AN IVR CALL PERFORMANCE CLASSIFICATION SYSTEM USING COMPUTATIONAL INTELLIGENT TECHNIQUES

Pretesh Bhoola Patel

A thesis submitted to the Faculty of Engineering and the Built Environment, University of the Witwatersrand, in fulfillment of the requirements for the degree of Doctor of Philosophy.

Johannesburg, 2009.

Declaration

I declare that this thesis is my own, unaided work, except where otherwise acknowledged. It is being submitted for the degree of Doctor of Philosophy to the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination in any other university.

Signed this __ day of _____20__

_____
Pretesh Bhoola Patel.

Abstract

Speech recognition adoption rate within Interactive Voice Response (IVR) systems is on the increase. If implemented correctly, businesses experience an increase of IVR utilization by customers, thus benefiting from reduced operational costs. However, it is essential for businesses to evaluate the productivity, quality and call resolution performance of these self-service applications. This research is concerned with the development of a business analytics for IVR application that could assist contact centers in evaluating these self-service IVR applications. A call classification system for a pay beneficiary IVR application has been developed. The system comprises of field and call performance classification components. 'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' field classifiers were developed using Multi-Layer Perceptron (MLP) Artificial Neural Network (ANN), Radial Basis Function (RBF) ANN, Fuzzy Inference System (FIS) as well as Support Vector Machine (SVM). Call performance classifiers were also developed using these computational intelligent techniques. Binary and real coded Genetic Algorithm (GA) solutions were used to determine optimal MLP and RBF ANN classifiers. These GA solutions produced accurate MLP and RBF ANN classifiers. In order to increase the accuracy of the call performance RBF ANN classifier, the classification threshold has been optimized. This process increased the classifier accuracy by approximately eight percent. However, the field and call performance MLP ANN classifiers were the most accurate ANN solutions. Polynomial and RBF SVM kernel functions were most suited for field classifications. However, the linear SVM kernel function is most accurate for call performance classification. When compared to the ANN and SVM

field classifiers, the FIS field classifiers did not perform well. The FIS call performance classifier did outperform the RBF ANN call performance network. Ensembles of MLP ANN, RBF ANN and SVM field classifiers were developed. Ensembles of FIS, MLP ANN and SVM call performance classifiers were also implemented. All the computational intelligent methods considered were compared in relation to accuracy, sensitivity and specificity performance metrics. MLP classifier solution is most appropriate for 'Say account' field classification. Ensemble of field classifiers and MLP classifier solutions performed the best in 'Say amount' field classification. Ensemble of field classifiers and SVM classifier solutions are most suited in 'Select beneficiary' and 'Say confirmation' field classifications. However, the ensemble of call performance classifiers is the preferred classification solution for call performance.

Acknowledgements

I would like to thank my supervisor, Prof. Tshilidzi Marwala, for his unending assistance, guidance, support and tremendous insight into this research project. I have learnt how to conduct world class research from him. This is a priceless gift that will never be forgotten.

My sincere thanks go to my parents, Bhoola Patel and Yoginiben Bhoola Patel, for their infinite support, patience as well as encouragement during the course of this research. I would also like to thank my fiancée, Deepa Jaga, for continuously reminding me of the importance of this work. She has also been amazingly patient and understanding during the course of this work. My thanks also go to my brother, Ketan Bhoola Patel, who continuously reinforced the idea of hard work will be rewarded. I also thank my sister, Anisha Bhoola Patel and her husband Hanish Patel for their extreme support.

I would like to also acknowledge the support and guidance provided by my colleagues at Intelleca, a division of the Bytes Technology Group. They have continuously reminded me of the value of this research. I sincere thank you goes to Carlos Goncalves and Jivko Mladenov.

Most importantly, I would like to thank my Lord, Bhagwan Swaminarayan and my Guru, Pramukh Swami Maharaj. They have made me achieve all that I have and it is due to Them that I am where I am.

I dedicate this thesis to Bhagwan Swaminarayan, Pramukh Swami Maharaj, my parents and fiancée.

Table of contents

List of Figures

List of Tables

Nomenclature

Abbreviation	Description
AA	Auto Attendants
AI	Artificial Intelligence
ANI	Automatic Number Identification
ANN	Artificial Neural Network.
ASR	Automated Speech Recognition
CEA	ClickFox Customer Experience Analytics
CPU	Central Processing Unit
CRM	Client Relationship Management
CSA	Customer Service Agent
DNIS	Dialed Number Identification Service
DTMF	Dual Tone Multi-Frequency
ERM	Empirical Risk Minimization
ERP	Enterprise Resource Planning
ETL	Extract Transform Load
FIS	Fuzzy Inference System.
FSVM	Fuzzy Support Vector Machines
GA	Genetic Algorithm
GMM	Gaussian Mixture Model

Abbreviation	Description
GUI	Graphical User Interface
HME	Hierarchical Mixture of Expert
HTML	Hypertext Markup Language
IVR	Interactive Voice Response
MLP	Multi-Layer Perceptron
RBF	Radial Basis Function
RMS	Root Mean Square
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
SOM	Self-Organizing Map
SRM	Structural Risk Minimization
SV	Support Vector
SVM	Support Vector Machine
TTS	Text-to-speech
VIA	Call-Analytics Virtual Interactive Analyst
VoiceXML	Voice Extensible Markup Language
VRU	Voice Response Units
VUI	Voice User Interface

Chapter 1

Business Analytics and Interactive Voice Response systems

1.1 Introduction

Customer satisfaction fosters loyalty, increases the probability of selling additional products and services as well as reduces the chances of competitive replacement. However, customer dissatisfaction results into direct revenue losses due to customer churn and indirect losses such as damage to reputation. Improving the customer experience is a vital priority for contact centers across different industries.

In order to provide customers with access to convenient and reliable information fast, Interactive Voice Response (IVR) systems have been adopted by businesses. If implemented correctly, these systems can assist in improving the customer experience (Nichols, 2006). This thesis details the implementation of a caller classification system that can be utilized within a business analytics for IVR solution. The objective of the solution is to assist contact centers in determining caller experience metrics, thereby assisting businesses in identifying areas of improvement within self-service applications.

This chapter begins with an explanation of IVR systems that will examine the benefits and current deployment techniques employed by businesses. Thereafter, business analytics for IVR is defined and current products that are available will be described. The chapter also states the research objectives, thesis contribution and the importance of the research. The structure of the thesis is also explained. The chapter ends with a list of publications that were published as a result of this research.

1.2 Interactive Voice Response systems

Interactive Voice Response (IVR) market is a rapidly growing contact center technology sector. The IVR market yielded approximately two billion US dollars in revenue in 2007, thus establishing this market as the second largest contact center technology sector (DMG Consulting LLC, 2009). Due to the current economic condition, businesses are under tremendous pressure to reduce operational expenses. When deployed correctly, IVR systems can automate twenty to ninety five percent of incoming calls, thus resulting in dramatic reduction in operational expenses (DMG Consulting LLC, 2009).

An IVR system is an automated telephony system that interacts with callers, gathers relevant information and routes calls to the appropriate destinations (Nichols, 2006). The inputs to the IVR system can be voice, Dual Tone Multi-Frequency (DTMF) keypad selection or a combination of the two. IVR systems can provide appropriate responses in the form of voice, fax, callback, e-mails and other media (Nichols, 2006). An IVR system solution may consist of telephony equipment, software applications, databases and supporting infrastructure. A major objective of an IVR system is to improve customer experience, while lowering operating costs.

Initially, Auto Attendants (AA) and Voice Response Units (VRU) were used to provide menu options and scripting tools to direct callers to certain queues as well as provide information to callers with minimal interaction (Ascent Group, Inc, 2008).

IVR systems provided the capability to integrate enterprise information systems and to interact with callers to tailor questions as well as responses according to customer requirements. Today, Advanced Speech Recognition (ASR) tools utilized within the IVR systems provide conversational interactions with callers, thus providing an effective method of gathering caller input information for customization (Ascent Group, Inc, 2008).

IVR system acceptance by businesses has taken more than a decade to grow. However, many businesses identified that at the expense of automation, their customers were being alienated. As a result, during the recent years, the importance of the relationship between customer experience, customer satisfaction and profitability have been emphasized by many businesses (Ascent Group, Inc, 2008). These businesses began actively addressing IVR system usability through customer-friendly application call flow designs. Due to continuous improvement processes implemented by businesses that enhance the IVR usability and functionality, the number of calls handled completely within IVR systems has increased. However, the initiative for call automation has been moderated as there are businesses that also deploy IVR systems as an option for customers to help themselves, rather than forcing callers to utilize IVR system services (Ascent Group, Inc, 2008).

IVR technology provides businesses more cost effective management through call segmentation, automated call handling and informational messaging. IVR systems can also assist businesses manage peak call volumes, enabling contact centers to respond to a large number of customers. IVR technology provides customers twenty four hour services as well as privacy (Ascent Group, Inc, 2008).

Additionally, new standards in speech technology are providing substantial advances in the predominantly proprietary IVR technology market (Ascent Group, Inc, 2008). In order to effectively as well as efficiently interact with customers, new voice technologies are emerging that allow businesses to leverage the internet and

telecommunications infrastructure. Technologies such as Voice Extensible Markup Language (VoiceXML) are enabling the development of solutions that execute on multiple platforms, thus providing businesses with the capabilities of managing web-based and IVR self-service applications utilizing the same infrastructure as well as language (VoiceXML Forum, 2009). As a result, businesses have the capabilities of providing feature rich value added self-service applications faster with reduced implementation costs, thus resulting in an increase in IVR application deployments.

VoiceXML is a scripting language utilized for defining voice enabled IVR applications (VoiceXML Forum, 2009). It is the 'Hypertext Markup Language (HTML)' for telephony based speech applications. VoiceXML hides the complexities of the telephony platform from IVR application developers. The language enables easier IVR voice application integration with internet-based applications. VoiceXML enables IVR applications to be developed in an environment familiar to web developers. The major objective of VoiceXML is to provide the advantages of web-based development and content delivery to IVR applications. VoiceXML utilizes ASR and DTMF for user input. Prerecorded audio and Text-to-speech (TTS) attributes are employed as output. The VoiceXML technology is proposed by the VoiceXML forum (VoiceXML Forum, 2009). VoiceXML is an international standard for defining telephony based voice applications.

Currently, IVR systems have been deployed in a number of industries such as financial services, telecommunication, manufacturing, insurance, utilities, consumer products as well as entertainment. The major motivation for IVR system implementation is automation and customer satisfaction (Ascent Group, Inc, 2008). Typical self-service IVR applications that are provided to callers are account inquiry, account payment, fault reporting, status inquiry, ordering of products or services, automated assistance in resolving technical problems as well as general company information (Ascent Group, Inc, 2008).

Majority of businesses implement IVR systems to selectively force callers through these self-service applications prior to interacting with a Customer Service Agent (CSA). This deployment strategy increases system utilization and success, particularly for callers that are unfamiliar with the system options and functionality provided. This strategy encourages IVR usage for routine tasks while complex inquiries are resolved through CSA interaction. During peak call volume periods, businesses can also selectively force callers through self-service IVR applications as well as after office hours. However, when the contact center is not experiencing large call volumes, the caller can be presented with an option whether or not to use the self-service applications (Ascent Group, Inc, 2008).

Many businesses also deploy IVR systems to provide optional services to callers. Businesses also implement these systems for efficient call routing (Ascent Group, Inc, 2008). There are businesses that have the capabilities to route a percentage of calls received to an IVR hosted by an outsourced vendor, transparent to the customer. This capability is usually found in large contact centers. In order to improve the efficiency of the contact centers during peak call volume periods, this option is exercised (DMG Consulting LLC, 2009).

In 2006 it has been reported that the speech technology adoption rate was on the increase (Nichols, 2006). This trend continues as indicated in (Global Industry Analysts, 2008) and (Datamonitor, 2008). Speech recognition provides another dimension to IVR system design. ASR self-service applications can dramatically improve IVR utilization. The benefits provided by speech recognition driven self-service applications are shorter call durations, increased usage, natural conversation interactions and, therefore, increased customer satisfaction. Due to the implementation of ASR IVR applications, businesses have reported an increase in IVR utilization from thirty five to seventy percent (Ascent Group, Inc, 2008). Therefore, if implemented correctly, callers prefer speech-enabled IVR applications.

Today, customers interact with many businesses that provide excellent services. These interactions set customer expectations. As a result, in order to outperform in the current market, all customer-facing technologies should be scrutinized to ensure that these implementations support business service strategies and, therefore, deliver the expected, if possible, preferred customer experience.

In order to achieve this, in relation to IVR systems, best practices state that businesses should evaluate the performance of the self-service applications as the business would CSA productivity, quality and call resolution (Ascent Group, Inc, 2008). Businesses should have the capabilities to measure the IVR system performance from the perspective of the caller that is the influence the automated application had on accomplishing the objective of the customer. The best results are achieved through constant monitoring and refinement of IVR applications (Miller, 2007).

Business analytics for IVR can provide contact centers with these essential capabilities. These solutions compute performance measures such as caller disconnects at the various stages of the applications. This assists businesses in determining caller satisfaction or dissatisfaction with the system. For example, due to frustration experienced, the customer may have given up and therefore abandoned the call. The caller may have received the required information and, as a result, ended the call. Business analytics for IVR solutions also provide transfers to CSA statistics, which aid contact centers to determine if the caller is provided with sufficient information to complete transactions. These measures also assist in determining whether or not customers are familiar with the application call flows. This research entails the development of such a business analytics for IVR solution that employs computational intelligent methodologies.

The section that follows examines business analytics for IVR.

1.3 Business analytics for IVR

In this research, business analytics for IVR is defined as systems that effectively use IVR application data to determine the manner in which callers are utilizing the automated system and, through the use of this information, assists contact centers in identifying areas for Voice User Interface (VUI) improvement. As a result, business analytics for IVR solutions assist contact centers in enhancing the customer experience by analyzing the automated applications from the perspective of the customer. An increase in the performance of the automated IVR services can result in satisfied customers, increase in customer retention, increase in self-service containment, decrease call duration and lower contact center costs.

Business analytics for IVR assist contact centers in determining the number of tasks that were completed, the number of callers that selected to transfer to CSA and the number of caller disconnects. These solutions also provide further details such as the tasks where the caller disconnect or transfer to CSA occurred. The reason for the transfer to CSA such as due to system errors or callers preferring to interact with CSA is also provided. Currently, there are business analytics for IVR solutions that also provide implementation detail measurements such as speech recognition accuracy per task as well as out of grammar or invalid option selections (Miller, 2007).

Typical business analytics for IVR solutions analyze information such as transcribed voice recordings, application data and caller or customer feedback (Miller, 2007). IVR application data includes log files and reporting database information. Caller or customer feedback for IVR applications are usually determined through the use of automated call surveys. Once, the caller has completed a transaction with the automated service, these applications are typically presented to the customer.

However, there are contact centers that use outbound IVR applications to conduct the survey after the caller has disconnected (opinion-8, 2009).

Business analytics for IVR solutions provide contact centers the ability to monitor specific events within the IVR applications. These solutions are the basis for predicting and enhancing the quality of future customer interactions. In order to achieve this, business analytics for IVR solutions utilize "insight engines" that process databases consisting of call detail records, stored transcribed voice recordings or utterances and other call interaction information to identify patterns that result into crucial information that assist contact centers to manage future interactions. A major objective of these solutions is to generate sufficient detailed information to assist contact centers isolate decision points that indicate customer dissatisfaction.

Majority of business analytics for IVR solutions provide the capabilities to determine system status, call progress and exit analysis information (Miller, 2007). System status information provides statistics such as the duration various components of the IVR system were enabled and active as well as disabled. This information also provides statistics that indicate where failures are occurring. Call progress information assists in determining call patterns that illustrate the manner in which customers are proceeding through the IVR applications. Exit analysis assists contact centers in determining the major technique calls within the IVR applications at specific modules are terminated that is due to caller disconnects or transfer to CSA.

Majority of the current business analytic for IVR solutions employ a four-stage process towards customer experience improvement (Miller, 2007). This process involves isolating self-service difficulties, capturing customer care concerns, increasing the duration in IVR systems and identifying time as well as the cause of exit from the IVR systems. Isolating self-service difficulties involve detecting modules within the IVR applications high levels of caller disconnects or transfers to CSA are occurring due to customer frustration experienced. For example, when an

IVR application does not recognize a callers input due to the caller saying too few digits in the response to account number information queried, the IVR application continuously re-prompts a caller with no reason for the unsuccessful attempt would be a cause of caller frustration.

IVR system caller satisfaction levels are mainly determined by the manner in which self-service solutions address customer expectations. Capturing customer care concerns entails correlating call survey results with specific characteristics of the customer interaction. This involves analyzing customer feedback and utilizing this information to identify modules within the self-service applications that require enhancements to improve the caller experience. Increasing the duration in IVR systems is an objective of many contact centers as this may correlate to callers successfully using the IVR applications. The contact center is reducing costs as these queries do not require interaction with CSA, which has a high cost. However, if this caller behaviour is also accompanied with high levels of frustration, the IVR application is analyzed to determine the cause. Identifying the time and cause of exit from the IVR applications assists in determining whether the call ended successfully or the call transferred to a CSA for completion.

Typically, business analytics for IVR solutions begin an implementation by reviewing IVR application reporting or logging information as well as IVR application documentations to identify specific events that map crucial modules within the automated systems. IVR application documentations may include functional and technical specifications. Application logging events that identify critical caller interaction such as speech recognition events are also identified. The mapped events are then utilized to define important business events such as business success task completion modules. Each contact center may interpret these events differently. For example, a contact center may regard transfer to CSA as a successful task completion as the caller exercised the available “transfer me” option and the IVR solution successfully executed the transfer procedure. However, another contact

center may interpret the transfer to CSA event as unsuccessfully task completion as the transaction had not been completed within the IVR solution. A CSA concluded the task. As a result, the outcome of this process can be business specific. Thereafter, the information determined is utilized to compute statistics that provide insight into the performance of the IVR applications. These are then presented to businesses through reports, websites or dashboards. An iterative analyze of these statistics assist in identifying areas of improvement.

Currently, there are many third-party vendors that provide business analytics for IVR solutions. However, ClickFox Customer Experience Analytics (CEA), Call Analytics Virtual Interactive Analyst (VIA), pureXML business analytics for IVR and VoiceObjects analyzer solutions are examined in this research.

PureXML provides a hosted web based business analytics for IVR solution. As a result, the solution caters for remote access. In order to implement the analytics solution, pureXML follows a process that maps IVR logging information such as application metric files into a standard analytics Extensible Markup Language (XML) structure. An Extract Transform Load (ETL) process is employed to populate the standard XML structure. Thereafter, businesses have secure access to a web based visual representation of caller activity per module or node. The representation illustrates all IVR application nodes mapped. When a particular module is selected, the previous node or nodes that could lead to the current module is displayed as well as the node or nodes that could follow. The relevant module caller behaviour statistics calculated are also illustrated. As a result, pureXML provides the contact center with a detail interactive call flow illustration (pureXML, 2007).

The pureXML business analytics for IVR solution computes caller disconnects, difficulties experienced and transfer to CSA caller behaviour statistics. These statistics are calculated per IVR application module. Caller difficulties represent events such as unsuccessful caller input recognition and callers not responding to

automated prompt. The solution also provides the business with transaction complete success and failure metrics. As a result, in order to deliver a successful pureXML analytics solution, the accuracy of the mapping process of log event files to an XML structure is crucial.

ClickFox CEA is an analytical solution that employs a patented "customer behavior pattern recognition engine", which identifies and models the actual paths that occur within the channels that form a customer interaction across the enterprise (Clickfox, 2009). ClickFox offers hosted as well as on premises deployments. Similar to the pureXML solution, the ClickFox CEA solution implementation conducts a detailed analysis of IVR application logged information and supporting design documentation such as functional specification to map logged information to key business performance indicators. Thereafter, a robust interactions repository tailored to business requirements and processes is created. This forms the foundation of all analysis conducted. The patented technology utilized by ClickFox CEA aggregates data across all customers and all communication access points such as websites and IVRs, to produce a visual illustration of how customers are interacting with these systems across the enterprise. ClickFox assists businesses in discovering opportunities for improvements by identifying correlations and trends across all customer interactions as well as by determining optimal methods for handling each type of inquiry or interaction.

ClickFox CEA provides businesses with capabilities to conduct various types of analysis (Clickfox, 2009). Traffic analysis provides individual customer experience information such as the time taken before, after and between modules within a process. Task analysis enables businesses to examine dominant as well as unique process flow paths, caller difficulties, caller disconnect and transfer to CSA events. This analysis provides insights into customer behaviour and the impact of this behaviour on operational outcomes. Pattern analysis entails the constant evaluation of current and past behaviour to determine common customer behaviour characteristics.

This analysis is accomplished through the use of a series of automated patented technologies and processes. ClickFox CEA also utilizes an Artificial Intelligence (AI) recommendations engine that identifies discrepancies in system design, customer experience and the manner in which different customers interact with these systems (Clickfox, 2009). The AI recommendation engine provides recommendations on new, enhanced or simplified paths that effectively as well as efficiently processes customer inquiries or interactions within these applications. The business analysis views such as high level trending dashboards can be populated by analytical data computed during task and traffic analysis. These views can also incorporate the results returned by the AI recommendation engine, thus providing contact centers with the ability to examine crucial information such as key performance indicators, essential process flows, and self-service completion rates.

VoiceObjects Server is a component of VoiceObjects 7 family of products (VoiceObjects, 2009). It is phone application server that provides businesses such as enterprises and telecommunication carriers with the capabilities to develop VoiceXML self-service applications, which during a live call, utilizing information from the Client Relationship Management (CRM) solution or other databases, forms personalized dialogs with a customer. VoiceObjects 7 family of products also provides businesses to develop multimodal phone applications that could employ voice, video, graphics and text interfaces. Integration between the VoiceXML applications and CRM solutions, Enterprise Resource Planning (ERP) solutions as well as other databases is provided by Web service support. This enables self-service applications to access required customer as well as organization resource information. VoiceObjects 7 family of products also provides businesses with an execution environment for these applications that permits remote management, monitoring of multimodal applications as well as online application maintenance for enhancements. Detailed application analysis is also provided (VoiceObjects, 2009).

Enterprise Edition and Network Edition versions of VoiceObjects 7 family of products are available (VoiceObjects, 2009). Network Edition, together with the capabilities provided in the Enterprise Edition, provides businesses such as telecommunication carriers with a multi-tenant environment, thus enabling these businesses to host various phone applications for other organizations.

VoiceObjects Analyzer is the component within the VoiceObjects 7 suite of software that provides businesses with self-service application analysis (VoiceObjects, 2009). It is a complete service analysis environment that can be utilized throughout the enterprise to determine system usage, system as well as application performance, caller bahaviour and speech recognition success. The analytics capabilities of VoiceObjects Analyzer are based on the statistics functionality of VoiceObjects Infostore. VoiceObjects Infostore, the application logging component within the VoiceObjects 7 family of products, enables businesses to store data that is retrieved and transformed from VoiceObjects Servers in a standardized data model (VoiceObjects, 2009).

VoiceObjects Analyzer is designed to process this data into information that assists businesses to gain insight into factors that influence the profitability of their services. Through the use of VoiceObjects Analyzer, businesses can continuously tune and enhance the deployed self-service applications to improve customer acceptance of these services. VoiceObjects Analyzer provides businesses with information that assists contact centers in determining the self-service task completion rates, the dominant paths callers are using within self-service applications, application dialog usage trends, the application system error rate and caller duration statistics. Physical server hardware analysis capabilities are also provided such as workload on servers, current port utilization and balance of load on clusters.

The key feature of VoiceObjects Analyzer is the real-time reporting capabilities. Due to VoiceObjects Infostore, the data source used by VoiceObjects Analyzer, being

automatically populated with application and system usage information as a caller accesses the self-service applications, the time consuming ETL process is eliminated. When applications are enhanced or extended to provide additional services, modifications to the logging database schema is also no longer required.

Call Analytics VIA is an IVR application and Computer Telephony Integration (CTI) analytics solution (Call Analytics, Inc, 2007). The solution provides caller behaviour statistics required by call center managers, telephony managers, IVR application developers and department managers. The solution provides call center managers with the ability to track the progress of callers through automated IVR self-service solutions. The automated alerts that can be scheduled to be sent to telephony managers when specific defined thresholds are exceeded such as the number of ports available on an IVR platform has exceeded seventy five percent, allows the managers to obtain complete call data when required. IVR application developers are provided with information to determine the bottle-necks within these applications, thus enabling the developers to optimize the self-service solutions. Call Analytics VIA provides reports that assist department managers to ensure that their customers are satisfied with the IVR solutions provided. The Speech module captures complete call utterances for further analysis (Call Analytics, Inc, 2007).

