new SpO$_2$ monitors were connected to the main patient monitors to include their data in the collection system, which previously could not be captured after the old SpO$_2$ monitors were replaced. The PIC allowed the patient monitoring data of all patients to be viewed in real-time. The PIC then exported the data for each patient every minute to a CDR for long term storage and research analysis. Patient information and test results from ADT and laboratory systems were also sent to the CDR for storage. While the capture of ADT and laboratory data was automated, capturing patient monitor data was not, and relied on clinical staff connecting data capture cables, and manually entering patient information into the PIC.

In 2009, a new NICU unit was built in which new equipment provided by Draeger Medical was installed. This move also increased the number of beds in the NICU from 16 to 20 beds. The current data collection system has been operating since the move in 2009. The new patient monitors send data within the NICU’s network to a central station using industry standard cables and Transmission Control Protocol (TCP) and Internet Protocol (IP) networking protocols. The central station then broadcasts the data via industry standard HL7 messages to the Information Systems (IS) department’s MDLink HL7 processing server, which acts as a gateway for all HL7 messages, and then on to the CDR as seen in Figure 3.1. Data from the Laboratory and ADT systems are also sent to the MDLink server, and then on to the CDR. All patients are entered into the new central station as part of a routine process. This greatly increased the amount of data being collected for each patient, and also increased the accuracy, since the system records when a patient moves to a different
In June 2010, a new project at CHEO called eClinDoc was launched to move nursing notes and charting from paper to electronic storage. This would theoretically allow the charting data to be captured and stored in the CDR, and used as part of the CDS system to potentially improve the accuracy of risk estimation models.
Unfortunately, the eClinDoc project was not built using industry standard technology and protocols, but with a proprietary system for inputting, storing, and extracting data. After many discussions with the IS department, it was determined that the data could not be extracted and sent via HL7 to the CDR. It may be possible to generate custom reports to extract some of the data from eClinDoc, but this cannot be done in an automated way, and must be manually extracted and imported into the CDR, which makes using the data in real-time for a CDS system not feasible. Any changes to eClinDoc requires custom work to be completed by the vendor at great expense. While this potentially rich set of data is now being captured electronically, it is not possible for researchers to easily access or use the data on external systems.

Previous systems used proprietary networking hardware and protocols, and proprietary databases, which made accessing the data and upgrading the systems difficult. We need to develop systems that support industry standards and open source technologies that will provide the flexibility to upgrade components in the future and allow any software or system to access the data. More detailed information about the evolution of the data collection systems at CHEO along with data collection statistics for each system can be found in Appendix B.3. A number of lessons were learned from the integration process of the CDR with proprietary systems, and have been detailed in Appendix B.4.
Chapter 4

Methodology

4.1 Introduction

This chapter describes the methodology used for the research and development of a real-time CDR system capable of collecting, storing, and retrieving data from patient monitors, ADT, and laboratory results within in a NICU setting (Gilchrist et al., 2008, 2010, 2011b). It also outlines the approach for new neonatal mortality risk models that use summary data (data from admission until up to the first 72 hours after NICU admission) and real-time data segments (data broken down into time segments extracted from the entire duration of the patient’s NICU stay) stored in the CDR, with the ability to provide results in real-time (Gilchrist et al., 2011a).

4.2 Hybrid-EAV Storage Format

After experiencing the challenges and frustration of upgrading proprietary data collection systems (Appendix B.3.1), and not finding any affordable solutions that used
industry standard technology and protocols, it became clear that we needed to use a new approach. This section describes the design criteria of a new hybrid EAV storage format and the methodology to evaluate three important aspects of its performance: storage size, query speed, and query complexity.

4.2.1 Storage Format Design

To manage the large volumes of data we were collecting from the NICU, and be able to access the data in real-time, it was essential to have an optimised CDR storage structure. During the course of a two year period (2010 and 2011) there were 864 unique physiological and laboratory result attributes being used in the NICU. After the first two months of 2010, there were 308 unique attributes with an average of 25 new attributes being added every month. With such a high degree of attribute instability, a storage format that accommodates the fast changing medical domain was required.

The EAV storage format is commonly used in the medical domain, because the design allows the addition of new attributes without having to change the physical database schema, which can be time consuming and costly to update, and test the database and applications that use the database. After investigating the limitations of the commonly used simple EAV and multi-data type EAV formats described in Section 2.1.3, we set out find a way to combine the advantages of both types in a single new EAV format. The advantage of the simple EAV format is that it only uses one table to store the data, so it is simple to access, and does not require searching for or retrieving data from multiple tables, which will incur a performance penalty. The advantage of the multi-data type EAV format is that data do not need to be
converted to do comparisons or mathematical operations since they are already stored in the correct data type.

For the new hybrid EAV format to achieve the advantages of both common formats, it must allow the use of multiple data types in a single table to provide simpler and faster queries, and provide the ability to use multiple data types without having to first convert the data.

4.2.2 Measures of Performance

We hypothesised that our novel hybrid EAV design that allows the use of multiple data types in a single table, would provide simpler and faster queries. The performance of the three EAV formats (hybrid, simple, and multi-data type) was evaluated using queries of millions of clinical data points of real patients captured from patient monitors and other systems in the NICU. The performance was empirically compared using the two CDR Performance data sets (5.4M and 16.7M data points) collected at different times in the NICU at CHEO. Details of these data sets can be found in Section 3.1.2.

4.2.2.1 Storage Size

Storage size is an important performance measure for a CDR storage format. Since large quantities of data will be stored, it is important to see if the amount of storage space required for the hybrid EAV design differs greatly from the others or if it remains efficient as the database grows in size. Storage performance was measured by taking all data points in the CDR (for each data set) and storing them in a separate database for each of the three EAV formats. The size in megabytes of the resulting database's
storage space for each EAV format was noted.

4.2.2.2 Query Speed

Query speed is very important in a real-time environment. If queries to the CDR are too slow to provide data in real-time, it does not matter how efficient the storage size is, the storage format will not be useful. To evaluate the speed with which the queries could return results, we looked at three different scenarios that are relevant to medical data retrieval. All queries were performed three times, and the average speeds were used as results. Query caching on the CDR database was disabled, so that the second and third queries would require full searches instead of just retrieving the results from the cache.

Querying all data for a single patient

To build models that estimate the risk of clinical events, it is important to extract all the data stored for individual patients in their original data types. The speed of querying all data points for one patient was measured for each of the EAV formats. From the 5.4M data point set, the sample patient (Patient A) queried had 282,584 data points including integer, text and floating point values. Of the 16.7M data points in the second data set, the sample patient (Patient B) queried had 296,829 data points including integer, text and floating point values. For use in real-time CDS systems, this test is the most important for gauging how fast a risk estimation can be performed for each patient. This test indicates the minimum amount of time required to provide data to the model before it can calculate a risk estimation.
Calculating pulse pressure

An example of a clinical use of a CDR would be to calculate the pulse pressure, that is, the numeric difference between systolic blood pressure and diastolic blood pressure (Pastor-Barriuso et al., 2003). Widened pulse pressure can be a predictor of heart problems especially in older adults and premature babies (Glynn et al., 2000; Ellison et al., 1983). An example in premature neonates is the presence of a Patent Ductus Arteriosus (PDA) (Ellison et al., 1983). This involves the patency of a vessel outside of the heart that remains open during fetal life to send blood away from the fluid filled lungs, but should close shortly after birth when the lungs fill with air and the vessel is no longer needed. In preterm infants, this vessel can remain open, leading to congestive heart failure. It is difficult to determine whether a PDA is the reason for the deteriorating condition of a preterm infant based on clinical signs alone. A widening of the pulse pressure measurement is one early clinical sign that helps clinicians relate the deterioration to PDA. A definitive diagnosis is made by an echocardiogram of the heart, however, this requires highly trained personnel and special equipment, and is often difficult to obtain quickly. Timely diagnosis of this condition leads to more effective resource utilisation and optimal timing of treatment.

The speed of querying all systolic and diastolic blood pressure data from the CDR, and then calculating the pulse pressure was measured for each of the EAV formats being tested. Of the 5.4M data points in the first data set, there were 16,956 blood pressure data points stored from the patient monitors for all patients. Of the 16.7M data points in the second data set, there were 3,239,316 blood pressure data points stored from the patient monitors for all patients.
Finding patients with respiratory decompensation

Premature infants or those born with infection or diseases that affect the lung often need mechanical ventilation to ensure adequate lung oxygen delivery and waste product (carbon dioxide) removal. When monitoring an infant on mechanical ventilation, multiple variables are tracked to ensure adequate support and to determine when changes are needed. As respiratory decompensation escalates, PCO$_2$ builds up in the blood, lowering blood pH. Bicarbonate (HCO$_3^-$) rises, and these changes can be measured by sampling the blood from an artery or capillary (Taeusch et al., 2005). When a critical amount of carbon dioxide builds up to lower the pH beyond a threshold level, the physician must increase the ventilator support to prevent respiratory failure or arrest. These thresholds can vary slightly for each baby, but typically would be at a PCO$_2$ greater than 60 mmHg and an HCO$_3^-$ greater than 19 mmHg that has resulted in a pH less than 7.25.

The speed of querying all patients to find the ones with respiratory decompensation was measured for each of the EAV formats being tested. In the 5.4M data point CDR, there were 2023 data points each of PCO$_2$, HCO$_3^-$, and pH. In the 16.7M data point CDR, there were 1272 data points each of PCO$_2$, HCO$_3^-$, and pH. This type of query uses “AND” conditions between attributes in the same and different database tables. For the simple and hybrid formats, which use only one table, the “AND” condition is compared for PCO$_2$ and pH floating point attributes and HCO$_3^-$ integer attributes within the same table. The multi-data type format contains several tables therefore the “AND” condition compares attributes between the integer and floating point tables. This type of query provides a good measure of comparing speed efficiency between EAV formats, since the intersection of multiple queries needs to be
completed.

4.2.2.3 Query Complexity

The design of a storage format can affect the complexity of the query to retrieve data. Storage formats that use multiple tables for example, tend to have more complex queries since data must be retrieved from more than one table to provide the requested results. More complex queries are more difficult to write and can also take longer to execute. Evaluating the complexity of all three EAV formats is important to ensure the new hybrid EAV format does not introduce unnecessary complexity in the design.

Data are extracted from the CDR using the SQL (ANSI/ISO/IEC 9005-2003, 2003), which is supported by most of the popular databases available including MySQL that was used for these tests. Sometimes researchers and clinicians need to query data from patient physiological data or laboratory results that fall within a range of values. An example would be searching for all patients in the NICU with a respiration rate of less than 40 or greater than 60 breaths per minute, or who have hypothermic temperatures between 32.0 and 36.4 degrees Celsius. A query to return such results was created for the three EAV formats to compare the complexity measured in terms of the number of query elements. We considered a simple query like extracting all data for a single patient, as well as more complex queries looking for patients with attribute values within a specified range, where any match would suffice (OR), and where all conditions needed to match (AND).
CHAPTER 4. METHODOLOGY

4.3 Real-Time CDR

Current clinical central stations are designed for clinical use to see trends, and not for research purposes. Therefore, they only store physiological data for short periods of time, typically for several days. Clinical information in hospitals is usually stored in different locations. At CHEO, the physiological data is stored in the NICU, the laboratory results are stored on a different system in another section of the hospital. Admissions and discharge information are stored on a third system, and patient discharge diagnoses are stored in a fourth system in the Health Records department. Accessing all the required data from four different systems that use equipment from four different vendors is challenging. Fast access to clinical data is necessary to perform risk estimations of medical events in real-time. A new system is required to store enough data to analyse and create models, and to be able to access the data in real-time to detect medical events as they occur, and deliver intelligent alerts. With only clinical trends available, it is not possible to achieve these research goals.

This section describes the design criteria of a new real-time CDR. With an efficient hybrid EAV storage format as a foundation, research was conducted for a new CDR design that is robust and flexible enough to store real-time data from patient monitors, ADT, and laboratory results in a NICU setting for a period of years using a standard desktop PC without the need of any specialised commercial software. The design needed to support receiving multiple connections from different sources simultaneously to communicate with different medical data sources from multiple vendors in real-time on a continuous basis. It was also imperative that the CDR design be compatible with industry standards to support receiving data from, and making data available to the many different vendors used in the medical domain. This will al-
low for a simpler upgrade process if new equipment or software is purchased by the hospital, unlike our experience using proprietary systems (see Appendix B).

4.3.1 Data Collection

The first important function of a CDR is to collect data from various medical systems in the hospital. The most common way for medical devices to communicate is using the HL7 industry standard for messaging (HL7, 2009). HL7 specifies formatting of messages using components in a defined sequence. For example, an Observation Result (ORU) message can be sent containing observations from physiological monitors or clinical laboratory results. At CHEO, the patient monitors in the NICU, the ADT, and laboratory systems all support sending data using HL7 messages. The Draeger patient monitors also support a proprietary method for transmitting data, but if this method were to be implemented, then a change in hardware vendors in the future would require that the data collection interface for the CDR be completely re-written to support the new hardware. Instead, an interface supporting the HL7 standard should be used. This ensures that any application or system that conformed to the HL7 standard could communicate with the CDR with minimal or no changes.

4.3.2 Data Storage

After data are received at the CDR, the data needs to be efficiently stored for later access and retrieval. Large amounts of data are typically stored in a database, and managed through a relational DBMS such as MySQL, Oracle, SQL Server, or Sybase. While the HL7 standard exists for communication between medical devices, a standard called SQL (ANSI/ISO/IEC 9005-2003, 2003) exists for communication with
databases. This query language allows a standardised way to store data in a database, and also retrieve data from a database. The CDR should use a DBMS that is compatible with the SQL standard. This would ensure that any system or software supporting SQL can be used to access data in the CDR, either locally or remotely over a network. By conforming to the SQL standard, it also allows the CDR to be upgraded or modified to use a different DBMS without having to modify any applications that use the CDR.

After data have been submitted to a DBMS using SQL, the data must be stored using a specified format. To obtain the real-time efficiency required for processing large amounts of medical data, the hybrid EAV format we developed should be integrated into the DBMS.

4.3.3 Patient Privacy

Research Ethics Boards (REBs) take patient privacy issues very seriously. Before access is given to patient data, researchers must first prove to the REB that they have a plan to protect privacy. When dealing with electronic patient data, this usually means de-identifying the data, so even if lost or stolen, the data cannot be attributed to a specific patient. The current proprietary clinical information systems are meant to work inside the clinical environment, and so do not usually have the ability to de-identify patient information. Maintaining patient privacy is critical for researchers to be able to access medical data. With these types of systems, exported data must be sanitised manually or with a separate system to remove patient identifiers and personal information. This can be a time consuming extra step that the researchers must perform before they can start working with the data or removing it from the
clinical environment. It also opens the possibility to mistakes being made, and the data not being properly de-identified.

To minimise the chance of private patient information being exposed, and prevent privacy laws from being breached, the CDR should have a facility built-in that automatically protects patient privacy without any further processing or input required from the users. This would also have the added benefit of providing researchers and physicians immediate access to the real-time data, as it would be available in an already protected form. An REB approved de-identification process can be used for the automated process in the CDR, which means that all users will be using the same standardised method.

4.3.4 Data Retrieval

Now that medical data has been collected and stored in the CDR, it must also be easily retrievable for researchers to select the data sets they need to complete their studies. Since the CDR design uses the SQL standard for storing and retrieving data, any SQL compliant software can be used to extract data from the CDR. Many commercial and free open-source tools such as HeidiSQL (Ansgar Becker, 2012) provide powerful and easy to use interfaces to extract and analyse data in SQL compliant systems. Data could also be retrieved directly from inside a user built research application such as a CDS system that requires real-time data updates. If a data mining tool such as a DT or ANN does not support the SQL standard directly, a tool such as HeidiSQL can be used to construct SQL queries that communicate with the CDR to extract and export data in any required format such as a comma separated flat file that can then be imported into most types of systems. The CDR design should allow the extraction
of data using different strategies to limit the amount of exported information for large data sets.

4.3.4.1 Patient Complete Stays

For testing and training of models to estimate the risk of clinical outcomes, the design should allow for the ability to only extract data for patients who have complete stays (i.e., they are not currently admitted to the unit, where data is being collected).

4.3.4.2 Time Spans

The CDR should have the ability to only extract data before a specific date, after a specific date, or between specific dates. This feature is important when training models to extract data from specific time periods. It is also useful for real-time models to get the latest data for a patient from the CDR, such as data for a patient in the last 12 hours, without having to access the data from their entire stay. For example, this would allow the extraction of all data for patient X, only for a 2 month period starting at date Y. It should also allow for the extraction of data for patient X in the last 5 minutes to feed the most recent results for patient X to a CDS system.

4.3.4.3 Data Frequency

Using the support for time spans, it should be possible to specify the data frequency for the attribute being extracted. If for example heart rate data is collected once every minute, but the analysis only requires data hourly, the design should allow extraction of the data entries that are greater than or equal to 60 minutes apart from the last entry found. It should also be possible to find the average value within each
CHAPTER 4. METHODOLOGY

60 minute time span and extract that value if an average is desired.

4.3.4.4 Meta-Data Descriptions

Numerical attribute values are sometimes used in databases for fast machine processing, but make it difficult for human readability. The CDR design should support the ability to add text names and descriptions, so that humans can more easily read and understand the data content.

4.4 Development of New Neonatal Mortality Risk Models

While neonatal risk estimation models such as SNAP (Richardson et al., 1993, 2001), CRIB (Cockburn et al., 1993), and NTISS (Gray et al., 1992) use scores to rank patients based on severity of illness, they are not designed for specific units or regions. They are not accurate enough to estimate outcomes for single patients, and cannot be easily updated to reflect new practices (Frize et al., 1995, 2006). Physicians and the medical team need more patient-specific information to help make treatment decisions. One of the limitations of current neonatal mortality risk estimation models that can estimate outcomes for single patients is that they rely on static historical data from only the first 12 hours of life (Townsend and Frize, 2008). A new type of real-time risk estimation model that generates patient-specific risk estimations on medical outcomes would provide physicians with the potential to improve patient outcomes. This new approach could use up-to-date patient data, as it occurs, to better evaluate the current state of the patient being monitored and changes in that
CHAPTER 4. METHODOLOGY

state when it occurs. By analysing the data in real-time, the risk estimation models are expected to provide the unique ability to detect clinical events as they happen and deliver immediate feedback to physicians.

We have researched methods for developing new models to estimate mortality risk using summary data and real-time data segments from a single, outborn-only NICU. The summary data models used data from admission until up to the first 72 hours after NICU admission, and the real-time models used data broken down into time segments extracted from the entire duration of the patient’s NICU stay. We also developed a baseline model using CNN summary data from the first 12 hours after NICU admission. We compared the mortality models to find the best mortality risk estimation, that is, the one that has a minimal number of false positives. We also identified the attributes that were most often included in the best mortality risk estimation models.

4.4.1 Data Sets

We used the 18 SNAP attributes that were available to us electronically from two different data sets for this work: (1) summary data from the first 12 hours following NICU admission collected by the CNN from multiple NICUs, and (2) a data set of real-time medical measurements collected from CHEO, a single outborn-only NICU. The list of attributes can be seen in Table 4.1. More details of these data sets can be found in Section 3.1.

One attribute that is not being used for mortality risk estimation is birth weight. Although it has been found to be a strong predictor for neonatal mortality, the correlation to mortality is so strong that it was not used in the SNAP (Richardson et al.,
CHAPTER 4. METHODOLOGY

Table 4.1: List of 18 attributes used in Mortality models

<table>
<thead>
<tr>
<th>Attribute List</th>
</tr>
</thead>
<tbody>
<tr>
<td>lowest mean blood pressure</td>
</tr>
<tr>
<td>lowest heart rate</td>
</tr>
<tr>
<td>lowest serum glucose</td>
</tr>
<tr>
<td>lowest serum sodium</td>
</tr>
<tr>
<td>lowest pO2 level</td>
</tr>
<tr>
<td>lowest serum pH</td>
</tr>
<tr>
<td>lowest blood hematocrit</td>
</tr>
<tr>
<td>lowest absolute neutrophil count</td>
</tr>
<tr>
<td>lowest platelet count</td>
</tr>
</tbody>
</table>

1993) and SNAP-II (Richardson et al., 2001) scoring system in order to find other predictors of mortality. Richardson et al. found that, “The association between illness severity and in-hospital mortality is independent of, and additive to, the mortality risks inherent in birth weight alone” (Richardson et al., 1993). By not including birth weight, we hope to discover other attributes which may be important in estimating the risk of mortality that may otherwise be missed. Once mortality models are created and tested without birth weight, adding the attribute in the future could be considered to see if results are improved.

4.4.2 Data Preparation

Missing values are common in medical data. Few patients have every test performed, so a number of laboratory results are missing in many patient records. In consultation
CHAPTER 4. METHODOLOGY

with our clinical partners, patient records missing more than 50% of the attributes were discarded. Approximately 15% of patients in the CNN data set were excluded from consideration because they were missing >50% of the attributes we were using for model development. In the CHEO data set, vital signs are recorded every minute, whereas some lab results are only recorded once or twice a day. Depending on the time segment used with the CHEO data set, more patients may be excluded due to missing data if no laboratory tests were recorded during the specified time frame. By excluding patients who have fewer measurements, we are focusing our work on a sicker patient population (Ennett et al., 2001).

When using data extracted directly from medical measurement devices, artifacts can occur. For example, if a patient is moved while being examined, the patient’s heart rate and/or respiratory rate may increase due to the stress caused by the interaction, which may not accurately represent the patient’s condition at that instant. To adjust for this factor, when developing the risk estimation models using the CHEO risk estimation data sets, we used the median of the top or bottom five values recorded for vital signs for the highest or lowest attribute measurements, respectively. A heart rate of 0 may occur when an electrocardiogram lead moves or falls off. Our clinical partners advised that we classify heart rate measurements less than 30 or greater than 340 beats per minute, and respiratory rate measurements greater than 130 respirations per minute as artifacts and discard them.