The solution also provides a dashboard view real-time status of the IVR applications and platform performance. This information is displayed in Snapshot, Counter and Trend sections (Call Analytics, Inc, 2007). The Snapshot section summaries the current self-service success rates as well as the overall health of the IVR applications. Snapshot section also includes errors and contained call information. The Counter section illustrates the number of calls received grouped by language selection, account lookup success, menu utilization, transactions as well as Dialed Number Identification Service (DNIS). The Trend section displays graphs of total calls received, menu utilization statistics and calls per DNIS for daily, weekly, monthly or quarterly periods. Call Analytics VIA alerting capabilities are defined on the

dashboard and assist in communicating early warnings of potential IVR application as well as platform issues.

The dashboard view, predefined and ad-hoc reports can be accessed utilizing a Graphical User Interface (GUI). Call Analytics VIA also allows reports to be exported to Microsoft Excel for further analysis (Call Analytics, Inc, 2007). The solution also has the capabilities of filtering data in combinations of call date time, caller response input, prompt visited, IVR application name, IVR name, IVR port, transfer to CSA extension number, Automatic Number Identification (ANI), DNIS or call end reason. Furthermore, when a report has been defined, it can be saved for future use (Call Analytics, Inc, 2007).

Call Analytics VIA solution consists of standard predefined reports such as Call end summary, Payments report, Error report, Port capacity report, Frequent caller by account number or ANI report, Total number of call reports, Contained call summary report, Transfer by CSA extension summary report and Total calls per Prompt summary report. The Total number of call reports is daily, weekly, monthly and yearly (Call Analytics, Inc, 2007).

Call Analytics VIA solution can accommodate reporting for multiple IVR platforms. Call Analytics offers hosted as well as on premises deployments. Due to the solution comprising of four components, flexibility is provided in deployment. The solution can be implemented utilizing a single or multiple servers. Call Analytics VIA system comprises of Database, Call Analyst, Call Logger and Utilization Service components (Call Analytics, Inc, 2007).

However, these solutions are expensive (pureXML, 2007), (Clickfox, 2009), (VoiceObjects, 2009) (Call Analytics, Inc, 2007). A number of businesses do implement a solution that has been created by internal resources. These solutions are the competition to the third-party solutions. These solutions can also be expensive.

The following section details the research objectives and contribution.

1.4 Research objectives and thesis contribution

Earlier in this chapter, section 1.2 gave an introduction to IVR systems. These systems may experience large call volumes. Business analytics for IVR solutions have been implemented to better understand the caller experience as well as behaviour within these systems. The emergences of computational intelligent methods such as Artificial Neural Network (ANN), Fuzzy Inference System (FIS) and Support Vector Machine (SVM) have presented an alternate approach to modeling, which is potentially beneficial. These methods are capable of utilizing large data sets to derive relationships within the data presented.

This research entails the development of a business analytics for IVR application that employs AI methodologies. The classification system is to provide businesses with the capabilities to measure business intelligence performance metric levels such as customer satisfaction, call containment, task completion, efficiency and usability.

Call containment and task completion metrics assist businesses to determine the percentage of callers completing transactions within the IVR successfully, without interacting with a CSA. Efficiency and usability metrics provide businesses with indications of the call durations and difficulties callers experience within the automated applications. In relation to the caller experience, the customer satisfaction metric provides the contact center with a single inclusive indicator of the complete call performance. This metric is based on the above performance measures. The call performance classification system should also provide businesses with implementation detail performance indicators that assist businesses to improve the IVR application performance and therefore caller experience. Metric levels such as

field performance, field attempts and field recognition levels provide IVR application developers with the ability to identify areas of improvement rapidly.

In order to effectively improve customer experience within IVR systems, businesses require not only information such as the number of calls that were contained within the IVR applications and transferred to a CSA, businesses also require detailed measures. The objective of the call classification system is to provide the metrics to improve IVR applications in relation to customer experience and therefore customer satisfaction. As a result, the classification system is to provide businesses with the capabilities to measure essential metrics required by contact centers to enhance IVR applications. Measures such as the number of calls that completed transactions successfully within the automated application as well as the number of calls that were abandoned or transferred to CSA due to recognition difficulties are also to be provided.

The aim of this research is thus to:

1 Design a business analytics for IVR solution based on computational intelligence to assist contact centers in determining IVR application performance in relation to caller experience.

2 Develop a component within the proposed business analytics for IVR solution that assists contact centers to compute implementation detail performance indicators using computational intelligent techniques such as ANN, FIS and SVM. As a result, provide IVR application developers with the capability to identify areas of improvement rapidly.

3 Utilizing ANN, FIS and SVM to develop a component within the proposed business analytics for IVR solution that provides contact centers with the capabilities of determining the complete call performance.

4 Propose and implement an ensemble of classifiers for each of the proposed components.

5 Compare the classifiers implemented to determine the superior approach for this application problem.

The major contribution of this thesis is to, therefore, illustrate how computational intelligence methods can be utilized to model caller behaviour based on application logging information to provide businesses with performance indicators. These performance metrics will assist contact centers to improve the caller experience and therefore customer satisfaction.

1.5 Importance of the research

As earlier stated in previous sections, IVR systems experience large call volumes. These call volumes result in vast amounts of application logging events that detail caller interaction. In order to assist contact centers in interpreting and thus effectively utilizing this information, business analytics for IVR solutions have been developed as presented in section 1.3. In this research, computational intelligence methods are utilized to model caller interaction information. As a result, this research proposes another approach into caller behaviour modeling. The other main contributions of this research are:

- Introduce a new research direction into caller interaction modeling through the utilization of computational intelligent methodologies.

- Introduce a new research direction into caller experience performance modeling through the utilization of computational intelligent methodologies.

- Providing an application to understand caller behaviour within IVR systems, which can subsequently be used by businesses to improve caller experience and, therefore, increase caller satisfaction levels. The application will also assist businesses in reducing resources as well as time spent in analyzing information to understand caller behaviour within IVR systems.

- Investigate the capabilities of ANN, FIS and SVM classification techniques in categorizing data extracted from application logging event files.

1.6 Structure of the thesis

Chapter 1 of this thesis has presented vital information that is required to understand the research problem of concern. As a result, IVR systems and business analytics for IVR solutions have been examined. This chapter also states the major objectives and research contributions of this work.

Chapter 2 provides a background on the computational intelligent techniques to be utilized within the proposed IVR caller classification application. These techniques include ANN, Genetic Algorithm (GA), FIS and SVM. A thorough analysis of these techniques is presented.

Chapter 3 describes the proposed call classification system. The source of the data, the application logging event file, is examined. Thereafter, the architecture of the call classification system is presented, thus providing detail into the components of the system. Selection of data and preprocessing techniques utilized are also described.

Chapter 4 examines the components implemented by utilizing ANN computational intelligent techniques. The results yielded by the developed ANN classifiers are

illustrated. The findings of the investigation are presented together with concluding remarks.

Chapter 5 describes the implementation process of the components utilizing FIS computational intelligent techniques. The results achieved by the developed FIS classifiers are examined together with concluding remarks.

Chapter 6 details the utilization of SVM methods in implementing the call classification system components. The results yielded by the developed SVM classifiers are illustrated. The findings of the investigation are examined together with concluding remarks.

Chapter 7 compares the ANN, FIS and SVM classifiers to determine the superior computational intelligent approach for this research problem. Ensembles of classifiers are also presented. The results obtained are illustrated together with conclusions drawn from the analysis.

Finally, Chapter 8 presents the overall conclusion of the thesis, which illustrates the manner in which the research objectives have been achieved. Also, possible further research work is proposed in this chapter.

1.7 Publications

From this research, the following journal and conference publications were made:

- Patel, P. B and Marwala, T:2009, Caller Behaviour Classification: A Comparison of SVM and FIS Techniques, *Advances in Computational Intelligence*, *Springer-Verlag*, vol.116/2009, pp.199-208.

- Patel, P.B and Marwala, T.: 2009, Caller Interaction Classification: A Comparison of Real and Binary Coded GA-MLP Techniques, *Advances in Neuro-Information Processing: Lecture Notes in Computer Science*, *Springer-Verlag*, vol. 5507, pp. 728-735.

- Patel, P. B. and Marwala, T.: 2008, Interactive Voice Response field classifiers, *2008 IEEE International conference on Systems, Man and Cybernetics*, pp. 3425-3430.

- Patel P. B. and Marwala, T.: 2009, Genetic Algorithms, Neural Networks, Fuzzy Inference System, Support Vector Machines for Call performance classification, *IEEE 2009 International Conference on Machine Learning and Applications*, accepted, to be published.

- Patel P. B. and Marwala, T.: 2009, Caller behaviour classification using computational intelligent methods, *International Journal of Neural Systems (IJNS)*, under review.

Chapter 2

Computational Intelligent techniques

2.1 Introduction

This chapter examines the computational intelligent techniques that have been considered in the research. Artificial Neural network (ANN), Genetic Algorithm (GA), Fuzzy Inference System (FIS) and Support Vector Machines (SVM) are therefore examined.

2.2 Artificial neural network

Due to the difficulty and complexity of statistical techniques as well as the high level of proficiency required to utilize such methods, there has been a significant increase in the usage of ANNs. This increase has also been attributed to the fact that ANNs can be applied to virtually every field in industry. For example, ANNs can be utilized in medical diagnosis, machine fault diagnosis, fingerprint recognition as well as financial creditworthiness evaluation applications. These networks can also be employed in product line development to control the quality of products manufactured. ANN research has gathered enormous momentum in recent years. As a result, this field of study has been introduced in many universities.

ANNs were introduced based on the understanding of neurology in the early 1940s. They have been motivated by the fact that scientists are challenged to effectively utilize machines on tasks currently solved by humans (Smith, 2003), (Orr, 2006), (Bishop, 1995). ANNs can be considered as an exceptionally robust data-modeling tool that consists of a network of interconnected simple processors or units, which individually operate on local data and together these units capture as well as numerically represent the intricate input output relationships of complex systems (Neuro Solution Technologies, 2009), (Haykin, 1998). These networks are data-mining techniques that have been inspired by the desire to develop artificial systems capable of performing 'intelligent' computations similar to those performed within the human brain. An ANN acquires its knowledge through repeated presentations of data. It 'learns' by adjusting the weights of the network connections, which is similar to adjusting the synaptic weights within the inter-neuron connections within the human brain (Neuro Solution Technologies, 2009). The ANN creates its own organization or representation of the application problem information during training from the data observed. Thereafter the network will exhibit some capability for

generalization in obtaining rather accurate outputs when presented with new unseen data.

An advantage of ANNs is their ability to represent both linear as well as non-linear relationships. As a result, these networks are able to approximate any computable function to arbitrary precision and are known as universal approximators (Bishop, 1995). ANNs are effective within application problems in which an algorithmic solution cannot be formulated. ANNs have the capabilities of adaptive learning; the network learns how to do tasks based on training data or initial experience (Bishop, 1995). ANNs require short computational times for modeling of systems.

ANNs do not experience the many drawbacks statistical techniques on handling data possess (Smith, 2003), (Orr, 2006). Statistical methods do impose restrictions on the number of input data. However, ANNs do not. Statistical regressions are performed utilizing unrealistic simple dependency linear and logarithmic functions. ANNs do not require intensive mathematical techniques to transform data. However, statistical methods do require intensive mathematical transformations. Due to ANNs non-linear nature, these methodologies are better able to account for complexity of human behaviour and also these techniques provide tolerance to missing or erroneous information.

The integration of ANNs into the modern environment is a major challenge in industry. This is due to ANNs, when applied to large scale problems, at times, become unstable. Also, due to ANNs negligence of the effects of noise, these networks, at times, do not react appropriately to abrupt changes within the application problem data. ANNs are also viewed as black boxes with unknown rules that are utilized internally.

2.2.1 Artificial neural network architectures

There exists a great diversity of ANN architectures, such as:

- Multi-Layer Perceptron (MLP)
- Radial Basis Function (RBF)
- Recurrent Neural Network (RNN)
- Hierarchical Mixture of Expert (HME)
- Self-Organizing Map (SOM)

However, the most common, and often used in practical applications, is that of a feed-forward structured neural network (Nabney, 2002), (Bishop, 1995). It has been stated that these ANN architectures with a single hidden layer, provided with sufficient data, can be used to model any function (Beale and Jackson, 1990). It should be mentioned that there are situations where it is necessary or worthwhile in utilizing two or more hidden layers. This is dependent on the primary purpose of the ANN within the application.

Among the family of feed-forward structured networks, the MLP and the RBF ANN architectures are possibly the most extensively employed ANNs in pattern classification (Nabney, 2002). MLP networks employ correlation-based algorithms, whereas RBF networks utilize distance-based algorithms (Reyneri and Sgarbi, 1997). Both networks function in a supervised manner. Due to the non-linear capabilities of these networks, they are said to be excellent universal approximators that provide highly accurate solutions (Bishop, 1995). As a result, these networks produce very practical tools for classification and inverse problems.

Due to the fact that RNN require to be unfolded (Bishop, 1995), these ANN are complex to design. HME networks result in ANNs that over-fit the application problem data and SOM networks have characteristics of overcrowding as well as underutilization of neurons within the ANN (Haykin, 1998), (Bishop, 1995). As a

result, the MLP and RBF feed-forward structured ANNs have been considered in this research. These ANNs will be examined in the following section.

Multi-Layer Perceptron (MLP)

The MLP network evolved from the combination of many simple components. The most fundamental of these is the mathematical model of the neuron. In 1943 McCullock and Pitts proposed this neural model, which then formed the basis for formal calculus of brain activity (Chen, 1991). In 1958 Rosenblatt introduced the Perceptron model. This was an elementary visual system that could be taught to recognize a limited class of patterns (Chen, 1991). It was this model that then formed the foundation upon which most forms of AI were born (Huang and Lippmann, 1988). A perceptron can be considered as a device that computes the weighted sum of its inputs. It then propagates this sum through an activation function to produce the output. This activation function can be linear or nonlinear (Chen, 1991). However, a network of linear perceptrons was found to have serious computational limitations (Chen, 1991). These limitations were overcome by adding layers of nonlinear perceptrons that resulted in the MLP ANN.

Figure 2.1 illustrates a feed-forward structured network. As mentioned above, the MLP ANN has a feed-forward structured network architecture whereby each unit receives inputs only from lower layer units. Feed-forward structured networks do not have connections between units in the same layer. These networks usually comprises of input, hidden and output layers, all of which are interconnected with respect to the hidden layer.

The training of these networks is accomplished through backpropagation and a complex nonlinear hidden as well as output weights optimization. At iterations, the error of the network is assessed by forward propagating the inputs through the

network and the derivative of this error is calculated with respect to each weight within the network.

The error function generally used in ANN computation is the squared difference between the actual and desired outputs. Optimization techniques, such as the scaled conjugate gradient method, are then used to minimize the error function by altering the weights, initially in the output layer and then the hidden layer. Essentially, the error is backpropagated from the output of the network, through the output weights and to the hidden weights (Bishop, 1995).

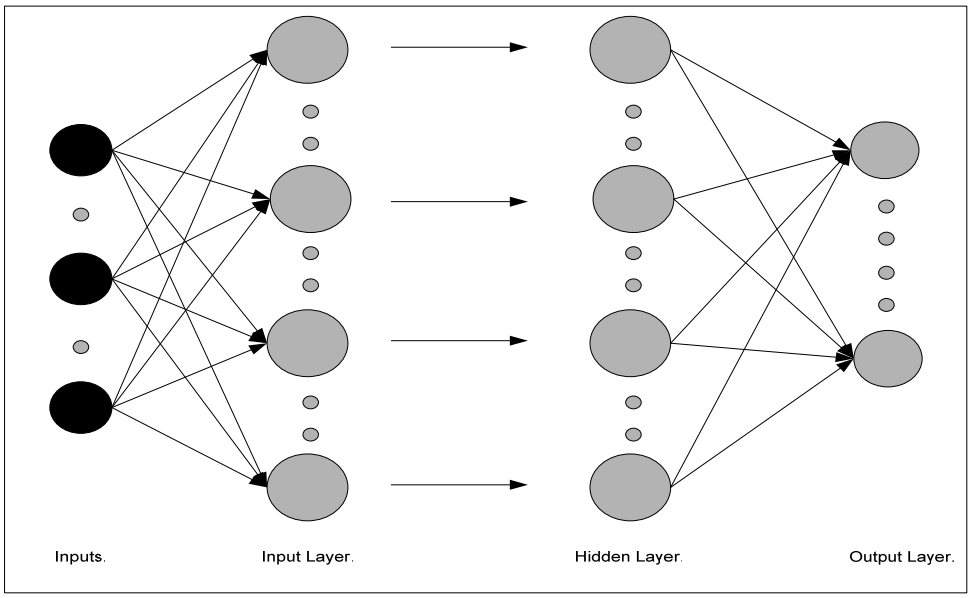

Figure 2.1: Feed-forward neural network topology

During the development of the MLP ANN, over-fitting as well as under-fitting should be avoided. This can be accomplished by dividing the data into three sets. Over-fitting occurs when the network does not generalize but rather tends to memorize the training data. Under-fitting occurs when the network does not follow the data at all (Bishop, 1995). The data is divided into training, validation and test sets. The training data set is used to train the ANN to find the general pattern between its inputs and outputs. The validation data set is used to assess the network and the test data is used to confirm the prediction quality of the developed networks.

The hidden and output layers contain activation functions. The choice of the hidden-unit activation function for the MLP network is mainly dependent on the application of the network (Bishop, 1995). However, it has been determined that the hyperbolic tangent activation function offers a practical advantage of giving rise to faster convergence during training (Nabney, 2002). There are three major forms of the MLP network output-unit activation function. These are the linear, logistic sigmoidal and softmax activation functions (Nabney, 2002). It has been stated that the appropriate selection of the MLP network output-unit activation function for a pattern classification problem is the logistic sigmoidal function (Nabney, 2002). Equation 2.2.1 illustrates this activation function.

$$y_k = \frac{1}{1+\exp(-a_k)} \qquad (2.2.1)$$

where,

a = output value of hidden layer activation function

k = 1,...,c

c = total number of network outputs

y = output value of activation function

Radial Basis Function (RBF)

The RBF networks have become a popular alternative to the MLP network approach (Haykin, 1998). RBF networks are inspired from traditional statistical classification techniques (Haykin, 1998). These are based on Cover's theorem on the separability of patterns. This theorem states that nonlinearly separable patterns can be separated linearly if the pattern is cast nonlinearly into a higher dimensional space. Therefore, the RBF network converts the pattern to a higher dimension after which it classifies the pattern linearly (Haykin, 1998).

The RBF network hidden layer, utilizing a set of basis functions, performs a nonlinear mapping from the input space into a higher dimensional space in which the patterns become linearly separable. In order to accomplish this, the RBF network employs a Gaussian hidden-unit activation function. The output layer usually implements a linear weighted sum of the hidden layer outputs (Haykin, 1998). As a result, a linear activation function is utilized within the RBF network output layer. Figure 2.2 illustrates the RBF ANN architecture.

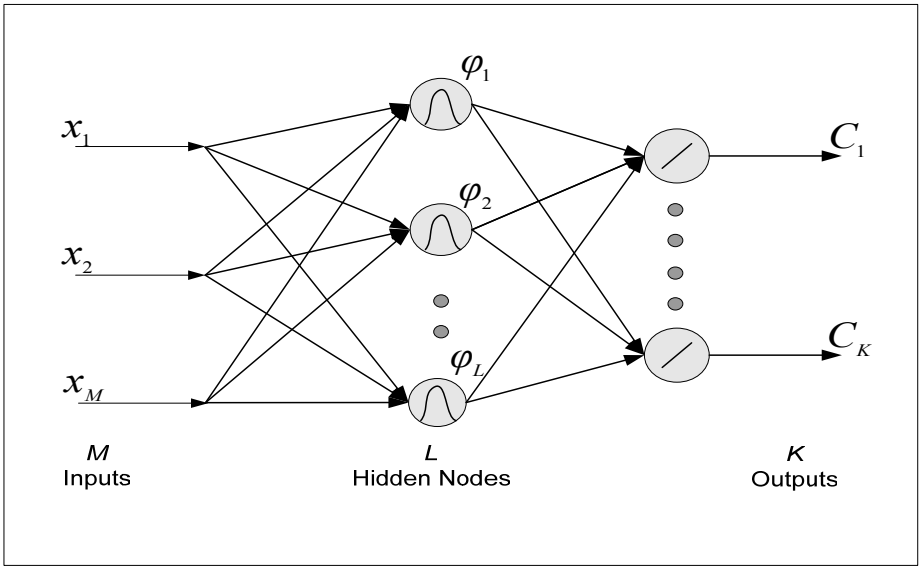

Figure 2.2: Radial Basis Function Artificial Neural Network

Similar to the MLP ANN implementation, during the development of the RBF ANN, over-fitting and under-fitting should be avoided. This can be accomplished by dividing the data into training, validation and test sets. The training data set is used to train the RBF ANN to identify the general pattern between its inputs and outputs. The validation data set is employed to evaluate the network and the test data is utilized to confirm the prediction quality of the developed RBF ANN.

2.3 Genetic Algorithm (GA)

Utilizing a form of directed random search process, evolutionary methods conduct an exploration to identify optimum solutions, without previous problem knowledge. GA solutions are evolutionary techniques employed to resolve optimization problems (Goldberg, 1989). Due to the intuitiveness, ease of implementation and ability to effectively solve highly nonlinear problems that are typical of complex engineering systems, these algorithms have been popular in academia and industry. After the original work conducted by Holland (Holland, 1975), GA solutions were developed. These are powerful set of stochastic global search techniques that have yielded good results in a range of problems across different industries (Rababaah et al., 2005), (Selouani and O'Shaughnessy, 2003), (Sasaki et al., 2001).

GA solutions have been considered as an optimization technique in this research due to its superiority over other optimization techniques (Michalewicz, 1992). Due to GA solutions considering a population of candidate solutions rather than a single candidate solution, the GA methodology differs from conventional optimization techniques. The populations of candidate solutions undergo a process of reproduction of individuals that advances individuals with better fitness values than the other individuals in the previous generation (Michalewicz, 1992). GA solutions provide an alternate technique to the existing conventional optimization methods in resolving problems where these methods are inappropriate. GA techniques are feasible and

optimum solutions than conventional methods. GA solutions transform the optimization problem into an appropriate form as opposed to other evolutionary approaches that process the problem unchanged.

GA solutions have been employed in resolving engineering problems that are complex and difficult to solve using conventional optimization methods. These algorithms manage, maintain and manipulate populations of candidate solutions utilizing a survival of the fittest strategy in their quest for an optimal better solution. In order to improve successive generations, the fittest individuals of any population tend to reproduce. However, as evident in biological evolution, inferior individuals can also survive and reproduce. The implementation of GA solutions involves the identification and determination of the chromosome representation, selection function, genetic operators, initialization function, termination function as well as evaluation function to be used (Holland, 1975), (Booker et al., 1989). A description of these essential components within GA solutions follows.

2.3.1 Chromosome representation

The chromosome representation scheme defines the manner in which the problem is structured in the GA solution. The representation used also governs the genetic operators that are to be employed (Booker et al., 1989). An individual or chromosome comprises of a sequence of genes. Various types of chromosome representations such as binary digits, floating point numbers, integers, symbols and matrices exist.

In traditional GA solutions, binary representation has been used for chromosomes (Houck et al., 1995). This results in an even discrete depiction of the real optimization problem. Within these binary coded GA solutions, binary substrings representing each parameter with a desired precision are concatenated to form a chromosome. Therefore, a large number of variables in a real world problem would result in

chromosomes encoded in long strings. Also, there is a discrepancy between the binary representation space and the actual problem space. For example, two points close to each other in the real space might be far away in the binary represented space. However, natural representations are more efficient and may result in better solutions.

In order to resolve these problems, floating point representation of parameters as a chromosome is utilized (Michalewicz, 1996). It has been stated, in relation to Central Processing Unit (CPU) time, floating point or real coded representations are more efficient. This representation also produces higher precision with more consistent results across replications (Michalewicz, 1996). In these real coded GA solutions, a chromosome is coded as a finite length string of the real numbers corresponding to the real world problem variables. Real coded GA solutions are robust, accurate as well as efficient because they are conceptually closest to the real world problem and moreover, the string length reduces to the number of variables. It has been reported that the real coded GA solutions outperformed binary coded GA solutions in many design problems (Janikow and Michalewicz, 1991). This research will determine if this is true in relation to the application problem of concern.

2.3.2 Selection function

In order to produce successive generations, the selection of individuals is important within GA solutions. The selection function identifies the individuals that will survive and proceed onto the next generation. A probabilistic selection is performed based upon the fitness of the individual such that the superior individuals have a higher possibility of being selected and therefore advancing onto the next generation. Several selection methods such as roulette wheel selection and its extensions, scaling techniques, tournament, elitist models and ranking selection techniques do exist (Houck et al., 1995). However, in this research normalized geometric ranking and

tournament selection functions are employed within the GA solutions developed. As a result, these selection functions are described below.

Ranking selection function methods utilize the evaluation function to map individual solutions to a completely ordered set, thus allowing minimization and negativity. When all individual solutions are sorted, these selection function techniques assign P_i based on the rank of individual solution i. Normalized geometric ranking selection function, defines P_i for each chromosome or individual solution using equation (2.3.1) (Houck et al., 1995).

$$P_i = q'(1-q)^{r-1} \qquad (2.3.1)$$

$$q' = \frac{q}{1-(1-q)^P} \qquad (2.3.2)$$

where,

q = probability of selecting the best individual

r = rank of the individual, where best equals 1

P = population size

The tournament selection function is a common selection mechanism used within GA solutions (Booker et al., 1989). This selection function method is simple to implement and is efficient for both non-parallel and parallel architectures. Similar to the ranking methods, the tournament selection function method only requires the evaluation function to map individual solutions to a partially ordered set. However, this selection technique does not assign probabilities to the individual solutions. This selection method begins by randomly selecting a number of individuals from the current population. The number of individuals is set by the tournament size. In this research, three individual solutions competed within each tournament. The selection method then compares the fitness of these individuals competing and inserts the

individual solution with the best fitness value into the new population. This process is repeated until an appropriate population is achieved.

2.3.3 Genetic operators

Crossover and mutation genetic operators are employed to provide basic search mechanisms for the GA solution. These operators produce new solutions based on existing individual solutions in the population. Crossover genetic operators employ two individuals as parents to produce two new individual solutions. However, mutation operators alter one individual to yield a single new solution. Simple, arithmetic and heuristic crossover operators are usually utilized. Commonly employed mutation operators are boundary, non-uniform, uniform and multi-non-uniform operators. However, the types of operators used are dependent on the chromosome representation employed within the GA solution. In this research, binary and real coded chromosome representations were considered. As a result, the binary coded GA solutions employed binary mutation and simple cross over genetic operators. However, the real coded GA solutions utilized non-uniform mutation and arithmetic cross-over genetic operators (Houck et al., 1995). Therefore, these genetic operators are described in this thesis.

Equation (2.3.3) illustrates the binary mutation that entails flipping each bit in every individual within the population with probability p_m (Chang and Lin, 2001).

$$x'_i = \begin{cases} 1 - x_i, & \text{if } U(0,1) < p_m \\ x_i, & \text{otherwise} \end{cases} \qquad (2.3.3)$$

Equation (2.3.4) and equation (2.3.5) illustrates the simple crossover genetic operator that employs randomly generated number r from a uniform distribution between 1 and m to create two individuals (Chang and Lin, 2001).

$$x'_i = \begin{cases} x_i, & \text{if } i < r \\ y_i, & \text{otherwise} \end{cases} \tag{2.3.4}$$

$$y'_i = \begin{cases} y_i, & \text{if } i < r \\ x_i, & \text{otherwise} \end{cases} \tag{2.3.5}$$

As illustrated in equation (2.3.6), the non-uniform mutation genetic operator employed by the real coded GA solutions selects one variable, *j*, and sets this variable equal to a non-uniform random number. This is mathematically represented as follows (Chang and Lin, 2001):

$$x'_i = \begin{cases} x_i + (b_i - x_i) f(G) & \text{if } r_1 < 0.5, \\ x_i - (x_i + a_i) f(G) & \text{if } r_1 \geq 0.5, \\ x_i, & \text{otherwise} \end{cases} \tag{2.3.6}$$

where,

$$f(G) = (r_2 (1 - \frac{G}{G_{\max}}))^b \tag{2.3.7}$$

r_1, r_2 = uniform random numbers between 0 and 1.

G = current generation.

$G_{\max}$ = maximum number of generations.

b = shape parameter.

Equation (2.3.8) and equation (2.3.9) illustrates the arithmetic crossover operator that produces complimentary linear combinations of the parents (Chang and Lin, 2001).

$$\overline{X'} = r\overline{X} + (1-r)\overline{Y} \tag{2.3.8}$$

$$\overline{Y'} = (1-r)\overline{X} + r\overline{Y} \tag{2.3.9}$$

2.3.4 Initialization, termination and evaluation functions

GA solutions may begin with an initial population. This population can be randomly generated or derived from other methods. GA solutions advance from generation to generation, ending when a termination criteria is achieved. The termination condition could be the maximum number of generations, population convergence criteria, lack of improvement of the best solution for a specified number of generations or achieving a specific targeted value within an objective function. Evaluation or objective functions of a variety of forms can be utilized within GA solutions. These functions enable the mapping of populations into a partially ordered set. Figure 2.2 illustrates the common GA solutions optimization process followed.

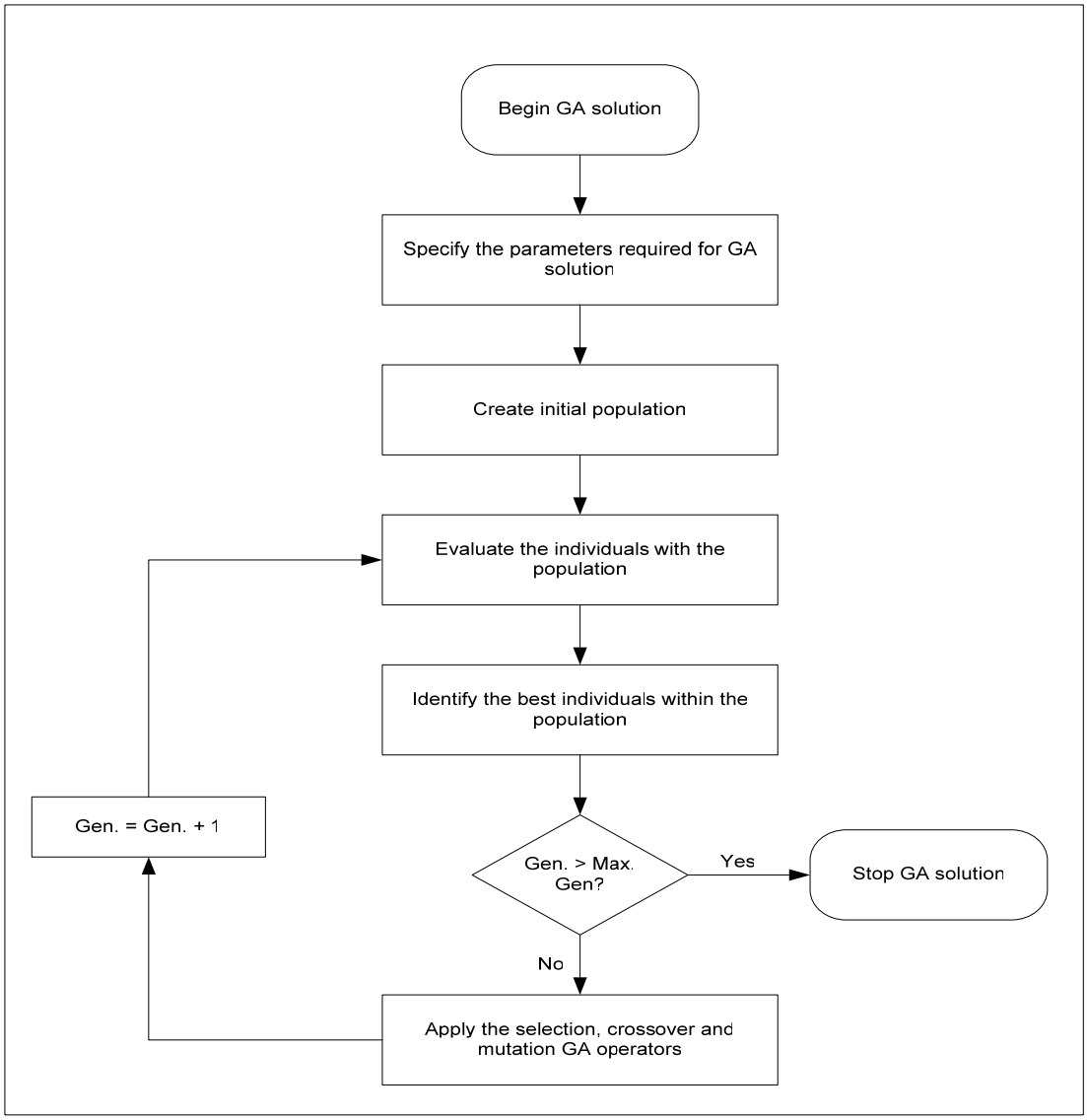

Figure 2.3: GA solution optimization process

2.4 Fuzzy Inference System (FIS)

Fuzzy logic was originally developed by Dr. Lotfi Zadeh. He published his seminal work on fuzzy set in 1965. In 1973 he proposed his theory of fuzzy logic (GuruNet Corp, 2009).

Instead of classifying membership as either true or false as in a classical logic system, in fuzzy set theory, which is the foundation of FISs, an input can belong to one or more fuzzy sets with a degree of membership (Kasabov, 1996). The degree of membership is defined by fuzzy membership functions. Fuzzy logic also allows conclusions to be reached from inputs with a gradation of truth. Membership can be viewed as a representation of the "possibility" of association with the particular set (Kasabov, 1996).

One of the major advantages of fuzzy logic is its ability to be developed on top of the experience of experts within an industry (Uhrig and Tsoukala, 1997). In order to accomplish this, it uses heuristic rules to describe the available expert knowledge. These fuzzy inference rules are expressed in the form "IF A THEN B", where A is the premise and B is the consequence. The actions of the rules are executed or "fired" when the degree of membership of the inputs exceed certain threshold values. The threshold values define the minimum required membership of the inputs that an expert would expect for the particular rule to be executed and are generally defined by subjective criteria. Conflicting rules are allowed to fire jointly (Kasabov, 1996).

FISs are processes that utilize fuzzy logic to formulate a mapping from a given input to an output (The Mathworks, 1995). The mapping then provides a foundation from which decisions can be made. FISs have been successfully applied in fields such as automatic control, data classification, decision analysis and expert systems. Due to its multidisciplinary nature, these systems are associated with many names, such as

fuzzy-rule-based systems, fuzzy expert systems, fuzzy modeling, fuzzy logic controllers as well as fuzzy systems (The Mathworks, 1995).

A FIS involves Fuzzification, Inference and Defuzzification processes (The Mathworks, 1995). The Fuzzification process is a mapping from the observed input to the fuzzy sets defined in the corresponding universe. Inference process is a decision making logic that utilizes the fuzzy inference rules to determine fuzzy outputs corresponding to fuzzified inputs. Defuzzification produces non-fuzzy outputs (The Mathworks, 1995).

There are two popular types of FISs. These are the Mamdani-type and Sugeno-type inference systems (The Mathworks, 1995). Mamdani-type inference system is the most commonly employed fuzzy methodology. It was proposed by Ebrahim Mamdani in 1975 (The Mathworks, 1995). The proposed methodology is based on a paper by Lofti Zadeh in 1973 on fuzzy algorithms for complex systems and decision processes. In the Mamdani-type inference system the fuzzy sets from the consequent of each rule are combined through the aggregation operator and the resulting fuzzy set is defuzzified to yield the output of the system (The Mathworks, 1995). The Sugeno-type or Takagi-Sugeno-Kang method of inference was introduced in 1985 (The Mathworks, 1995). In this type of FIS, the consequent of each rule is a linear combination of the inputs. The output is a weighted linear combination of the consequents. This inference methodology is similar to the Mamdani-type process in many respects. The initial Fuzzification and Inference processes of the inference techniques are exactly the same. These inference systems vary in the manner their outputs are determined. The Sugeno output membership functions are either linear or constant (The Mathworks, 1995). Mamdani-type inference systems are widely accepted, intuitive and well-suited to human input. However, Sugeno method of inference is computationally efficient; performs well with linear, optimization as well as adaptive techniques and is well-suited to mathematical analysis (The Mathworks,

1995). As a result, the FISs developed in this research utilized Sugeno-type inference systems.

Clustering of numerical data establishes the foundation of many classification and system modeling applications. The objective of clustering is to locate natural groupings in a set of given inputs such that similar inputs are congregated in the same class (The Mathworks, 1995). Utilizing data clustering to obtain fuzzy inference rules provides an advantage in that the resultant rules are more tailored to the data than a FIS generated without clustering (The Mathworks, 1995).

There are two popular data clustering techniques. These are the fuzzy c-means and subtractive data clustering techniques (The Mathworks, 1995). The fuzzy c-means technique, introduced by Jim Bezdek in 1981, entails each data point belonging to a cluster to some degree that is specified by a membership grade (The Mathworks, 1995). This data clustering technique provides a method that illustrates the ability to group data points that populate a multidimensional space into a specific number of unique clusters. Fuzzy c-means technique requires two predefined clusters that are intended to indicate the mean location of each cluster (The Mathworks, 1995). Every data point is assigned a membership grade for each cluster. Due to the cluster centers and the membership grades for each data point being updated iteratively, the technique moves the cluster centers to the correct locations within the data set. This iteration involves minimizing a function that represents the distance from any given data point to a cluster center weighted by the membership grade of that data point (The Mathworks, 1995).

Subtractive data clustering technique is a modified form of the Mountain Method for cluster estimation (Yen and Wang, 1999). In this method, each data point is considered as a potential cluster center and defines a measure of the potential of a data point (Chiu, 1994). The measure of potential for a given point is a function of its distances to all other data points. A point with many neighbouring points will have a

high potential value. After the potential of every data point has been computed, the point with the largest potential value is selected as the first cluster center. Thereafter, in order to determine the next cluster and its center, all the data points in the vicinity of the first cluster center, which is determined by a radius of influence or cluster radius, is removed. This process is iterated until all the input data are within a cluster radius of a cluster center (The Mathworks, 1995). Specifying a small cluster radius will usually yield many clusters in the data. However, specifying a large cluster radius will result in few cluster centers in the data (The Mathworks, 1995). The Sugeno -type inference systems developed in this research, employed subtractive clustering.

2.5 Support Vector Machine (SVM)

SVM, introduced within the framework of Statistical Learning Theory, is a technique proposed by Vapnik in 1995 for resolving classification, regression and density estimation problems (Vapnik, 1995), (Vapnik, 1998). It has been demonstrated that SVM techniques have outperformed traditional neural network methods in relation to generalization capabilities in a number of different application domains (Cortes and Vapnik, 1995). Due to the techniques many attractive features and powerful classification as well as regression capabilities, the use of SVMs has gained popularity as an alternative approach to fuzzy logic, neural networks and genetic algorithms. These traditional methodologies generally experience the presence of local minima, structure selection problems (number of hidden layers, number of hidden nodes, number of rules, population size) and over-fitting training data (Gunn, 1998). Similar to MLP and FIS, SVM maps the input vector to a feature space, thus enabling the classification or regression processes to be performed efficiently.

SVM based learning is established on the Structure Risk Minimization (SRM) principle. This principle has been demonstrated to be superior to the traditional

Empirical Risk Minimization (ERM) principle that is employed by conventional neural networks (Cortes and Vapnik, 1995). SRM principle minimizes the upper bound of the generalization error on Vapnik-Chernoverkis dimension, whereas the ERM principle minimizes the training error. The objective of SRM principle is to achieve an optimum value of the structural risk function whereas the ERM principle minimizes the empirical risk function. In other words, SRM principle is concerned with the minimization of both the capacity of the learning machine and the training error. However, ERM only minimizes the training error. This enables SVMs to generalize information contained implicitly in the training data set optimally (Gunn, 1998). As a result, SVMs have good generalization capabilities. SVMs have many advantages in solving small sample size, nonlinear and high dimensional pattern recognition problems (Taylor and Cristianini, 2000).

SVMs utilize kernel functions to transform data in the input space to a higher dimensional feature space where an optimal hyperplane that maximizes the margin between classes is determined (Taylor and Cristianini, 2000). The maximum margin hyperplane provides the largest separation between the decision classes. Support vectors (SV) are the training examples that are closest to the maximum margin hyperplane. Figure 2.4 illustrates the SVM process.

The kernel function is a key technology of SVM. The type of kernel function will affect the classifier learning and generalization capabilities. Different kernel functions will result in different SVM classifiers. In this research the linear, polynomial, Radial Basis Function (RBF) and sigmoid kernel functions have been considered.

Equation (2.5.1), (2.5.2), (2.5.3) and (2.5.4) below illustrate the linear, polynomial, RBF and sigmoid kernel functions, respectively.

$$K(\mathbf{x}_i\mathbf{x}_j) = \mathbf{x}_i^T\mathbf{x}_j \quad (2.5.1)$$

$$K(\mathbf{x}_i\mathbf{x}_j) = (\gamma\mathbf{x}_i{}^T\mathbf{x}_j + r)^d, \gamma > 0 \quad (2.5.2)$$

$$K(\mathbf{x}_i\mathbf{x}_j) = \exp(-\gamma \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2), \gamma > 0 \quad (2.5.3)$$

$$K(\mathbf{x}_i\mathbf{x}_j) = \tanh(\gamma\mathbf{x}_i{}^T\mathbf{x}_j + r) \quad (2.5.4)$$

where,

$K(\mathbf{x}_i\mathbf{x}_j)$ is the kernel function

γ, r and d are kernel parameters.

Currently, in order to determine the optimal kernel parameter pairs, there no general rules. As a result, as proposed in (Hsu et al., 2003), a grid search is performed to optimize these values.

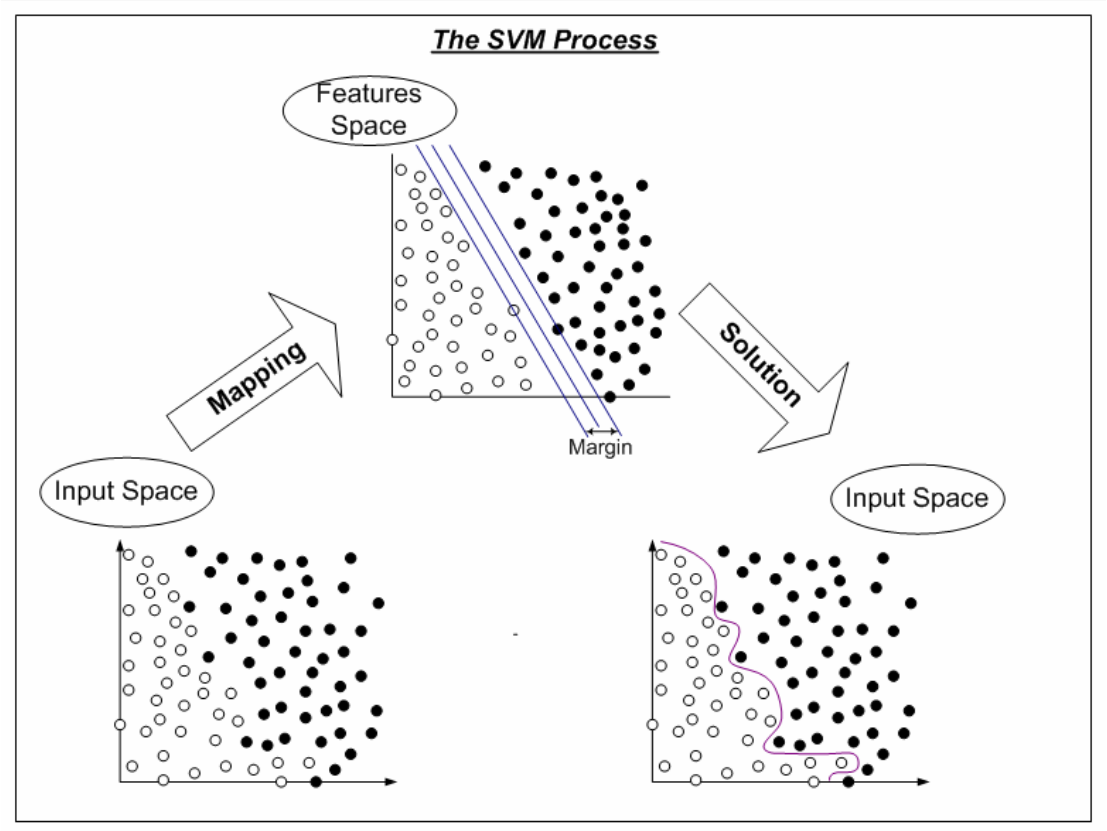

Figure 2.4: Support Vector Machine process

In order to resolve the multi-class problem, the “one-against-one” approach in which classifiers are constructed and each trained with data from two different classes has been utilized in this research (Chang and Lin, 2001). The final classification decision is made based on a majority vote among the binary classifiers.

SVMs resolve regression problems by transforming input data onto high dimensional feature space utilizing a nonlinear mapping. Thereafter, in this feature space, linear regression is performed.

Chapter 3

The call classification system

3.1 Introduction

Businesses may have the best products, lowest prices and most intelligent employees, but if potential customers perceive that the business does not have the capacity to complete the project, the companies will lose these customers. In the current economic condition, businesses are required to be aggressive in increasing the profile of their brand, establish a good reputation as well as presence in their market sector.

Although, businesses may spend precious capital into these ventures, there are more efficient and inexpensive ways to achieve the same results through the use of affordable latest technologies such as Interactive Voice Response (IVR) systems. A telephone is more than a phone. It is a major interaction point to the customer for the business. It is the "front door" of the company. The manner in which calls from customers are answered and the manner in which the calls are processed is an instant measure of the business efficiency and customer relations attitude.

By using the correct combination of new call handling tools, a small business can also project professionalism and competence from the first crucial customer interaction. Intelligent phone systems incorporate, accommodate and integrate a wide range of key business processes that have resulted in significant increase in IVR system implementations (Jones, 2008).

However, there are many businesses that provide IVR systems to customers, which are poorly designed or implemented that result in caller frustration when utilizing the automated solutions (Faulkner, 2006). An example of a cause of caller frustration would be a voice prompt that plays a detailed description of the options available. The descriptions are long, thus resulting in caller confusion as the caller cannot remember all the options presented. Another example is an IVR application that does not provide the caller sufficient time to respond to a prompt. As a result, these customers would probably end the call or request for a transfer to a Customer Service Agent (CSA).

The aim of this research is to develop a call classification application, using computational intelligent methods, which could assist businesses in quantifying caller behaviour within their IVR systems. It is anticipated that this application would be used in conjunction with other customer behaviour analysis techniques such as listening to recorded calls. As a result, this application should be used to confirm the IVR system performance in relation to customer interaction.