To obtain more reliable mortality risk estimates, the 5-by-2 cross validation technique was used to validate the models. The data were first separated into mortality and non-mortality data sets (step 1). These sets were then randomly split into equal sized training and test sets, with the first mortality and non-mortality sets merged for
CHAPTER 4. METHODOLOGY

training, and the second sets merged for testing (step 2). This step ensures that there are an equal number of mortality and non-mortality cases in the test and training sets. This resulting test and training set is labelled Set #1. This final test and training set #1 are then reversed, so that the test set is used for training and the training set for testing, and labelled Set #2 (step 3). This process is completed a total of five times, to create ten separate training and testing sets (Sets #1 through #10), and the average of all results was recorded. Advantages of the 5-by-2 cross validation method are large training and test sets, and using all data points for both training and testing, so important patient records that lead to better risk estimations are not left out (Dietterich, 1998). Figure 4.1 illustrates the 5-by-2 cross validation process. When developing and testing the models using the time segment data sets, we kept all data from a single patient together to prevent having data from one patient in both the training and test sets at the same time, which would affect the model's classification performance.

4.4.3 Risk Estimation Models

To develop the neonatal mortality risk estimation models, we used C5.0, a data mining software tool that can generate decision trees and use them to make risk estimations (Rulequest Research, 2009a). C5.0 was chosen because it is very fast and we are looking to achieve real-time performance from our models. It is available as free open source software, and provides an Application Programming Interface (API) that allows easy integration of models created with C5.0 into any software, which makes it ideal for using with a CDS. C5.0 handles missing values automatically, which is important when dealing with real-time data streams, as imputations of missing values
would otherwise have to be calculated on the fly before feeding the data to the model. The decision trees produced by C5.0 give human readable results that are easy for clinicians to understand. Decision trees were generated using C5.0 with the option to use boosting, where 10 DTs are created and vote to decide the classification of a
CHAPTER 4. METHODOLOGY

given case. The performance results were generated by classifying the data in the test data set, which had not been previously seen by C5.0.

To compare the performance of the C5.0 mortality models to at least one other classification method, the best overall mortality risk estimation model produced by C5.0 is compared to an ANN model, created using the same data. We chose the ANN research framework developed by MIRG (Ennett et al., 2004; Rybchynski, 2005) that uses a MLP feedforward network with backpropagation. It exhaustively modifies the training parameters in an automated fashion to find the optimal ANN structure in networks ranging from zero to $2n + 1$ hidden nodes, where $n$ is the number of inputs. Missing values must be replaced when using ANNs, therefore, they were imputed using a hybrid ANN-CBR approach, which uses mean value of the k-nearest neighbours of the patient case (Ennett, 2003; Ennett and Frize, 2003; Ennett et al., 2008). The MIRG ANN classification tool was developed using MatLab code, which requires a commercial license to use. Since the tool does not automatically handle missing data, a separate system needs to be devised and integrated to perform high quality imputations. With the k-nearest neighbours approach, a large amount of data for previous patients needs to be kept and searched to find the relevant matching cases, and perform the imputation before the data can be used by the ANN model. These issues make it more challenging to integrate the tool into an automated real-time CDS environment.

Fifteen neonatal mortality risk estimation models were developed in three major categories: (1) baseline, (2) summary data, and (3) real-time. The baseline model uses CNN data from the first 12 hours of NICU admission. The summary data models use the CHEO data sets with data from the first 12 hours of NICU admission up to
The first 72 hours of admission. Finally, the real-time models use the CHEO data sets with data from the entire patient stay, broken down into segments of varying time periods in length from 12 hours to 48 hours. A summary of the 15 models can be found in Table 4.2 which use the 18 attributes listed in Table 4.1 (for further details see Table B.1).

### Baseline Risk Estimation Model with CNN Data

Previously, MIRG created a neonatal mortality risk estimation model using the CNN data set with 27 of the SNAP attributes (Townsend and Frize, 2008). To see if a reduced set of attributes could still predict mortality with clinically useful results, a C5.0 decision tree model was developed using only the 18 attributes in Table 4.1. We used this model developed with CNN summary data as a baseline to compare with the performance of the models developed using summary data and time segment data.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Model Type</th>
<th>Time Frame</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN</td>
<td>Baseline</td>
<td>First 12 hours</td>
<td>18</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary</td>
<td>First 12 hours</td>
<td>18</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary</td>
<td>First 24 hours</td>
<td>18</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary</td>
<td>First 48 hours</td>
<td>18</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary</td>
<td>First 72 hours</td>
<td>18 + Gender</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary</td>
<td>First 48 hours</td>
<td>18 + CO₂</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary</td>
<td>First 48 hours</td>
<td>18 + Std Dev</td>
</tr>
<tr>
<td>CHEO 2011</td>
<td>Summary</td>
<td>First 48 hours</td>
<td>18</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>12-hour segments</td>
<td>Entire stay</td>
<td>18</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>24-hour segments</td>
<td>Entire stay</td>
<td>18</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>48-hour segments</td>
<td>Entire stay</td>
<td>18</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>48-hour segments</td>
<td>Entire stay</td>
<td>18 (stable, unstable)</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>48-hour segments</td>
<td>Entire stay</td>
<td>18 (stable+non-stable, unstable)</td>
</tr>
<tr>
<td>CHEO 2011</td>
<td>48-hour segments</td>
<td>Entire stay</td>
<td>18</td>
</tr>
</tbody>
</table>
CHAPTER 4. METHODOLOGY

from CHEO. Our intention was that the CHEO models should be at least as good as the CNN model.

4.4.3.2 Summary Data Risk Estimation Models with CHEO Data

For the summary data models developed using the CHEO data, one record for each patient was created that summarised the attributes over the specified time period. We looked at time periods from NICU admission to 12, 24, 48, and 72 hours after admission. We wanted to see how well the 18 SNAP attributes could predict mortality over longer time periods from admission. We developed four separate C5.0 decision tree models with these summary data of the attributes listed in Table 4.1. We compared the models to see which one produced the best mortality risk estimation, and which attributes were used in the decision tree most often. The CHEO summary data models were also compared to the CNN baseline risk estimation model to see if there are any changes in the risk estimation accuracy.

After comparing these models, we created three more mortality models using attributes not included in the 18 SNAP attributes that were previously used. In addition to the 18 SNAP attributes, a model was created using the gender of the patient, as our physician partners noted that female and male patients tend to have different outcomes. A second model was created using the lowest Carbon Dioxide (CO$_2$) lab result, which our physician partners thought might have an impact on mortality. Finally, a third model was created adding the standard deviation of six SNAP attributes, which measured both a highest and lowest value to see if the amount of variation of the data during the summary period might lead to better risk estimation. Those six attributes were mean blood pressure, heart rate, respiration rate, blood
CHAPTER 4. METHODOLOGY

Summary Data (First 12, 24, 48, 72 hours)

<table>
<thead>
<tr>
<th>Patient #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient #2 (Removed, &gt;50% missing data)</td>
</tr>
<tr>
<td>Patient #3</td>
</tr>
<tr>
<td>Patient #4 (Removed, &gt;50% missing data)</td>
</tr>
<tr>
<td>Patient #5</td>
</tr>
<tr>
<td>Patient #6</td>
</tr>
</tbody>
</table>

Time Segment Data (12, 24, 48 hours)

<table>
<thead>
<tr>
<th>Patient #1 (Time Seg. 1)</th>
<th>Patient #1 (Time Seg. 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient #2 (Removed, &gt;50% missing data)</td>
<td></td>
</tr>
<tr>
<td>Patient #3 (Time Seg. 1)</td>
<td>Patient #3 (Time Seg. 2)</td>
</tr>
<tr>
<td>Patient #3 (Time Seg. 3)</td>
<td>Patient #4 (Time Seg. 4)</td>
</tr>
<tr>
<td>Patient #5 (Time Seg. 1)</td>
<td>Patient #5 (Time Seg. 2)</td>
</tr>
<tr>
<td>Time Seg. 3 (Removed)</td>
<td>Time Seg. 3 (Removed)</td>
</tr>
<tr>
<td>Patient #4 (Time Seg. 4)</td>
<td></td>
</tr>
</tbody>
</table>

Figure 4.2: Sample diagram of Summary and Time Segment data sets.

hematocrit, serum glucose, and serum sodium.

4.4.3.3 Real-Time Risk Estimation Models with CHEO Data

The real-time models used time segments over the patient’s entire NICU stay at CHEO. For example, when we used a 12-hour time segment, one record for each time segment period was created with the attributes for each patient. If a patient was in the NICU for 30 hours, the patient has 3 associated data records: (1) hours 0 to 12; (2) hours 12 to 24; and (3) hours 24 to 30. The number of records per patient depended on the length of time the patient spent in the unit. Figure 4.2 shows a sample diagram of how the summary and time segment data sets were created. The length of stay for patients who survived ranged from 1 to 193 days with a median ± IQR stay of 7.0 ± 13.3 days, while the patients who died ranged from 1 to 176 days with a median ± IQR stay of 8.0 ± 16.5 days.
CHAPTER 4. METHODOLOGY

We used the C5.0 algorithm to create the real-time mortality risk estimation models for three different length time segments: 12, 24, and 48 hours. To account for potential artifacts for these models, we calculated the median of the top 5 highest and/or lowest values of the vital signs (heart rate, mean blood pressure, and respiratory rate) over each of the time segments. We only used the highest and/or lowest lab values for each patient during the time segment.

We then created two more mortality models to see if splitting the patients into groups of stable (survivors), non-stable (mortality patient data up to the last 48 hours of admission), and unstable (mortality patients only the last 48 hours of admission) could improve results. These tests may also show how important data before the last 48 hours of admission are for estimating mortality risk. The first model used just two categories: stable, and unstable patients, with the non-stable time segments discarded. The second model used all three categories, with time segments from stable and non-stable patients combined together as survivor cases, and time segments from unstable patients as the mortality cases.

4.4.3.4 CHEO 2010 vs 2011 Data Sets

To see how well the mortality risk estimation model works with future data, the CHEO 2010 data set was used to train a C5.0 decision tree mortality model. This time the entire 2010 data set was used for training, while the first three months of 2011 were used as a test set to simulate new patients arriving in the future. The original 2011 data set had only 1 mortality case and 75 survival cases resulting in a 1.3% mortality rate, which is considerably lower than the 2010 data set of 7.6%. For summary data models, to balance out the data set, 7 mortality cases and 35 non-mortality cases
from the end of 2010 were moved to the 2011 data set, which resulted in a 6.8% mortality rate in both data sets. The modified 2010 data set contained \( N=266 \) (18 mortality) cases and the modified 2011 data set contained \( N=118 \) (8 mortality) cases. To make a fair comparison with the real-time data sets, time segments from the same 7 mortality cases and 35 non-mortality cases from the end of 2010 were moved to the 2011 real-time data set. The modified real-time 2010 data set contained \( N=1,133 \) (112 mortality) time segments and the modified real-time 2011 data set contained \( N=444 \) (36 mortality) time segments.

The summary data time span and time segment size that produced the best overall mortality risk estimation models generated by C5.0 for both the summary data model and the real-time data model will be used for this comparison.

4.4.4 Measures of Performance

Our clinical partners feel that clinically useful mortality risk estimations have greater than 90% specificity and a sensitivity of at least 60%. Sensitivity, specificity, Positive Predictive Value (PPV), and Negative Predictive Value (NPV) are usual determinants of performance, especially for clinicians. To this we add the F1 score and Matthews Correlation Coefficient (MCC). The F1 score is a measure of the risk estimation's accuracy that uses both precision (positive predictive value) and recall (sensitivity) to calculate the score, where 0 is the worst score, and 1 is the best (Chinchor, 1992). It is defined as the harmonic mean of precision (P) and recall (R). MCC measures the quality of binary classifications. It combines the sensitivity and specificity into a single measure that is balanced even when the classes are of different sizes (Baldi et al., 2000). MCC provides a correlation coefficient between the predicted and observed
classifications, and returns a value between -1 and 1, with -1 being an inverse risk estimation, 0 a random risk estimation, 1 a perfect risk estimation. Measures such as accuracy are not useful if the two classes differ greatly in size such as with patient mortality data.

A confusion matrix created from the results of the classification process contains the number of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) values. Figure 4.3 shows a diagram of a confusion matrix for mortality, which is used to calculate the various measures of performance listed in equations 4.1 through 4.7.
CHAPTER 4. METHODOLOGY

Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (4.1)

\text{Sensitivity} = R = \frac{TP}{TP + FN} \quad (4.2)

\text{Specificity} = \frac{TN}{TN + FP} \quad (4.3)

PPV = P = \frac{TP}{TP + FP} \quad (4.4)

NPV = \frac{TN}{TN + FN} \quad (4.5)

\text{MCC} = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (4.6)

F1 = \frac{2 \times P \times R}{P + R} \quad (4.7)

The PPV is the proportion of positive risk estimations that are correctly predicted. This measure reflects the probability that a positive risk estimation will give the correct condition being tested for. The NPV is the proportion of negative risk estimations that are correctly predicted. These values are directly proportional to the prevalence of the condition being tested (Altman and Bland, 1994). The predictive values depend heavily on the prevalence of the condition being tested. The lower the prevalence of the condition, the more likely a negative risk estimation will be correct and the less likely a positive risk estimation will be correct. PPV and NPV values found in one data set will not apply universally. For a condition with very low prevalence, the PPV will not approach 1 even if specificity and sensitivity are high, leading to many positive risk estimations that are false positives. Due to the dependence on prevalence, PPV and NPV are only useful if the ratio of the number of patients in one class and the number of patients in the other class is equivalent.
to the prevalence of the condition being tested. For example, if the condition being tested is mortality, the ratio of patients who survive and die in the data set should be equivalent to prevalence of mortality for that population.

4.4.5 Imbalanced Data Sets

An imbalanced data set contains one outcome that is less prevalent than any other outcome. Mortality is a rare outcome in the NICU. As a result, the data set contains a much smaller number of mortality cases compared to survivors, which leads to an imbalanced data set. Previous research has shown that imbalanced data can make it more challenging for pattern classification systems to classify the less prevalent outcome, which is often the outcome of interest (Japkowicz, 2000; Ennett et al., 2001; Chawla et al., 2004; Visa and Ralescu, 2005). Previously, MIRG looked at using random re-sampling of the rare outcome until an artificial level of 20% was achieved to help deal with an imbalanced data set when using ANNs (Ennett and Frize, 2000). We completed some additional research to determine if over-sampling, under-sampling, or a combination of both is beneficial with decision trees for two different outcomes: mortality, and IVH.

Using mortality data from the CNN database, we compared random over-sampling, random under-sampling, and a combination of both with the C5.0 decision tree algorithm. The database contained 5% cases of patients who died ($N=662$) with the remaining 95% of patients who survived ($N=12,937$). The data was split, so that two thirds of mortality and non-mortality cases were placed in a training data set and the other one third of cases were placed in a test data set. The training data set contained 441 mortality cases and 8,642 non-mortality cases to keep the same
CHAPTER 4. METHODOLOGY

5% ratio. The data in the test set is never seen by the C5.0 algorithm while it is generating a decision tree.

An imbalanced data set from the CNN database containing patients with IVH was also compared with C5.0 using the same three methods as with mortality. The database contained 4% cases of patients with IVH \((N=506)\) with the remaining 96% of patients without IVH \((N=13,078)\). The data was split, so two thirds of IVH and non-IVH cases were placed in a training set. The training data set contained 337 IVH cases and 8,718 non-IVH cases. The training data was used to create random under-sampled data at 10%, 20%, 30%, 40%, and 50% of the less prevalent outcome, random over-sampled data at 10%, 20%, 30%, 40%, and 50% of the less prevalent outcome, and a combination of random under-sampled and random over-sampled data at 10%, 20%, 30%, 40%, and 50% of the less prevalent outcome. Decision trees were generated using C5.0 with the option to use boosting, where 10 DTs are created and vote to decide the classification of a given case. The performance results were generated by classifying the data in the test data set, which had not been previously seen by C5.0.

4.4.6 Risk Categories

When the C5.0 DT algorithm classifies a case using one of the mortality risk estimation models, it provides a classification (mortality, non-mortality) and also a probability of how confident it was at making that classification. Our clinical partners have specified that they prefer to see a risk category such as low, moderate, and high instead of a probability value. To use risk categories, thresholds must be chosen to separate probabilities to determine below which value is low, and above which
value is high.

Research was conducted to see what kind of result could be obtained when using various threshold values. The C5.0 DT mortality model using boosting and 10% over-sampled data created from the same CNN data set in Section 4.4.5 was used for these tests. The test set contained 221 mortality cases and 4,313 non-mortality (survivor) cases. The DT model was used to classify each of the 4,534 cases. The classification and the confidence probability were recorded. Each result was then assigned to the category low, moderate, or high. The experiment was repeated four times to evaluate four different confidence probability thresholds used for the categories:

1. Low $\leq$ 10% and High $\geq$ 90%
2. Low $\leq$ 20% and High $\geq$ 80%
3. Low $\leq$ 30% and High $\geq$ 70%
4. Low $\leq$ 40% and High $\geq$ 60%

4.4.7 Usage of Neonatal Risk Models with Real-Time Data

The neonatal mortality risk models will generate patient-specific risk estimations for physicians with the potential to improve patient outcomes. For the model to estimate risk in real-time, data for the specific patient must first be retrieved from the CDR. The data are then used by the risk estimation model to provide the physician with the mortality risk estimation of their patient using the latest data available. All of these steps must be completed within a time frame of no more than one minute to be considered real-time.
CHAPTER 4. METHODOLOGY

The mortality risk estimation model requires the most up-to-date data for a patient to provide an estimate of their current condition related to mortality risk. The CDR database was designed to be very fast to extract all data for an individual patient. Since the interface for accessing data in the CDR uses the industry standard SQL query language, an SQL query can be constructed to request the data required for Patient X. The CDR should be able to return the specific attributes required for the mortality model for Patient X in real-time. These data are then used by the mortality model to estimate the risk of mortality for that patient. To determine if data can be retrieved and used by the mortality model in real-time, three patients (Patients X, Y, and Z) were randomly selected from the CHEO 2011 risk estimation data set, which contained a total of 62 million data points. To simulate retrieving data for one type of mortality model, we chose to use the 48-hour summary, and 48-hour time segment types, starting 48 hours before the data collection stopped for that patient. Patient X had a total of 344,361 data points with 22,253 data points in the 48-hour time range used. Patient Y had a total of 58,398 data points with 19,527 in the 48-hour range, and Patient Z had 153,810 total data points with 13,307 points in the 48-hour time range.

An SQL query that collects the required 18 SNAP variables (see Table 4.1) for the mortality model was constructed to extract the data for a single patient during the specified 48-hour period. The SQL query contains 48 components to find the maximum or minimum values of the SNAP attributes such as blood pressure, and also extract values that are required to calculate certain attributes such as the highest immature/total neutrophil (HITN) ratio. For example, to calculate this ratio, values for neutro bands, neutro polys, metamyelocytes, and myelocytes must all be extracted.
from the same laboratory report and then used in equation 4.8:

\[
HTN_{ratio} = \frac{\text{neutrobands} + \text{metamyelocytes} + \text{myelocytes}}{\text{neutrobands} + \text{metamyelocytes} + \text{myelocytes} + \text{neutropolys}} \tag{4.8}
\]

This query was run three times for each patient and the average execution time of the three runs, along with the standard deviation, were recorded. Query caching for the MySQL database was disabled, so that all three runs would be independent and not use any cached information. The data retrieved from the query was then used as input to the mortality risk estimation DT model to measure the time it takes to classify the data and return a confidence probability. The risk estimation calculation was repeated three times, and the average execution times with standard deviation were recorded. Adding the query extraction time and model performance time provides the total time required to calculate a mortality risk estimation, which we are measuring to see if it meets our real-time performance criteria of one minute.

4.4.8 Updating Neonatal Risk Models

A general issue with using risk estimation models in CDS systems is the need to update the models in the future, as new data are collected over time. Ideally, the ability to update a model with new data should be as simple and automated as possible. Since both the CDR and neonatal risk estimation models use real-time data, and the CDR is being designed to store years worth of data, this should be possible. We explore methods using the CDR and DT risk models to create a model update strategy. The criteria for a model update strategy requires the following components:

1. The ability to extract the necessary attributes from the CDR that are needed
CHAPTER 4. METHODOLOGY

to create new mortality models.

2. The ability to export the data into the necessary format that is supported by the pattern classification tool that will be used to create the model.

3. The ability to take the exported data files and generate ten data sets to perform the 5-by-2 cross validation process.

4. The ability to generate a new mortality risk estimation model from the ten data sets and output the results.

5. The ability to incorporate the new mortality model into the CDS system.
Chapter 5

Results

5.1 Introduction

This chapter presents the results of the hybrid EAV storage format design and performance evaluation (Gilchrist et al., 2011b, 2010), and describes the design and implementation of the real-time CDR (Gilchrist et al., 2011b, 2008). It also provides the results of the new neonatal mortality risk models that use summary data (data from admission until up to the first 72 hours after NICU admission) and real-time data segments (data broken down into time segments extracted from the entire duration of the patient’s NICU stay) stored in the CDR (Gilchrist et al., 2011a). Finally it describes the data access performance achieved to deliver the necessary information to the mortality model to estimate a patient’s risk in real-time.
5.2 Hybrid EAV Storage Format Results

5.2.1 Storage Format Design

The novel hybrid EAV storage format combines the best of the simple EAV format (Table 2.6), which uses a single table for storing data, and the multi-data type EAV format (Table 2.7), which stores data in their original data types. Table 5.1 shows how this is accomplished by using multiple value columns in a single table (Gilchrist et al., 2011b). By using NULL values in the columns that are not used, less space is wasted as a NULL value requires very little memory. The hybrid EAV format uses integer (4 bytes), floating point (4 bytes), and text (2 to N bytes) values in a single table.