3.2 Source of the data

As the developed system is to be used to identify trends of caller behaviour within the IVR Extensible Markup Language (VoiceXML) applications, the classifiers are trained based on data extracted from IVR log event files. These files are generated by the IVR platform as specific events occur during a call to the system. Events such as call begin, form enter, form select, Advanced Speech Recognition (ASR) events, transfer events and call end events are written to the logs (VoiceGenie Technologies, 2005). Figure 3.1 illustrates an extract from an IVR log event file. Descriptions of the events that were utilized to generate the data sets used to develop the classification application are provided in Table 3.1.

```
2008-08-01/16:38:53.007 METRIC 003C0089-0C1C19F6 form_select :Confirm:BLOCK
2008-08-01/16:38:53.010 METRIC 003C0089-0C1C19F6 form_select :FldConfirmTransaction:FIELD
2008-08-01/16:38:53.276 METRIC 003C0089-0C1C19F6 prompt /usr/local/phoneweb/tmp/003C0089-0C1C19F6-0001//promptsfile.003C0089-0C1C19F6-0001.11
                                                 :audio|http://10.208.26.24/jsp/VoiceGenie/Audio/Shireen/ENGLISH/Speech/PNA001.wav;audio
                                                 |http://10.208.26.24/jsp/VoiceGenie/Audio/Shireen/ENGLISH/Speech/SCT001.wav;
2008-08-01/16:39:12.173 METRIC 003C0089-0C1C19F6 prompt_end done
2008-08-01/16:39:12.173 METRIC 003C0089-0C1C19F6 input_start :VOICE
2008-08-01/16:39:12.173 METRIC 003C0089-0C1C19F6 prompt /usr/local/phoneweb/tmp/003C0089-0C1C19F6-0001//promptsfile.003C0089-0C1C19F6-0001.12
                                                 :audio|http://10.208.26.24/jsp/VoiceGenie/Audio/Shireen/ENGLISH/Speech/SCT020A.wav;audio
                                                 |http://10.208.26.24 /jsp/VoiceGenie/Audio/Shireen/ENGLISH/Speech/SCT020C.wav;audio
                                                 |http://10.208.26.24/jsp/VoiceGenie/Audio/Shireen/ENGLISH/Speech/SCT021.wav;
2008-08-01/16:39:29.816 METRIC 003C0089-0C1C19F6 prompt_end done
2008-08-01/16:39:30.615 METRIC 003C0089-0C1C19F6 asr_trace bargein:_bargein_
2008-08-01/16:39:32.789 METRIC 003C0089-0C1C19F6 asr_trace ASR_NOMATCH_WITH_NBEST:results:+<_gram5>the guide|18|<the guide>
2008-08-01/16:39:32.791 METRIC 003C0089-0C1C19F6 event nomatch:1|REJECTED(ASR)
2008-08-01/16:39:32.791 METRIC 003C0089-0C1C19F6 event_handler_enter :nomatch maxspeechtimeout|http://10.208.26.24/jsp/VoiceGenie/Speech/Controls
                                                 /Common/ConfirmTransactionControl/SayConfirmTransaction.jsp?TransactionType=TRF&MistakeCnt=0&
                                                 AMOUNT=5009%2E00&Incomer=null&AccountNumber=133967805&AccountStyleDescription=APP&
                                                 AccountIndexValue=2&AccountNumberTo=5221261092987044&AccountStyleDescriptionTo=CRC&
                                                 AccountIndexValueTo=3&BENCODE=null&BENNAME=null&CUSTREF=null&PHONENUMBER=null&IncreaseDecrease
                                                 =null#ConfirmTransactionDetails.FldConfirmTransaction
2008-08-01/16:39:32.792 METRIC 003C0089-0C1C19F6 goto :#ConfirmTransactionDetails
2008-08-01/16:39:32.792 METRIC 003C0089-0C1C19F6 form_enter :ConfirmTransactionDetails
2008-08-01/16:39:32.796 METRIC 003C0089-0C1C19F6 form_select :Transfer:BLOCK
2008-08-01/16:39:32.799 METRIC 003C0089-0C1C19F6 form_select :TransferFromAccountNo:BLOCK
2008-08-01/16:39:32.803 METRIC 003C0089-0C1C19F6 form_select :TransferToAccount:BLOCK
2008-08-01/16:39:32.805 METRIC 003C0089-0C1C19F6 form_select :Confirm:BLOCK
2008-08-01/16:39:32.807 METRIC 003C0089-0C1C19F6 form_select :FldConfirmTransaction:FIELD
2008-08-01/16:39:32.959 METRIC 003C0089-0C1C19F6 input_start :VOICE
2008-08-01/16:39:32.959 METRIC 003C0089-0C1C19F6 prompt /usr/local/phoneweb/tmp/003C0089-0C1C19F6-0001//promptsfile.003C0089-0C1C19F6-0001.13
                                                 :audio|http://10.208.26.24/jsp/VoiceGenie/Audio/Shireen/ENGLISH/Speech/GEN003.wav;audio
                                                 |http://10.208.26.24/jsp/VoiceGenie/Audio/Shireen/ENGLISH/Speech/SCT024.wav;
2008-08-01/16:39:40.061 METRIC 003C0089-0C1C19F6 asr_trace bargein:_bargein_
2008-08-01/16:39:40.085 METRIC 003C0089-0C1C19F6 prompt_end asrbargein:_bargein_
2008 08 01/16:39:43.450 METRIC 003C0089 0C1C19F6 asr_trace ASR_NOMATCH_WITH_NBEST:results:+<_gram5>turn the guide on|1|<turn the guide on>
2008-08-01/16:39:43.451 METRIC 003C0089-0C1C19F6 event nomatch:2|REJECTED(ASR)
2008-08-01/16:39:43.452 METRIC 003C0089-0C1C19F6 event_handler_enter :nomatch maxspeechtimeout|http://10.208.26.24/jsp/VoiceGenie/Speech/Controls
                                                 /Common/ConfirmTransactionControl/SayConfirmTransaction.jsp?TransactionType=TRF&MistakeCnt=0&
                                                 AMOUNT=5009%2E00&Incomer=null&AccountNumber=133967805&AccountStyleDescription=APP&
                                                 AccountIndexValue=2&AccountNumberTo=5221261092987044&AccountStyleDescriptionTo=CRC&
                                                 AccountIndexValueTo=3&BENCODE=null&BENNAME=null&CUSTREF=null&PHONENUMBER=null&IncreaseDecrease
                                                 =null#ConfirmTransactionDetails.FldConfirmTransaction
2008-08-01/16:39:43.453 METRIC 003C0089-0C1C19F6 goto :#ConfirmTransactionDetails
2008-08-01/16:39:43.453 METRIC 003C0089-0C1C19F6 form_enter :ConfirmTransactionDetails
2008-08-01/16:39:43.455 METRIC 003C0089-0C1C19F6 form_select :Transfer:BLOCK
2008-08-01/16:39:43.458 METRIC 003C0089-0C1C19F6 form_select :TransferFromAccountNo:BLOCK
2008-08-01/16:39:43.462 METRIC 003C0089-0C1C19F6 form_select :TransferToAccount:BLOCK
2008-08-01/16:39:43.465 METRIC 003C0089-0C1C19F6 form_select :Confirm:BLOCK
2008-08-01/16:39:43.467 METRIC 003C0089-0C1C19F6 form_select :FldConfirmTransaction:FIELD
2008-08-01/16:39:43.619 METRIC 003C0089-0C1C19F6 input_start :VOICE
2008-08-01/16:39:43.620 METRIC 003C0089-0C1C19F6 prompt /usr/local/phoneweb/tmp/003C0089-0C1C19F6-0001//promptsfile.003C0089-0C1C19F6-0001.14:
                                                 audio|http://10.208.26.24/jsp/VoiceGenie/Audio/Shireen/ENGLISH/Speech/GEN005.wav
2008-08-01/16:39:51.551 METRIC 003C0089-0C1C19F6 asr_trace bargein:_bargein_
2008-08-01/16:39:51.552 METRIC 003C0089-0C1C19F6 prompt_end asrbargein:_bargein_
2008-08-01/16:39:53.720 METRIC 003C0089-0C1C19F6 asr_trace ASR_NOMATCH_WITH_NBEST:results:+<_gram3>consultant|16|<consultant>
2008-08-01/16:39:53.723 METRIC 003C0089-0C1C19F6 event nomatch:3|REJECTED(ASR)
2008-08-01/16:39:53.724 METRIC 003C0089-0C1C19F6 event_handler_enter :noinput nomatch maxspeechtimeout|http://10.208.26.24/jsp/VoiceGenie/Speech
                                                 /Controls/Common/ConfirmTransactionControl/SayConfirmTransaction.jsp?TransactionType=TRF&
                                                 MistakeCnt=0&AMOUNT=5009%2E00&Incomer=null&AccountNumber=133967805&AccountStyleDescription=
                                                 APP&AccountIndexValue=2&AccountNumberTo=5221261092987044&AccountStyleDescriptionTo=CRC&
                                                 AccountIndexValueTo=3&BENCODE=null&BENNAME=null&CUSTREF=null&PHONENUMBER=null&IncreaseDecrease
                                                 =null#ConfirmTransactionDetails.FldConfirmTransaction
2008-08-01/16:39:53.724 METRIC 003C0089-0C1C19F6 event SystemError:1
2008-08-01/16:39:53.724 METRIC 003C0089-0C1C19F6 event_handler_enter :SystemError|http://10.208.26.24/jsp/VoiceGenie/Speech/Common/RootDoc.jsp
2008-08-01/16:39:53.725 METRIC 003C0089-0C1C19F6 submit :http://10.208.26.24/jsp/VoiceGenie/Common/ErrorNavigators.jsp#ErrorSpeechMainMenu
                                                 |namelist|ActionEvent;undefined;
```

Figure 3.1: Extract of an IVR log event file

Table 3.1: Descriptions of log events used to generate data sets

Event name	Description
asr_trace	Provides information on ASR events such as prompt barge-in and successful recognition.
Event	Provides information on an event that is thrown. This log entry is used to determine difficulty statistics such as 'no matches', 'no inputs', 'maximum speech timeout' as well as 'system error' statistics.
incall_end	This log event indicates the reason for terminating the calling. The event indicates if the caller disconnected, the application ended the call or the call ended due to a system error.
transfer_result	This log event indicates that the call is transferred to a CSA. The event also indicates whether or not the transfer is successful.
form_select	Form_select log event indicates the fields that the call accessed. This event is used to identify the 'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation'. This log event also assisted in determining Dual Tone Multi-Frequency (DTMF) transfers.
incall_begin	The incall_begin log event marks the start of an inbound call to the IVR. The incall_begin , incall_end and transfer_result log events were used to calculate the duration of a call.

As mentioned, VoiceXML applications are voice-based dialog scripts that consist of form and menu elements. The form and menu elements are used to group input as well as output sections together. A block element is used to specify executable content to be executed in the order stated. Block elements are used to query backend databases for information required. A field element is used to obtain and interpret

user input information. As a result, form elements may contain block and field elements (VoiceGenie Technologies, 2009).

Field elements may comprise of elements that facilitates querying the caller for the required information and reacting to caller behaviour experienced. The sequence of elements that occur within the parent field element is illustrated in Figure 3.2.

Prompt elements provide voice applications the ability to execute specific audio files that contain the recorded query. Depending on the answer provided by the caller, the application directs the caller to a specific call flow. This logic will be contained within the filled elements. Elements such as goto, return and submit are used to manage application call flows. Filled elements that occur within field elements allow the voice applications to respond to information received. Catch elements contained within field elements provide the VoiceXML application with the capabilities to handle specific events such as "no matches", "no inputs" and caller transfer requests. Figure 3.3 consists of an example of VoiceXML field code. When this VoiceXML application is executed, the corresponding log events generated are illustrated in Figure 3.4.

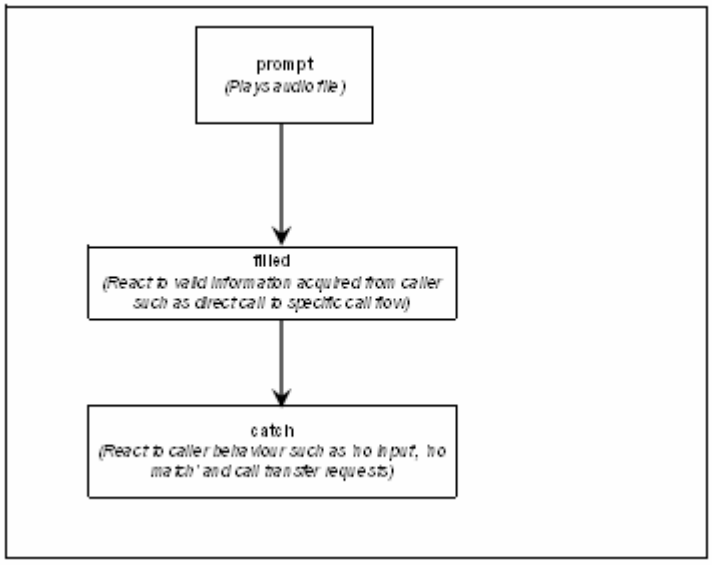

Figure 3.2: Sequence of child elements within parent field element

```
<field slot = "result"  name = "FldSayAmount" >
  <grammar src = "/jsp/VoiceGenie/Grammars/ourCurrency.xml" />
  <prompt>
    <audio expr = "'/jsp/VoiceGenie/Audio/' + SPEECHVOICEARTIST + '/' + SPEECHPROMPTLANGUAGE + '/Speech/SAM002.wav'">
      How much would you like to pay
    </audio>
  </prompt>
  <filled>
    <assign name = "SayAmount_Status" expr = "'SUCCESS'" />
      <return namelist = "SayAmount_Status FldSayAmount$ FldSayAmount ApplicationLastResult" />
  </filled>
  <catch event = "noinput" count = "1">
    <prompt>
      <audio expr = "'/jsp/VoiceGenie/Audio/' + SPEECHVOICEARTIST + '/' + SPEECHPROMPTLANGUAGE + '/Speech/SAM009.wav'">
        To continue with this transaction you need to tell me what amount you would like to use.
        Just say an amount as you normally would. For instance, say "three hundred rand". Say that amount now.
      </audio>
  </prompt>
  </catch>
  <catch event = "noinput" count = "2">
    <prompt>
      <audio expr = "'/jsp/VoiceGenie/Audio/' + SPEECHVOICEARTIST + '/' + SPEECHPROMPTLANGUAGE + '/Speech/GEN002.wav'">
           I still don't think you said anything.
      </audio>
    </prompt>
    <assign name = "SayAmount_Status" expr = "'SpeechDifficulties'" />
    <goto nextitem = "DTMF_Amount" />
  </catch>
  <catch event = "nomatch maxspeechtimeout" count = "1" >
    <prompt>
      <audio expr = "'/jsp/VoiceGenie/Audio/' + SPEECHVOICEARTIST + '/' + SPEECHPROMPTLANGUAGE + '/Speech/GEN003.wav'">
        I'm not sure what you said. Please say the amount that you want to use now.
      </audio>
    </prompt>
  </catch>
  <catch event = "nomatch maxspeechtimeout" count = "2" cond="GUIDESETTING=='OFF'">
    <prompt>
      <audio expr = "'/jsp/VoiceGenie/Audio/' + SPEECHVOICEARTIST + '/' + SPEECHPROMPTLANGUAGE + '/Speech/GEN005.wav'">
        I'm still not sure what you said.
      </audio>
    </prompt>
    <assign name = "SayAmount_Status" expr = "'SpeechDifficulties'" />
    <goto nextitem = "DTMF_Amount" />
  </catch>
</field>
```

Figure 3.3: Example of VoiceXML field code

```
2008-08-01/16:38:45.043 METRIC 003C0089-0C1C19F6 form_select :FldSayAmount:FIELD
2008-08-01/16:38:45.076 METRIC 003C0089-0C1C19F6 input_start :VOICE
2008-08-01/16:38:45.076 METRIC 003C0089-0C1C19F6 prompt /usr/local/phoneweb/tmp/003C0089-0C1C19F6-0001
                                                 //promptsfile.003C0089-0C1C19F6-0001.10:audio|http://10.208.26.24/jsp/VoiceGenie/Audio/Shireen
                                                 /ENGLISH/Speech/SAM002.wav;
2008-08-01/16:38:46.791 METRIC 003C0089-0C1C19F6 prompt_end done
2008-08-01/16:38:48.608 METRIC 003C0089-0C1C19F6 asr_trace bargein:_bargein_
2008-08-01/16:38:51.626 METRIC 003C0089-0C1C19F6 asr_trace ASR_DONE:results:+<_gram1>5009.00|73|<five thousand nine>
2008-08-01/16:38:51.627 METRIC 003C0089-0C1C19F6 filling :SayAmount.FldSayAmount:FIELD:5009.00
2008-08-01/16:38:51.628 METRIC 003C0089-0C1C19F6 input_end :5009.00
2008-08-01/16:38:51.628 METRIC 003C0089-0C1C19F6 filled_enter ALL:FldSayAmount
```

Figure 3.4: Corresponding IVR log field events generated

The research conducted entailed the creation of a call classification system for a pay beneficiary self-service IVR application. The data utilized in the creation of the system has been provided by Intelleca, a division of the Bytes Technology Group. As a result, 'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation'

field classifiers were created. Caller behaviour per field is unique. For example, at a 'Say confirmation' field the caller is required to say 'yes' or 'no'. However, the caller is requested to say the currency value at the 'Say amount' field. As a result, the duration to complete the VoiceXML application field is much shorter at the confirmation field. Therefore, each classifier is trained with data relevant to the field.

3.3 Architecture of the call classification system

The classification system developed consists of two components, the field and the call performance classification components. The field classification component consists of classifiers that categorize caller behaviour at a field within the IVR automated applications into specific interaction classes. A call performance classifier utilizes these interaction classes to evaluate the performance of the customer call in relation to caller behaviour.

As a result, the call performance classes can assist in determining trends of caller behaviour within the self-service systems. For example, the caller behaviour classification application can identify calls that transferred or disconnected at the final task of the automated process as well as calls where the Advanced Speech Recognition (ASR) performed poorly. Thereafter, analysts can listen to a sample of these calls and then determine the reason for this. The caller behaviour classification system can also identify the field that resulted in the majority of the callers transferring to a customer service agent or caller disconnecting. The field interaction classes can elaborate on the reason for the caller behaviour experienced.

In order to develop such an application, the classification of data must be accurate. As a result, field classifiers as well as call performance classifiers that were developed utilizing Artificial Neural Network (ANN), Fuzzy Inference System (FIS), Genetic Algorithm (GA) and Support Vector Machine (SVM) techniques are compared to

determine the most appropriate methodology. Ensembles of classifiers were also considered.

As mentioned in section 2.2.1, Multi-Layer Perceptron (MLP) and Radial Basis Function (RBF) ANN architectures considered, are feed-forward structures whereby each unit receives inputs only from lower layer units.

GA solutions are known to be robust optimization procedures based on the mechanism of the natural evolution. GA solutions have the capability of locating a global optimum as these procedures do not use any derivative information and GA solutions search from multiple points.

A fundamental method in data mining and pattern recognition is clustering of data. Fuzzy clustering involves the natural grouping of data in a large data set and provides a basis for constructing rule-based fuzzy model (Elmzabi et al., 2007). Fuzzy c-means, mountain clustering, subtractive clustering and entropy-based fuzzy clustering are among the fuzzy clustering algorithms used. In this research subtractive clustering is of concern.

As mentioned in section 2.5, SVMs perform well for modeling challenging high-dimensional data. SVMs have been used successfully in text mining (Joachims, 1998), image mining (Tolambiya and Kalra, 2008), bioinformatics (Ramaswamy et al., 2001) and information fusion (Pal and Mather, 2004). SVM performance has been demonstrated to be superior to the performance of decision trees, neural networks and Bayesian techniques (Joachims, 1998), (Ramaswamy et al., 2001), (Pal and Mather, 2004).

The classification of data into various classes has been an important research area for many years. ANNs have been applied to pattern classification (Patel and Marwala, 2008). Research has also been conducted on fuzzy classification. This resulted in

many algorithms, such as fuzzy K-nearest neighbour (Keller et al., 1985) and fuzzy c-means (Bezdek, 1981), being applied to classification problems. Fuzzy systems constructed using GA solutions have been utilized (Russo, 1998). Fuzzy neural networks have also been employed in pattern classification applications (Patel and Marwala, 2006). SVMs have been applied to multi-category classification problems (Hsu and Lin, 2002). These classification tasks have also been implemented by combining multiple simpler specialized classifiers (Zadrozny, 2002).

The caller behaviour classification system developed is illustrated in Figure 3.5. As mentioned the system comprises of two components. Figure 3.5 and Table 3.2 show the inputs as well as outputs of the field classification component. These specific inputs have been selected to characterize the caller experience at a field within a VoiceXML application. The outputs of the classifiers summarize the caller field behaviour through the use of interaction classes.

The confidence input illustrates the IVR speech recognition probability. The value is a percentage. The larger the percentage, the greater the probability the system interpreted the caller successfully. A caller may answer a question the VoiceXML application prompts with a response that the application does not accommodate. These events are represented by the no match inputs. In general, most VoiceXML applications accommodate three no match events per field. On a third no match event, the call is transferred to a DTMF field. If the caller fails to complete the DTMF field successfully within the first attempt, the call is transferred to a customer service agent. The same process is used for the third no input and maximum speech timeout events. The no match field classifier inputs assist in identifying callers that misunderstood the VoiceXML prompt as well as unique responses that the VoiceXML application can use to improve field recognition coverage.

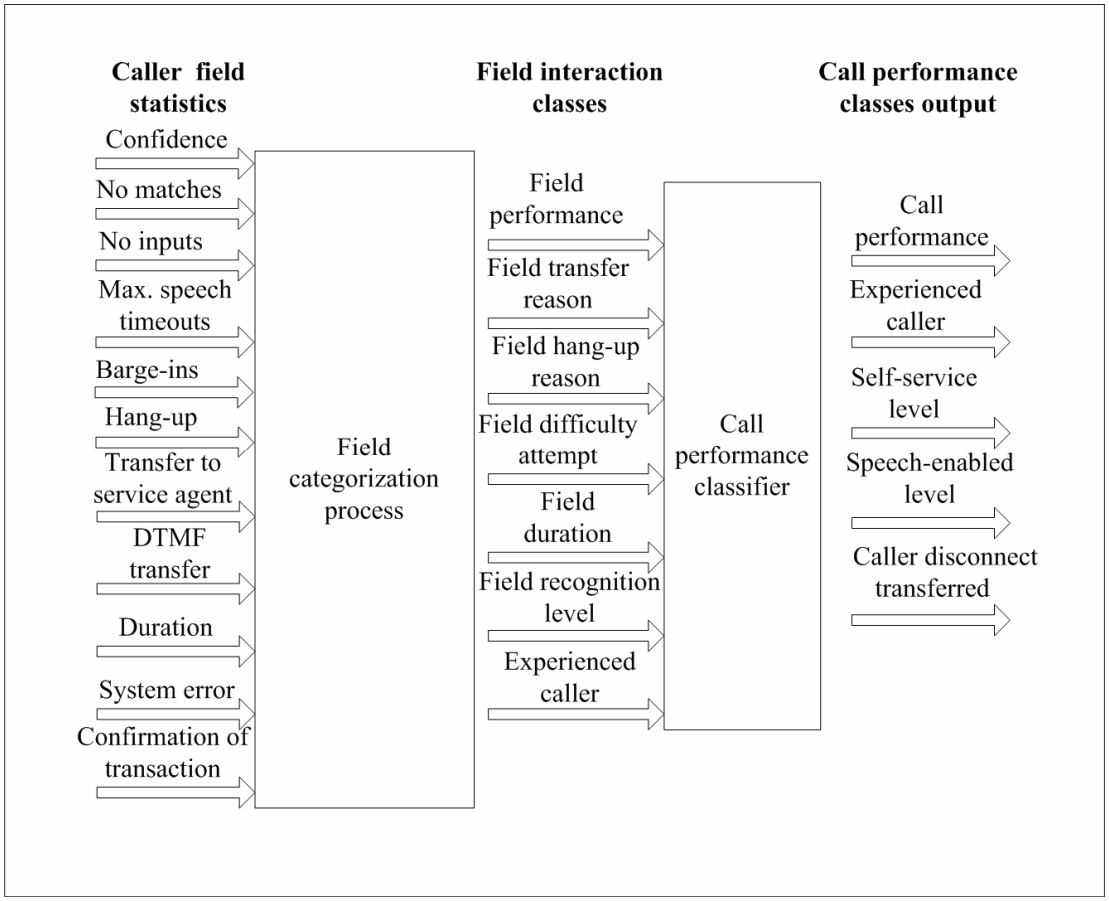

Figure 3.5: Call classification system

In response to a prompt, a caller may remain silent. These events are represented by the no input parameters. VoiceXML applications normally accommodate three no input events on each field. These input parameters assist in identifying callers that were confused when prompted with the automated application question. As a result, the caller remained silent.

Callers may reply to VoiceXML applications by talking beyond the allocated timeout period of the field. These events are represented by the maximum speech timeout input parameters of the field classifiers (Nuance, 2002). Maximum speech timeout input parameters are important as they assist in determining whether the timeout periods are adequate for callers to complete their responses.

Table 3.2: Field classification component inputs and outputs

Inputs	Outputs	Output interaction class
Confidence	Field performance	Good, acceptable, investigate, bad
No matches	Field transfer reason	Unknown, difficulty
No inputs	Field hang-up reason	Unknown, difficulty
Max speech timeouts	Field difficulty attempt	Attempt 1, attempt 2, attempt 3
Barge-ins	Field duration	High, medium, low
Hang-up	Field recognition level	High, medium, low
Transfer to Service Agent	Experienced caller	True, false
DTMF transfer		
Duration		
System error		
Confirmation of transaction		

Barge-in input parameters illustrate whether or not a caller interrupted the application while the automated question prompt played. Caller disconnects, transfer to DTMF, transfers to CSA and system errors are represented by the hang-up, DTMF transfer, transfer to service agent and system error input parameters, respectively. These inputs can also assist in determining the level of difficulty the caller experienced in the field. The duration input parameter illustrates the time the caller spent completing the field.

Confirmation of transaction represents whether or not the caller verified the application recognition as being true.

The field performance output interaction class of the classifier will illustrate whether the caller behaviour is good, acceptable, investigate or bad. The field transfer reason and field hang-up reason interaction classes attempt to identify the motivation for the transfer to CSA or caller disconnect, respectively. Field difficulty attempt interaction class computes the number of difficulty events that occurred during the field interaction. The field duration as well as field recognition level classes illustrate three categories of performance; low, medium and high. As a result, these output parameters will assist in characterizing the caller experience at a VoiceXML field. Experienced caller output parameter categorizes whether or not the caller is a regular user of the application and is therefore comfortable with the application call flow. In determining the number of experienced callers, the contact center can determine the usage of the application.

The function of the call performance classifier is to provide a summarized performance evaluation of the complete call based on all fields accessed during the call. Figure 3.5 and Table 3.3 illustrates the inputs as well as outputs of the component. The call performance output class provides a measure of the call performance based on field performance interaction classes. Experienced caller output parameter assists in determining whether the caller is familiar with the IVR application as the caller has used the application previously. Self-service level illustrates a measure of the extent of the IVR application usage before the application ended the call, caller disconnects or transfer to CSA event occurs. Speech-enabled level output parameter illustrates three categories of performance; good, acceptable or investigate. This provides a measure of the number of fields the caller completed successfully using ASR. The caller disconnect transferred call performance output parameter identifies the field a caller disconnect or transfer event occurred utilizing the field classifier disconnect and transfer interaction classes.

Therefore, the self-service level call performance output classes provide businesses with call containment and task completion metrics. The field transfer reason, hang-up reason, difficulty attempt as well as duration field performance interaction classes assists the contact center in measuring efficiency and usability for each task within the IVR application. The caller disconnect transferred call performance output class also assists in measuring efficiency of the complete call. The field performance interaction as well as the call performance output class provides businesses with the capabilities to quantify customer satisfaction.

Table 3.3: Call performance classification component inputs and outputs

Inputs (per field)	Outputs	Output performance class
Field performance	Call performance	Good, acceptable, investigate, bad
Field transfer reason	Experienced caller	True, false
Field hang-up reason	Self-service level	Good, acceptable, investigate, bad
Field difficulty attempt	Speech-enabled level	Good, acceptable, investigate
Field duration	Caller disconnect transferred	Field Say Account transfer Field Say Account caller disconnect Field Say Amount transfer Field Say Amount caller disconnect Field Select Beneficiary transfer Field Select Beneficiary caller disconnect Field Say Confirmation transfer Field Say Confirmation caller disconnect None
Field DTMF transfer		
Field recognition level		
Experienced caller		

3.4 Selection and preprocessing of data

As previously stated, the data utilized in implementing the field classification component classifiers has been based on data extracted from IVR log event files. A business intelligence solution that involved Extract, Transform and Load (ETL) processes has been developed to extract as well as compute information such as recognition confidence values, duration values and call completion information. This information is stored within a database and is then manipulated utilizing specific rules to generate the data sets. Rules such as if no hang-up, transfer to CSA, DTMF transfer, system error, no inputs, no matches or maximum speech timeouts occur, but the confidence level at the field is greater than 80%, the duration to complete the field is less than the average field duration and the field confirmation is true, the field performance interaction class would be computed as 'good', were followed. Table 3.4 illustrates the rules that were followed to create the respective field performance interaction classes.

The call performance classification component input data set values were calculated using the interaction classes computed based on the rules mentioned above. Similar rules such as when all the field classifiers compute the field performance interaction class as good, acceptable, investigate or bad the call performance level would be good, acceptable, investigate or bad, respectively, were used to create the call performance classification output data set. The call performance output class provides a measure of the call performance based on field performance interaction classes. The experienced call output class used the field classifier experienced caller outputs to generate an output value. If two or more of the field classifiers experienced caller output is set to true, the experienced call output class would also be true. The self-service level, speech enabled level and caller disconnect transferred output parameters utilized the field classifier disconnect and transfer interaction classes to compute an output value. However, the speech enabled level output value calculation

is also based on DTMF transfer information. Table 3.5 illustrates the rules that computed the call performance classification output classes.

No match, no input and maximum speech timeout information has been presented to the field classifiers, using a binary notation. These inputs are presented by three digit binary words. For example, if a ‘no match 1’ and a ‘no match 2’ occur at a field, the binary notation will be ‘011’, the sum of the binary notations of ‘No match 1’ ‘No match 2’ and ‘No match 3’. A similar binary notation is employed for the no input and maximum speech timeout classifier inputs. The barge-in, hang-up, transfer to CSA, DTMF transfer, system error and confirmation of transaction input information were represented by bit binary words. A similar binary notation scheme has also been utilized to interpret and present the interaction classes to the call performance component. This data presentation method has also been employed in interpreting the call performance classification component output values. Table 3.6 comprises of the binary notations utilized to present the inputs to the field classifiers.

Table 3.4: Field performance classifier interaction output class rules

Outputs	Output interaction class	Conf.[1]	No match	No input	Max. speech timeout	Barge - in	Hang - up	Transfer to CSA	DTMF transfer	Duration	System Error	Confirm transaction
Field performance	Good	>80	0	0	0	_[2]	0	0	0	< average	0	1
	Acceptable	80 to 70	<=2	<=2	<=2	-	0	0	-	< average	0	1
	Investigate	<= 70	<=2	<=2	<=2	-	1	0	-	>= average	-	-
	Bad	<= 70	<=2	<=2	<=2	-	1	1	-	>= average	-	0
Field transfer reason	Unknown	-	< 2	< 2	< 2	-	-	1	-	-	-	-
	Difficulty	-	>= 2	>= 2	>= 2	-	-	1	-	-	-	-
Field hang-up reason	Unknown	-	< 2	< 2	< 2	-	1	-	-	-	-	-
	Difficulty	-	>= 2	>= 2	>= 2	-	1	-	-	-	-	-
Field difficulty attempt	Attempt 1	-	1	1	1	-	-	-	-	-	-	-
	Attempt 2	-	2	2	2	-	-	-	-	-	-	-
	Attempt 3	-	2	2	2	-	-	-	-	-	-	-
Field duration	High	-	-	-	-	-	-	-	-	> average	-	-
	Medium	-	-	-	-	-	-	-	-	= average	-	-
	Low	-	-	-	-	-	-	-	-	< average	-	-
Field recognition level	High	>= 90	-	-	-	-	-	-	-	-	-	-
	Medium	90 to 70	-	-	-	-	-	-	-	-	-	-
	Low	<= 70	-	-	-	-	-	-	-	-	-	-
Experienced caller	True	> 80	<=1	<=1	<=1	1	0	0	0	< average	0	1
	False	<= 80	> 1	> 1	> 1	0	0 or 1	0 or 1	0 or 1	>= average	0 or 1	0 or 1

[1] Conf. represents Confidence Levels
[2] – represents not applicable.

Table 3.5: Rules followed to compute call performance classifier output classes

Outputs	Performance class	Rules
Call performanc e	Good	'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' field performance is "Good"
	Acceptable	'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' field performance is "Acceptable"
	Investigate	'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' field performance is "Investigate"
	Bad	'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' field performance is "Bad"
Experienced caller	True	Sum of 'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' experienced caller interaction classes is greater than or equal to 2
	False	Sum of 'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' experienced caller interaction classes is less than 2
Self-service level	Good	No transfer to CSA or caller disconnect occurs at any of the modules within the application.
	Acceptable	Transfer to CSA or caller disconnect occurs at 'Say confirmation' module.
	Investigate	Transfer to CSA occurs at any of the modules within the application.
	Bad	Transfer to CSA or caller disconnect occurs at any of the modules within the application.
Speech-enabled level	Good	No DTMF transfer occurs on any of the modules within the application.
	Acceptable	No DTMF transfer or DTMF transfer occurs at 'Say amount' module.
	Investigate	DTMF transfer occurs on any of the modules within the application.
Caller disconnect transferred	Field Say Account transfer	Transfer to CSA occurs at 'Say account' module.
	Field Say Account caller disconnect	Caller disconnect occurs at 'Say account' module.
	Field Say Amount transfer	Transfer to CSA occurs at 'Say amount' module.
	Field Say Amount caller disconnect	Caller disconnect occurs at 'Say amount' module.
	Field Select Beneficiary transfer	Transfer to CSA occurs at 'Select beneficiary' module.
	Field Select Beneficiary caller disconnect	Caller disconnect occurs at 'Select beneficiary' module.
	Field Say Confirmation transfer	Transfer to CSA occurs at 'Say confirmation' module.
	Field Say Confirmation caller disconnect	Caller disconnect occurs at 'Say confirmation' module.
	None	No Caller disconnect or transfer to CSA occurred.

Table 3.6: Binary notation of inputs to field classifiers

Inputs	Input values	Binary notation
Confidence	-	-
No matches	No match 1	001
	No match 2	010
	No match 3	100
No inputs	No input 1	001
	No input 2	010
	No input 3	100
Max speech timeouts	Max speech timeout 1	001
	Max speech timeout 2	010
	Max speech timeout 3	100
Hang-up	True	1
	False	0
Barge-ins	True	1
	False	0
Transfer to Service Agent	True	1
	False	0
DTMF transfer	True	1
	False	0
Duration	-	-
System error	True	1
	False	0
Confirmation of transaction	True	1
	False	0

Table 3.7 and table 3.8 illustrate the binary notation used to represent the interaction and call performance classes.

Table 3.7: Binary notation of field output interaction

Outputs	Output interaction class	Binary notation
Field performance	Good	0001
	Acceptable	0010
	Investigate	0100
	Bad	1000
Field transfer reason	Unknown	10
	Difficulty	01
Field hang-up reason	Unknown	10
	Difficulty	01
Field difficulty attempt	Attempt 1	001
	Attempt 2	010
	Attempt 3	100
Field duration	High	100
	Medium	010
	Low	001
Field recognition level	High	100
	Medium	010
	Low	001
Experienced caller	True	1
	False	0

The confidence and duration input parameters of the field classifiers were preconditioned by normalizing the data. Normalizing the data entails manipulating the data sets such that the values within the sets are between zero and one. The purpose of normalizing the data sets is to modify the variable levels to a reasonable value. If such a transformation is not employed, the value of the variable could be too large for the network to process, especially when several layers of nodes within a

neural network are involved (Baum and Haussler, 1989). Normalizing the data sets also reduces the fluctuation and noise within the data (Baum and Haussler, 1989). There are a variety of practical reasons that illustrate normalizing the data sets can result in faster training and reduce the chances of obtaining local optima. Some of these reasons include better numerical conditioning (Hessian matrices), better weight initialization values and better weight decay estimates (Baum and Haussler, 1989).

The field classification component classifiers developed were trained utilizing the normalized data sets. Normalization is accomplished by acquiring the minimum and maximum values within the data sets. These values are then utilized to compute the normalized values. Due to the binary word representation utilized to present the remaining field classifier inputs and the interaction classes computed, normalization of these values is not necessary.

In order to ensure that over-fitting and under-fitting were avoided, the data has been divided into three sets. The data is divided into training, validation and test sets. The training data set is used to train the algorithms to identify the general classification groups within the data. The validation data set is used to assess the classifier and the test data is used to confirm the classification capability of the developed models. This method is employed in the implementation of the field and call performance classification components.

Table 3.8: Binary notation of call performance classification classes

Outputs	Output interaction class	Binary notation
Call performance	Good	0001
	Acceptable	0010
	Investigate	0100
	Bad	1000
Experienced caller	True	1
	False	0
Self-service level	Good	0001
	Acceptable	0010
	Investigate	0100
	Bad	1000
Speech-enabled level	Good	001
	Acceptable	010
	Investigate	100
Caller disconnect transferred	Field Say Account transfer	000000001
	Field Say Account caller disconnect	000000010
	Field Say Amount transfer	000000100
	Field Say Amount caller disconnect	000001000
	Field Select Beneficiary transfer	000010000
	Field Select Beneficiary caller disconnect	000100000
	Field Say Confirmation transfer	001000000
	Field Say Confirmation caller disconnect	010000000
	None	100000000

Chapter 4

Artificial Neural Networks classifiers

4.1. Introduction

Multi-Layer Perceptron (MLP) and the Radial Basis Function (RBF) Artificial Neural Network (ANN) architectures were utilized in the development of both the field and call performance classification components. The MLP and RBF ANN architectures are possibly the most extensively employed ANNs in pattern classification (Nabney, 2002). Due to the non-linear capabilities of these computational intelligent methods, they are said to be excellent universal approximators that provide highly accurate solutions. As a result, these networks produce very practical tools for classification and inversion problems (Bishop, 1995).

This section details the implementation of the MLP and RBF ANN classifiers.

4.2. MLP and RBF classifiers

It has been stated that a network with one hidden layer, provided with sufficient data, can be used to model any function (Beale and Jackson, 1990). As a result, the MLP and RBF ANN architectures employed consisted of only one hidden layer. The MLP network hidden layer consists of non-linear activation functions. The choice of the activation function is mainly dependant on the application of the ANN (Bishop, 1995). However, it has been found that the hyperbolic tangent activation function offers a practical advantage of giving rise to faster convergence during training (Nabney, 2002). As a result, this function has been utilized within the MLP networks. The MLP network output layer also consists of activation functions. It has been stated that the appropriate selection of the output-unit activation function for a classification problem is the logistic sigmoidal function (Nabney, 2002). As a result, this function has been employed within the output layer of the MLP network. The RBF network that has been developed contained a Gaussian activation function within its hidden layer and a linear activation function within its output layer.

The MLP and RBF ANN implementation process involved the optimization of the network architectures. As a result, this entailed the identification of the correct number of hidden neurons that would yield the most accurate results. Binary and real coded Genetic Algorithm (GA) solutions were employed to optimize the field as well as call performance ANN classifier architectures. Populations of MLP and RBF ANN individuals were generated by the GA solutions. Thereafter, the GA solutions were evaluated utilizing performance metrics. The best performing classifiers were identified. Depending on the accuracy of the ANN classifiers, if unacceptable accuracy values were achieved, the classification threshold employed by the ANN was optimized. The ANN classifiers were then compared to determine the most accurate solution as well as the most appropriate ANN for this problem of concern. Figure 4.1 illustrates the ANN implementation process followed.

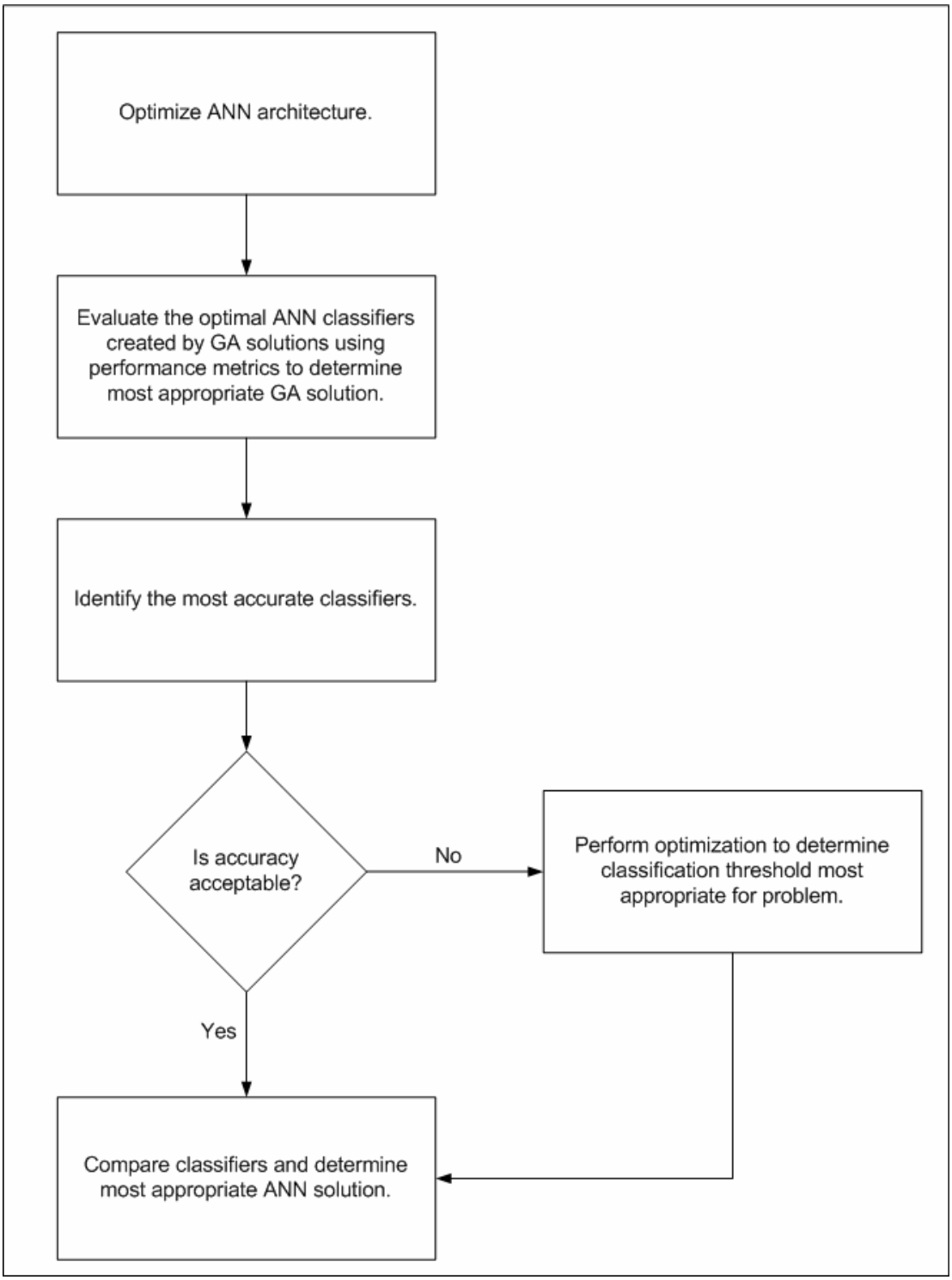

Figure 4.1: Artificial Neural Network implementation process followed

4.2.1 Optimization of Artificial Neural Network architecture

GA solutions are known to be robust optimization procedures based on the mechanism of the natural evolution. GA solutions have the capability of locating a global optimum as these procedures search from multiple points. In traditional GA solution, binary representation has been used for chromosomes. Floating point representation, real coded GA solution, of parameters as a chromosome has also been used (Houck et al., 1995).

An error function that mapped the number of hidden nodes to the accuracy of the developed network is used as the evaluation function for the GA solution. The fitness of the individuals within a population is determined by calculating the accuracy of the ANNs when presented with validation and test data sets. The minimum value of these accuracies determined the fitness of the individual. The ANNs were trained using the training data set. The outputs of the ANNs were interpreted by utilizing a classification threshold value of 0.5. This implies that if the classifier outputs a value less than 0.5, the output will be regarded as a zero. Similarly, if the output value is larger than or equal to 0.5, the output will be interpreted as a one.

Since this is a classification implementation, the accuracy of the networks developed can no longer be calculated utilizing the sum of square error of the difference between the target and the network output values. Instead, a confusion matrix is employed to identify the number of true and false classifications that are generated by the ANN developed. This is then utilized to calculate the true accuracy of the ANN classifiers, using the following equation:

$$Accuracy = \sqrt{\frac{TP \times TN}{(TP + FN) \times (FP + TN)}} \qquad (4.2.1)$$

where,

TP is the true positive (1 classified as a 1),

TN is the true negative (0 classified as a 0),

FN is the false negative (1 classified as a 0),

FP is the false positive (0 classified as a 1).

The GA solution produced twenty five generations of ten ANN individuals within the population. The GA solutions were limited to produce ANN individuals with the number of hidden nodes between five and one hundred. Due to the predictive capabilities or generalization capabilities reducing as the number of intermediate units increase, ANNs with hidden nodes greater than one hundred were not considered. More hidden nodes increases the dimensionality of the function being fitted, enabling easier training which results from higher training capacity. However, this detrimentally affects the generalization capabilities of the network. A major consideration when developing a suitable ANN for a classification application is to make a trade-off between convergence and generalization (Baum and Haussler, 1989).

In order to produce successive generations, selection of individuals is important in a GA. The selection function determines which of the individuals will survive and move on to the next generation. A probabilistic selection is performed based upon the fitness of an individual such that the superior individuals have a higher chance of being selected. There are several schemes for the selection process. Roulette wheel selection and its extensions, scaling techniques, tournament, normal geometric ranking and elitist models are examples of selection functions used (Houck et al., 1995). The selection approach assigns a probability of selection to each individuals based on its fitness value. GA solutions that used normalized geometric ranking and tournament selection functions were compared in this research.

Crossover and mutation operators provide basic search mechanism of the GA. Crossover operators transform two individuals into two new individuals, while mutation operators alter one individual to produce a single solution. In this research, binary coded GA solution utilized binary mutation and the real coded GA solution used non-uniform mutation genetic operators. Arithmetic and simple cross-over operators have been used within the real coded and binary GA solution, respectively.

4.2.2 Evaluation of Genetic Algorithm solutions employed

Binary and real coded GA solution that used normalized geometric ranking as well as tournament selection functions were compared in terms of computational efficiency and quality of the GA solution. Computational efficiency, in this context, is defined as the number of generations the GA utilized to converge to the most optimal number of hidden nodes. Table 4.1 illustrates the results of the MLP and RBF ANN hidden nodes optimization using GA solution. During the field 'Say account' MLP classifier optimization, it is evident that the binary coded GA solution and the real coded GA solution converged to an optimal value by generation twenty two and nine, respectively. As a result, with regards to the field 'Say account' MLP classifier, the real coded GA solution outperformed the binary coded GA solution. However, during the field 'Say amount' MLP classifier optimization, the GA solution using tournament selection function outperformed the GA solution that employed normalized ranking selection function. The real coded GA that used normalized ranking selection function converged the fastest to an optimal number of field 'Select beneficiary' and 'Say confirmation' MLP classifier hidden nodes solution. However, the GA that employed binary coded tournament selection function also yielded the optimal field 'Say confirmation' MLP classifier number of hidden nodes, utilizing the same number of generations.

In terms of computational efficiency, the binary coded GA and the real coded GA that used normalized geometric ranking selection function outperformed all the GA solutions considered in optimizing the field 'Say account' and 'Say amount' RBF classifier architectures, respectively. However, the real coded GA that employed tournament selection function performed the best in optimizing the field 'Select beneficiary' and 'Say confirmation' RBF classifier architectures.

During the call performance classification RBF ANN architecture optimization, it has been determined that the binary and real coded GA solutions converged to a solution at generation one. However, the call performance classification MLP ANN architecture optimization achieved different results. Table 4.1 illustrates the results of these network implementations. As a result, in terms of computational efficiency, all GA solutions considered in developing the call performance classification component to optimize the RBF architecture performed equally well. However, when optimizing the MLP number of hidden nodes in relation to computational efficiency, the binary coded GA that employed normalized geometric ranking selection outperformed all GA solutions considered. This algorithm converged to a solution at generation one.

Quality of the GA solution is the confirmation that the optimal number of hidden nodes computed by the GA is truly the most suitable value. In order to determine the quality of the GA solutions, field MLP and RBF as well as call performance classifiers were created containing the optimal number of hidden nodes calculated by the algorithms.