<table>
<thead>
<tr>
<th>Patient ID</th>
<th>Attribute ID</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Integer)</td>
<td>(Integer)</td>
<td>(Text)</td>
<td>(Integer)</td>
<td>(Float)</td>
</tr>
<tr>
<td>4 bytes</td>
<td>4 bytes</td>
<td>2..N bytes</td>
<td>4 bytes</td>
<td>4 bytes</td>
</tr>
<tr>
<td>1</td>
<td>1 (Name)</td>
<td>Joe Public</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>1</td>
<td>2 (Heart Rate)</td>
<td>NULL</td>
<td>120</td>
<td>NULL</td>
</tr>
<tr>
<td>2</td>
<td>1 (Name)</td>
<td>Jane Private</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>2</td>
<td>3 (Temperature)</td>
<td>NULL</td>
<td>NULL</td>
<td>36.5</td>
</tr>
</tbody>
</table>

The hybrid EAV design can support any data type by including a separate column for the data types to be used. The single table design avoids the need to access and join multiple tables when retrieving data, and having multiple value columns allows data to remain in their original data types, so conversion is not required to perform comparisons and mathematical calculations. Table 5.2 summarises the different characteristics of a conventional database and the hybrid, simple, and multi-data type EAV designs.
The performance of the hybrid EAV format was compared to the simple EAV and multi-data type EAV formats to determine how well it performed using the novel design (Gilchrist et al., 2011b, 2010). Two CDR performance data sets (Section 3.1.2) from the CHEO NICU database were used for this evaluation. The first contained 5.4M data points from 157 patients while the second had 16.7M points from 170 patients. Three aspects of the hybrid EAV format’s design were evaluated: storage size, query speed, and query complexity. For all tests, the same data were stored in the three different EAV formats to compare their performance. The EAV tables

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Conventional</th>
<th>Simple EAV</th>
<th>Multi-data Type EAV</th>
<th>Hybrid EAV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of tables</td>
<td>One</td>
<td>One</td>
<td>One table for each data type</td>
<td>One</td>
</tr>
<tr>
<td>Adding new attributes changes data structure</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Treatment of empty cells</td>
<td>Wasted space if not using NULL values</td>
<td>No empty cells</td>
<td>No empty cells</td>
<td>Use NULL values for empty cells</td>
</tr>
<tr>
<td>Data stored in original data type</td>
<td>Yes</td>
<td>No, all stored as text</td>
<td>Yes, separate table for each data type</td>
<td>Yes, separate column for each data type</td>
</tr>
<tr>
<td>Transformation required for calculations</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Effect of large number of attributes</td>
<td>Break table into smaller tables</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Degree of query complexity</td>
<td>Low</td>
<td>Moderate</td>
<td>High</td>
<td>Moderate</td>
</tr>
</tbody>
</table>

Table 5.2: Summary of Data Structure Characteristics
CHAPTER 5. RESULTS

were created using the open-source MySQL 5.1.46 database software (MySQL, 2010) with InnoDB tables (a transaction safe storage engine with crash-recovery abilities to protect data) on a commonly found Windows XP 64bit PC with 4GB of RAM and quad-core Intel Core2 Q6600 processor at 2.4GHz.

5.2.2 Storage Size Results

While all three EAV formats stored the same 16.7M points of data, the simple EAV format used the least amount of space at 1264MB, the hybrid EAV format used 1271MB, and the multi-data type format used the most space at 1276MB. The hybrid format used a single table, and could store the data in their original data type. The simple format used a single table, but stored everything as text, which was less compact for storing numbers such as integers or floating point values more than 4 digits long. Most of the integer and floating point data in the CDR were 2 or 3 digits long, which explained why the simple format was more compact than the hybrid and multi-data type formats. The multi-data type format required 1% more space than the simple format with some extra overhead required to manage multiple tables.

When storing the other data set with 5.4M data points, the hybrid EAV format used the least amount of space at 414MB, the simple EAV format with 417MB of space, and the multi-data type format used the most space at 421MB. After the data set tripled in size, the hybrid format used 0.6% more storage space than the simple format. The multi-data type format continued to use the most storage space. Given the small difference in storage requirements, we consider that all three formats required approximately the same storage space. However, these data sets contained data for a small number of patients. We anticipate the difference in storage space
5.2.3 Query Speed Results

5.2.3.1 Querying all data for a single patient

The time taken for querying all 296,829 data points from the 16.7M data point set from Patient B was 0.02 seconds (sec) for the simple EAV format, 0.02 sec for the hybrid EAV format, while the multi-data type format took 8.44 sec, which was considerably slower for this type of query. The time taken to query the 5.4M data point set for all 282,584 data points from Patient A was 0.02 sec for the hybrid EAV format, 0.02 sec for the simple EAV format, while the multi-data type format took 6.43 sec. The multi-data type format needs to combine separate tables using UNION operators when querying all data from a patient; this is not needed for the simple or hybrid formats, which explains why they are essentially instantaneous. With three times more data, the hybrid and simple EAV formats continue to have near-instantaneous query times, while the multi-data type format slows as the number of data points to search increases.

5.2.3.2 Calculating pulse pressure

The performance results of querying 3,239,316 data points for calculating the pulse pressure of all patients shows that all three EAV formats have similar performance levels with the hybrid at 0.24 sec being twice as fast as the multi-data type (0.49 sec); the simple format (0.59 sec) was slightly slower than the others. With the 5.4M data point set, the results of querying 16,956 data points for calculating the pulse pressure of all patients are similar and almost instantaneous for all three EAV formats. The
multi-data type took 0.15 sec, the hybrid was 0.19 sec and the simple was 0.23 sec. To identify the pulse pressure, the data stored as text need to be converted to integers. The extra time needed to cast these values into a different type of data format explains why the simple format EAV was slower than the other two approaches, which can use the values in their original data type. The performance of the hybrid format is faster than the multi-data type and simple formats. As the database grows, the query time increases slightly for each format, but takes only a quarter of a second for the hybrid format.

5.2.3.3 Finding patients with respiratory decompensation

Querying the 1272 blood gas laboratory results in the 16.7M data point set to find patients who are suffering from respiratory decompensation resulted in a performance of 0.02 sec for the multi-data type, 20.05 sec for the hybrid, and 23.98 sec for the simple EAV formats. Performance results with the 5.4M data point set were much faster. The multi-data type took 0.03 seconds, the hybrid took 2.75 sec and the simple took 3.29 sec.

This query, where “AND” conditions needed to be met for three different variables (PCO$_2$, HCO$_3$, and pH), required the intersection of three sub-queries. With MySQL, this was achieved with the use of INNER JOIN statements. The multi-data type format had near instantaneous query performance since the integer, floating point, and text data are separated into individual tables. The query had to join the very large integer data table (n=16,608,990), which contained HCO$_3$ data with the smaller floating point table (n=22,643) for PCO$_2$ and again the floating point table for pH. Joining a small table twice to a large one takes very little time. With the hybrid
format, all of the data are in the same table, which meant the query had to join a very large table (n=16,690,455) three times to access all three variables, taking considerably more time. The simple format must do the same thing, but also convert the data stored as text into integers and floating point values, which resulted in the slowest performance. When the data in the set tripled, the time per query greatly increased as the JOIN statements had to merge larger data sets. The multi-data type remained the same speed since it was combining only small data sets to the large set, while the simple and hybrid formats were 7.3 times slower with 16.7M data points. The performance of the hybrid format was faster than the simple format, but significantly slower than the multi-data type in this case, since floating point data were used in the query that allowed the multi-data type to take advantage of its separate tables.

It should be noted that if all three variables had of been integer values, the multi-data type EAV format would have also needed to join three large tables. To compare performance results, where all three EAV formats needed to combine the large integer data table for each attribute in the query, further tests were conducted using the same 1272 blood gas results but compared the PO₂, total carbon dioxide (T CO₂), and HCO₃ components, which were all integers. The hybrid format had the fastest query time of 23.10 sec, the simple format took 27.29 sec, and the multi-data type format took 27.80 sec. When the multi-data type format has to combine large tables, it becomes as slow as the simple format. When only 5.4M data points are in the data set, the multi-data type (2.74 sec) has a similar performance to the hybrid (2.91 sec) with both performing better than the simple (3.35 sec), since they do not need to convert the data type first.
5.2.4 Query Complexity Results

The number of elements in an SQL query does not change as the size of the database increases; therefore the results for complexity queries are identical for both the 5.4M and 16.7M data point databases.

5.2.4.1 Querying all data for a single patient

Querying all data stored for a single patient is one of the simpler types of queries to construct, but is commonly used in research. In the simple EAV table, most of the data in this set should actually be integers (>97%), although they are stored as text to accommodate all data types. To illustrate a query that readily permits manipulation of the numerical data, as is commonly needed when identifying patients with specific clinical conditions, we tested the SQL query used to extract and convert Patient A’s data from text to an unsigned integer, as shown in Table 5.3.

Table 5.3: SQL queries for querying all data for a single patient

<table>
<thead>
<tr>
<th>Simple EAV</th>
<th>Multi-data Type EAV</th>
<th>Hybrid EAV</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELECT Attribute, value_integer, NULL AS value_float, value_integer AS value_float,</td>
<td>SELECT Attribute, value_integer, NULL AS value_float, value_integer AS value_float,</td>
<td>SELECT Attribute, value_integer, value_float,</td>
</tr>
<tr>
<td>CAST(value_text AS UNSIGNED) FROM eav_multidatatype_integer PatientID = PatientA; WHERE</td>
<td>WHERE PatientID = PatientA UNION SELECT Attribute, value_float, NULL FROM eav_multidatatype_float</td>
<td>value_text FROM eav.hybrid WHERE</td>
</tr>
<tr>
<td>PatientID = PatientA; WHERE</td>
<td>NULL, value_float, NULL FROM eav_multidatatype_float</td>
<td>PatientID = PatientA;</td>
</tr>
<tr>
<td>value_text, NULL, NULL FROM eav_multidatatype_text WHERE</td>
<td>WHERE PatientID = PatientA UNION SELECT Attribute, value_text, NULL, NULL FROM</td>
<td></td>
</tr>
<tr>
<td>value_text, NULL, NULL FROM eav_multidatatype_text WHERE</td>
<td>eav_multidatatype_text</td>
<td></td>
</tr>
<tr>
<td>WHERE PatientID = PatientA;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Splitting the queries into individual elements, the hybrid EAV format had the least complex query with only 11 elements. The simple EAV format was slightly more complex needing 12 elements due to the fact that most data stored as text must be converted to a different data type. The multi-data type query was the most
complex with 39 elements. For the multi-data type format, three separate tables need to be queried and the result sets combined using the UNION operator, so it required nearly four times as many elements in the query as the other formats.

### 5.2.4.2 Extracting patients with attribute values within a specified range (OR condition)

A more complex SQL query was written to search for all patients in the NICU who have an abnormal respiration rate of less than 40 or greater than 60 breaths per minute, or who have hypothermic temperatures between 32.0 and 36.4 degrees Celsius, as shown in Table 5.4.

**Table 5.4: SQL queries for extracting patients with attribute values within a specified range (OR condition)**

<table>
<thead>
<tr>
<th>Simple EAV</th>
<th>Multi-data Type EAV</th>
<th>Hybrid EAV</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELECT PatientID, Attribute, value_text FROM eav_simple</td>
<td>SELECT PatientID, Attribute, value_integer, NULL AS value_float, value_text</td>
<td>SELECT PatientID, Attribute, value_integer, value_float, value_text</td>
</tr>
<tr>
<td>WHERE (Attribute = RespRate AND (CAST(value_text AS UNSIGNED) &lt; 40 OR CAST(value_text AS UNSIGNED) &gt; 60)) OR (Attribute = Temperature AND (CAST(value_text AS DECIMAL(5,1)) &gt;= 32.0 AND CAST(value_text AS DECIMAL(5,1)) &lt;= 36.4));</td>
<td>OR (value_integer &lt; 40 OR value_integer &gt; 60) UNION</td>
<td>(Attribute = RespRate AND value_integer &lt; 40 OR value_integer &gt; 60) UNION</td>
</tr>
<tr>
<td>AND (CAST(value_text AS DECIMAL(5,1)) &gt;= 32.0 AND CAST(value_text AS DECIMAL(5,1)) &lt;= 36.4));</td>
<td>value_float, NULL FROM eav_multidatatype.float WHERE Attribute = Temperature AND value_float &gt;= 32.0 AND value_float &lt;= 36.4);</td>
<td>AND value_float &gt;= 32.0 AND value_float &lt;= 36.4);</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The hybrid EAV format provides the simplest query with 32 elements since there is only one table to deal with, and the data are already in the proper data types for performing the threshold comparisons. The simple EAV format is more complicated with 44 elements. While only one table and value column exist in the simple EAV
format, the data must first be cast into integer and floating point values to do range comparisons. The multi-data type format is the most complex with 45 elements, since two separate tables need to be queried and the result sets are combined using the UNION operator.

5.2.4.3 Extracting patients with attribute values within a specified range (AND condition)

A complex SQL query was written to search for all patients suffering from respiratory decompensation. This required the blood gas laboratory results of the patient to have a PCO$_2$ greater than 60 mmHg, an HCO$_3$ greater than 19 mmHg, and a pH less than 7.25. Since the laboratory values needed to be from the same blood test, we needed to include the time stamp, as shown in Table 5.5.

The hybrid EAV format and multi-data type format have equally complex queries with 63 elements each. Since three JOINs need to be performed to complete this query, the multi-data type is not at a disadvantage using three different tables. The simple EAV format has the most complex query with 71 elements. Extra elements are required to cast the text values into floating point and integer values making it more complex than the others.

For simple queries, the multi-data type format has significantly more complex queries than the simple EAV or hybrid EAV formats due to its combination of multiple tables. When the query becomes more complicated (as for an OR condition), the difference in complexity is lower. When finding patients with abnormal respiration rates or hypothermic temperatures, the complexity of the queries for the simple EAV and multi-data type formats was essentially the same. The extra elements in the
Table 5.5: SQL queries for extracting patients with attribute values within a specified range (AND condition)

<table>
<thead>
<tr>
<th>Simple EAV</th>
<th>Multi-data Type EAV</th>
<th>Hybrid EAV</th>
</tr>
</thead>
</table>
| SELECT t3.PatientID, t3.value.timestamp, t1.value
| AS PCO2, t2.value text as pH, t3.value text AS HCO3 FROM env_simple t1 INNER JOIN eav_simple t2 ON t1.PatientID = t2.PatientID AND t1.value.timestamp = t2.value.timestamp INNER JOIN eav_simple t3 ON t1.PatientID = t3.PatientID AND t1.value.timestamp = t3.value.timestamp WHERE CAST(t1.value AS UNSIGNED) > 60 AND CAST(t2.value AS pH) > 7.25 AND t3.value > 19; | SELECT t3.PatientID, t3.value.timestamp, t1.value_float AS PCO2, t2.value_float as pH, t3.value_integer AS HCO3 FROM env_multidatatype_float t1 INNER JOIN eav_multidatatype_float t2 ON t1.PatientID = t2.PatientID AND t1.value.timestamp = t2.value.timestamp INNER JOIN eav_multidatatype_integer t3 ON t1.PatientID = t3.PatientID AND t1.value.timestamp = t3.value_timestamp WHERE t1.value_float > 60 AND t2.value_float > 7.25 AND t3.value > 19; | SELECT t3.PatientID, t3.value.timestamp, t1.value_float AS PCO2, t2.value_float as pH, t3.value_integer AS HCO3 FROM eav_hybrid t1 INNER JOIN eav_multidatatype_float t2 ON t1.PatientID = t2.PatientID AND t1.value.timestamp = t2.value.timestamp INNER JOIN eav_multidatatype_integer t3 ON t1.PatientID = t3.PatientID AND t1.value_timestamp = t3.value_timestamp WHERE t1.value_float > 60 AND t2.value_float > 7.25 AND t3.value > 19; |

query required for casting the values into the proper data type with the simple EAV format was equivalent to the extra elements needed to search and combine results from multiple tables in the multi-data type format. For even more complex queries (as an AND condition), the complexity between the three EAV formats was almost equal. The hybrid and multi-data type formats had equal complexity and the simple format was slightly more complex. Data do not need to be cast or pulled from multiple tables in the hybrid EAV format, so the overall complexity of the hybrid EAV query is lower than the rest.
5.2.5 EAV Performance Summary

We compared the performance of three different types of EAV storage formats for a CDR with two different data sets (5.4M and 16.7M data points). Our hybrid EAV approach provided a nice balance between the simple and multi-data type formats and scaled well as the amount of data in the data set increased. Focusing mainly on the larger data set, Table 5.6 shows that the hybrid format uses less storage space than the multi-data type and only slightly more than the simple format. It performs better than the multi-data type format when querying all data from a single patient, and better than the simple format when performing more complex queries and calculations. Overall, the hybrid format uses less complex queries than the multi-data type and the simple formats.

Table 5.6: Summary of EAV Format Performance Results

<table>
<thead>
<tr>
<th>Testing Criterion</th>
<th>Data set (5.4 million data points)</th>
<th>Data set (16.7 million data points)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Simple EAV</td>
<td>Multi-data Type EAV</td>
</tr>
<tr>
<td>Storage space required (MB)</td>
<td>417</td>
<td>421</td>
</tr>
<tr>
<td>Query Speed: Querying all data for a single patient (sec)</td>
<td>0.02</td>
<td>6.43</td>
</tr>
<tr>
<td>Query Speed: Calculating pulse pressure (sec)</td>
<td>0.23</td>
<td>0.15</td>
</tr>
<tr>
<td>Query Speed: Finding patients with respiratory decompensation (sec)</td>
<td>3.29</td>
<td>0.03</td>
</tr>
<tr>
<td>Query Speed: Querying three integer values (sec)</td>
<td>3.35</td>
<td>2.74</td>
</tr>
<tr>
<td>Query Complexity a: Querying all data for a single patient (# elements)</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Query Complexity a: OR query (# elements)</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Query Complexity a: AND query (# elements)</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

aResults not dependent on data set size.
CHAPTER 5. RESULTS

When the data set increased from 5.4M data points to 16.7M data points, the performance ranking of the hybrid EAV format for query speed and complexity stayed the same compared to the simple and multi-data type formats. This indicates that the hybrid format should continue to provide a good balance as the data scales even further. It is important to note that all query speeds took less than 30 seconds using the hybrid EAV format, which means the performance far exceeds the 1 minute maximum time allowed for real-time operation. The most important query used by real-time risk estimation models for individual patients was almost instantaneous at 0.02 seconds.

5.3   Real-Time CDR Design Results

A new real-time CDR was designed to store the large amounts of patient data collected from patients in the NICU. This section describes the details of the design for data collection, data storage, patient privacy, and data retrieval (Gilchrist et al., 2011b, 2008).

5.3.1   Data Collection

Data collection from the medical systems at CHEO was accomplished using the HL7 messaging standard. The CDR received and processed medical data using MIRTH, an open-source HL7 interface engine (Mirth Corporation, 2009). MIRTH facilitated the routing, filtering, and transformation of messages over a variety of protocols with support for numerous standards. MIRTH filtered data from different sources, pulling out relevant information, and then storing it in an SQL compatible database. MIRTH is a freely available and very popular for processing HL7 data. While the software is
open-source, support contracts are available for MIRTH.

The HL7 messages from the various hospital medical systems were received by the CDR and processed by MIRTH, which is compatible with versions 2 and 3 of HL7 standard. MIRTH supports multiple channels for receiving HL7 data from different sources and different source types, such as database, flat file, file transfer, web access, and TCP network connections. It provides basic filtering of HL7 data for simple configurations and allows the use of JavaScript to write code for more advanced filtering capabilities. MIRTH allows access to individual components in an HL7 message. Figure 5.1 shows how MIRTH transforms a raw HL7 message into easy to understand and accessible components.

<table>
<thead>
<tr>
<th>Raw HL7 Message</th>
<th>Filtered Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSH</td>
<td>~&amp;</td>
</tr>
</tbody>
</table>

**Patient ID:** 000000003  
**Patient Name:** FirstName LastName  
**Patient Bed:** NICU  
**Observation Time:** 20090306172200  
**Observation[1] ID:** 188  
**Observation[1] Name:** SpO2  
**Observation[1] Value:** 97  
**Observation[1] Units:** %  
**Observation[2] ID:** 44  
**Observation[2] Name:** Pulse  
**Observation[2] Value:** 133  
**Observation[2] Units:** bpm

**Figure 5.1:** MIRTH HL7 Filtering.
5.3.2 Data Storage and Patient Privacy

The CDR design provides easy integration for current and future systems, since it uses an SQL standards-based database server. It was not designed for a specific purpose, so it is generalisable for any situation where time series data needs to be collected and stored. All data can be stored, including the automatic addition and tracking of new attributes, and the data can be used for different purposes. Specific data can be extracted into a smaller database for a more specialised purpose if required. No specific database software package or vendor is required to implement the CDR design. Any relational database software, whether open source or proprietary, that supports the SQL standard can be employed, which allows organisations using the CDR to select the right software to meet their operational, security, and encryption requirements. This design offers a long-term sustainable solution that will be suitable for many years.

The open-source MySQL database software (MySQL, 2010) was chosen for the CDR. It is freely available, widely supported, and many tools and applications exist that are compatible. Support contracts can be also obtained for MySQL. Researchers can use a variety of applications to access the data, and with terabytes of cost-effective storage capacity available, years of data can be kept depending on the needs of researchers. The new hybrid EAV model was integrated into the CDR database to allow for the handling of data types in their original form with a single database table (recall Table 5.1). This should allow for higher performance queries to the database when searching for ranges of values or any kind of operation that applies specifically to the data type whether it be text or numeric in nature since it does not have to be converted first. The use of NULL values in MySQL does not take up any additional
disk space, so using extra columns in this manner should have very little effect on disk usage.

After data arrived at the CDR and was processed by MIRTH, the data were split into private patient information (such as patient ID, name, birth date), and de-identified medical data (such as heart rate values and oxygen levels). The private data were written to a segregated private portion of the MySQL database (as seen in the "Private Patient Data" block of Figure 5.2) while the de-identified data were written to a segregated de-identified portion of the database (as seen in the "De-Identified Medical Data" block of Figure 5.2). Patients in the de-identified portion of the medical database are identified by a Study ID, which is generated by the system and does not relate in any way to their Patient ID. The data are, therefore, de-identified before they get written to the database. This allows immediate extraction and usage of data from the CDR, since no further processing is required.