Table 4.1: ANN classifier implementation results

GA	Field	MLP				RBF			
		Hidden nodes	Gen[1]	Accuracy		Hidden nodes	Gen[1]	Accuracy	
				Validation	Test			Validation	Test
Field classification component									
Binary coded normalized geometric ranking selection	Say account	7	22	0.9833	0.9507	66	13	0.9478	0.9363
	Say amount	96	21	0.9095	0.9694	**94**	**16**	**0.9143**	**0.9603**
	Select beneficiary	9	22	0.9537	0.9654	61	18	0.9401	0.9529
	Say confirmation	34	19	0.9556	0.9018	73	25	0.9641	0.9033
Binary coded tournament selection	Say account	13	22	0.9782	0.9431	96	18	0.9492	0.9358
	Say amount	95	3	0.9092	0.9699	53	5	0.9123	0.9600
	Select beneficiary	**6**	**7**	**0.9628**	**0.9722**	83	16	0.9426	0.9534
	Say confirmation	**50**	**6**	**0.9557**	**0.9021**	77	16	0.9517	0.9025
Real coded normalized geometric ranking selection	Say account	**8**	**9**	**0.9844**	**0.9513**	70	24	0.9493	0.9371
	Say amount	**9**	**18**	**0.9199**	**0.9796**	31	4	0.9125	0.9595
	Select beneficiary	40	6	0.9499	0.9635	26	19	0.9391	0.9552
	Say confirmation	54	6	0.9567	0.9020	88	24	0.9632	0.9041
Real coded tournament selection	Say account	**8**	**9**	**0.9844**	**0.9513**	**94**	**15**	**0.9499**	**0.9376**
	Say amount	**9**	**1**	**0.9199**	**0.9796**	68	9	0.9117	0.9588
	Select beneficiary	44	23	0.9494	0.9670	**68**	**8**	**0.9443**	**0.9573**
	Say confirmation	**50**	**21**	**0.9557**	**0.9021**	**60**	**12**	**0.9588**	**0.9048**
Call performance classification component									
Binary coded normalized geometric ranking selection	-	75	1	0.9891	0.9896	40	1	0.8281	0.8186
Binary coded tournament selection	-	47	16	0.9890	0.9901	40	1	0.8281	0.8186
Real coded normalized geometric ranking selection	-	**10**	**21**	**0.9904**	**0.9918**	25	1	0.8281	0.8186
Real coded tournament selection	-	69	6	0.9888	0.9893	25	1	0.8281	0.8186

[1]Gen represents Generation

The real coded GA utilizing tournament selection function is best suited for optimizing the field classifier architectures as this algorithm computed the optimal number of hidden nodes that produced the most accurate classifiers. This is true for the field 'Say account', 'Say amount' and 'Say confirmation' MLP classifiers. This is also valid for the field 'Say account', 'Select beneficiary' and 'Say confirmation' RBF classifiers. However, the binary coded GA that used the tournament selection function returned the optimal number of hidden nodes that resulted in the most accurate field 'Select beneficiary' MLP classifier. This GA also computed the same optimal number of 'Say confirmation' MLP hidden nodes as the real coded GA that employed tournament selection function, which produced the best field 'Say confirmation' classifier. Similarly the real coded GA that used normalized geometric ranking selection function yielded the same number of field 'Say amount' MLP hidden nodes as the real coded GA that utilized tournament selection function. However, the binary coded GA that used normalized geometric ranking selection function returned the optimal number of hidden nodes that produced the most accurate field 'Say amount' RBF classifier.

It is evident that, during the call performance RBF ANN number of hidden nodes optimization, the binary coded and real coded GA solutions computed different optimal values. The binary coded GA solutions yielded forty number of hidden nodes and the real coded GA solutions calculated twenty five number of hidden nodes as optimal. During this optimization process, it has been determined that both numbers of hidden nodes achieve the same validation and test data set accuracies. As a result, in relation to quality of the GA solution, all GA solutions perform equally well when optimizing the call performance RBF classifier architectures.

It is also evident, in relation to the call performance RBF ANN number of hidden nodes optimization, when comparing the GA solutions considered, the real coded GA solution yielded a smaller number of hidden nodes value that achieved the same accuracy on both data sets.

Table 4.1 shows the number of hidden nodes that resulted in the most accurate call performance MLP ANN classifier is ten. This number of hidden nodes creates a network that performs accurately on both the validation and test data sets. As a result, this classifier has good generalization capabilities. Therefore, in terms of quality of GA solution, the real coded normalized geometric ranking selection GA is most suited in optimizing the call performance MLP network architecture. However, this algorithm converged to this solution at generation twenty one of twenty five.

4.2.3 Optimization of classification threshold

In order to improve the classification accuracy of the call performance RBF ANN implementations, investigations were conducted to optimize the classification threshold of the RBF ANN. These experiments resulted in a significant improvement in performance on both the validation and test data sets. When utilizing a classification threshold of 0.5, the field MLP ANN and call performance classifier implementations achieved accuracy values larger than eighty five percent. Similarly, the field RBF ANN classifiers yielded accuracy values greater than eighty five percent. As a result, the classification thresholds of 0.5 are appropriate for the field and call performance MLP ANN classifiers as well as field RBF ANN classifier.

Classification threshold has been optimized by minimizing an error function that mapped the classification thresholds to the sum of the sensitivity and specificity of the developed call performance classifiers. In this research, sensitivity is defined as the probability that the classifier categorizes a set of inputs to the correct specific interaction or call performance classes. Specificity is defined as the probability that the classifier indicates that a set of inputs does not correctly belong to specific interaction or call performance classes. The former measure describes the effectiveness of the classifier at categorizing output classes correctly, while the latter characterizes the performance of the classifier at discarding the other output classes.

This optimization process involved varying the classification threshold from 0.1 to 0.7 in iterations of 0.01 for both validation and test data sets. During this process, the optimized hidden nodes of twenty five have been used. For each of the threshold values the accuracy, sensitivity and specificity of the call performance RBF ANN are calculated. The accuracy of the classifier is calculated using, equation (4.2.1) above. The sensitivity and specificity of the RBF ANN is calculated using equation (4.2.2) and equation (4.2.3) below, respectively.

$$Sensitivity = \frac{TP}{TP + FN} \tag{4.2.2}$$

$$Specificity = \frac{TN}{TN + FP} \tag{4.2.3}$$

where,

TP is the true positive (1 classified as a 1),

TN is the true negative (0 classified as a 0),

FN is the false negative (1 classified as a 0),

FP is the false positive (0 classified as a 1).

The optimization criteria employed in this investigation required the identification of the threshold value that yielded the maximum of the sum of sensitivity and specificity. This minimizes the mean of the error rate for positive classifications and the error rate for negative classifications. This optimization criteria is equivalent to determining the point on the Receiver Operating Characteristic (ROC) where the tangent has a slope of one (Freeman and Moisen, 2008). It is also equivalent to maximizing the Youden's index or the true skill statistic.

Table 4.2 illustrates the call performance RBF ANN implementation threshold values that resulted in the largest sum of sensitivity and specificity value for the validation as well as test data sets. As illustrated in the table, the optimal threshold value of 0.65 is an appropriate value for this classification problem.

Table 4.2: Results of call performance RBF ANN classifier threshold optimization

Threshold	Accuracy	Sensitivity	Specificity	Sensitivity + Specificity
Validation				
0.10-0.11	0.6583	0.4408	0.9831	1.4240
0.12-0.18	0.6869	0.4831	0.9765	1.4596
0.19	0.7170	0.5313	0.9678	1.4990
0.20-0.31	0.7530	0.5914	0.9586	1.5500
0.32-0.35	0.7853	0.6535	0.9436	1.5971
0.36-0.57	0.8281	0.7385	0.9285	1.6670
0.58-0.64	0.8394	0.7838	0.8990	1.6827
0.65-0.70	**0.9068**	**0.9784**	**0.8404**	**1.8188**
Test				
0.10-0.11	0.6507	0.4328	0.9783	1.4111
0.12-0.18	0.6789	0.4741	0.9721	1.4462
0.19	0.7097	0.5223	0.9643	1.4866
0.20-0.31	0.7469	0.5833	0.9563	1.5396
0.32-0.35	0.7795	0.6452	0.9419	1.5870
0.36-0.57	0.8186	0.7238	0.9258	1.6496
0.58-0.64	0.8306	0.7688	0.8973	1.6661
0.65-0.70	**0.9019**	**0.9677**	**0.8406**	**1.8083**

Therefore, the optimal RBF call performance classifier achieved an accuracy of ninety one percent and ninety percent on validation and test data sets, respectively. This demonstrates that this classifier has good generalization capabilities. However, the call performance MLP ANN with ten number of hidden nodes is approximately nine percent more accurate than the RBF ANN classifier.

4.2.4 Comparison of field and call performance Artificial Neural Network classifiers

Figure 4.2 and figure 4.3 illustrates the results of the field MLP classifier implementations for validation and test data sets, respectively. These classifiers employed the hyperbolic tangent hidden layer activation function and the logistic sigmoidal output layer activation function. The field classifiers consisted of seventeen inputs and eighteen outputs. The field MLP classifiers were trained utilizing the scaled conjugate gradient algorithm.

When comparing validation and test data set classification performance, it is evident that the 'Say account' and 'Say confirmation' field classifiers yielded larger validation data set accuracies. The 'Say account' and 'Say amount' MLP classifier that were optimized by real coded GA solutions that used normalized geometric ranking as well as tournament selection functions produced the most accurate classifiers. As a result, the MLP ANN, utilizing eight hidden nodes, is most appropriate for 'Say account' field classification. Similarly, the MLP ANN that employed nine hidden nodes is best suited in 'Say amount' field classification.

It is evident from figure 4.2 that the binary coded GA solution that employed tournament selection produced the most optimal 'Select beneficiary' MLP classifier. Therefore, this MLP ANN that contained six hidden nodes is most appropriate for 'Select beneficiary' field classification. In relation to 'Say confirmation' field classification, as illustrated in figure 4.2, the MLP classifiers optimized utilizing GA solutions that employed tournament selection functions yielded the best results. As a result, these MLP classifiers, which consisted of fifty hidden nodes, are best suited to 'Say confirmation' classification.

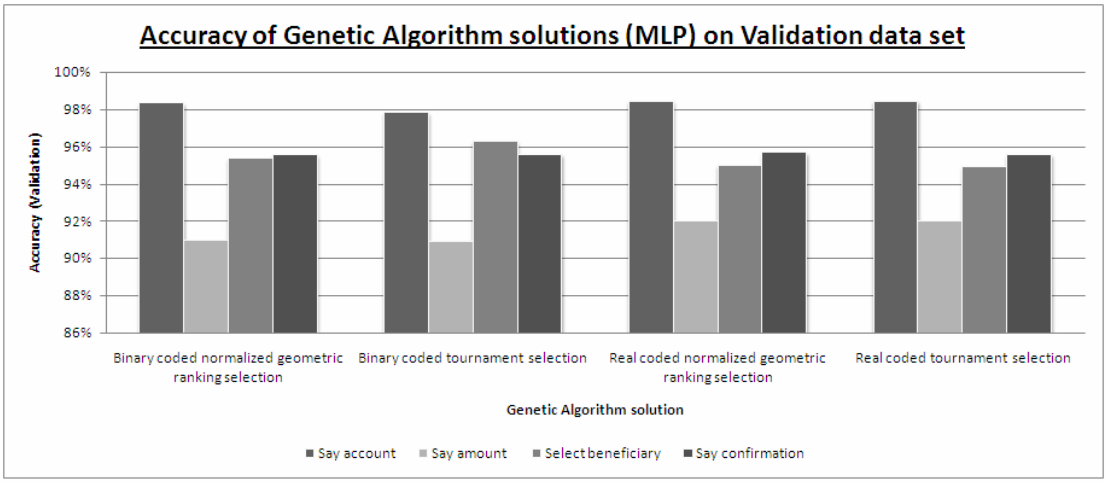

Figure 4.2: MLP ANN Field classifier results (Validation)

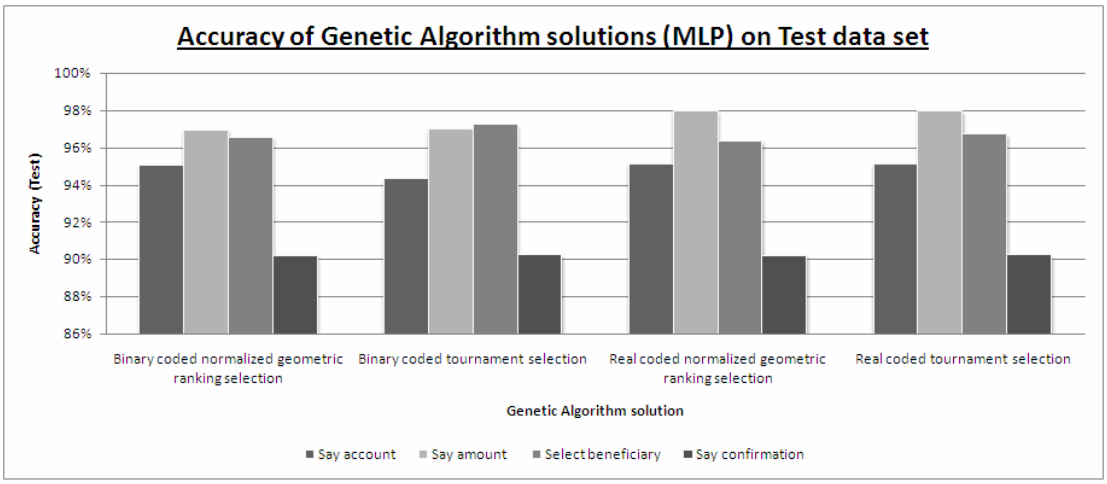

Figure 4.3: MLP ANN Field classifier results (Test)

Figures 4.4 and 4.5 illustrate the results of the RBF ANN implementations for validation and test data sets, respectively. The field RBF classifiers contained a Gaussian activation function within its hidden layer and a linear activation function within its output layer. Similar to the field MLP classifiers, the scaled conjugate gradient algorithm was utilized in training these classifiers. The field RBF classifiers also consisted of seventeen inputs and eighteen outputs.

Similar to the MLP ANN implementation, when comparing validation and test data set classification performance, it is evident that the 'Say account' and 'Say confirmation' field classifiers yielded larger validation data set accuracies. The 'Say amount' and 'Select beneficiary' classifiers performed better on the test data set. Similar behaviour has been noted in the field MLP classifier implementation results.

As illustrated in figure 4.5, the 'Say account' RBF classifier optimized using the real coded GA solution that employed tournament selection achieved the best results on test data. Therefore, this field RBF classifier, consisting of ninety four hidden nodes is best suited in resolving this classification problem. However, the binary coded GA solution that utilized normalized geometric ranking selection yielded the most accurate 'Say amount' RBF classifier. As a result, this field RBF classifier, which also employed ninety four hidden nodes, is most appropriate for this problem.

Similar to the 'Say account' RBF solutions, the 'Select beneficiary' and 'Say confirmation' RBF classifiers optimized using the real coded GA solution that utilized tournament selection achieved the best classification results. Therefore, the 'Select beneficiary' RBF classifier, which consisted of sixty eight hidden nodes, is best suited to this problem. Similarly, the 'Say confirmation' RBF classifier that utilized sixty hidden nodes is most appropriate for this problem.

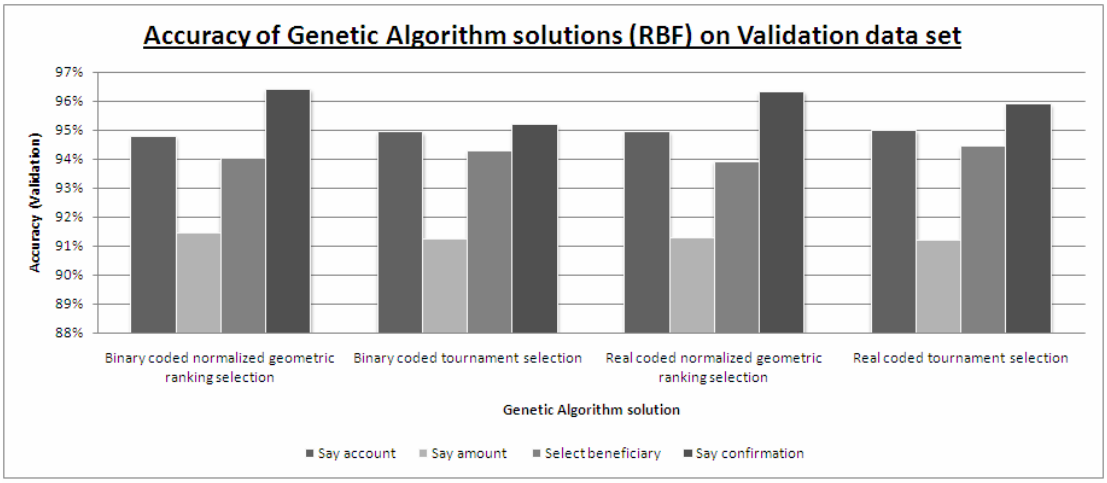

Figure 4.4: RBF ANN Field classifier results (Validation)

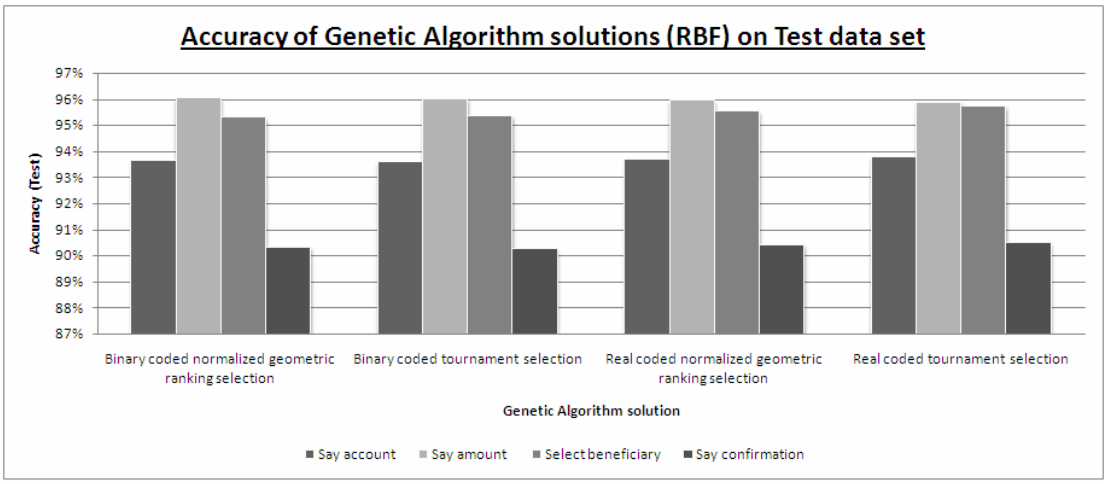

Figure 4.5: RBF ANN Field classifier results (Test)

It is evident from figure 4.6, that the field MLP classifiers implemented outperform the field RBF classifiers. As illustrated in figure 4.6, ANN field classifiers are most accurate in 'Select beneficiary' field classification. This is evident on both validation and test data sets. However, these ANN field classifiers are least accurate in 'Say confirmation' field classification.

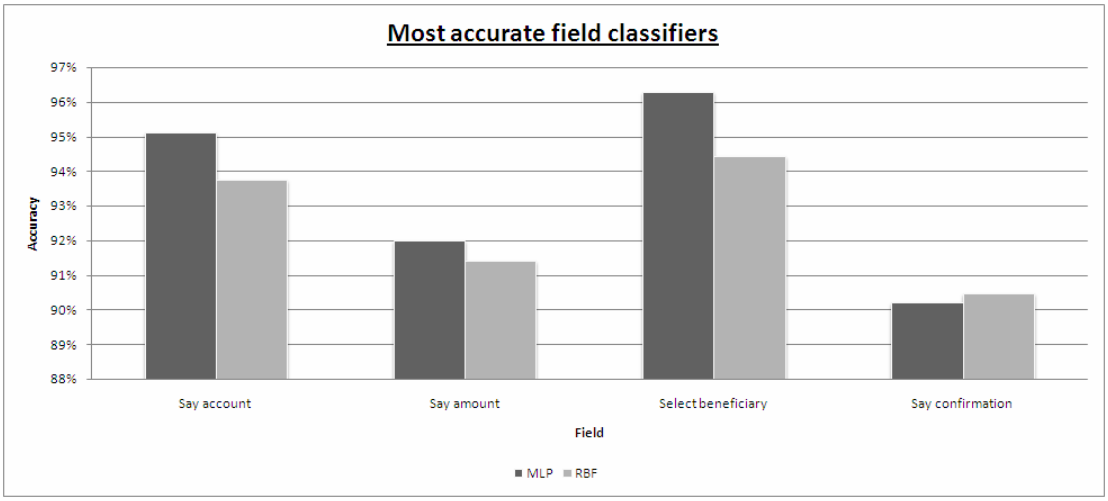

Figure 4.6: Most accurate ANN field classifiers

Figure 4.7 and figure 4.8 illustrate the call performance MLP and RBF classifier implementation results. The call performance MLP classifiers also employed the hyperbolic tangent hidden layer activation function and the logistic sigmoidal output layer activation function. However, the RBF classifiers contained a Gaussian activation function within its hidden layer and a linear activation function within its output layer. The call performance classifiers consisted of seventy six inputs and twenty one outputs. The inputs of the call performance classifier were the eighteen field interaction classes per field classifier and the field DTMF transfer status per field. These classifiers were also trained utilizing the scaled conjugate gradient algorithm.

It is evident from these figures that the MLP classifiers outperform the RBF classifiers. The real coded GA solution that utilized normalized geometric ranking selection produced the most optimal MLP call performance classifier. As a result, this classifier, which consisted of ten hidden nodes, is most appropriate for call performance classifications.

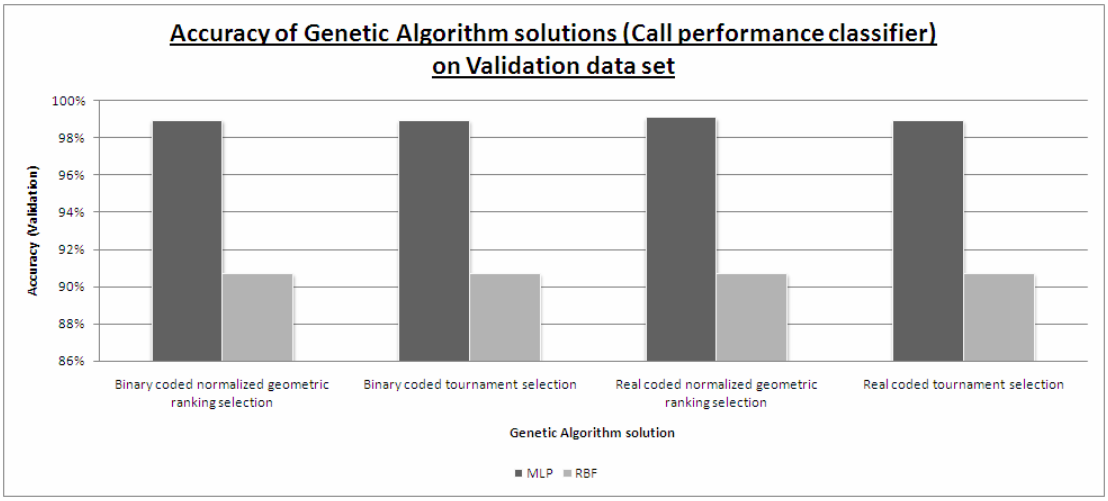

Figure 4.7: Call performance classifier results (Validation)

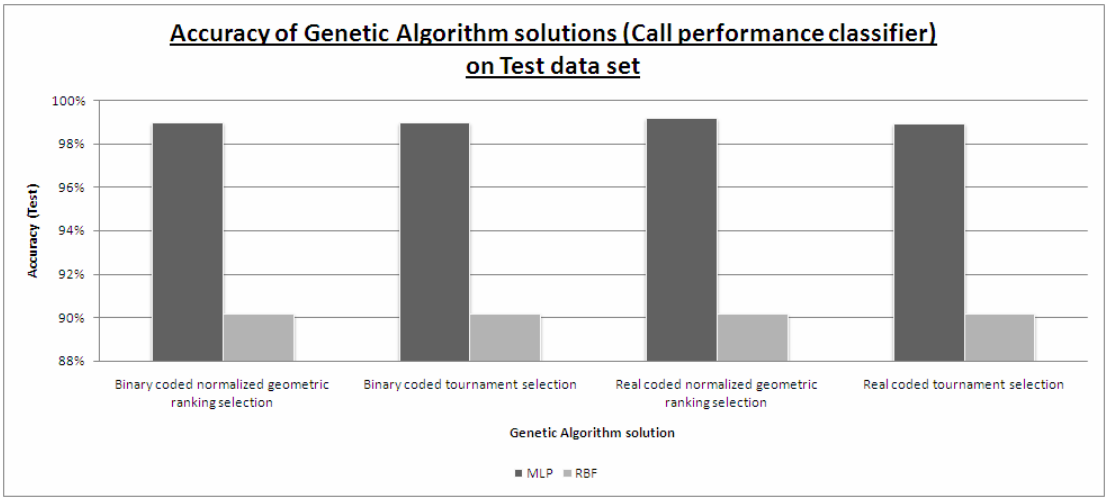

Figure 4.8: Call performance classifier results (Test)

Chapter 5

Fuzzy Inference System classifiers

5.1. Introduction

Fuzzy Inference Systems (FIS) were also considered in the development of both the field and call performance classification components. As mentioned in Chapter 2, FIS techniques are computational intelligent techniques that use fuzzy logic to formulate a mapping from a given input to an output (The Mathworks, 1995). These mappings provide a foundation that is utilized to make decisions.

This chapter examines the FIS classifiers developed during this study.

5.2. FIS classifiers

Clustering of numerical data establishes the basis of many classification and system modeling applications. As previously stated in Chapter 2, the primary purpose of a clustering technique is to locate natural groupings in a set of presented inputs with the objective of congregated similar inputs in the same class (The Mathworks, 1995). When employing clustering methods to compute fuzzy inference rules, the resultant rules are specifically tailored to the data. As a result, this is an advantage when compared to a FIS developed without clustering (The Mathworks, 1995).

The fuzzy c-means and subtractive data clustering techniques are the two most popular methods used (The Mathworks, 1995). The quality of the fuzzy c-means method depends strongly on the choice of the number of centers and the initial cluster positions (Chiu, 1994). This method is also known to possess the "curse of dimensionality" (Chiu, 1994). This implies that the number of rules increases exponentially as the input data increases in size. As a result, due to these problems, the FIS method utilized in the development of the field and call performance classifiers, used subtractive clustering to generate the required membership functions as well as set of fuzzy inference rules.

The development of the field and call performance FIS classifiers involved the optimization of the inference system architecture. This entailed the optimization of the cluster radius used within these components. Thereafter, depending on the accuracy achieved, in order to improve the accuracy values achieved, the classification thresholds were optimized. The implementation process concludes by comparing the various inference systems developed to determine the optimal field and call performance FIS classifiers. Figure 5.1 illustrates the process followed.

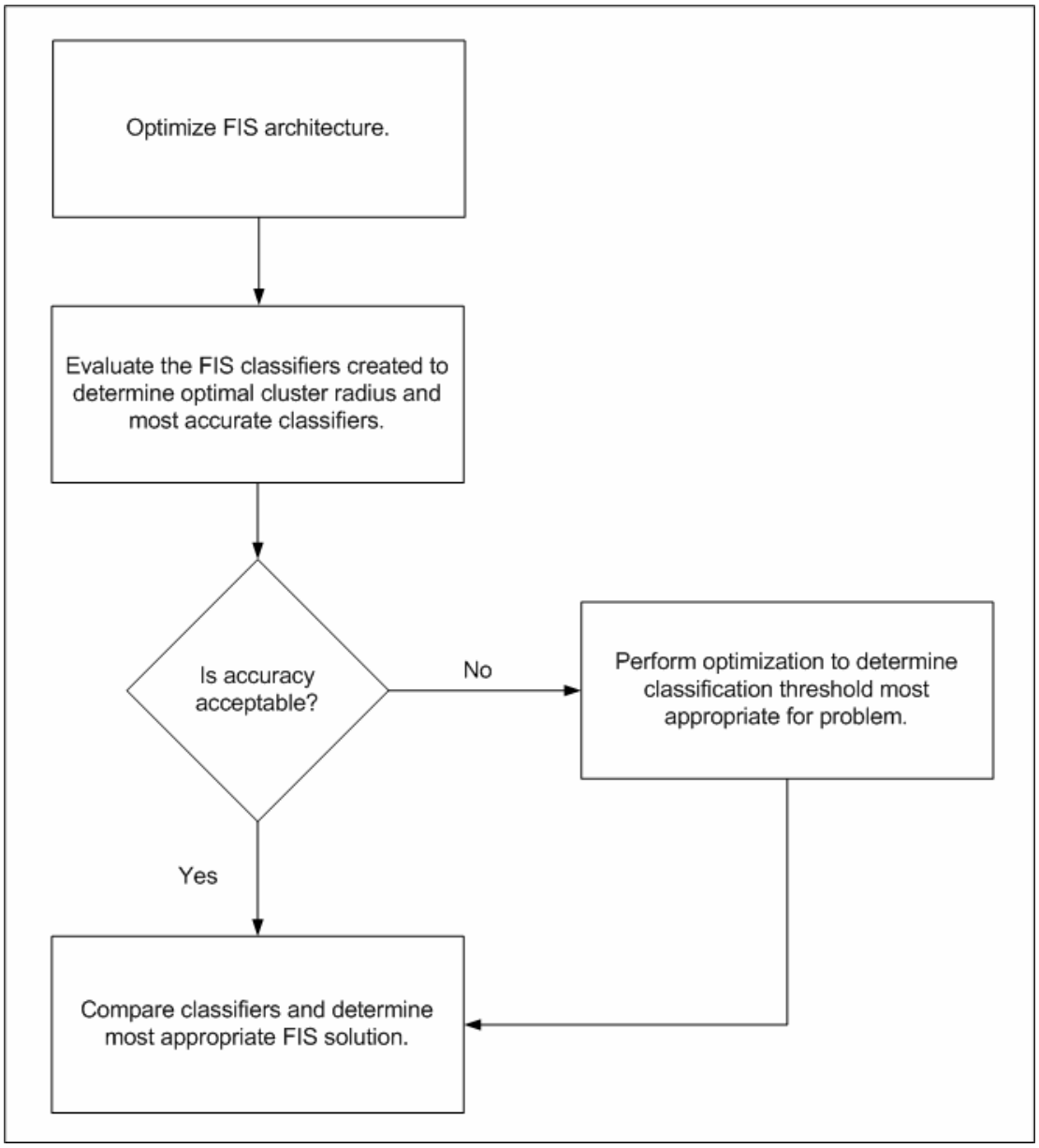

Figure 5.1: FIS system implementation process followed

5.2.1 Optimization of the Fuzzy Inference System architecture

The development of the field and call performance FIS classifiers involved the optimization of the cluster radius used within these components. The cluster radius has been optimized by minimizing an error function that mapped the radius to the

accuracy of the developed inference systems. This process used the validation data sets. The optimization process followed entailed the construction of various field and call performance inference systems with the cluster radius ranging from 0.01 to one. During the cluster radius optimization, classification threshold of 0.5 has been employed.

Table 5.1 illustrates the cluster radii that resulted in the most accurate field classifiers. As shown in the table, field 'Say account' FIS classifier proved to be the most accurate, yielding an accuracy of seventy eight percent on validation data set. However, the field 'Say amount' FIS classifier is the least accurate, producing an accuracy of 63.11% on validation data set.

Table 5.1 also illustrates the results of the call performance FIS classifier cluster radius optimization. It has been determined that cluster radii of 0.13 and 0.14 achieved the same validation and test data set accuracies. These cluster radii also achieve the best accuracies of ninety two percent and ninety one percent on validation and test data sets, respectively.

As a result of the accuracy values achieved, the classification thresholds were optimized.

Table 5.1: Results of FIS cluster radius optimization

Radius	Field	Accuracy	
		Validation	Test
Field classification component			
0.16	Say account	0.7800	0.8723
0.26	Say amount	0.6311	0.9566
0.40	Select beneficiary	0.7288	0.9339
0.78	Say confirmation	0.7074	0.8674
Call performance classification component			
0.13	-	0.9152	0.9108
0.14	-	0.9152	0.9108

5.2.2 Optimization of the classification threshold

In order to attempt to improve the accuracy of the FIS field and call performance classifiers, the classification threshold is optimized. Similar to the Artificial Neural Network (ANN) threshold optimization procedure followed previously in Chapter 4, the classification threshold is optimized by minimizing an error function that mapped the classification thresholds to the sum of the sensitivity and specificity of the developed FIS classifiers. The process is performed on the validation and test data sets.

This classification threshold optimization process entailed varying the classification threshold from 0.1 to 0.7 in iterations of 0.01. The procedure used the optimized field FIS classifier cluster radii illustrated in Table 5.1. However, the process employed to optimize the call performance FIS classifier classification threshold used a cluster radius of 0.14. For each of the threshold values the accuracy, the sensitivity and the

specificity of the FIS is calculated using, equation (4.2.1), (4.2.2) and (4.2.3) stated in Chapter 4, respectively.

Table 5.2 illustrates the threshold values that resulted in the largest sum of sensitivity and specificity value for the validation and test data sets. It is evident that by identifying suitable threshold values, the validation data set accuracy of the field classifiers has improved.

As illustrated in Table 5.2, the accuracy of the 'Say account' FIS classifier has improved by 2.68% on validation data and the 'Say amount' FIS classifier has become 19.54% more accurate on validation data. Similarly, the 'Select beneficiary' and 'Say confirmation' FIS classifiers are 5.55% and 8.77% more accurate on validation data.

After optimizing the classification threshold of the field FIS classifiers, the performance of these inference systems on test data decreased. However, as illustrated in Table 5.2, the optimized field FIS classifiers perform similarly on both validation and test data. As a result, it can be concluded that due to the optimization of classification threshold, the field inference systems have improved generalization capabilities.

As shown in Table 5.2, the original threshold value of 0.5 is an appropriate value for the call performance classification problem. As a result, the optimization of classification threshold did not improve the performance of this inference system.

Table 5.2: Results of FIS threshold optimization

Radius	Threshold	Field	Accuracy	Sensitivity	Specificity	Sensitivity + Specificity
Field classification component						
Validation						
0.16	0.16	Say account	0.8068	0.7200	0.9943	1.7143
0.26	0.15	Say amount	0.8265	0.7022	0.9860	1.6882
0.40	0.11	Select beneficiary	0.7843	0.6217	0.9900	1.6117
0.78	0.21	Say confirmation	0.7951	0.7138	0.9958	1.7096
Test						
0.16	0.16	Say account	0.8077	0.6615	0.9863	1.6478
0.26	0.15	Say amount	0.8254	0.6847	0.9951	1.6798
0.40	0.11	Select beneficiary	0.7782	0.6273	0.9908	1.6181
0.78	0.21	Say confirmation	0.7947	0.6576	0.9604	1.6180
Call performance classification component						
Validation						
0.14	0.10-0.24	-	0.6846	0.4853	0.9658	1.4511
	0.25-0.33		0.6849	0.4857	0.9658	1.4515
	0.34-0.49	-	0.6827	0.4838	0.9633	1.4471
	0.5	-	0.9152	0.9455	0.8858	1.8313
	0.51-0.60	-	0.9148	0.9452	0.8853	1.8305
	0.61-0.66	-	0.9147	0.9452	0.8852	1.8304
	0.67-0.70	-	0.9145	0.9458	0.8842	1.8300
Test						
0.14	0.10-0.24	-	0.6846	0.4826	0.9710	1.4536
	0.25-0.33		0.6847	0.4828	0.9710	1.4538
	0.34-0.49	-	0.6831	0.4814	0.9695	1.4509
	0.5	-	0.9108	0.9333	0.8888	1.8221
	0.51-0.60	-	0.9106	0.9331	0.8886	1.8217
	0.61-0.66	-	0.9106	0.9331	0.8886	1.8217
	0.67-0.70	-	0.9099	0.9333	0.8871	1.8204

5.2.3 Comparison of field and call performance Fuzzy Inference System classifiers

Figure 5.2 illustrate the results of the field FIS classifier implementations for validation and test data sets, respectively. The field FIS classifiers consisted of seventeen inputs and eighteen outputs. The field 'Say account', 'Say amount' and 'Say confirmation' FIS classifiers contained five rules. However, the field 'Select beneficiary' FIS classifier contained four rules.

When comparing the performance of the inference systems on validation and test data sets, it is evident that the optimized classifiers achieved similar accuracy values on both sets of data. Due to these results, it can be concluded that this classifier has good generalization capabilities.

The 'Say amount' FIS classifier is the most accurate inference system developed, producing an accuracy of 82.54% on test data. The least accurate is the 'Select beneficiary' FIS classifier, achieving an accuracy of 77.82% on test data.

It is also evident from figure 5.2, that the call performance FIS classifier developed achieved similar accuracy values on validation and test data. Due to this behaviour, this classifier has good generalization capabilities.

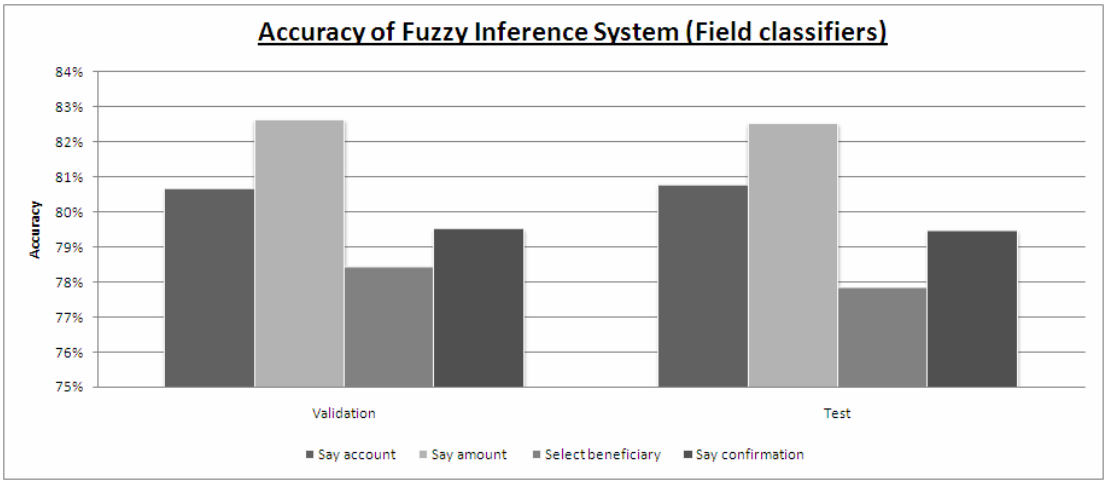

Figure 5.2: FIS Field classifier results

The call performance FIS classifier consisted of seventy six inputs and twenty one outputs. The classifier also contained three hundred and fifty eight rules.

The most accurate call performance FIS classifier yielded an accuracy of 91.52% and 91.08% on validation and test data sets, respectively.

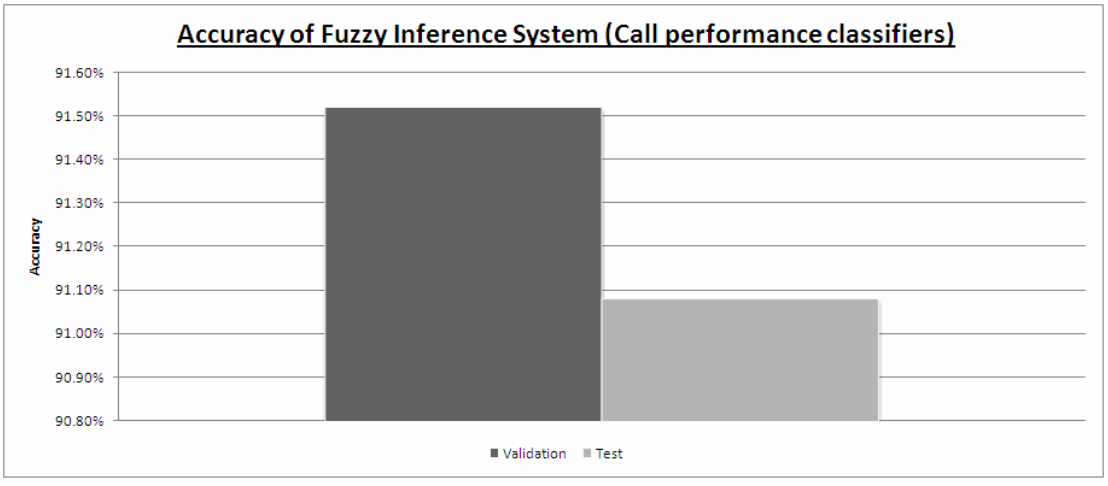

Figure 5.3: FIS Call performance classifier results

Chapter 6

Support Vector Machine classifiers

6.1. Introduction

During the development of the field and call performance components, Support Vector Machine (SVM) classifiers were also implemented. As previously stated, SVM is a reputable computational intelligent technique for resolving classification problems. SVMs have many advantages in resolving small sample size, nonlinear and high dimensional pattern recognition problems (Taylor and Cristianini, 2000).

This chapter describes the implementation methodology executed in developing SVM field and call performance classifiers.

6.2. SVM classifiers

SVM utilizes support vector (SV) kernel functions to map the data in the input space to a higher dimensional feature space where the problem can be processed in a linear form (Taylor and Cristianini, 2000). As a result the kernel function is a key technology of SVM. The type of kernel function will affect the classifier learning and generalization capabilities. Different kernel functions will construct different SVM classifiers. This research considers the linear, polynomial, Radial Basis Function (RBF) and sigmoid kernel functions.

When the number of training instances is less than the number of features within the data, the linear kernel function is most appropriate (Hsu et al., 2003). However, the RBF kernel function has the ability to accommodate non-linear relationships between input instances and output classes. The sigmoid kernel function behaves similar to the RBF kernel functions for certain parameters. The RBF kernel function has less hyperparameters than the polynomial kernel function (Hsu et al., 2003).

SVM implementation process involved creating field and call performance classifiers that employed the kernel functions mentioned above. The validation and test data set performance metrics of the resulting SVM classifiers were then compared to determine the kernel function most suitable for this application. As a result, this involved the selection of an appropriate kernel function that would result in classifiers with excellent generalization capabilities. Thereafter, depending on the accuracy achieved, in order to improve the accuracy values achieved, the classification thresholds were optimized. The implementation process concludes by comparing the various SVM classifiers developed to determine the optimal field and call performance SVM solution. Figure 6.1 illustrates the implementation process followed.

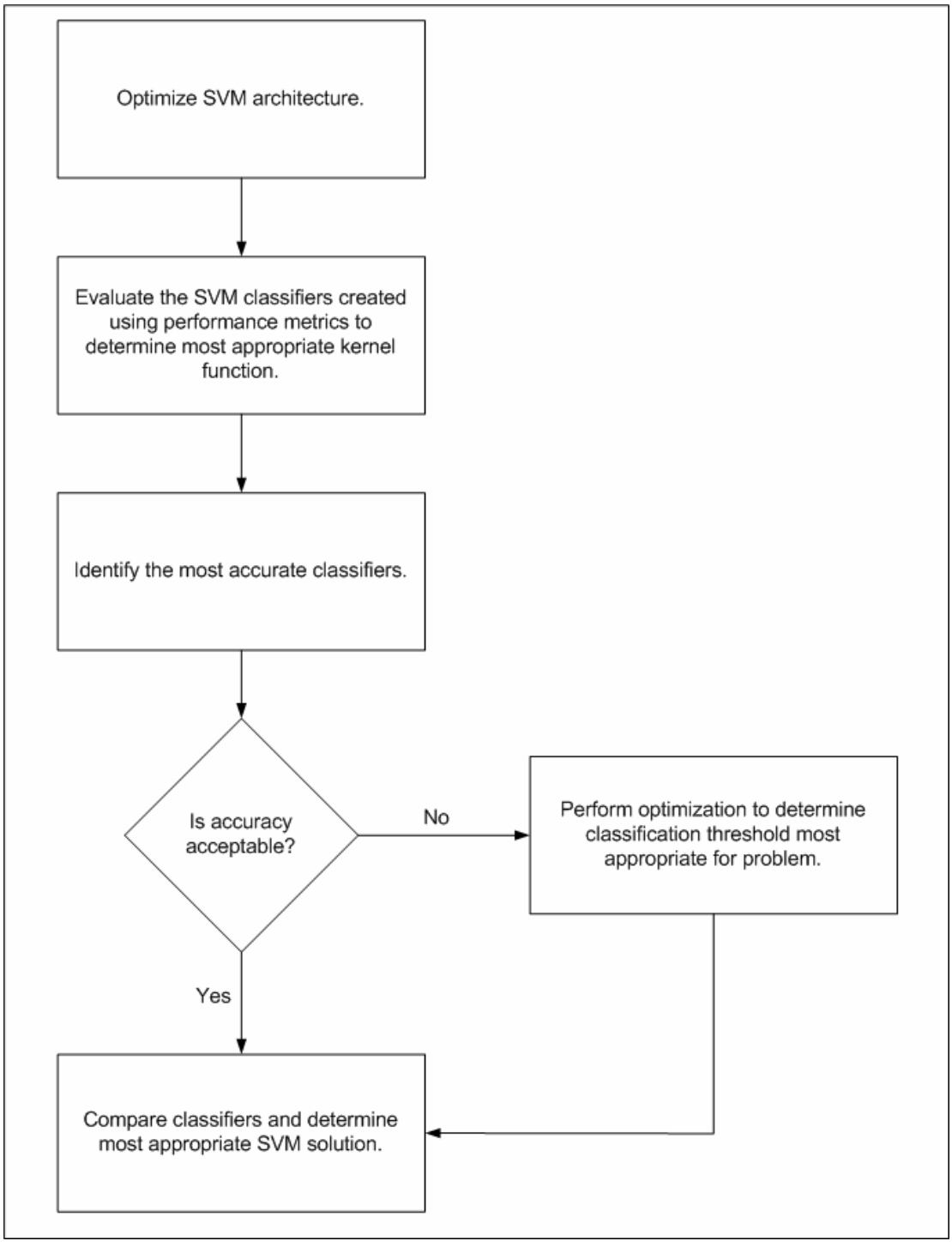

Figure 6.1: SVM implementation process

6.2.1 Optimization of the Support Vector Machine classifier architecture

Utilizing the training data set, the SVM classifiers that employed linear, polynomial, RBF and sigmoid kernel functions were trained. SVM classifiers with different kernel function parameters were created. Thereafter, the validation and test data sets were presented to the models. The accuracy of the developed classifiers was calculated for the validation and test data sets using equation (4.2.1) stated in Chapter 4. Grid-search method described in (Hsu et al., 2003) was conducted to identify the good parameters. SVMs that resulted in the largest accuracy value, when presented with the validation and test data sets, were analyzed.

It is evident from Table 6.1 that exceptional results were obtained that yielded field and call performance classifiers with good generalization capabilities.

It is evident that the polynomial kernel function resulted in the most accurate field 'Say account' and 'Say amount' SVM classifier. The 'Say account' SVM classifier that used the polynomial kernel function is, on average, 3.32% more accurate on test data than the other 'Say account' SVM classifiers developed. Similarly, when comparing 'Say amount' SVM classifiers developed, the 'Say amount' SVM classifier that utilized the polynomial kernel function is, on average, 1.19% more accurate on validation data.

Table 6.1: Results of SVM implementation

Kernel function	Field	C[1]	γ [2]	r^2	d^2	Accuracy	
						Validation	Test
Field classification component							
Linear	Say account	7	-	-	-	0.9708	0.8820
	Say amount	2.99	-	-	-	0.9086	0.9708
	Select beneficiary	150	-	-	-	0.9555	0.9651
	Say confirmation	3	-	-	-	0.9605	0.9050
Polynomial	**Say account**	**10**	**0.1461**	**0**	**4**	**0.9223**	**0.9191**
	Say amount	**150**	**0.07**	**0**	**3**	**0.9246**	**0.9428**
	Select beneficiary	25500	0.1	0	2	0.9630	0.9649
	Say confirmation	1000	0.07	0	3	0.9629	0.9068
Radial Basis Function	Say account	102500	0.0598	-	-	0.9800	0.8942
	Say amount	29000	0.091	-	-	0.9185	0.9780
	Select beneficiary	**10100**	**1**	**-**	**-**	**0.9641**	**0.9724**
	Say confirmation	**1900**	**1.05**	**-**	**-**	**0.9637**	**0.9095**
Sigmoid	Say account	101	0.0705	0	-	0.9694	0.8816
	Say amount	5.8	0.07	0	-	0.9110	0.9648
	Select beneficiary	149.5	0.06	0	-	0.9535	0.9635
	Say confirmation	140	0.05	0	-	0.9595	0.9041
Call performance classification component							
Linear	**-**	**1**	**-**	**-**	**-**	**0.9911**	**0.9923**
Polynomial	-	40	0.06	0	2	0.9830	0.9821
Radial Basis Function	-	1	0.07	-	-	0.9756	0.9736
Sigmoid	-	10	0.029	0	-	0.9886	0.9890

[1]C is the penalty parameter of the error term (Hsu et al., 2003).