Figure 5.3 shows the CDR database design with three main tables: Patient Iden-
Figure 5.3: Extended Entity Relation (EER) Diagram for CDR.

tity, EAV Attributes, and EAV Data. The schema for creating this CDR design in an SQL compliant database has been included below. It was created automatically from the Extended Entity Relation (EER) diagram created with the MySQL Workbench 5.1 software:

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';

CREATE SCHEMA IF NOT EXISTS 'cdr' DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci ;
USE 'cdr';

-- Table 'cdr'. 'eav_data'
```
CHAPTER 5. RESULTS

CREATE TABLE IF NOT EXISTS 'cdr'. 'eav.data' ( 'eavID' BIGINT UNSIGNED NOT NULL AUTO_INCREMENT , 'StudyID' INT UNSIGNED NOT NULL DEFAULT 0 , 'AttributeID' INT UNSIGNED NOT NULL DEFAULT 1 , 'value_timestamp' TIMESTAMP NULL DEFAULT NULL , 'value_int' INT NULL DEFAULT NULL , 'value_float' FLOAT NULL DEFAULT NULL , 'value_char' VARCHAR(255) NULL DEFAULT NULL , PRIMARY KEY ('eavID') , INDEX 'studyID_attribID_timestamp' ('StudyID' ASC, 'AttributeID' ASC, 'value_timestamp' ASC) ) ENGINE = InnoDB;

-- Table 'cdr'. 'eav_attributes'

CREATE TABLE IF NOT EXISTS 'cdr'. 'eav.attributes' ( 'AttributeID' INT UNSIGNED NOT NULL AUTO_INCREMENT , 'AttributeClinicalID' VARCHAR(45) NOT NULL DEFAULT '' , 'AttributeName' VARCHAR(45) NOT NULL DEFAULT '' , 'AttributeType' ENUM('INTEGER', 'FLOAT', 'VARCHAR', 'TIMESTAMP') NOT NULL DEFAULT 'INTEGER' , 'AttributeUnits' VARCHAR(45) NULL DEFAULT '' , 'AttributeCodingSystem' VARCHAR(45) NULL DEFAULT '' , PRIMARY KEY ('AttributeID') , INDEX 'attribute.clinicalID.name' ('AttributeClinicalID' ASC, 'AttributeName' ASC) ) ENGINE = InnoDB;

-- Table 'cdr'. 'patient.identity'

CREATE TABLE IF NOT EXISTS 'cdr'. 'patient.identity' ( 'PatientID' INT UNSIGNED NOT NULL DEFAULT 0 , 'StudyID' INT UNSIGNED NOT NULL AUTO_INCREMENT , 'PatientLastName' VARCHAR(45) NULL DEFAULT NULL , 'PatientFirstName' VARCHAR(45) NULL DEFAULT NULL , 'PatientDateOfBirth' DATE NULL DEFAULT NULL , 'PatientGender' ENUM('FEMALE', 'MALE', 'UNKNOWN') NULL DEFAULT NULL , 'PatientAdmitDate' TIMESTAMP NULL DEFAULT NULL , 'PatientDischargeDate' TIMESTAMP NULL DEFAULT NULL , PRIMARY KEY ('StudyID', 'PatientID') , INDEX 'last_first_dob' ('PatientLastName' ASC, 'PatientFirstName' ASC, 'PatientDateOfBirth' ASC) , INDEX 'admit_discharge' ('PatientAdmitDate' ASC, 'PatientDischargeDate' ASC) )
CHAPTER 5. RESULTS

ENGINE = InnoDB;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

5.3.2.1 Patient Identity Table

The patient identity table contains information related to patients such as hospital ID, name, date of birth, and gender. This table is stored in the CDR located at the hospital to link the de-identified data in the other tables back to the real patient to confirm diagnoses or other information. Only users with the appropriate REB approval would have access to this table containing sensitive patient information.

To de-identify the data, a unique study ID is generated for each patient. The study ID starts at 1 for the first patient, and automatically increments for each successive patient. To maximise privacy, the study ID should be randomly generated and a verification done to ensure that the value is not already being used. It is the key used to link a patient in the de-identified portion of the data to a real hospital patient. This allows us to easily separate the patient identity table from the other tables when only exporting de-identified data is required, for data mining or other analyses. A database index is created for this table to allow for fast queries of a patient by last name, last and first names, or last and first names and date of birth. A second index is also created for fast queries of patient admit and discharge dates. Indexes are used to quickly find rows in the database with specific column values. If no index exists, the database search has to start with the first row and then read every entry in the patient identity table to find the relevant rows. The larger the database table, the more time the non-indexed search will take. Since patient ID is a primary key and
automatically indexed, queries for it are also fast. Table 5.7 shows an example of data stored in the patient identity table.

### 5.3.2.2 EAV Attributes Table

The EAV attributes table contains information related to the types of attributes stored in the database. This allows new attribute types to be added easily without having to change the database design at all. This feature is important for medical data as new technology and new types of testing will introduce new attribute types to be stored in the database. For example if patient weight was not previously recorded in the database, but this is later deemed to be important, a new attribute type can be created called “Weight” and added to the table. The database table structure itself does not need to be updated; only a new row needs to be added to the table.

Each attribute is assigned a unique attribute ID. An attribute name is also stored to provide a text-based description of the attribute, such as “Heart Rate”. The attribute data type is also stored to indicate whether the data value being stored for the attribute is an integer, floating point number, text, or a timestamp. The attribute units are stored to provide a description of the unit of measurement being
CHAPTER 5. RESULTS

Table 5.8: EAV Attributes Table Sample

<table>
<thead>
<tr>
<th>Attribute ID</th>
<th>Attribute Name</th>
<th>Attribute Type</th>
<th>Attribute Units</th>
<th>Attribute CodingSystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Heart Rate</td>
<td>INTEGER</td>
<td>bpm</td>
<td>LOINC</td>
</tr>
<tr>
<td>2</td>
<td>Weight</td>
<td>FLOAT</td>
<td>kg</td>
<td>LOINC</td>
</tr>
</tbody>
</table>

used, such as “bpm (beats per minute)”. Finally, the attribute coding system is stored to indicate what kind of coding system was used for the attribute, which may change from vendor to vendor. No patient identifying information is stored in this table, so it is safe to export and remove from the hospital or for research by hospital staff. Indexes are created for this table to allow fast queries by attribute id or attribute name. Table 5.8 shows an example of data stored in the EAV attributes table.

5.3.2.3 EAV Data Table

The EAV data table contains information related to the captured data from patient monitors, the ADT and Lab systems, and discharge diagnosis reports. This is the main storage area for all the raw research data. The table contains columns for each of the data types supported by the CDR. This means the EAV data table supports any data that can be saved in timestamp, integer, floating point, or text format. Data for new kinds of attributes can easily be stored as long as they can be saved in one of the supported data types.

Each record is assigned a unique eavID to distinguish between all other data records. The study ID is stored to identify, which patient the data belongs to. The attribute ID is stored to identify the type of data that is found in the record, providing a link back to the EAV attributes table to match the corresponding attribute such as “Heart Rate”. The timestamp, int, float, and char columns store data for the
appropriate attribute data type being recorded such as date and time, integer values, floating point number, and text based data. The timestamp field will always be populated either with a timestamp from the patient monitor or ADT/Lab record, or with the current time that the data was entered into the database. Only one of the other fields: int, float, and char will contain data, depending on the type; the rest will be NULL. The NULL value is used instead of 0 since it takes up no space in the database. It is also important to note that the EAV data table requires the EAV attributes table to be present, so that information about the attribute type can be obtained. When exporting the EAV data table, to help improve patient privacy, time stamp fields can be modified to contain the relative time to the first time stamp for each patient in the database. For example, if the first time stamp for the patient being exported was March 12, 2012 at 00:00:00 hours, then all time stamps for that patient would be re-calculated to be the relative time from that first time stamp. The first time stamp would be recorded from some chosen starting point such as January 1, 1970 at 00:00:00. If a record was being exported that had the time stamp March 15, 2012 at 12:00:00 hours, it would then be converted to a relative time stamp for 3.5 days later from the same chosen starting point, so January 4, 1970 at 12:00:00. This way all patient time stamps are stored, staring at January 1, 1970 at 00:00:00, so that the original time stamp does not reveal when the patient was actually admitted to the hospital in the exported data which could provide information to re-identify him/her.

Indexes are created for this table to allow fast queries by study ID, study ID and attribute ID, or study ID and attribute ID and time stamp. For example, this would allow a query to find all data belonging to a patient with study ID = X quickly. It
CHAPTER 5. RESULTS

Table 5.9: EAV Data Table Sample

<table>
<thead>
<tr>
<th>eav ID</th>
<th>Study ID</th>
<th>Attribute ID</th>
<th>val_time</th>
<th>val_int</th>
<th>val_float</th>
<th>val_char</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>20081201141731</td>
<td>137</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>20081201141810</td>
<td>135</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2</td>
<td>20081201141810</td>
<td>NULL</td>
<td>2.273</td>
<td>NULL</td>
</tr>
</tbody>
</table>

would also allow a query to find all “Heart Rate” data for a patient with study ID = X quickly. Finally it would allow a fast query to find all “Heart Rate” data on a specific date or range of dates for a patient with study ID = X. Table 5.9 shows an example of data stored in the EAV data table.

5.3.3 Data Retrieval

The CDR was designed to allow the extraction of data using different strategies to limit the amount of exported information for large data sets. Since the CDR design supports the SQL standard, any existing SQL compliant software can be used or new software written to access and extract data from the CDR. SQL queries can be constructed to extract data using strategies like those outlined below. Once extracted, data can also be stored in any required format (such as a comma separated flat file) that can then be imported into systems that do not directly support SQL. It could also be used directly in an application such as a CDS system to regularly update risk models that provide the current status of each patient to physicians.

5.3.3.1 Patient Complete Stays

The Patient Identity table contains PatientAdmitDate and PatientDischargeDate columns, which by default contain the value NULL. When extracting data, it is
possible to check the patient’s record first to see if those values are both not NULL, which implies that both an admit and discharge date have been recorded and the patient’s stay is complete. This way patients who are currently admitted to the NICU can be ignored. An example SQL query to show only patients with full stays is:

```sql
SELECT * FROM 'patient_identity' WHERE 'PatientAdmitDate' IS NOT NULL AND 'PatientDischargeDate' IS NOT NULL;
```

### 5.3.3.2 Time Spans

With the `value_timestamp` column of the EAV Data table, the CDR has the ability to only extract data before a specific date, after a specific date, or between specific dates. For example an SQL query to find all data between January 1st, 2011 at 8am and January 7th at 4pm would look like.

```sql
SELECT * FROM 'eav_data' WHERE 'value_timestamp' BETWEEN '2011-01-01 08:00:00' AND '2011-01-07 16:00:00';
```

### 5.3.3.3 Data Frequency

Using the support for time spans, it is possible to specify the data frequency for the attribute being extracted. For example, with the CDR design, it is possible to find the average heart rate (\(AttributeID = 1\)) of a patient (\(StudyID = 1\)) over a 60 minute time span (January 1st, 2011 at 8am to January 1st, 2011 at 9am). Heart rates are stored as integers, so SQL query would be accessing the `value_int`, `StudyID`, `AttributeID`, and `value_timestamp` columns in the CDR database using:
SELECT AVG('value_int') FROM 'eav_data' WHERE 'StudyID' = 1 AND 'AttributeID' = 1 AND 'value_timestamp' BETWEEN '2011-01-01 08:00:00' AND '2011-01-01 09:00:00';

5.3.3.4 Meta-Data Descriptions

The EAV data table contains only a numeric Attribute ID value, which is designed to be fast for machine processing, but not for human readability. The EAV attributes table however contains text columns to describe the name or description of the attribute (such as Heart Rate), the data type (such as integer), the type of units that attribute is measured in (such as beats per minute (bpm)), and the coding system used by the attribute (such as Logical Observation Identifiers Names and Codes (LOINC)). LOINC is a universal coding system for identifying laboratory and clinical observations that can be used to standardise terms when using multiple independent systems (LOINC, 2012). This data can easily be extracted automatically with the raw data, so that humans can more easily read and understand the data content.

5.4 Neonatal Mortality Risk Model Results

This section describes the results of the new neonatal mortality risk models (Gilchrist et al., 2011a) that use summary data (data from admission until up to the first 72 hours after NICU admission) and real-time data segments (data broken down into time segments extracted from the entire duration of the patient’s NICU stay) from the CNN and CHEO data sets. The DT models are compared to one created using an ANN. Results are also provided using newer 2011 CHEO data to see how well the DT model performs when using new data. Methods are examined for improving
DT results with imbalanced data sets, and various threshold values are explored for creating risk categories for results instead of providing only raw number results. Finally, data retrieval and usage performance are measured to validate whether the mortality model can estimate a patient’s risk in real-time.

5.4.1 Baseline Risk Estimation Model with CNN Data

The baseline mortality risk estimation model used the summary CNN data collected during the first 12 hours after NICU admission. While the original CNN database contained 20,488 patients, there were 19,814 patient cases that had mortality results recorded giving a 3.8% mortality rate (750 patients). After removing 3,124 patient cases due to missing more than 50% of the attributes, the CNN data contained 17,364 total patient cases with a 4% mortality rate (687 patients). The resulting model achieved 97% mean specificity and 56% mean sensitivity on the test data, which did not meet the clinical usefulness guideline set by our clinical partners. The F1 score of 0.24 and MCC of 0.28 shows that the mortality model is not simply making random risk estimations. Boosted DTs produced better results than single DTs, therefore, only the boosted results are shown.

5.4.2 Summary Data Risk Estimation Models with CHEO Data

We created mortality models using CHEO data from the first 12, 24, 48, and 72 hours after NICU admission to see if the results improved when using more up-to-date patient data. After removing 102 patient cases that had more than 50% of the attributes missing, the CHEO summary data set contained 256 patient cases with a
9% mortality rate (24 patients). Of the four mortality models, the model using only data from first 12 hours had the best mean sensitivity at 73%, followed by the model using data from the first 72 hours with 68%, while the other two models had 63% sensitivity. The mean specificity between models was almost identical with the first 48 hours achieving 94%, the first 24 and 72 hours 93%, and the first 12 hours 92%. For mean MCC and F1 scores, the first 48 hours had the best results with 0.45 and 0.45, followed by the first 72 hours with 0.41 and 0.40, the first 24 hours with 0.37 and 0.35, and finally the first 12 hours with 0.34 and 0.28, respectively. Full results can be seen in Table 5.11. Boosted DTs produced better results than single DTs, therefore, only the boosted results are shown.

With the goal of minimizing false positives, the model using data from the first 48 hours after NICU admission had the best performance, with the highest specificity, PPV, MCC, and F1 scores, although all four models provided results that met the minimum clinical usefulness criteria set by our clinical partners. Of the ten mortality models created with the 5-by-2 cross validation method, all 18 attributes were used in at least one of the models, but some attributes were used considerably more often. The lowest mean blood pressure was used in all ten mortality models. The lowest serum pH, highest immature/total neutrophil ratio and lowest serum sodium were used in 90% of the models, and highest serum glucose and highest respiratory rate were used in 70% of the models. The usage of the remaining attributes is summarised in Table 5.10.

Using the risk estimation model with the best performance (first 48 hours after admissions) as a starting point, three more mortality models were created to see if using extra attributes in addition to the 18 SNAP variables could improve perfor-
Table 5.10: Attribute Usage in Best Decision Tree Models Using CHEO Data

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Summary First 48 Hours Usage</th>
<th>48-Hour Time Segments Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>lowest mean blood pressure</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>lowest serum pH</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>highest immature/total neutrophil ratio</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>lowest serum sodium</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>highest serum glucose</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>highest respiratory rate</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>lowest serum glucose</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>highest serum sodium</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>lowest heart rate</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>lowest pO₂ level</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>highest blood hematocrit</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>highest pCO₂ level</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>highest mean blood pressure</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>lowest blood hematocrit</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>lowest white blood cell count</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>highest heart rate</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>lowest absolute neutrophil count</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>lowest platelet count</td>
<td>1</td>
<td>10</td>
</tr>
</tbody>
</table>
CHAPTER 5. RESULTS

mance. The first new model added the attribute gender to the 48-hour data set. The resulting model had an equal mean specificity to the original 48-hour model, but a lower mean sensitivity at 60% showing that less of the mortality cases were correctly identified as such. PPV (0.38) and NPV (0.97) were the same as the original model. MCC was slightly lower at 0.43 and F1 score was identical. A similar trend was seen with the second model, which added lowest CO₂ as an attribute, with mean sensitivity (59%), PPV (0.37), MCC (0.42), and F1 scores (0.43) being lower than the original model and mean specificity (94%) and NPV (0.97) values being equal. The third model, which added the standard deviation of six attributes (mean blood pressure, heart rate, respiration rate, blood hematocrit, serum glucose, and serum sodium) showed a slight decrease in performance with mean sensitivity (62%), mean specificity (94%), PPV (0.36), NPV (0.97), MCC (0.42), and F1 score (0.43). None of the extra attributes improved the results of the current best mortality model using decision trees. The C5.0 decision tree learning process rarely selected one of the new attributes to be used in the tree structure and when it did, the results led to lower performance than using the original 18 SNAP variables on their own.

5.4.3 Real-Time Risk Estimation Models with CHEO Data

Three mortality risk estimation models were created using CHEO real-time data containing the attributes listed in Table 5.10 for the patients' entire NICU stays using time segments of 12, 24, and 48 hours. Time segments are data from the patient's entire NICU stay at CHEO, broken down into segments of the specified time period in length. For example, with 12 hour time segments, a patient with an NICU stay for 30 hours would have 3 associated data records: (1) hours 0 to 12; (2)
CHAPTER 5. RESULTS

hours 12 to 24; and (3) hours 24 to 30. After removing 37 patient cases that had more than 50% of the attributes missing, the CHEO time segment data set contained 321 patient cases with an 8% mortality rate (27 patients). Due to missing attributes, 6,091 segments were removed from the 12-hour segment data, 2,200 segments from the 24-hour segment data, and 1,048 segments from the 48-hour segment data. The 12-hour segment data set consisted of 2398 segments of data with 20% (400 segments) of the data belonging to mortality cases. The 24-hour segment data set consisted of 1834 segments of data with 16% (258 segments) of the data belonging to mortality cases, and finally the 48-hour segment data set consisted of 1245 segments of data with 13% (140 segments) of the data belonging to mortality cases.

Table 5.11 shows that the performance as measured by the mean sensitivity improved as the length of the time segment increased with 59% for 12-hour segments, 64% for 24-hour segments, and 66% for 48-hour segments. The mean specificity also increased with the length of the time segment resulting in 85% for 12-hour segments, 88% for 24-hour segments, and 91% for 48-hour segments. For mean MCC and F1 scores, the 48-hour segments had the best result with 0.34 and 0.36, the 24-hour segments with 0.34 and 0.33, and the 12-hour segments with 0.25 and 0.24. The 48-hour time segment model performed the best, having better results for every category measured. At 66% mean sensitivity and 91% mean specificity, it is the only real-time model that is within the required performance range to be useful for our clinical partners. Similar to the CHEO 48-hour summary data models, all 18 attributes from Table 5.10 were used in at least one of the 10 models created with the 5-by-2 cross validation process for the CHEO 48-hour time segment models, but for the time segment models more than half of the attributes were used in all 10 cross
validation models. Table 5.10 summarises the attribute usage for the ten mortality models using a 48-hour time segment with the CHEO data. Boosted DTs produced better results than single DTs, therefore, only the boosted results are shown.

Using the data from the 48-hour time segment model, which performed best, two more mortality models were tested to see if splitting the patients into groups of stable (survivors), non-stable (mortality patient data up to the last 48 hours of admission), and unstable (mortality patients only the last 48 hours of admission) could improve results. The models were developed using the same 18 SNAP attributes as the 48-hour time segment mortality model. The first model used just two categories: stable, and unstable patients, with the non-stable time segments removed. The use of only stable and unstable patients led to a significant increase in mean specificity (98%), a slight increase in mean NPV (1.0), but a large drop in mean sensitivity (49%), bringing the results below the minimum requirements to be clinically useful. The mean PPV (0.18), MCC (0.27), and F1 scores (0.24) were lower than the original 48-hour time segment model. The second model used time segments from the stable and non-stable patients combined, and unstable patients. The resulting model had higher mean specificity (98%) and mean NPV (0.99), but significantly lower mean sensitivity (23%), mean PPV (0.15), mean MCC (0.17), and mean F1 score (0.22) compared to the original 48-hour time segment model. Removing the time segments before the last 48 hours of admission greatly decreased most of the performance measures of the model, and combining those time segments with the non-mortality cases decreased the performance even further. This suggests that while the specificity improved, data from before the last 48 hours of admission for mortality cases is important in detecting true mortality cases, and should be taken into considering when developing mortality
models.

5.4.4 CHEO 2010 vs 2011 Data Set Results

We created C5.0 decision trees using the CHEO 2010 data set as the training set and the CHEO 2011 data set as the test set to measure the performance of the mortality risk estimation models for future NICU patients. For comparison, we used the 48-hour summary data time span and 48-hour time segment size, which produced the best mortality models with the CHEO 2010 data set alone. When using the 48-hour summary data, the single DT produced a better result than the boosted one. The single DT mortality model had 75% sensitivity, 96% specificity, 0.38 PPV, 0.99 NPV, 0.50 F1 score, and 0.52 MCC, which exceeded the minimum clinical usefulness requirements. The DT required only 3 attributes to achieve this level of performance: lowest serum pH, lowest blood pressure, and lowest heart rate. Figure 5.4 shows the mortality risk estimation decision tree. Using 48-hour summary data from the entire CHEO 2010 data set for training allowed the DT to create a model that could estimate risk mortality of the patients in the CHEO 2011 data set better than using half of the 2010 data for training and half for testing.

A boosted DT produced a better model than a single DT when using the 48-hour time segment data of the patient’s entire stay. This boosted model had 50% sensitivity, 93% specificity, 0.22 PPV, 0.98 NPV, 0.21 F1 score, and 0.30 MCC, which is not good enough to be clinically useful. The 48-hour summary data model using the CHEO 2011 data set outperformed using time segments from the entire stay.
Figure 5.4: CHEO 48-Hour Summary Data Mortality Risk Estimation Model Decision Tree.

5.4.5 DT Model Comparison

Table 5.11 summarises the performance results for the ten neonatal mortality risk estimation models created with the C5.0 decision tree algorithm using the 18 SNAP
attributes listed in Table 5.10, three models that used additional attributes, and two models that used the CHEO 2011 data set.

The results show that the data we collected prospectively in the CDR from the NICU at CHEO can produce mortality models as good or better than those developed with the retrospective CNN data set. With access to data beyond the first 12 hours from NICU admission, we produced a mortality model that performs better than the summary data models that only use the first 12 hours of data. These results showed that using data within the first 48 hours from NICU admission produced the best results overall.

The time segment data set, which used all data for each patient’s entire stay in the NICU was broken into time segments of 12, 24, or 48 hours. It seems the significantly larger data set using time segments led to an increase in false positives, which reduced the specificity compared to the summary data models. Traditional pattern classification techniques, such as decision trees, do not easily handle multiple sets of data points associated with the same patient. Each time segment of data was treated as if it belonged to a separate patient by the decision tree algorithm. When looking at the entire patient admission, some time segments of data will contain ranges of values from mortality patients who are similar to those who survived, during periods when the patients were less ill. This most likely led to the decision tree having a more difficult time discriminating between the vital signs and laboratory results of patients who died versus those who survived. The summary data models had higher specificity values compared to the time segment models. Since the summary data models were looking at highest or lowest values over an extended period and there was only one set of data points per patient, the decision tree seemed to be able to
### Table 5.11: Summary of Results for DT Neonatal Mortality Risk Estimation Models

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Model Type</th>
<th>Time Frame</th>
<th># Patients (% Mortality)</th>
<th>Sens.</th>
<th>Spec.</th>
<th>PPV</th>
<th>NPV</th>
<th>MCC</th>
<th>F1 score</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN</td>
<td>Summary</td>
<td>First 12 hours</td>
<td>17,364 (4%)</td>
<td>0.56 ± 0.04</td>
<td>0.97 ± 0.00</td>
<td>0.16 ± 0.03</td>
<td>0.99 ± 0.00</td>
<td>0.28 ± 0.00</td>
<td>0.24 ± 0.03</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary</td>
<td>First 12 hours</td>
<td>256 (9%)</td>
<td>0.73 ± 0.25</td>
<td>0.92 ± 0.01</td>
<td>0.18 ± 0.09</td>
<td>0.99 ± 0.01</td>
<td>0.34 ± 0.11</td>
<td>0.28 ± 0.11</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary</td>
<td>First 24 hours</td>
<td>256 (9%)</td>
<td>0.63 ± 0.26</td>
<td>0.93 ± 0.01</td>
<td>0.27 ± 0.14</td>
<td>0.98 ± 0.02</td>
<td>0.37 ± 0.14</td>
<td>0.35 ± 0.13</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary</td>
<td>First 48 hours</td>
<td>256 (9%)</td>
<td>0.63 ± 0.19</td>
<td>0.94 ± 0.01</td>
<td>0.38 ± 0.15</td>
<td>0.97 ± 0.02</td>
<td>0.45 ± 0.12</td>
<td>0.45 ± 0.12</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary</td>
<td>First 72 hours</td>
<td>256 (9%)</td>
<td>0.68 ± 0.23</td>
<td>0.93 ± 0.01</td>
<td>0.32 ± 0.16</td>
<td>0.98 ± 0.02</td>
<td>0.41 ± 0.14</td>
<td>0.40 ± 0.14</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary + Gender</td>
<td>First 48 hours</td>
<td>256 (9%)</td>
<td>0.60 ± 0.14</td>
<td>0.94 ± 0.01</td>
<td>0.38 ± 0.15</td>
<td>0.97 ± 0.02</td>
<td>0.43 ± 0.12</td>
<td>0.45 ± 0.12</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary + CO₂</td>
<td>First 48 hours</td>
<td>256 (9%)</td>
<td>0.59 ± 0.13</td>
<td>0.94 ± 0.01</td>
<td>0.37 ± 0.16</td>
<td>0.97 ± 0.02</td>
<td>0.42 ± 0.11</td>
<td>0.43 ± 0.13</td>
</tr>
<tr>
<td>CHEO 2010</td>
<td>Summary + std dev</td>
<td>First 48 hours</td>
<td>256 (9%)</td>
<td>0.62 ± 0.18</td>
<td>0.94 ± 0.02</td>
<td>0.36 ± 0.17</td>
<td>0.97 ± 0.02</td>
<td>0.42 ± 0.12</td>
<td>0.43 ± 0.13</td>
</tr>
<tr>
<td>CHEO 2011</td>
<td>Summary</td>
<td>First 48 hours</td>
<td>384 (7%)</td>
<td>0.75 ± 0.12</td>
<td>0.96 ± 0.04</td>
<td>0.38 ± 0.09</td>
<td>0.99 ± 0.00</td>
<td>0.52 ± 0.00</td>
<td>0.50 ± 0.00</td>
</tr>
</tbody>
</table>

Statistics presented are mean ± standard deviation. CHEO = Children's Hospital of Eastern Ontario; CNN = Canadian Neonatal Network; MCC = Matthews correlation coefficient; Sens. = Sensitivity; Spec. = Specificity; NPV = negative predictive value; PPV = positive predictive value.

The CHEO time segments data set had 321 patients with an 8% mortality rate, but for these models it makes more sense to present the relative number of time segments from the patients who died rather than the overall mortality rate.
distinguish the patients who survived more easily from those who died leading to the observed increase in specificity.

Our clinical partners agree that a risk estimation model that provides an accurate mortality risk estimate 48 hours after a patient’s admission to the NICU would be clinically useful. During the first few days of an infant’s admission, the medical team will typically watch the baby’s response to therapy, and gather more information about complications before trying to base patient management decisions on predicted outcomes.

5.4.6 ANN Model Comparison

The MIRG ANN research framework was used to generate a mortality risk estimation model to compare with the performance of the best DT model, which used summary data from the first 48 hours of admission to the NICU. Using the same CHEO 2010 data set with 18 SNAP attributes, the ANN research framework automatically modified the training parameters in networks from 0 to 37 hidden nodes. The ten resulting ANN models from the 5-by-2 cross validation method had mean sensitivity 68% ± 14%, and mean specificity 97% ± 3% with the number of hidden nodes ranging from 1 to 28 (mean 12 ± 9). The 48-hour summary data ANN model had both higher mean sensitivity and mean specificity compared to the 48-hour summary data CHEO 2010 DT model.

A second ANN model was created using the CHEO 2010 data set, using only the 3 SNAP attributes (lowest serum pH, lowest blood pressure, and lowest heart rate) that were identified as a minimum data set using 48-hour summary data from the CHEO 2011 data set with DTs. The resulting ANN models had mean sensitivity 62%
± 14%, and mean specificity 99% ± 2% with networks ranging from 2 to 7 hidden nodes (mean 4 ± 2). Compared to the ANN model with 18 SNAP attributes, the sensitivity was reduced and the specificity increased slightly. This confirms, with a second pattern classification technique, that a minimum data set of the 3 SNAP attributes can produce clinically useful mortality risk estimation models when using summary data from the first 48 hours after admission.

5.4.7 Imbalanced Data Set Results

Using C5.0 boosted decision trees, the performance results for methods to deal with imbalanced mortality data sets, which contains a much smaller number of mortality cases compared to survivors, are found in Table 5.12. The boosted decision trees provided better overall results compared to the single DT with C5.0, therefore, only results from the boosted trees are reported. Using random under-sampling, MCC, F1 score, and sensitivity performance decreased as the percentage of the less prevalent outcome increased. Specificity remained very similar for all methods. For random over-sampling, MCC, F1 score, and sensitivity increased up to 30% of the less prevalent outcome then decreased at 40% and beyond. The F1 score and MCC increased for the combination of random over and under-sampling as the percentage of the less prevalent outcome increased while sensitivity had a downward trend in the same period. The best performance measured by MCC and F1 score were achieved using 50% random over and under-sampling, the best sensitivity was with 10% random over-sampling, and best specificity was with 50% random under-sampling.

When using boosted decision trees with C5.0 using IVH data, no decision tree could be generated from the raw non-resampled data. Performance results show
Table 5.12: Imbalanced Data Results for Mortality

<table>
<thead>
<tr>
<th>Method</th>
<th>MCC</th>
<th>F1 score</th>
<th>Sensitivity</th>
<th>Specificity</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Resample</td>
<td>0.36</td>
<td>0.33</td>
<td>0.62</td>
<td>0.96</td>
</tr>
<tr>
<td>Under 10%</td>
<td>0.45</td>
<td>0.47</td>
<td>0.52</td>
<td>0.97</td>
</tr>
<tr>
<td>Under 20%</td>
<td>0.41</td>
<td>0.44</td>
<td>0.37</td>
<td>0.98</td>
</tr>
<tr>
<td>Under 30%</td>
<td>0.37</td>
<td>0.39</td>
<td>0.27</td>
<td>0.98</td>
</tr>
<tr>
<td>Under 40%</td>
<td>0.35</td>
<td>0.35</td>
<td>0.23</td>
<td>0.99</td>
</tr>
<tr>
<td>Under 50%</td>
<td>0.31</td>
<td>0.31</td>
<td>0.20</td>
<td>0.99</td>
</tr>
<tr>
<td>Over 10%</td>
<td>0.40</td>
<td>0.38</td>
<td><strong>0.64</strong></td>
<td>0.96</td>
</tr>
<tr>
<td>Over 20%</td>
<td>0.38</td>
<td>0.38</td>
<td>0.56</td>
<td>0.96</td>
</tr>
<tr>
<td>Over 30%</td>
<td>0.42</td>
<td>0.41</td>
<td>0.60</td>
<td>0.97</td>
</tr>
<tr>
<td>Over 40%</td>
<td>0.38</td>
<td>0.37</td>
<td>0.57</td>
<td>0.96</td>
</tr>
<tr>
<td>Over 50%</td>
<td>0.34</td>
<td>0.34</td>
<td>0.51</td>
<td>0.96</td>
</tr>
<tr>
<td>Over &amp; Under 10%</td>
<td>0.38</td>
<td>0.38</td>
<td>0.57</td>
<td>0.96</td>
</tr>
<tr>
<td>Over &amp; Under 20%</td>
<td>0.37</td>
<td>0.39</td>
<td>0.48</td>
<td>0.97</td>
</tr>
<tr>
<td>Over &amp; Under 30%</td>
<td>0.39</td>
<td>0.41</td>
<td>0.44</td>
<td>0.97</td>
</tr>
<tr>
<td>Over &amp; Under 40%</td>
<td>0.42</td>
<td>0.45</td>
<td>0.46</td>
<td>0.97</td>
</tr>
<tr>
<td>Over &amp; Under 50%</td>
<td><strong>0.45</strong></td>
<td><strong>0.48</strong></td>
<td>0.41</td>
<td>0.98</td>
</tr>
</tbody>
</table>

(see Table 5.13) that no major changes in specificity were seen. MCC and F1 score increased when going from 10% random under-sampling to 20% random under-sampling, but then gradually decreased as the percentage of the less prevalent outcome increased. The specificity continuously decreased over the same range. There was a significant drop in MCC, F1 score, and specificity going from 10% random over-sampling to 20%, but then gradually increased as the percentage of the less prevalent outcome increased. While the specificity dropped using the random over and under-sampling method as the percentage of the less prevalent outcome increased, the MCC and F1 score gradually increased over the same period. The best performance measured by MCC was achieved using 30% random under-sampling, the best F1 score was with 20% random under-sampling, the best sensitivity was with 10% random over-sampling, and best specificity was with 50% random under-sampling.
CHAPTER 5. RESULTS

Table 5.13: Imbalanced Data Results for IVH

<table>
<thead>
<tr>
<th>Method</th>
<th>MCC</th>
<th>F1 score</th>
<th>Sensitivity</th>
<th>Specificity</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Resample</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Under 10%</td>
<td>0.16</td>
<td>0.19</td>
<td>0.21</td>
<td>0.97</td>
</tr>
<tr>
<td>Under 20%</td>
<td>0.23</td>
<td><strong>0.26</strong></td>
<td>0.19</td>
<td>0.98</td>
</tr>
<tr>
<td>Under 30%</td>
<td><strong>0.24</strong></td>
<td>0.25</td>
<td>0.16</td>
<td>0.98</td>
</tr>
<tr>
<td>Under 40%</td>
<td>0.23</td>
<td>0.24</td>
<td>0.15</td>
<td>0.98</td>
</tr>
<tr>
<td>Under 50%</td>
<td>0.22</td>
<td>0.22</td>
<td>0.13</td>
<td><strong>0.99</strong></td>
</tr>
<tr>
<td>Over 10%</td>
<td>0.10</td>
<td>0.07</td>
<td><strong>0.35</strong></td>
<td>0.96</td>
</tr>
<tr>
<td>Over 20%</td>
<td>0.04</td>
<td>0.03</td>
<td>0.13</td>
<td>0.96</td>
</tr>
<tr>
<td>Over 30%</td>
<td>0.07</td>
<td>0.07</td>
<td>0.16</td>
<td>0.96</td>
</tr>
<tr>
<td>Over 40%</td>
<td>0.08</td>
<td>0.08</td>
<td>0.21</td>
<td>0.96</td>
</tr>
<tr>
<td>Over 50%</td>
<td>0.11</td>
<td>0.10</td>
<td>0.22</td>
<td>0.97</td>
</tr>
<tr>
<td>Over &amp; Under 10%</td>
<td>0.16</td>
<td>0.14</td>
<td>0.34</td>
<td>0.97</td>
</tr>
<tr>
<td>Over &amp; Under 20%</td>
<td>0.20</td>
<td>0.22</td>
<td>0.27</td>
<td>0.97</td>
</tr>
<tr>
<td>Over &amp; Under 30%</td>
<td>0.16</td>
<td>0.19</td>
<td>0.20</td>
<td>0.97</td>
</tr>
<tr>
<td>Over &amp; Under 40%</td>
<td>0.17</td>
<td>0.21</td>
<td>0.18</td>
<td>0.97</td>
</tr>
<tr>
<td>Over &amp; Under 50%</td>
<td>0.20</td>
<td>0.23</td>
<td>0.18</td>
<td>0.97</td>
</tr>
</tbody>
</table>

The results were different between mortality and IVH data sets. This shows that the best methods to deal with imbalanced data can change depending on the data used. To obtain the best possible classifier, several methods to deal with imbalanced data should be tried to find the one that best suits the data and classifier being used for each unique data set.

5.4.8 Risk Category Results

The C5.0 DT mortality model created with boosting and 10% over-sampled data from the CNN data set was used to process the test set with $N=221$ mortality cases (5%) and $N=4,313$ non-mortality cases (95%). This DT model, chosen because it produced the best results from the CNN data set, had 64% sensitivity and 96% specificity. The data were processed with the DT using 4 different threshold settings, and estimated
there would be 92 mortality cases and 4,442 non-mortality cases in the test set. The results listed in Table 5.14 show that the DT confidence probability threshold chosen will have a varying degree of impact on the amount of true mortality cases categorised as low risk, and the true non-mortality cases categorised as high risk for mortality. A threshold of Low $\leq$ 20% and High $\geq$ 80% for example, means that if the DT result had a confidence probability of 20% or lower, the result would be classified in the low risk category, if the DT confidence probability was 90% or higher, the result would be classified in the high risk category, and any probability in between would be classified in the moderate risk category. With the threshold set to Low $\leq$ 10% and High $\geq$ 90%, 79% of the mortality cases are listed as high or moderate risk for mortality while 82% of the non-mortality cases are listed as low risk. Increasing the threshold to Low $\leq$ 20% and High $\geq$ 80% captures 64% of the mortality cases as high or moderate risk, and 92% of the non-mortality cases as low risk. A threshold of Low $\leq$ 30% and High $\geq$ 70% flags 49% of the mortality cases as high or moderate risk, and 96% of the non-mortality cases as low risk. Finally, the Low $\leq$ 40% and High $\geq$ 60% threshold captures 36% of the mortality cases as high or moderate risk and 98% of the non-mortality cases as low risk.

As the range of Low and High risk thresholds increase, the percentage of mortality cases categorised as high or moderate risk decrease, but the percentage of non-mortality cases categorised as low risk increases. A balance needs to be selected to present physicians with enough mortality cases in the high or moderate categories to make the estimations clinically useful, but not so high as to categorise too many non-mortality patients out of the low risk category. Our clinical partners chose the Low $\leq$ 20% and High $\geq$ 80% threshold to use as it categorises almost twice as many
## Table 5.14: Risk Category Threshold Results

<table>
<thead>
<tr>
<th>Case Outcome</th>
<th>Risk Threshold</th>
<th>Estimated Mortality Risk</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>High</td>
<td>Moderate</td>
</tr>
<tr>
<td><strong>Low</strong></td>
<td>Low ≤ 10% and High ≥ 90%</td>
<td>5 (2%)</td>
<td>169 (77%)</td>
</tr>
<tr>
<td><strong>Mortality</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Low ≤ 20% and High ≥ 80%</td>
<td>7 (3%)</td>
<td>135 (61%)</td>
</tr>
<tr>
<td></td>
<td>Low ≤ 30% and High ≥ 70%</td>
<td>20 (9%)</td>
<td>89 (40%)</td>
</tr>
<tr>
<td></td>
<td>Low ≤ 40% and High ≥ 60%</td>
<td>37 (17%)</td>
<td>43 (19%)</td>
</tr>
<tr>
<td><strong>Non-</strong></td>
<td>Low ≤ 10% and High ≥ 90%</td>
<td>1 (0%)</td>
<td>756 (19%)</td>
</tr>
<tr>
<td><strong>Mortality</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Low ≤ 20% and High ≥ 80%</td>
<td>1 (0%)</td>
<td>337 (8%)</td>
</tr>
<tr>
<td></td>
<td>Low ≤ 30% and High ≥ 70%</td>
<td>4 (0%)</td>
<td>150 (4%)</td>
</tr>
<tr>
<td></td>
<td>Low ≤ 40% and High ≥ 60%</td>
<td>15 (0%)</td>
<td>65 (2%)</td>
</tr>
</tbody>
</table>

Mortality cases as high or moderate risk (64%) compared to low risk (36%), and still selects almost all (92%) of non-mortality cases as low risk. If you move up one more threshold level to Low ≤ 30% and High ≥ 70%, the mortality cases are split almost evenly between high or moderate levels, and low risk levels, which makes it too difficult to discriminate between the levels.

### 5.4.9 Real-Time Data Usage Results

To calculate the total time required to generate a mortality risk estimation, two steps need to be completed. First, the data for the specific patient must be queried and extracted from the CDR. Second, the data must then be used in the DT model to create a classification result and confidence probability. The results of processing data from Patients X, Y, and Z three times to calculate a mortality risk estimation are found in Table 5.15. The mean time to extract 22,253 data points for Patient X during a 48-hour time period was 0.17 seconds. The DT processed the data to generate a result in a mean of 0.09 seconds for a mean total of 0.26 seconds. With Patient Y, 19,527 data points were extracted from the CDR in a mean time of 0.18
Table 5.15: Timing Results of Calculating Mortality Risk Estimations from 3 Patients

<table>
<thead>
<tr>
<th>Patient</th>
<th>CDR Query Time (sec)</th>
<th>DT Model Time (sec)</th>
<th>Total Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>0.17 ± 0.05</td>
<td>0.09 ± 0.00</td>
<td>0.26 ± 0.05</td>
</tr>
<tr>
<td>Y</td>
<td>0.18 ± 0.07</td>
<td>0.09 ± 0.00</td>
<td>0.27 ± 0.07</td>
</tr>
<tr>
<td>Z</td>
<td>0.16 ± 0.03</td>
<td>0.09 ± 0.00</td>
<td>0.24 ± 0.03</td>
</tr>
</tbody>
</table>

seconds and processed by the DT in a mean of 0.09 seconds for a mean total of 0.27 seconds. Patient Z’s 13,307 data points were extracted in a mean time of 0.16 seconds and processed by the DT in a mean of 0.09 seconds for a mean total of 0.24 seconds.

These results show that even with more than a year’s worth of patient data (62 million data points), the hybrid EAV design of the CDR along with the C5.0 DT mortality model were able to produce a result from start to finish in an average of one quarter of a second for randomly selected patients. The performance of this CDS system easily meets the requirements of providing results in real-time, which for the purpose of this thesis, was defined as completing a task within a one minute period. The real-time CDR and mortality risk estimation models provide the necessary framework to integrate patient-specific risk estimations in a CDS system that can be enabled to generate intelligent alerts to notify medical staff of important clinical events or a change in the status of a patient. An early prototype of a CDS system that provides patient-specific decision support for both physicians and parents to improve parent/clinician communication has been developed (Frize et al., 2011).

5.4.10 Updating Neonatal Risk Models Results

Risk estimation models in CDS systems need to be updated as new data becomes available in the future. For example, if a neonatal mortality risk estimation model was
created for the NICU at CHEO using the CHEO 2010 data set, after one year of use there would be another year’s worth of patient data available to create a potentially improved model. The design of the real-time CDR and DT risk estimation models allows this to be done in a simple, efficient, and mostly automated way. We have developed a risk model update strategy, taking advantage of the open source nature of the CDR components and the industry standard features such as the SQL query language. Using the 48-hour summary data mortality model created from the CHEO 2011 data set as an example, the risk model can be updated in five steps:

1. A SQL query was constructed once and can be re-used to process patient data on the CDR. The SQL query had 47 components that extracted the 18 required SNAP attributes from every patient for the first 48 hours after admission to the NICU. The data for each patient was then stored in a temporary database table on the CDR with one record per patient and 20 columns, which included the generated StudyID, the mortality outcome of the patient, and the 18 SNAP attributes. The SQL code to create the temporary database table is found in Appendix C.1 and the sample SQL query to accomplish this step is presented in Appendix D.4. This SQL query processed the 62 million data points from 568 patients in 47 minutes and 55 seconds.

2. A JavaScript component can be written in the open source MIRTH HL7 processing tool that uses SQL to extract every patient from the CDR temporary table that has data for at least 50% of the attributes, ignoring those that have less. First, it would extract all of the mortality patients, and store them in a comma-separated value flat file called mortality.csv. Second, it would extract all of the non-mortality patients, and store them in a comma-separated value
flat file called nonmortality.csv. These comma-separated flat files are the format that the C5.0 DT software is capable of reading. This step could also be accomplished by writing code in another programming language, such as C or C++, that has libraries capable of connecting to SQL compliant databases.

3. A script was written to process the two .csv flat files generated in the previous step to automatically create the necessary 5-by-2 cross validation training and test sets. It first randomised the mortality and non-mortality patients, then splits each file in half for train and test sets. The mortality and non-mortality training files were merged into one training data set called mortality_fold_1a.data. The mortality and non-mortality test files were merged into one test data set called mortality_fold_1a.test. The training set was then copied to mortality_fold_1b.test and the test set was copied to mortality_fold_1b.data. This process was repeated 4 more times to generate all ten 5-by-2 cross validation data sets for the C5.0 tool. A sample Bourne-Again Shell (BASH) script to accomplish this step is presented in Appendix D.1. The script takes approximately 8 seconds to generate all ten data sets.

4. The C5.0 tool is available as an open-source command line tool, which can be compiled and used on any platform with a C compiler. This made it possible for a script to be created that called the C5.0 tool to process the ten data sets that were created in the previous step. The script called the see5x executable using the correct command line options to create decision trees for each of the data files and output the results to a new file for review. This process was repeated adding the -b command line option to create boosted decision trees to see whether boosted or single DTs gave better results. A sample BASH
script to complete this step is presented in Appendix D.2. The script takes approximately 2 seconds to generate ten single decision trees and ten boosted decision trees from the data sets.

5. Steps 1 through 4 can be completely automated and scheduled to run on a regular basis such as monthly, quarterly, or yearly depending on the needs of the researchers or the users of the CDS system. The final step requires human intervention to review the results of the DT model output to ensure that the DTs created are clinically useful and that there were no errors in the process. If the new DTs are an improvement, the generated DT created by the C5.0 tool can be used to overwrite the existing DT in the real-time risk estimation model. With the new DT in place, the risk estimation code does not need to be updated, and the new DT will be used immediately without any further changes required. The next time a risk estimation is made using the CDS, the new improved DT will automatically be used by the system.

This strategy shows that the combination of real-time CDR and mortality risk estimation models in our CDS system can be used to update risk estimation models whenever required. Four out of the five steps for updating a model can be completely automated to provide a simple and efficient method to update models when new data are available. The automation also reduces the chance of error with all of the data processing and manipulation that is required, if the process was done manually.

5.4.11 Intended Use of Models

It is important to note that the risk estimations from these real-time mortality models should not solely be used for making decisions. They are intended to be an additional
piece of information for the clinical team to use along with laboratory results, examinations, and other test results. Our clinical partner intends to use the risk estimation results at the beginning of her shift to help determine which patients need attention first, and as a way to quickly determine if patients are improving or deteriorating.

An early prototype of a CDS system that provides patient-specific decision support for both physicians and parents to improve parent/clinician communication was developed using the CDR and real-time risk estimation models (Frize et al., 2011). Physicians are presented with a category of High, Moderate, or Low mortality risk for each patient. A specific confidence value is also presented, which indicates the percentage the mortality model is confident in its result. The sensitivity and specificity of the mortality model are provided so that physicians have enough information to decide how much weight to give to the model results compared to the rest of their evaluation of the patient. Colour coded icons allow the clinician to quickly see if the mortality risk estimation of the patient has increased or decreased, and by how much in the preceding 24 hours. A physician usability study for the prototype CDS system was conducted with neonatologists who work in the NICU at CHEO (Weyand et al., 2011; Weyand, 2011). The study showed that neonatologists found the tool easy to learn and use. They also found the tool would be useful for parents and physicians, and would be willing to use it, given the opportunity.
Chapter 6

Concluding Remarks

This chapter summarises the conclusions reached from the results presented in Chapter 5, specifies the contributions to knowledge provided by this research work, and describes ideas for future research work.

6.1 Conclusions

There were two main objectives of this thesis work: (1) to research a method to collect, store, and retrieve clinical data in a real-time environment, and (2) to devise neonatal mortality risk estimation models that used the collected data in real-time to assist clinical decision making. This section describes the important findings of this work and shows that the objectives were successfully completed.

1. A novel hybrid EAV data storage format that combined the advantages of two popular medical storage formats was successfully created and tested. It used similar storage space, but had less complex queries, and allowed faster access to large amounts of data for risk estimation models to operate in real-time. The
hybrid data storage format was capable of remaining efficient even after storing several years' worth of data with tens of millions of data points. The hybrid EAV format is very suitable for developing a CDR. The flexible nature of EAV will allow new medical equipment output and data types to be easily stored in the CDR for use in clinical decision support and research. Each patient being monitored has a different set of attributes associated with him/her, allowing for efficient use of space with this design. Queries performed on the CDR for clinical decision support will be related to a single patient, which EAV is optimised to handle.

2. A new approach to CDRs was designed that pulls information together from multiple systems into a single location using free open-source tools and industry standard protocols. The hybrid EAV storage format was integrated into the CDR design, which accommodates the fast changing medical domain, and supports the addition of new attributes automatically without the need for any database format changes. The data are organised for efficient use in both research and decision support. Data can be retrieved and exported from the CDR in real-time using the SQL standard, which means that any existing tools that support SQL or new software that is written to support SQL can be used to access the data in the CDR. The CDR collected more than two years' worth of data from the NICU in CHEO from patient monitors and laboratory results with over 90 million data points from over 1,000 patients. Since the CDR supports industry standards such as HL7 and SQL, and it uses the hybrid EAV structure that supports an unlimited number of attributes without having to change the database design, the CDR should be generalisable to use in any
hospital department that supports HL7 and requires storing data over time.

3. The CDR is capable of automatically segregating private patient information from raw data in real-time, before the data are stored. The private patient information is stored in a separate database table that can be restricted to only those people who need access, while the other data are de-identified using a non-attributable ID and stored in a database table with fewer restrictions. This provides the ability to give researchers and physicians access to the real-time raw data without any further processing and without violating patient privacy. **Objective 1 is complete.**

4. To help overcome the limitations of current scoring-based systems, the large amounts of patient data collected in the CDR were used to create neonatal mortality risk estimation models using DTs and ANNs that can estimate outcomes for a single patient. These novel models used real-time data from patients beyond the first 12 hours of life to provide risk estimations as new information became available. The mortality models used the most recent patient information quickly enough from the CDR to make risk estimations in real-time. We compared the performance of fifteen different neonatal mortality risk estimation DT models using 62 million data points (568 patients) in our CDR from the CHEO 2011 data set, 43 million data points (357 patients) in our CDR from the CHEO 2010 data set, and 17,364 patients from the CNN data set. The mortality model using CHEO 2011 summary data from the first 48 hours after NICU admission provided the best risk estimation model with the least number of false positives. It had a mean sensitivity of 75% with a mean specificity of 96%, exceeding the minimum performance guidelines requested from our clini-
cal partners (>60% sensitivity, >90% specificity). Three attributes were found to be most important to estimate the risk of mortality: lowest serum pH, lowest blood pressure, and lowest heart rate.

5. ANN mortality risk estimation models were also created to compare the performance against DTs. The best ANN model had slightly higher specificity at 97%, but lower sensitivity at 68% compared to the DT model. The ANN model confirmed the same three important attributes for mortality as the DT model: lowest serum pH, lowest blood pressure, and lowest heart rate.

6. We demonstrated that changing the amount of random over-sampling and under-sampling of data can effect the performance results of imbalanced data set for neonatal mortality and IVH with DTs. With mortality data from the CNN data set, the best overall results were obtained by using 10% random over-sampling of mortality cases. This increased the sensitivity of the DT model from 62% to 64% and specificity from 96% to 99%. With IVH, all of the over and under-sampling techniques produced a better model than with no re-sampling. Since the results change with each data set, it is important try several methods for dealing with imbalanced data sets to find the one that best suits the specific data set.

7. Our clinical partners prefer to see a risk category such as low, moderate, and high instead of only a probability value when the DT returns a result from a risk estimation. Four different risk thresholds were tested, and a risk category threshold of Low ≤ 20% and High ≥ 80% probability was chosen by our clinical partners. If using too broad a threshold range for the moderate category, too
many non-mortality patients are categorised as moderate risk, and if too narrow a threshold range is chosen, too many mortality cases are categorised as low risk for mortality. With the chosen threshold, almost twice as many mortality cases were categorised as high or moderate risk (64%) compared to low risk (36%), and still selects almost all (92%) of non-mortality cases as low risk.

8. We demonstrated that the combined CDR and CDS system can provide mortality risk estimation results in real-time (within one minute). By using three randomly selected patients in the CDR from the CHEO 2011 data set, we showed that the CDR could find and extract the 18 SNAP attributes necessary for the specified 48-hour time period of each patient, input the data into the DT mortality risk estimation model, and produce a result in less than 0.3 seconds on average. This performance level far exceeds our criteria of a one minute maximum delay for real-time, and will allow physicians or an automated CDS system to generate risk estimations of patient specific outcomes on demand.

9. To help physicians with the overload of information they face, the risk estimation models integrated the relevant available information to provide more information than the simple threshold-based alerts such as “heart rate low” that now occur in the NICU. Supporting multiple parameters more effectively used all the information that was available to provide physicians with an overall mortality risk assessment for individual patients. The mortality models can be integrated into a CDS system to provide physicians with access to patient-specific risk estimations of clinical outcomes in real-time. Real-time support facilitates the ability to update the risk estimation models in a mostly automated fashion when new data are available. This has the added benefit of reducing
the chance of error since the data do not need to be manually extracted or
manipulated to produce a new model. **Objective 2 is complete.**

### 6.2 Contributions to Knowledge

1. Designed and tested a new method to collect, store, and retrieve large amounts of data in real-time. The resulting new hybrid EAV data storage format combines the benefits of the simple EAV format and the multi-data type EAV format. It uses less complex queries, and provides faster access to the data, while using the same amount of storage space than previous EAV formats.

2. Facilitated medical research by pulling information together from multiple systems into a single location using free open-source tools and industry standard protocols. This new approach for CDR accommodates the fast changing medical domain by supporting the addition of new attributes automatically without the need for any database format changes. It is capable of providing researchers and physicians with real-time access to new and stored data without the need for further processing and without violating patient privacy, because it automatically segregates private patient information from raw data in real-time before the data are stored. We provided evidence that the hybrid EAV storage format remains efficient and continues to provide real-time access after storing years’ worth of data with tens of millions of data points. With each patient using 10 MB of storage (106,903 data points) on average, the CDR can store data for approximately 200,000 patients on a single common 2000 GB hard drive. The CHEO NICU would take approximately 500 years to reach this capacity with
their current admission rate.

3. Developed mortality risk estimation models using DTs and ANNs that are capable of estimating patient-specific outcomes in **real-time**. Previous research used the CNN database, collected within the first 12 hours of birth. This new approach used real-time data from patients beyond the first 12 hours of life to produce risk estimations as new information became available. This provides the unique ability to detect clinical events as they happen, and deliver immediate feedback to physicians. The best DT model had 75% sensitivity and 96% specificity, and surpassed the minimum clinical usefulness criteria set by our clinical partners, which was greater than 60% sensitivity and 90% specificity. The best ANN model had 62% sensitivity and 99% specificity, which also surpassed the minimum clinical usefulness criteria.

4. Provided evidence that data prior to the last 48 hours of a patient stay in the NICU contains important information for estimating the risk of mortality, since the DT mortality model decreased significantly in sensitivity (66% down to 49%) when these data were removed.

5. Provided further evidence that methods such as under-sampling and over-sampling data in imbalanced data sets can lead to improved results from pattern classification methods such as DTs.

6. Demonstrated that the CDR and mortality risk-estimation models provide a framework for a CDS system that can operate in real-time by extracting all necessary data from randomly selected patients in the CDR, and using that data to calculate risk estimation and to provide a result in less than a minute.
7. Demonstrated that support for real-time data in the design of the CDR and risk estimation models facilitate the ability to update the models in a mostly automated fashion when new data is available.

The combined CDR and CDS system is a unique research tool for clinical use and research purposes using data from infants in the NICU. It estimates critical clinical outcomes in real-time, and will provide alerts and warnings in the prototype implementation of the work.

6.3 Future Research

1. MIRG’s vision for a clinically useful CDS system would allow a physician to request a risk estimation for a patient on demand, or the CDS system would automatically check a risk estimation to see if an alert should be generated. In each case, the models would use data from the most recent time segment to make its prediction. A prototype should be implemented of the real-time CDS system framework with the ability to generate intelligent alerts and warnings from medical events as they occur, such as when the risk estimation for a patient changes significantly, or when a specified set of conditions for a specific patient transpires.

2. In conjunction with our clinical partners, we intend to run a clinical trial of the prototype using patients in the NICU at CHEO.

3. Try combining the use of both the time segment mortality model, which provided high specificity and NPV, with the higher sensitivity summary data model in a two stage approach. First, estimate the mortality risk with time segment
model. If the model classifies the patient as non-mortality, report a result of low risk for mortality and stop. If instead, the model classifies as mortality, the data are re-calculated using the summary data model, which performs better in this case, and present that model’s result as low, moderate, or high risk. Examine the overall results of the combined models to see if it improves performance compared to the summary data model alone.

4. Using the real-time combined CDR and CDS system framework as a foundation, explore creating real-time models for other outcomes such as length of stay, artificial ventilation duration, and a number of complications such as necrotizing enterocolitis (NEC), intraventricular hemorrhage (IVH), and bronchopulmonary dysplasia (BPD).

5. For physicians to better understand how the risk estimation is being calculated in the CDS system, explore additional information to present with the model results. One example could be displaying important portions of the decision tree with the risk estimation. This would allow physicians to quickly see that the risk estimation for an outcome such as mortality was obtained because the results from that specific patient followed the given path down the tree with various vital sign and lab results being above or below certain values.

6. Currently, we have only received the patient discharge diagnoses from the first three months of 2011 at CHEO. When the full data set for 2011 is available, new models should be investigated that use both the 2010 and 2011 data, and also compare 2010 to 2011.

7. Expand beyond the current 18 SNAP attributes used for generating the mor-
tality models. In the future, as other attributes become available for collection in the CDR, test their usefulness for estimating the risk of mortality and other outcomes.

8. Explore creating real-time models using techniques other than decision trees, such as recurrent artificial neural networks that can take advantage of multiple sets of data points per patient. Also investigate the use of time discords that analyse time series data to find sub-sequences that are least similar to the global time sequence. The ability to detect discords in medical time series may help improve the risk estimation of patient outcomes, and could be further explored.

9. A limitation of these real-time mortality models was that they required patient cases to have data for at least 50% of the attributes, which biased the data selected for developing the model. The next steps could be to include less ill infants from the testing data set to see whether the risk estimation model can be useful for assessing their mortality risk.

10. Confirm how well the CDR design generalises by testing the performance of the CDR and hybrid EAV format using different types of clinical data from different institutions. Also, to better understand how the performance is affected by various aspects of computer hardware such as CPU cache sizes and system memory, the performance should be tested on different hardware configurations.
References


REFERENCES


REFERENCES


Gilchrist, J., M. Frize, C. M. Ennett, and E. Bariciak (2011b, October). Perfor-
REFERENCES


REFERENCES


REFERENCES

on ACSW frontiers, Darlinghurst, Australia, pp. 227–232. Australian Computer Society, Inc.


REFERENCES

Master’s thesis, School of Information Technology and Engineering, University of Ottawa, Ottawa, Ontario, Canada.


Appendix A

Publications

The following are publications derived from this work, portions of which were used in Chapters 2, 3, 4, and 5:


Appendix B

Data Collection at CHEO

B.1 CNN and CHEO Database Statistics

Table B.1 provides an overview of the median values recorded in the CNN and CHEO databases for the patient populations used for the mortality risk estimation model development.

B.2 CHEO Risk Estimation Data Set Statistics

Table B.2 shows the number of patients that had data for each attribute of interest, the percentage of patients that were missing those attributes, minimum, maximum, and median values for individual attributes.
## APPENDIX B. DATA COLLECTION AT CHEO

### Table B.1: CNN and CHEO Database Statistics

<table>
<thead>
<tr>
<th>Attributes</th>
<th>CNN (1st 12 Hours)</th>
<th>CNN (1st 12 Hours)</th>
<th>CHEO (1st 12 Hours)</th>
<th>CHEO (Entire Stay)</th>
<th>CHEO (Entire Stay)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># Patients</td>
<td># Patients</td>
<td># Patients</td>
<td># Patients</td>
<td># Patients</td>
</tr>
<tr>
<td>total number of patients</td>
<td>17364</td>
<td>333</td>
<td>37.80%</td>
<td>357</td>
<td></td>
</tr>
<tr>
<td>female patients (%)</td>
<td>41.80%</td>
<td>38.40%</td>
<td>37.80%</td>
<td>135</td>
<td></td>
</tr>
<tr>
<td>highest mean blood pressure (mm Hg)</td>
<td>47.0 ± 14.0</td>
<td>59.0 ± 14.0</td>
<td>79.0 ± 26.0</td>
<td>343</td>
<td></td>
</tr>
<tr>
<td>lowest mean blood pressure (mm Hg)</td>
<td>36.0 ± 12.0</td>
<td>45.0 ± 14.0</td>
<td>37.0 ± 16.0</td>
<td>343</td>
<td></td>
</tr>
<tr>
<td>lowest white blood cell count (x10^9/L)</td>
<td>13.6 ± 9.5</td>
<td>12.4 ± 7.2</td>
<td>9.1 ± 5.8</td>
<td>244</td>
<td></td>
</tr>
<tr>
<td>highest serum glucose (mmol/L)</td>
<td>4.7 ± 2.1</td>
<td>5.3 ± 2.0</td>
<td>5.9 ± 2.9</td>
<td>320</td>
<td></td>
</tr>
<tr>
<td>lowest serum glucose (mmol/L)</td>
<td>3.2 ± 1.7</td>
<td>4.5 ± 1.7</td>
<td>3.7 ± 1.3</td>
<td>320</td>
<td></td>
</tr>
<tr>
<td>highest heart rate (bpm)</td>
<td>160.0 ± 21.0</td>
<td>179.0 ± 23.0</td>
<td>206.0 ± 25.0</td>
<td>345</td>
<td></td>
</tr>
<tr>
<td>lowest heart rate (bpm)</td>
<td>126.0 ± 18.0</td>
<td>102.0 ± 26.3</td>
<td>73.0 ± 28.0</td>
<td>345</td>
<td></td>
</tr>
<tr>
<td>lowest absolute neutrophil count (x10^9/L)</td>
<td>7.2 ± 7.9</td>
<td>6.8 ± 7.0</td>
<td>3.8 ± 5.2</td>
<td>287</td>
<td></td>
</tr>
<tr>
<td>highest immature total neutrophil ratio</td>
<td>0.11 ± 0.2</td>
<td>0.19 ± 0.2</td>
<td>0.11 ± 0.2</td>
<td>287</td>
<td></td>
</tr>
<tr>
<td>highest blood hematocrit (%)</td>
<td>50.0 ± 12.0</td>
<td>45.4 ± 0.1</td>
<td>44.7 ± 0.1</td>
<td>292</td>
<td></td>
</tr>
<tr>
<td>lowest blood hematocrit (%)</td>
<td>49.0 ± 11.0</td>
<td>45.0 ± 0.1</td>
<td>38.8 ± 0.2</td>
<td>292</td>
<td></td>
</tr>
<tr>
<td>lowest platelet count (x10^9/L)</td>
<td>212.0 ± 102.0</td>
<td>233.0 ± 125.0</td>
<td>222.0 ± 152.0</td>
<td>281</td>
<td></td>
</tr>
<tr>
<td>lowest serum pH (mm Hg)</td>
<td>7.3 ± 0.1</td>
<td>7.3 ± 0.1</td>
<td>7.3 ± 0.1</td>
<td>268</td>
<td></td>
</tr>
<tr>
<td>highest pCO2 level (mm Hg)</td>
<td>44.0 ± 15.0</td>
<td>40.0 ± 12.0</td>
<td>45.0 ± 15.0</td>
<td>269</td>
<td></td>
</tr>
<tr>
<td>lowest pO2 level (mm Hg)</td>
<td>53.0 ± 28.0</td>
<td>65.0 ± 40.5</td>
<td>54.0 ± 32.0</td>
<td>269</td>
<td></td>
</tr>
<tr>
<td>highest respiratory rate (rpm)</td>
<td>62.0 ± 30.0</td>
<td>120.0 ± 25.8</td>
<td>129.0 ± 1.0</td>
<td>344</td>
<td></td>
</tr>
<tr>
<td>highest serum sodium (mEq/L)</td>
<td>136.0 ± 6.0</td>
<td>134.0 ± 7.0</td>
<td>139.0 ± 5.0</td>
<td>324</td>
<td></td>
</tr>
<tr>
<td>lowest serum sodium (mEq/L)</td>
<td>136.0 ± 6.0</td>
<td>133.0 ± 7.0</td>
<td>132.0 ± 7.0</td>
<td>324</td>
<td></td>
</tr>
</tbody>
</table>

Values presented are median ± inter quartile range (IQR), unless otherwise noted.

Feature acronyms: pH = measure of acidity or alkalinity of blood sample; pCO2 = blood carbon dioxide level; pO2 = blood oxygen level.

Unit definitions: bpm = beats per minute; L = liters; mEq = milliequivalents; mm Hg = millimeters of mercury; mmol = millimole; rpm = respirations per minute.
### Table B.2: CHEO Risk Estimation Data Set Statistics

<table>
<thead>
<tr>
<th>Attributes</th>
<th># Patients</th>
<th>% Missing</th>
<th>Minimum Value</th>
<th>Maximum Value</th>
<th>Median ± IQR</th>
</tr>
</thead>
<tbody>
<tr>
<td>total number of patients</td>
<td>357</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>female patients</td>
<td>135</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>highest mean blood pressure (mm Hg)</td>
<td>343</td>
<td>3.9</td>
<td>27.0</td>
<td>126.0</td>
<td>79.0 ± 26.0</td>
</tr>
<tr>
<td>lowest mean blood pressure (mm Hg)</td>
<td>343</td>
<td>3.9</td>
<td>8.0</td>
<td>71.0</td>
<td>37.0 ± 16.0</td>
</tr>
<tr>
<td>lowest white blood cell count (x10⁹/L)</td>
<td>244</td>
<td>31.7</td>
<td>2.1</td>
<td>37.2</td>
<td>9.1 ± 5.8</td>
</tr>
<tr>
<td>highest serum glucose (mmol/L)</td>
<td>320</td>
<td>10.4</td>
<td>2.5</td>
<td>38.0</td>
<td>5.9 ± 2.9</td>
</tr>
<tr>
<td>lowest serum glucose (mmol/L)</td>
<td>320</td>
<td>10.4</td>
<td>0.8</td>
<td>13.0</td>
<td>3.7 ± 1.3</td>
</tr>
<tr>
<td>highest heart rate (bpm)</td>
<td>345</td>
<td>3.4</td>
<td>135.0</td>
<td>297.0</td>
<td>206.0 ± 25.0</td>
</tr>
<tr>
<td>lowest heart rate (bpm)</td>
<td>345</td>
<td>3.4</td>
<td>31.0</td>
<td>143.0</td>
<td>73.0 ± 28.0</td>
</tr>
<tr>
<td>lowest absolute neutrophil count (x10⁹/L)</td>
<td>287</td>
<td>19.6</td>
<td>0.0</td>
<td>5.1</td>
<td>3.8 ± 5.2</td>
</tr>
<tr>
<td>highest immature total neutrophil ratio</td>
<td>287</td>
<td>19.6</td>
<td>0.0</td>
<td>1.0</td>
<td>0.11 ± 0.2</td>
</tr>
<tr>
<td>highest blood hematocrit (%)</td>
<td>292</td>
<td>18.2</td>
<td>23.9</td>
<td>74.9</td>
<td>44.7 ± 0.1</td>
</tr>
<tr>
<td>lowest blood hematocrit (%)</td>
<td>292</td>
<td>18.2</td>
<td>16.8</td>
<td>65.9</td>
<td>38.8 ± 0.2</td>
</tr>
<tr>
<td>lowest platelet count (x10⁹/L)</td>
<td>281</td>
<td>21.3</td>
<td>4.0</td>
<td>630.0</td>
<td>222.0 ± 152.0</td>
</tr>
<tr>
<td>lowest serum pH (mm Hg)</td>
<td>268</td>
<td>24.9</td>
<td>6.8</td>
<td>7.5</td>
<td>7.3 ± 0.1</td>
</tr>
<tr>
<td>highest pCO₂ level (mm Hg)</td>
<td>269</td>
<td>24.6</td>
<td>24.0</td>
<td>133.0</td>
<td>45.0 ± 15.0</td>
</tr>
<tr>
<td>lowest pO₂ level (mm Hg)</td>
<td>269</td>
<td>24.6</td>
<td>12.0</td>
<td>174.0</td>
<td>54.0 ± 32.0</td>
</tr>
<tr>
<td>highest respiratory rate (rpm)</td>
<td>344</td>
<td>3.6</td>
<td>62.0</td>
<td>129.0</td>
<td>129.0 ± 1.0</td>
</tr>
<tr>
<td>highest serum sodium (mEq/L)</td>
<td>324</td>
<td>9.2</td>
<td>120.0</td>
<td>179.0</td>
<td>139.0 ± 5.0</td>
</tr>
<tr>
<td>lowest serum sodium (mEq/L)</td>
<td>324</td>
<td>9.2</td>
<td>112.0</td>
<td>149.0</td>
<td>132.0 ± 7.0</td>
</tr>
</tbody>
</table>

*a* Feature acronyms: pH = measure of acidity or alkalinity of blood sample; pCO₂ = blood carbon dioxide level; pO₂ = blood oxygen level.

*b* Unit definitions: bpm = beats per minute; L = liters; mEq = milliequivalents; mm Hg = millimeters of mercury; mmol = millimole; rpm = respirations per minute.
B.3 Evolution of Data Collection Systems at CHEO (2001 - Present)

Several data collection systems have been used in the NICU at CHEO since 2001. Initially a Philips DocVue system was installed that provided short-term clinical trends, which was replaced by a Philips Information Centre system capable of exporting patient data to an external system. Finally, a new NICU unit was built in which all new equipment provided by Draeger Medical was installed. In this section, the evolution of the data collection systems at CHEO over the past ten years is presented.

B.3.1 Philips DocVue Data Collection System (2001 - Sep 2008)

B.3.1.1 System Overview

The Philips DocVue data collection system was installed in 2001. It collected data from patient monitors in the NICU along with data from ADT, and laboratory (lab) systems. This system had a number of limitations that impacted the development of a CDS system. Some of the challenges were the lack of support from the vendors for the aging components, security issues with the unsupported operating system of the server, the inability to easily extract data from the system, and the limited storage capability (Gilchrist et al., 2008). The Philips DocVue system collected vital sign data such as heart rate, blood pressure, respiration rate, and blood SpO₂ levels in real-time, and sent it to a database server as shown in Figure B.1. Patient information and test results from ADT and lab systems were also sent to the DocVue database server.
A DocVue database client was used to view the patient data stored on the database server. This client showed clinical information about a patient such as admission and discharge dates, and the trend graphs of vital signs such as heart rate and blood pressure for a period of up to 4 days. It did not provide direct access to the raw data stored on the server. Figure B.2 shows the user interface for the DocView client.
APPENDIX B. DATA COLLECTION AT CHEO

B.3.1.2 Proprietary Technology of Philips Healthcare

A number of proprietary technologies were used in the Philips DocVue system which made it difficult to upgrade and to interface with the system properly. The network that connected the patient monitors to the database server was a proprietary Philips SDN technology that required a specialised network switch, cabling, and connectors. Without the use of a standard network technology such as the TCP and the IP, only equipment designed for the SDN network could be used with the system. This technology had limited adaptability to new medical equipment, which restricted the
number of new devices that could be integrated into the old system.

The database server itself used a proprietary DocVue database system that could only be accessed via the client software built for that database. Due to the proprietary nature, the data could not be directly retrieved for research analysis, and data queries were limited to those built into the existing client software. The current users of the system were limited to applications provided by the original designers since no new functionality could be added. The DocVue database client was designed to provide clinical information and trends, but could not be used for research purposes. As of 2008, the DocVue database client and server software are no longer supported by Philips Healthcare.

B.3.1.3 Unsupported Components by Microsoft

The database server ran on the Windows NT platform which is no longer supported by Microsoft. Security patches are no longer being produced for Windows NT and anti-virus and other security software are also no longer supported or updated. Due to this lack of protection, the system was vulnerable and if compromised, could start attacking and infecting other systems on the hospital network. The hospital IS department was concerned about the security implications of keeping this system connected to the hospital network.

B.3.1.4 Storage Capacity

Storage capacity of data on computer systems has greatly increased since the Philips DocVue system was installed in 2001. Compared to the 15 gigabytes that was typically available then, hard drives with thousands of gigabytes are readily available now.
APPENDIX B. DATA COLLECTION AT CHEO

With large amounts of data generated from real-time patient monitors, the design and storage capacity of the former database was limited to four days after which the oldest data was deleted. Though this was sufficient for the original design purpose, four days of data did not provide the ability to use the system for time-varying research, especially to study disease and clinical outcomes that occurred days or even weeks later.

B.3.2 Philips Information Centre (PIC) Data Collection System (Oct 2008 - Sep 2009)

B.3.2.1 System Overview

A new data collection system with updated components was installed at CHEO in October 2008 and is depicted in Figure B.3. While the proprietary SDN network and patient monitors remained due to high replacement costs, a new central collection station, called the PIC, replaced the Philips DocVue database server. The PIC was loaned by Philips Healthcare for one year. New SpO₂ monitors were connected to the main patient monitors to include their data in the collection system, which previously could not be captured after the old SpO₂ monitors were replaced. The PIC allowed the patient monitoring data of all patients to be viewed in real-time. The PIC then exported the data for each patient every minute to a CDR for long term storage and research analysis. Patient information and test results from ADT and lab systems were also sent to the CDR for storage.
Appendix B. Data Collection at CHEO

B.3.2.2 Industry Standard Technology

The new Philips Central Collection Station (CCS) and CDR both supported the TCP/IP networking standard (Socolofsky and Kale, 1991) which allowed for any modern computer or networking component to interface with the data collection system. All medical data transmitted between the new components of the data collection system.
APPENDIX B. DATA COLLECTION AT CHEO

system and CCS conformed to the HL7 industry standard for messaging (HL7, 2009). This ensured that any application that conformed to the HL7 standard could communicate with any of the components in the system. The CDR uses SQL compliant (ANSI/ISO/IEC 9005-2003, 2003) database software to store all the patient monitor data, ensuring that any software supporting SQL can be used to access data in the CDR, either locally on system or remotely over the network.

B.3.2.3 Integration Issues

Since the Philips DocVue database server would not allow external connections to access the stored data, it needed to be replaced entirely with the PIC central collection station. With support for HL7 messaging and TCP/IP networking, any compliant system could be used to receive the patient monitor data. A number of lessons were learned from this integration process which have been detailed in section B.4.

B.3.2.4 Data Collection Statistics

The CDR operated on an Intel Core2 quad-core system with 4GB of RAM and 750GB hard drive. There were no networking, processing, stability, or disk space issues during the collection period. The CDR processed over 6 million HL7 messages (approximately 200,000 data points per week) from the 16 bed NICU facility in the old unit. All raw HL7 messages were also being stored to disk as a backup in case the data needed to be re-sent to the CDR for testing or rebuilding the database. This allowed the HL7 messages to be re-processed by the CDR as if they were received over the network, simulating the real environment.

During the capture period, 308 patients were admitted to the NICU. Reporting of
laboratory results is automated, so there are data for all patients who had lab tests. Capturing of patient monitor data was not automated, and relied on clinical staff connecting data capture cables and entering patient information. There are patient monitoring data for 131 patients (43%). Discharge diagnosis reports are received for patients approximately two months after they are discharged. This is a manual process, where a monthly report is sent by the health records department and must be manually imported into the CDR. From the discharge diagnosis reports, there were 393 unique diagnosis codes.

The CDR used about 950MB of storage space (0.1% of capacity) during this one year period, showing it was capable of storing years worth of data using the current hardware. The data compressed to 65MB and was backed up daily by the hospital’s IS department. The data on the CDR system was also protected by single hard drive failure using Redundant Array of Independent Disks - Level 1 (RAID-1) which takes two hard drives and mirrors them (Advanced Computer & Network Corporation, 2009). If one hard drive fails, the data will be accessible on the second and the CDR can continue to operate as usual until a replacement disk is installed.

**B.3.3 Draeger Medical Data Collection System (Sep 2009 - Present)**

In September 2009, the NICU at CHEO was moved to a new facility with all new equipment manufactured by Draeger Medical. The patient monitors send data within the NICU’s network to a central station which then broadcasts the data via HL7 messages to the IS department’s MDLink HL7 processing server, which acts as a gateway for all HL7 messages, and then on to the CDR as seen in Figure 3.1. Data
APPENDIX B. DATA COLLECTION AT CHEO

from the Laboratory and ADT systems are also sent to the MDLink server and then on to the CDR.

Unlike the old NICU which lacked a central station for the medical staff to use, all patients are entered into the new central station as part of a routine process. This greatly increased the amount of data being collected for each patient, and also increased the accuracy, since the system records when a patient moves to a different bed location, and will encode the patient ID accordingly with the correct data. There are 20 beds in the new NICU, and each patient monitor sends data once every minute.

B.3.3.1 Industry Standard Technology

The new equipment in the NICU supports the HL7 messaging standard, making it very easy to upgrade the CDR to receive data from the new systems. Only minor modifications were required to the configuration of the CDR to accept a few additional fields in the HL7 messages from the new patient monitors. The CDR database itself did not need to be changed as the flexible EAV design and the use of an SQL standard database allowed it to continue to operate without any modification required. The use of industry standard technology and protocols has shown that even after a move to a new physical location with completely new hardware from a different manufacturer, the upgrade path to support the new system was painless and quick compared to the previous hardware upgrades and system changes at CHEO that used proprietary systems.
B.3.3.2 Updated Data Collection Statistics

The CDR is still being operated on the same Intel Core2 quad-core system with 4GB of RAM and 750GB hard drive. The system remains stable, and has not required any hardware upgrades to deal with the increased data storage or increased networking demands of the data feed in the new NICU. The CDR has processed over 85 million HL7 messages (approximately 800,000 data points per week) from the new 20 beds since the move. All raw HL7 messages continue to be stored to disk as a backup in case the data needed to be re-sent to the CDR for testing or rebuilding the database.

During the capture period, 765 patients were admitted to the NICU. The reporting of laboratory results continues to be automated, so there are data for all patients who had lab tests. Since the capturing of patient monitor data is now automated in the new NICU, any patient hooked up to a patient monitor is captured in the CDR. This automation accounts for the significant increase in captured data compared to the manual system in the previous NICU. Discharge diagnosis reports are still received manually for patients approximately two months after they are discharged and are manually imported into the CDR. There are currently 36 different physiological attributes, 828 laboratory result attributes, and 622 unique diagnosis codes being tracked by the CDR.

The CDR currently uses about 4GB of storage space (0.6% of capacity), showing it is capable of storing years worth of data using the current hardware even with the increased data demands of the new NICU. The data compresses to 247MB and is backed up daily by the hospital’s IS department. The data on the CDR system continues to be protected by single hard drive failure using RAID-1 which takes two hard drives and mirrors them (Advanced Computer & Network Corporation,
APPENDIX B. DATA COLLECTION AT CHEO

2009). Each patient in the CDR uses 10 MB of storage (106,903 data points) on average. With the 550 GB of available space currently on the CDR system, data for approximately 55,000 more patients can be stored. The CHEO NICU would take approximately 135 years to reach this capacity with their current admissions rate.

B.3.4 eClinDoc (June 2010 - Present)

In June 2010, a new project at CHEO called eClinDoc was launched to move nursing notes and charting from paper to electronic storage. This would theoretically allow the charting data to be captured and stored in the CDR and used as part of the CDS system to potentially improve the accuracy of risk estimation models. Some of the new variables stored in eClinDoc are listed in Table B.3.

Unfortunately, the eClinDoc project was not built using industry standard technology and protocols, but with a proprietary system for inputting, storing, and extracting data. After many discussions with the IS department, it was determined that the data could not be extracted and sent via HL7 to the CDR. It may be possible to generate custom reports to extract some of the data from eClinDoc but this cannot be done in an automated way, and must be manually extracted and imported into the CDR which makes using the data in real-time for a CDS system unfeasible. Any changes to eClinDoc requires custom work to be completed by the vendor at great expense. While this potentially rich set of data is now being captured electronically, it is not possible for researchers to easily access or use the data on external systems.
Table B.3: eClinDoc NICU Variables

<table>
<thead>
<tr>
<th>Variable Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apgar Score (5Minute)</td>
</tr>
<tr>
<td>Apnea Spells</td>
</tr>
<tr>
<td>Birth Weight</td>
</tr>
<tr>
<td>Current Weight</td>
</tr>
<tr>
<td>Feed Volume</td>
</tr>
<tr>
<td>Gastric Aspirates</td>
</tr>
<tr>
<td>Gestational Age at birth</td>
</tr>
<tr>
<td>Head Circumference</td>
</tr>
<tr>
<td>Mean Airway Pressure</td>
</tr>
<tr>
<td>Medications</td>
</tr>
<tr>
<td>Score for Neonatal Acute Physiology (SNAP)</td>
</tr>
<tr>
<td>Seizure Episodes</td>
</tr>
<tr>
<td>Temperature</td>
</tr>
<tr>
<td>Urine Output</td>
</tr>
<tr>
<td>Vomiting</td>
</tr>
</tbody>
</table>

B.4 Lessons Learned from CDR Integration at CHEO

B.4.1 Integration Issues

The new SpO₂ monitors at CHEO were selected for the NICU by respiratory therapists without consultation with the physicians or the research team. This resulted in new equipment being not compatible with the patient monitors, thus at one point, SpO₂ data could no longer be collected and stored by the system. The incorporation of all stakeholders into the procurement process could have greatly reduced integration issues, while still meeting the needs of the respiratory therapists and patients. After spending a considerable amount of time and effort, a solution was eventually found to capture the data. A special module for the patient monitor had to be or-
ordered that accepts an external data cable. A custom cable was made to interface the data cable of the SpO₂ monitor with the connector on the module of the patient monitor. The data from the SpO₂ monitor are now sent to the patient monitor which is relayed to the central collection station.

B.4.2 User and Staff Buy-In to the Data Collection System

It is extremely important to have a good relationship with the users of the system and staff working with the equipment. When all team members are aware of the research project and understand the usefulness and importance of the project, it is more likely that events such as equipment purchases would include consulting all of the stakeholders. This would help avoid potential compatibility and integration problems with any new components. Moreover, engaged staff are more diligent to connect a patient’s monitoring equipment to the data collection port, so data are captured and recorded for each patient. Also, engaged users of the system are more consistent in keeping the correct patients assigned to the data being collected from the different monitor stations, when patients are moved throughout the day.

Weekly Reports were generated for the manager of the NICU to determine what percentage of the patient population was being captured by the CDR. Data capture relied on both the nursing staff to connect the monitors to a data port and the Patient Service Clerks (PSCs) entering the patient information into the central collection station. If the data port was plugged in but the PSC did not enter the patient ID, it was not possible for the CDR to determine who the data belonged to and could not store it. Also, if a patient was moved to a different bed location, which occurs frequently after treatment has been performed, the PSC must update their
information on the central station or the CDR will not recognize to whom the new data stream belongs. From April to August 2009, the percentage of patients captured averaged about 45%. Figure B.4 shows the number of patients captured during that period. While the unit was nearly full (20 beds) for most of the period, the number of patients captured varied greatly depending on which staff were working that day.

Providing visual feedback of patient capture performance numbers to the staff on a weekly basis increased the number of patients being captured compared to the previous period where this information was not being tracked.

It is also imperative that users be given clear guidance on who they should contact if any questions arise or if support for the system is needed. Often a quick phone call to the right person can reduce frustration and help with compliance.
B.4.3 User and Staff Buy-In to the Research

It is important that the researcher is available to the clinical staff who interface with the system. Direct communication with the users allows the researcher to better understand how a system optimized for data collection and research analysis may affect those on which the proper operation of the system depends. Since the success of the data collection depends largely on the user’s willingness to carry out extra procedures in an extremely fast-paced environment, the researcher must be willing to find solutions which transfer much of the onus to the system rather than the staff. The physical presence of the researcher, and discussions with the staff about the importance of their involvement led to greater willingness to be part of the team. For the clinical staff, knowing their active participation leads to better quality data, which could lead to research that one day benefits them and their patients may increase acceptance of new procedures.

B.4.4 IT and Vendor Support

When running a research project, it is very useful to have full support from both the IT department of the participating organization, and also the vendors of the products being used. The legacy system was originally set up with the help of the IT department, but no clear responsibilities were assigned for supporting the system. After several years, the system suffered a hardware failure, and since the IT department was not supporting the system, they did not have the time nor the resources to repair it. As the vendor no longer supported the system, a support contract could not be attained by this route either. In the end, the responsibility fell on the academic researcher (JG) to spend valuable time repairing the system.
B.4.5 Industry Standards

Using industry standard hardware and software components greatly increases the ease and ability to upgrade the system in the future. It allows forward compatibility with multiple existing software and future packages. While improvements are being made in the medical field, there are still many proprietary technologies in use. Since the legacy database server was using a proprietary database, the system had to be restored to access the data through the database client. If an industry standard SQL database had been used, numerous methods would have existed to access the data directly without having to repair the entire system. Open source software can also provide interesting solutions. An open source database server such as MySQL is available at no cost and will run on all major systems. Many free tools for use with MySQL exist, and there is a large community that can provide support. Since the source code for the software is available, it can be conveniently updated and adapted to changing needs.

B.4.6 Backup Strategies

While seemingly obvious, it is imperative to have a good backup strategy in place. Research projects are often not given the same kind of care and attention as full production systems so sound IT practices like regular backups and archiving of data may not be performed. The hardware failure of a hard drive, for example, could be catastrophic; lost data which was collected in real-time can never be replaced, and it may take months or years to collect a new database of equivalent richness and depth. Data storage is inexpensive and numerous methods for automatically backing up data are available such as Redundant Array of Independent Disks (RAID) and often built
APPENDIX B. DATA COLLECTION AT CHEO

into the systems themselves (Advanced Computer & Network Corporation, 2009).
Appendix C

CDR Database Schema

C.1 CDR Export Temporary Table Schema

This SQL creation schema will create the temporary table used to extract data from the CDR for processing with pattern classification tools.

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';

CREATE SCHEMA IF NOT EXISTS 'cdr_exported_data' DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci ;
USE 'cdr_exported_data' ;

-- Table 'cdr_exported_data'.nicu_cnn_48hour'

CREATE TABLE IF NOT EXISTS 'cdr_exported_data'.'nicu_cnn_48hour' ( 'studyID' INT UNSIGNED NOT NULL ,
'hbloodp' INT NULL ,
'lbloodp' INT NULL ,
'hgluc' FLOAT NULL ,
'lgluc' FLOAT NULL ,
'hheartr' INT NULL ,
'lheartr' INT NULL ,
```

188
APPENDIX C. CDR DATABASE SCHEMA

'hhema' FLOAT NULL,
'lhema' FLOAT NULL,
'hitnrat' FLOAT NULL,
'hpc02' FLOAT NULL,
'hrespr' INT NULL,
'hssodium' INT NULL,
'lsodium' INT NULL,
'lanc' FLOAT NULL,
'lplt' INT NULL,
'lpo2' INT NULL,
'lserrum' FLOAT NULL,
'lwbc' FLOAT NULL,
'mortality' INT NOT NULL DEFAULT -1,
PRIMARY KEY ('studyID')
ENGINE = InnoDB
COMMENT = 'NICU CDR database with 18 SNAP attributes for 48 hours';

SET SQL_MODE=OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=OLD_UNIQUE_CHECKES;
Appendix D

Useful Data Processing Scripts

D.1 BASH Script to Create DT 5-by-2 Cross Validation Data Sets

This BASH script will take the exported patient data from the CDR and create the 10 necessary train and test data sets to use the C5.0 DT tool with the 5-by-2 cross validation method:

```bash
#!/bin/sh
NUMFOLDS=5
COUNT=1
while (( COUNT <= $NUMFOLDS ))
do
  echo "Creating fold #: $COUNT"
  # Randomly sort patients who died
  sort -R mortality.csv > temp_died.csv
  # Split file in half for train and test sets
  numlines=$(wc -l < temp_died.csv)
  echo "  mortality.csv has: $numlines entries"
  mod=$(($numlines % 2))
  if [ "$mod" -eq 0 ]
```
then
    div=$((($numlines / 2)))
else
    div=$((($numlines / 2)+1))
fi
echo " splitting half after entry #: $div"
split -1 $div temp_died.csv
mv xaa temp_died_train.csv
mv xab temp_died_test.csv

# Randomly sort patients who lived
sort -R nonmortality.csv > temp_lived.csv
# Split file in half for train and test sets
numlines2=$(wc -l < temp_lived.csv)
numlines2=$((numlines2+1))
echo " nonmortality.csv has: $numlines2 entries"
mod2=$((($numlines2 % 2))
if [ "$mod2" -eq 0 ]
then
    div2=$((($numlines2 / 2)))
else
    div2=$((($numlines2 / 2)+1))
fi
else
    div2=$((($numlines2 / 2)))
fi
echo " splitting half after entry #: $div2"
split -1 $div2 temp_lived.csv
mv xaa temp_lived_train.csv
mv xab temp_lived_test.csv

# Merge files
cat temp_died_train.csv > temp_train_fold_$COUNT.csv
cat temp_lived_train.csv >> temp_train_fold_$COUNT.csv
cat temp_died_test.csv > temp_test_fold_$COUNT.csv
cat temp_lived_test.csv >> temp_test_fold_$COUNT.csv

# Randomly sort patients to mix up lived and died within sets
sort -R temp_train_fold_$COUNT.csv > mortality_fold_$COUNT\_a.data
sort -R temp_test_fold_$COUNT.csv > mortality_fold_$COUNT\_b.data

# Reverse test and train for second test
cp mortality_fold_$COUNT\_a.data mortality_fold_$COUNT\_b.data
cp mortality_fold_$COUNT\_b.data mortality_fold_$COUNT\_a.data

# Remove temp files
rm -f temp_*

((COUNT += 1))
done
D.2 BASH Script to Call C5.0 Tool to Process Data Files

This BASH script will call the C5.0 tool to create both single and boosted decision trees using the ten 5-by-2 cross validation data sets and output the results to output files:

```bash
#!/bin/sh

# This script will run the C5.0/See5 decision tree software using the 5 by 2 cross validation. It will create normal DT models and also boosted models.

# Run regular C5.0 models
see5x -f mortality_fold_1_a > mortality_fold_1_a.out
see5x -f mortality_fold_1_b > mortality_fold_1_b.out
see5x -f mortality_fold_2_a > mortality_fold_2_a.out
see5x -f mortality_fold_2_b > mortality_fold_2_b.out
see5x -f mortality_fold_3_a > mortality_fold_3_a.out
see5x -f mortality_fold_3_b > mortality_fold_3_b.out
see5x -f mortality_fold_4_a > mortality_fold_4_a.out
see5x -f mortality_fold_4_b > mortality_fold_4_b.out
see5x -f mortality_fold_5_a > mortality_fold_5_a.out
see5x -f mortality_fold_5_b > mortality_fold_5_b.out

# Run boosted C5.0 models
see5x -b -f mortality_fold_1_b > mortality_fold_1_b.out.boost
see5x -b -f mortality_fold_1_a > mortality_fold_1_a.out.boost
see5x -b -f mortality_fold_2_a > mortality_fold_2_a.out.boost
see5x -b -f mortality_fold_2_b > mortality_fold_2_b.out.boost
see5x -b -f mortality_fold_3_a > mortality_fold_3_a.out.boost
see5x -b -f mortality_fold_3_b > mortality_fold_3_b.out.boost
see5x -b -f mortality_fold_4_a > mortality_fold_4_a.out.boost
see5x -b -f mortality_fold_4_b > mortality_fold_4_b.out.boost
see5x -b -f mortality_fold_5_a > mortality_fold_5_a.out.boost
see5x -b -f mortality_fold_5_b > mortality_fold_5_b.out.boost
```
D.3 BASH Script to Create 5-by-2 Cross Validation Data Sets using Matching Patients

This BASH script will create the 5 by 2 cross validation data sets by only using the same patients found in the reference data sets. This can be useful when trying to compare the performance of DT models using 12 hour and 48 hour summary data sets for example, to ensure the same patients that were randomly selected in the 12 hour data set will match those in the new 48 hour data set being compared:

```bash
#!/bin/sh
# This script will create the 5 by 2 cross validation data sets
# by only using the same patients found in the reference data sets.
#
# Usage: create_folds_matching_patients.sh reference_directory patient_file.csv
#
# Example: create_folds_matching_patients.sh ../CDR_12_hour nicu_48_hour.csv

NUMFOLDS=5
COUNT=1

while (( COUNT <= $NUMFOLDS ))
do
  echo "Creating fold #: $COUNT"
  # Create patient ID list file from reference fold
  cat $1/mortality_fold_$COUNT\_a.data | awk -F, '{print "\"$1\"",}'}
  > temp_train_fold_$COUNT.txt
  cat $1/mortality_fold_$COUNT\_a.test | awk -F, '{print "\"$1\"",}'}
  > temp_test_fold_$COUNT.txt

  # Copy only the lines that contain patient IDs found in temp file
  grep -f < $2 temp_train_fold_$COUNT.txt
  > mortality_fold_$COUNT\_a.data
  grep -f < $2 temp_test_fold_$COUNT.txt
  > mortality_fold_$COUNT\_a.test
```

This script takes two arguments: the directory containing the reference data sets and the patient file to match against. It creates 5 fold sets by using the same patients found in the reference data sets, ensuring consistency between the 12 hour and 48 hour data sets.
APPENDIX D. USEFUL DATA PROCESSING SCRIPTS

# Reverse test and train for second test
cp mortality_fold_$COUNT\_a.data mortality.fold_$COUNT\_b.test
cp mortality.fold.$COUNT\_a.test mortality_fold_$COUNT\_b.data

# Remove temp files
rm -f temp_*\*.txt

((COUNT += 1))
done

D.4 SQL Query to Extract Data from CDR for 48-Hour Summary Format

This SQL query extracts the 18 required SNAP attributes from every patient for the first 48 hours after admission to the NICU. The data for each patient is then stored in a temporary database table on the CDR with one record per patient and 20 columns which include the randomly generated StudyID, the mortality outcome of the patient, and the 18 SNAP attributes. This data can then be exported into a comma separated value format to be imported into pattern classification tools:

--
-- CDR Export to 48 Hour Summary format (18 SNAP Attributes in first 48 hours)
--

--
-- Set variables for the required attributes IDs
SET @bloodp = 6;   -- Mean Blood Pressure [NBP M=6]
SET @gluc = 18;    -- Glucose [GLUR=18]
SET @hr = 4;       -- Heart Rate [HR=4]
SET @hema = 73;    -- Hematocrit [HCT=73]
SET @nbands = 44;  -- Neutro Bands [NBANDS=44]
SET @meta = 41;    -- Metamyelocytes [META=41]
SET @myel = 43;    -- Myelocytes [MYEL=43]
SET @npolys = 45;  -- Neutro Polys [NPOLYS=45]
SET @pco2 = 17;    -- Partial Pressure of Carbon Dioxide [PCO2A=17]
SET @resp = 5;     -- Respiratory Rate [RESP=5]
APPENDIX D. USEFUL DATA PROCESSING SCRIPTS

---

SET @sodium = 35;  --- Sodium [NA=35]
SET @platelet = 79; --- Platelet Count [PLT=79]
SET @po2 = 23;  --- Partial Pressure of Oxygen [PO2A=23]
SET @serum = 22; --- Serum pH [PHA=22]
SET @wbc = 82;  --- White Blood Count [WBC=82]
SET @mortality = 681; --- Mortality [DispositionCode=681]
--- 7 = died, <7 = lived

-- Create temporary table with all patient min and max timestamps for first 48 hours in CDR
DROP TABLE IF EXISTS temp_time_48h_table;
CREATE TEMPORARY TABLE temp_time_48h_table
SELECT 'StudyiD', MIN('value_timestamp') AS min_time, ADDTIME(MIN('value.timestamp'), '48:00:00') AS max_time FROM 'eav_data' GROUP BY 'StudyiD';

-- Insert calculated highest and lowest blood pressure [NBP M=6] from temp table based on timestamps
INSERT INTO 'cdr_exported_data'.'nicu_cnn_48hour' ('studyiD', 'hbloodp', 'lbloodp')
SELECT tl.'StudyiD', MAX(tl.'value.int'), MIN(tl.'value_int') FROM 'eav_data' tl INNER JOIN temp_time_48h_table t2 ON tl.StudyiD = t2.StudyiD WHERE tl.'AttributelD' = ©bloodp AND tl.'value.timestamp' BETWEEN t2.min_time AND t2.max_time GROUP BY 'StudyiD'
ON DUPLICATE KEY UPDATE 'hbloodp'=VALUES('hbloodp'), 'lbloodp'=VALUES('lbloodp');

-- Insert calculated highest and lowest glucose [GLUR=18] from temp table based on timestamps
INSERT INTO 'cdr_exported_data'.'nicu_cnn_48hour' ('studyiD', 'hgluc', 'lgluc')
SELECT tl.'StudyiD', MAX(tl.'value.float'), MIN(tl.'value_float') FROM 'eav_data' tl INNER JOIN temp_time_48h_table t2 ON tl.StudyiD = t2.StudyID WHERE tl.'AttributelD' = ©glue AND tl.'value.timestamp' BETWEEN t2.min_time AND t2.max_time GROUP BY 'StudyiD'
ON DUPLICATE KEY UPDATE 'hgluc'=VALUES('hgluc'), 'lgluc'=VALUES('lgluc');

-- Insert calculated highest and lowest heart rate [HR=4] from temp table based on timestamps
SET @hr_max = 340;  --- Assume HR > 340 is outlier
SET @hr_min = 30;  --- Assume HR < 30 is outlier
INSERT INTO 'cdr_exported_data'.'nicu_cnn_48hour' ('studyiD', 'hheartr', 'lheartr')
SELECT tl.'StudyiD', MAX(tl.'value_int'), MIN(tl.'value_int') FROM 'eav_data' tl INNER JOIN temp_time_48h_table t2 ON tl.StudyiD = t2.StudyID WHERE tl.'AttributelD' = @hr AND tl.'value_int' < @hr_max AND tl.'value_int' > @hr_min AND tl.'value_timestamp' BETWEEN
t2.min_time AND t2.max_time GROUP BY 'StudyID'
ON DUPLICATE KEY UPDATE 'hheartr'=VALUES('hheartr'), 'lheartr'=VALUES('lheartr');

-- Insert calculated highest and lowest Hematocrit [HCT=73] from temp
-- table based on timestamps
INSERT INTO 'cdr_exported_data'.nicu_cnn_48hour ('studyID', 'hhema', 'lhema')
SELECT t1.'StudyID', MAX(t1.'value_float'), MIN(t1.'value_float')
FROM 'eav_data' t1 INNER JOIN temp_time_48h_table t2 ON t1.StudyID = t2.StudyID WHERE t1.'AttributeID' = @hema AND t1.'value_timestamp'
BETWEEN t2.min_time AND t2.max_time GROUP BY 'StudyID'
ON DUPLICATE KEY UPDATE 'hhema'=VALUES('hhema'), 'lhema'=VALUES('lhema');

-- Insert Highest immature/total neutrophil ratio from temp table based
-- on timestamps create temporary table with immature and neutrophil
-- data for first 48 hours in CDR default values are 0 since we need
-- to assume that if component in CBC is not present it implies 0 found
DROP TABLE IF EXISTS temp_neutro_48h_table;
CREATE TEMPORARY TABLE temp_neutro_48h_table (
    'StudyID' INT UNSIGNED NOT NULL DEFAULT 0,
    'value_timestamp' TIMESTAMP NULL DEFAULT NULL,
    'nbands' FLOAT NOT NULL DEFAULT 0,
    'meta' FLOAT NOT NULL DEFAULT 0,
    'myel' FLOAT NOT NULL DEFAULT 0,
    'npolys' FLOAT NOT NULL DEFAULT 0,
    PRIMARY KEY ('StudyID', 'value_timestamp'));
-- insert nbands into temp table
INSERT INTO temp_neutro_48h_table ('studyID', 'value_timestamp', 'nbands')
SELECT 'StudyID', 'value_timestamp', 'value_float' FROM 'eav_data'
WHERE 'AttributeID' = @nbands AND 'value_float' IS NOT NULL
ON DUPLICATE KEY UPDATE 'nbands'=VALUES('nbands');
-- insert meta into temp table
INSERT INTO temp_neutro_48h_table ('studyID', 'value_timestamp', 'meta')
SELECT 'StudyID', 'value_timestamp', 'value_float' FROM 'eav_data'
WHERE 'AttributeID' = @meta AND 'value_float' IS NOT NULL
ON DUPLICATE KEY UPDATE 'meta'=VALUES('meta');
-- insert myel into temp table
INSERT INTO temp_neutro_48h_table ('studyID', 'value_timestamp', 'myel')
SELECT 'StudyID', 'value_timestamp', 'value_float' FROM 'eav_data'
WHERE 'AttributeID' = @myel AND 'value_float' IS NOT NULL
ON DUPLICATE KEY UPDATE 'myel'=VALUES('myel');
-- insert npolys into temp table
INSERT INTO temp_neutro_48h_table ('studyID', 'value_timestamp', 'npolys')
SELECT 'StudyID', 'value_timestamp', 'value_float' FROM 'eav_data'
WHERE 'AttributeID' = @npolys AND 'value_float' IS NOT NULL
ON DUPLICATE KEY UPDATE 'npolys'=VALUES('npolys');
-- hitnrat = (neutro bands [NBANDS=44] + metamyelocytes [META=41]
APPENDIX D. USEFUL DATA PROCESSING SCRIPTS

-- + myelocytes [MYEL=43] / neutro bands [NBANDS=44] +
-- metamyelocytes [META=41] + myelocytes [MYEL=43] + neutro polys [NPOLYS=45])

INSERT INTO 'cdr_exported_data'.'nicu_cnn_48hour' ('studyID', 'hitnrat')
SELECT t1.StudyID, MAX((t1.nbands + t1.meta + t1.myel) / (t1.nbands + t1.meta + t1.myel + t1.npolys))
FROM temp_neutro_48h_table t1 INNER JOIN temp_time_48h_table t2 ON t1.StudyID = t2.StudyID WHERE t1.'value_timestamp' BETWEEN t2.min_time AND t2.max_time GROUP BY 'StudyID'
ON DUPLICATE KEY UPDATE 'hitnrat'=VALUES('hitnrat');

-- Insert calculated highest Partial Pressure Carbon Dioxide
-- [PCO2A=17] from temp table based on timestamps

INSERT INTO 'cdr_exported_data'.'nicu_cnn_48hour' ('studyID', 'hpco2')
SELECT t1.'StudyID', MAX(t1.'value_float') FROM 'eav_data' t1 INNER JOIN temp_time_48h_table t2 ON t1.StudyID = t2.StudyID WHERE t1.'AttributelD' = @pco2 AND t1.'value_float' IS NOT NULL AND t1.'value_timestamp' BETWEEN t2.min_time AND t2.max_time GROUP BY 'StudyID'
ON DUPLICATE KEY UPDATE 'hpco2'=VALUES('hpco2');

-- Insert calculated highest Respiratory Rate [RESP=5] from temp
-- table based on timestamps

SET @resp_max = 130; -- Assume RESP > 130 is outlier

INSERT INTO 'cdr_exported_data'.'nicu_cnn_48hour' ('studyID', 'hrespr')
SELECT t1.'StudyID', MAX(t1.'value_int') FROM 'eav_data' t1 INNER JOIN temp_time_48h_table t2 ON t1.StudyID = t2.StudyID WHERE t1.'AttributelD' = @resp AND t1.'value_int' < @resp_max AND t1.'value_timestamp' BETWEEN t2.min_time AND t2.max_time GROUP BY 'StudyID'
ON DUPLICATE KEY UPDATE 'hrespr'=VALUES('hrespr');

-- Insert calculated highest and lowest Sodium [NA=35] from temp
-- table based on timestamps

INSERT INTO 'cdr_exported_data'.'nicu_cnn_48hour' ('studyID', 'hsodium', 'lsodium')
SELECT t1.'StudyID', MAX(t1.'value_int'), MIN(t1.'value_int') FROM 'eav_data' t1 INNER JOIN temp_time_48h_table t2 ON t1.StudyID = t2.StudyID WHERE t1.'AttributelD' = @sodium AND t1.'value_timestamp' BETWEEN t2.min_time AND t2.max_time GROUP BY 'StudyID'
ON DUPLICATE KEY UPDATE 'hsodium'=VALUES('hsodium'), 'lsodium'=VALUES('lsodium');

-- Insert calculated lowest Absolute neutrophil count/Neutro Polys
-- [NPOLYS=45] from temp table based on timestamps

INSERT INTO 'cdr_exported_data'.'nicu_cnn_48hour' ('studyID', 'lanc')
SELECT t1.'StudyID', MIN(t1.'value_float') FROM 'eav_data' t1 INNER
JOIN temp_time_48h_table t2 ON t1.StudyID = t2.StudyID WHERE t1.'AttributeID' = @npolys AND t1.'value_float' IS NOT NULL AND t1.'value_timestamp' BETWEEN t2.min_time AND t2.max_time GROUP BY 'StudyID' ON DUPLICATE KEY UPDATE 'lane'=VALUES('lane');

--
-- Insert calculated lowest Platelet Count [PLT=79] from temp table based on timestamps
INSERT INTO 'cdr_exported_data'.'nicu_cnn_48hour' ('studyiD', 'lplt') SELECT tl.'StudyiD', MIN(tl.'value_int') FROM 'eav.data' tl INNER JOIN temp_time_48h_table t2 ON tl.StudyID = t2.StudyID WHERE tl.'AttributelD' = @platelet AND tl.'value_int' IS NOT NULL AND tl.'value_timestamp' BETWEEN t2.min_time AND t2.max_time GROUP BY 'StudyiD' ON DUPLICATE KEY UPDATE 'lplt'=VALUES('lplt');

--
-- Insert calculated lowest Partial Pressure of Oxygen [P02A=23] from temp table based on timestamps
INSERT INTO 'cdr_exported_data'.'nicu_cnn_48hour' ('studyiD', 'lpo2') SELECT tl.'StudyiD', MIN(tl.'value_int') FROM 'eav_data' tl INNER JOIN temp_time_48h_table t2 ON tl.StudyiD = t2.StudyID WHERE tl.'AttributelD' = @po2 AND tl.'value_int' IS NOT NULL AND tl.'value_timestamp' BETWEEN t2.min_time AND t2.max_time GROUP BY 'StudyiD' ON DUPLICATE KEY UPDATE 'lpo2'=VALUES('lpo2');

--
-- Insert calculated lowest Serum pH [PHA=22] from temp table based on timestamps
INSERT INTO 'cdr_exported_data'.'nicu_cnn_48hour' ('studyiD', 'lserum') SELECT tl.'StudyiD', MIN(tl.'value_floaf') FROM 'eav.data' tl INNER JOIN temp_time_48h_table t2 ON tl.StudyiD = t2.StudyID WHERE tl.'AttributelD' = @serum AND tl.'value_floaf' IS NOT NULL AND tl.'value_timestamp' BETWEEN t2.min_time AND t2.max_time GROUP BY 'StudyiD' ON DUPLICATE KEY UPDATE 'lserum'=VALUES('lserum');

--
-- Insert calculated lowest White Blood Count [WBC=82] from temp table based on timestamps create temporary table to hold results from both float and text fields since they are found in both and populate temp table with results from float field
DROP TABLE IF EXISTS temp_wbc_48h_table;
CREATE TEMPORARY TABLE temp_wbc_48h_table
SELECT tl.'StudyID', 'value_timestamp', tl.'value_float' FROM 'eav.data' tl INNER JOIN temp_time_48h_table t2 ON tl.StudyiD = t2.StudyID WHERE tl.'AttributelD' = @wbc AND tl.'value_float' IS NOT NULL AND tl.'value_timestamp' BETWEEN t2.min_time AND t2.max_time;

-- Function to strip the excess text from wbc count if in text field
DELIMITER $$
APPENDIX D. USEFUL DATA PROCESSING SCRIPTS

DROP FUNCTION IF EXISTS strip_wbc$$
CREATE FUNCTION strip_wbc(val VARCHAR(1024)) RETURNS FLOAT
BEGIN
DECLARE inti INTEGER DEFAULT 1;
DECLARE len INTEGER;
SET len=LENGTH(val);
IF len > 0 THEN
  WHILE(inti <= len) DO
    IF (STRCMP('\\',SUBSTRING(val,inti,1)) = 0) THEN
      SET inti=inti-1;
      RETURN LEFT(val,inti);
    END IF;
    SET inti=inti+1;
  END WHILE;
ELSE
  RETURN -1.0;
END IF;
RETURN 1.0;
END$$
DELIMITER ;

-- get data from text field and ignore results that need to be
-- verified, not valid, clotted or NSQ
INSERT INTO temp_wbc_48h_table ('studyID', 'value_timestamp', 'value.floal)
SELECT tl.'StudyID', 'value_timestamp', strip_wbc('value_char')
FROM 'eav_data' tl INNER JOIN temp_time_48h_table t2 ON tl.StudyiD
= t2.StudyID WHERE tl.'AttributelD' = ©wbc AND tl.'value_floal
IS NULL AND (t1.'value_timestamp' BETWEEN t2.min_time AND
AND t2.max_time) AND 'value_char' NOT LIKE '\To be verified\%' AND
'\value_char' NOT LIKE '\Not sufficient quantity\%' AND 'value_char'
NOT LIKE '\clotted\%' AND 'value_char' NOT LIKE '\not valid\%';

-- put data from temp table into final form
INSERT INTO 'cdr_exported_data'.nicu_cnn_48hour (studyiD, lwbc)
SELECT tl.'StudyID', MIN(tl.'value_floal') FROM temp_wbc_48h_table
tl INNER JOIN temp_time_48h_table t2 ON tl.StudyiD = t2.StudyID
WHERE tl.'value.floal IS NOT NULL AND tl.'value.timestamp'
BETWEEN t2.min_time AND t2.max_time GROUP BY 'StudyiD'
ON DUPLICATE KEY UPDATE 'lwbc'=VALUES('lwbc');

-- Insert Mortality (Disposition Code) [DispositionCode=681]
-- 7 = died, <7 = lived
-- Put into table as 1 = died
INSERT INTO 'cdr_exported_data'.nicu_cnn_48hour (studyID, mortality)
SELECT 'studyID', 1 AS mortality FROM eav.data WHERE 'AttributelD'
= @mortality AND value_int = 7
ON DUPLICATE KEY UPDATE 'mortality'=VALUES('mortality');

-- Put into table as 0 = survived
INSERT INTO 'cdr_exported_data'.nicu_cnn_48hour (studyID, mortality)
SELECT 'studyID', 0 AS mortality FROM eav_data WHERE 'AttributeID' = @mortality AND (value_int > 0 AND value_int < 7)
ON DUPLICATE KEY UPDATE 'mortality'=VALUES('mortality');