[2]Section 2.5 contains details of these parameters.

The SVM classifiers that used the RBF kernel function produced the most accurate 'Select beneficiary' and 'Say confirmation' SVM classifiers. When comparing 'Select beneficiary' SVM classifiers developed, the classifier that utilized the RBF kernel function is, on average, 0.68% more accurate on validation data. Similarly, the 'Say confirmation' SVM classifier that used the RBF kernel function is, on average, 0.42% more accurate than the other 'Say confirmation' SVM classifiers.

The linear kernel function resulted in the most accurate call performance SVM classifier. When comparing call performance SVM classifiers developed, this classifier is, on average, 0.87% and 1.07% more accurate on validation data and test data, respectively.

The field and call performance SVM classifiers created employed a classification threshold value of 0.5. This threshold value resulted in above ninety percent accurate classifications. This has been demonstrated on the training, validation and test data sets. As a result, this threshold value of 0.5 proved to be adequate for the SVM implementations.

6.2.2 Comparison of field and call performance Support Vector Machine classifiers

Figures 6.2 and 6.3 illustrate the results of the field SVM classifiers developed. The field SVM classifiers consisted of seventeen inputs and eighteen outputs. When comparing the performance of the field SVM classifiers on validation and test data, it is evident that depending on the kernel function employed within the model, larger accuracy values are achieved on test data.

The 'Say account' and 'Say amount' SVM classifiers that utilized a polynomial kernel function were the most accurate models. These SVM classifiers were 91.91%

and 92.46% accurate, respectively. SVM classifiers that employed RBF kernel functions were most suited to 'Select beneficiary' and 'Say confirmation' classification. These SVM classifiers were 96.41% and 90.95% accurate, respectively.

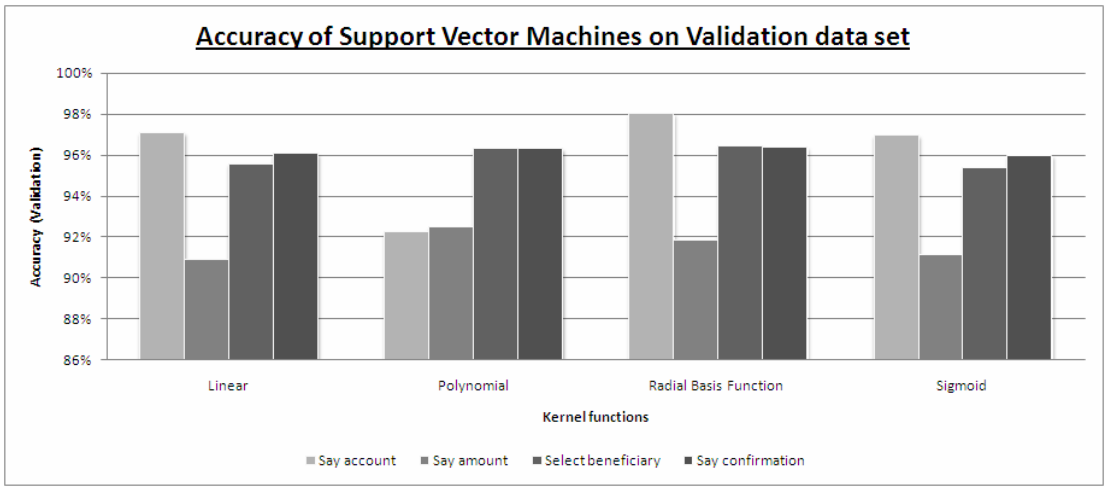

Figure 6.2: Support Vector Machines Field classifier results (Validation)

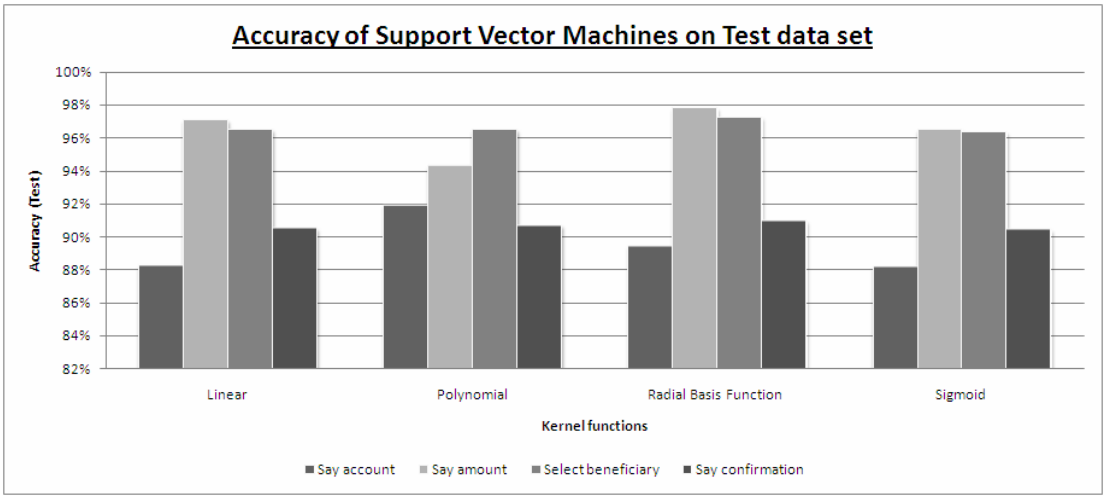

Figure 6.3: Support Vector Machines Field classifier results (Test)

Figure 6.4 illustrates the call performance SVM classifier implementation results. The call performance SVM classifier consisted of seventy six inputs and twenty one outputs. The SVM classifier achieved similar accuracy values on both validation and test data. As a result, good generalization capabilities have been exhibited.

The kernel functions used created call performance classifiers with accuracies larger than ninety five percent on validation and test data sets. It is evident that the linear kernel function resulted in the most accurate call performance classifier. The polynomial and sigmoid kernel function call performance classifiers were only approximately 0.90% and 0.25% less accurate on the test data set, respectively. However, the RBF kernel function call performance classifier is approximately 1.75% less accurate than the linear kernel function classifier.

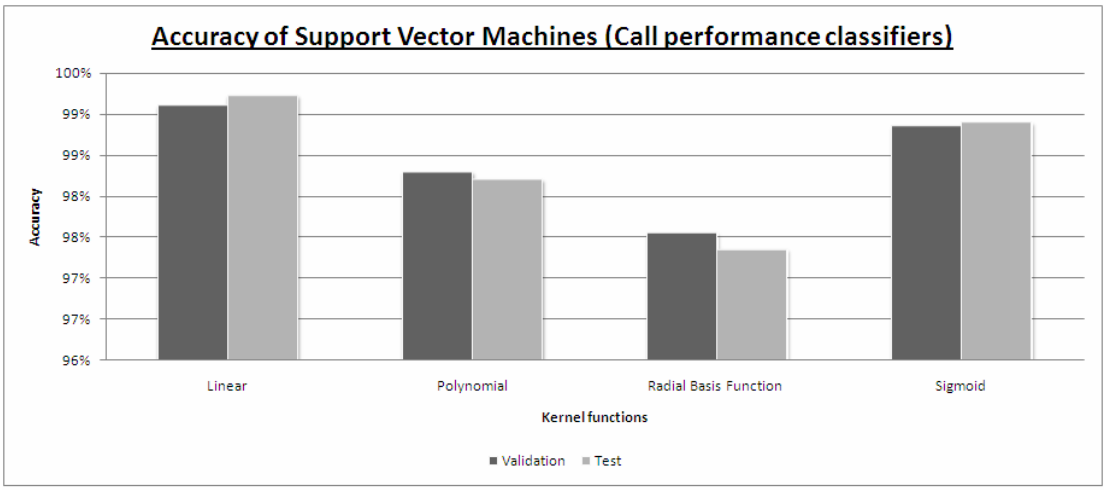

Figure 6.4: Call performance Support Vector Machines classifier results

Chapter 7

Comparison of the computational intelligent methods considered and the selection of the superior classifiers

7.1. Introduction

This research implemented popular computational intelligent techniques that yielded exceptional results in various application domains. As a result, differing approaches to resolve the application problems within this research were explored. Through the use of membership functions as well as inference engines equipped with a rule-based, Fuzzy Inference System (FIS) classifiers exploit the power of verbal descriptions. Multi Layer Perceptron (MLP) and Radial Basis Function (RBF) Artificial Neural Networks (ANN), store the desired form of knowledge within a massively interconnected layered feed-forward structure. Support Vector Machine (SVM), unlike the FIS technique, MLP and RBF ANN methods, exploits the Structural Risk Minimization (SRM) principle, thus enabling the minimization of the upper bound of a risk function and therefore achieving optimum classification functions.

This chapter compares the results yielded from the artificial intelligent methods considered.

7.2. Comparison of results achieved

The computational intelligent techniques considered produced highly accurate field and call performance classifiers. The MLP ANN, RBF ANN and the SVM implemented solutions achieved accuracy values larger than ninety percent on unseen validation as well as test data. However, the field FIS classifier yielded accuracies less than eighty five percent. Figure 7.1 and figure 7.2 illustrate the results of the field and call performance classifier implementations, respectively.

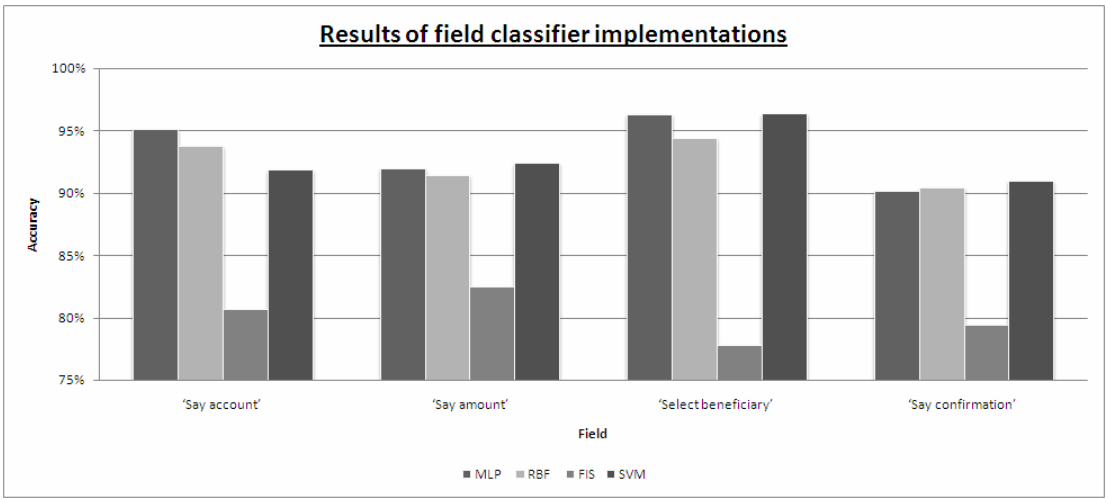

Figure 7.1: Results of field classifier implementations

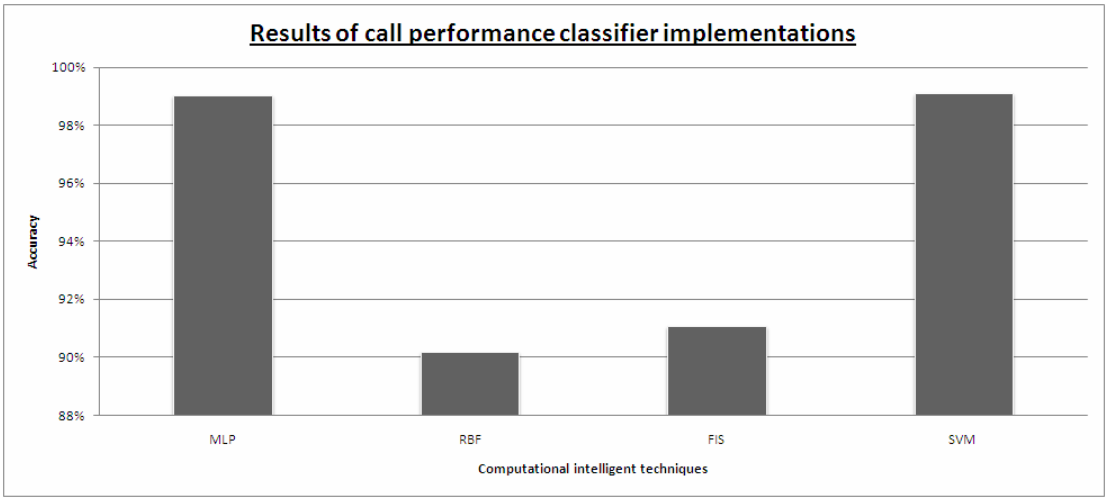

Figure 7.2: Results of call performance classifier implementations

Field 'Say amount', 'Select beneficiary' and 'Say confirmation' SVM classifiers achieved the best results on both validation and test data sets. However, field 'Say account' MLP classifiers produced the most accurate solutions, outperforming the field 'Say account' RBF and SVM classifiers on both validation and test data sets. The call performance SVM classifiers proved to be the most accurate, achieving an accuracy of 99.23% on test data.

7.3. Ensemble of classifiers

In order to improve the accuracy of the field and call performance classifiers, ensemble of networks has been considered. Ensembles of field 'Say account ','Say amount ','Select beneficiary' and 'Say confirmation' classifiers, consisting of the most accurate MLP ANN, RBF ANN as well as SVM networks, were developed. The call performance ensemble of classifiers considered consisted of the best MLP ANN, FIS and SVM models. Figure 7.3 and figure 7.4 details the ensemble of field and call performance classifiers used, respectively.

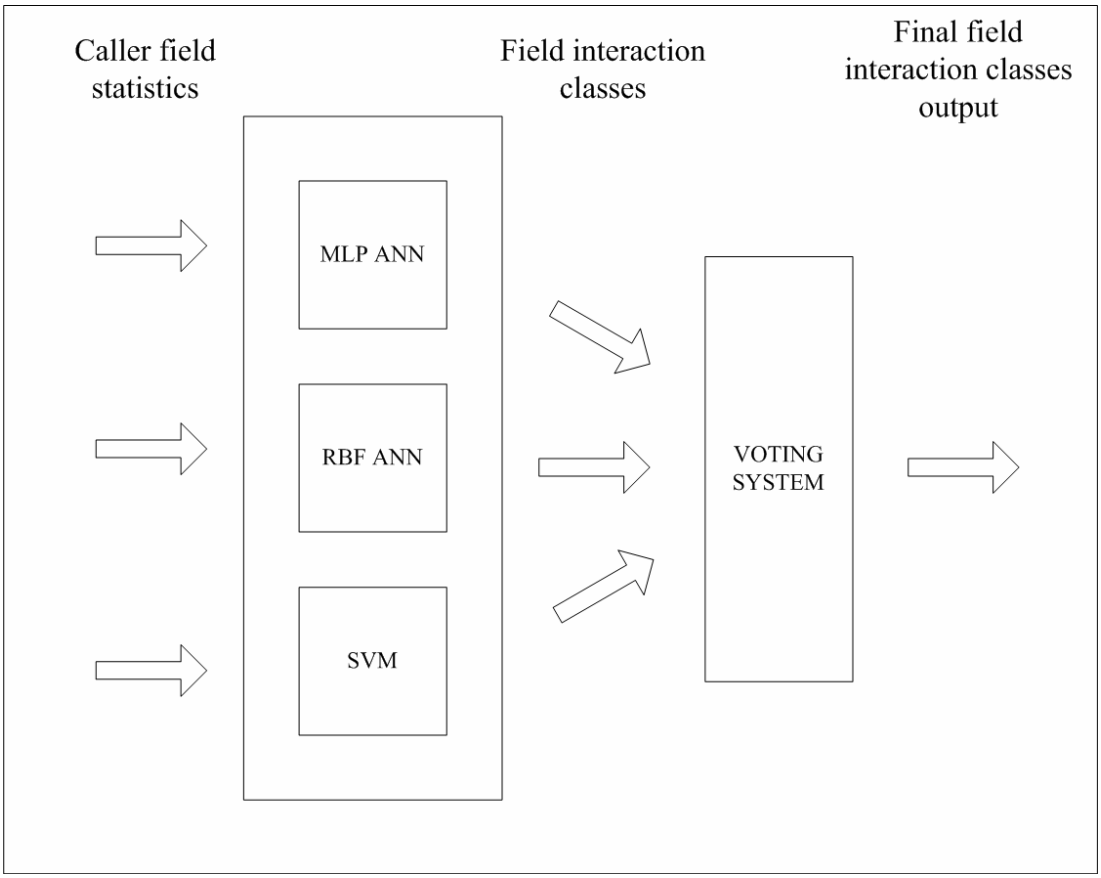

Figure 7.3: Ensemble of field classifiers

It has been stated that classifiers utilized simultaneously as committees or ensemble, will provide an average error that is lower than any individual classifier (Marwala, 2001), (Kittler, 1998), (Bishop, 1995), (Hansen and Salamon, 1990). A possible explanation for this is the fact that the set of inputs incorrectly categorized by the classifiers within the ensemble do not necessarily overlap. Therefore, a combination of networks as a classifier should outperform a single network classifier. Table 7.1 details the classifiers used within the ensembles.

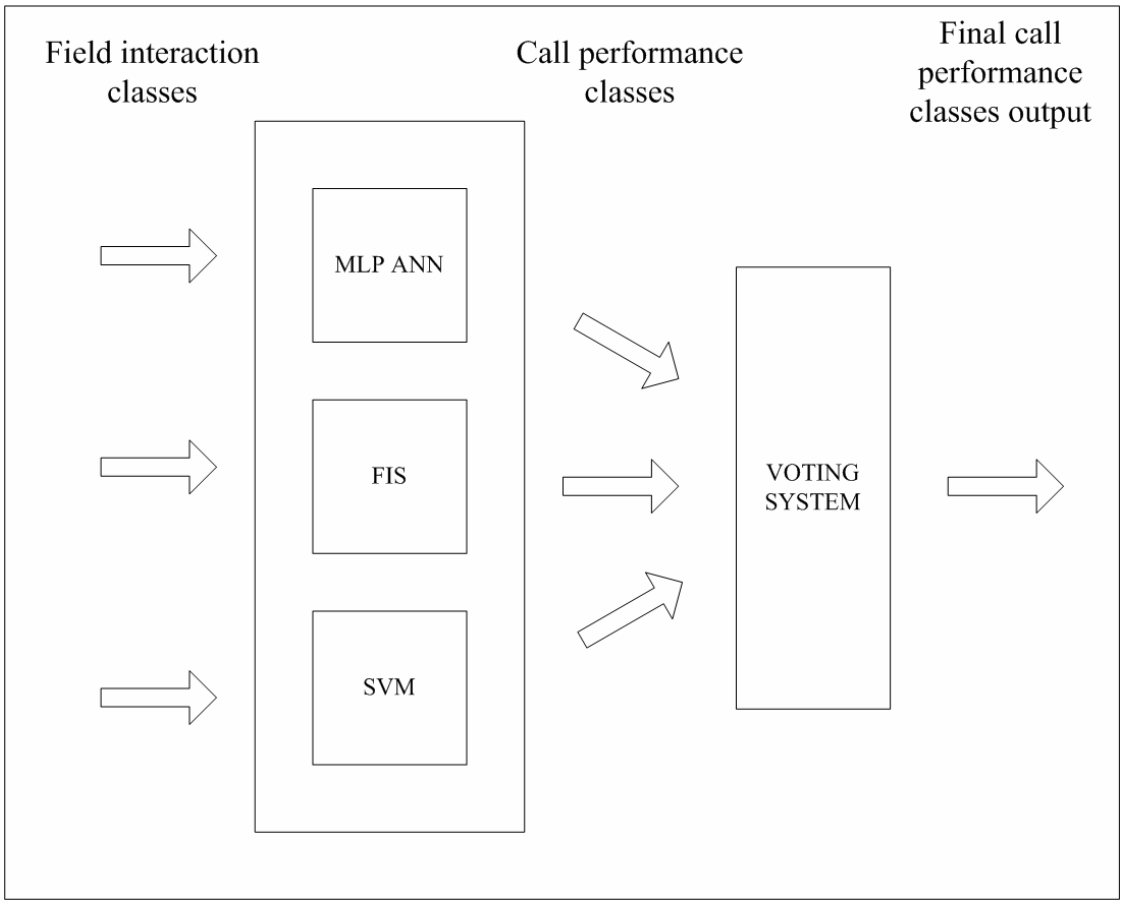

Figure 7.4: Ensemble of call performance classifiers

The outputs of classifiers were fed into a voting system. The voting system determined the final output of the ensemble. If the majority of the classifiers within the ensemble categorized an output into a certain class, the voting system would classify the output of the ensemble as the class. If all of the models within the ensemble classified an output into different classes, the voting system would classify the output of the ensemble as undecided.

Table 7.1: Classifiers used in Ensemble solution

Ensemble	Classifiers used	Accuracy	
		Validation	Test
'Say account' field Ensemble of classifiers	MLP ANN (8 hidden nodes)	0.9844	0.9513
	RBF ANN (94 hidden nodes)	0.9499	0.9376
	SVM (Polynomial)	0.9223	0.9191
'Say amount' field Ensemble of classifiers	MLP ANN (9 hidden nodes)	0.9199	0.9796
	RBF ANN (94 hidden nodes)	0.9143	0.9603
	SVM (Polynomial)	0.9246	0.9428
'Select beneficiary' field Ensemble of classifiers	MLP ANN (6 hidden nodes)	0.9628	0.9722
	RBF ANN (68 hidden nodes)	0.9443	0.9573
	SVM (RBF)	0.9641	0.9724
'Say confirmation' field Ensemble of classifiers	MLP ANN (50 hidden nodes)	0.9557	0.9021
	RBF ANN (60 hidden nodes)	0.9588	0.9048
	SVM (RBF)	0.9637	0.9095
Call performance Ensemble of classifiers	MLP ANN (10 hidden nodes)	0.9904	0.9918
	FIS (0.14 cluster radius)	0.9152	0.9108
	SVM (Linear)	0.9911	0.9923

As shown in Table 7.2, the ensemble of classifiers proved to be an accurate solution. These committees yielded large accuracy values on both validation and test data sets. In order to confirm the performance of the classifiers created, the sensitivity and specificity values were also compared.

Field 'Say account' MLP classifier is most accurate on validation and test data. When presented with validation data, the classifier achieved the largest sensitivity and specificity values. This classifier also yielded the largest sensitivity value on test data.

However, the field ‘Say account’ FIS classifier achieved the largest test data specificity value, one percent larger than the field MLP classifier. This indicates that the field FIS classifier has a larger negative classification rate on test data. The ensemble of ‘Say account’ classifiers is 0.74% less accurate than the MLP classifier. Therefore, the MLP classifier solution is the preferred classification solution for the ‘Say account’ field.

Ensemble of field ‘Say amount’ classifiers is the most accurate, achieving an accuracy of 92.61% on validation data. These classifiers also yielded the best sensitivity value on validation data. However, the field ‘Say amount’ MLP ANN achieved larger test data accuracy, only 0.67% more accurate then the ensemble of classifiers. This classifier also achieved the best test data sensitivity, only 0.47% larger then the ensemble of classifiers. The field ‘Say amount’ FIS classifiers achieved the best specificity values, but the lowest accuracy and specificity values. As a result, the ensemble of field classifiers as well as the MLP classifier solutions are preferred classification approaches for the ‘Say amount’ field.

The field ‘Select beneficiary’ SVM classifier is most accurate on both data sets. When presented with both the data sets, this classifier also yielded the largest sensitivity values. Similar to the field ‘Say amount’ FIS classifiers the field ‘Select beneficiary’ FIS classifier achieved the best specificity values, but the lowest accuracy and specificity values. It should be noted that the ensemble of field ‘Select beneficiary’ classifiers were only 0.19% and 0.01% less accurate on validation and test data sets. Therefore, the ensemble of classifiers and SVM classifier are the preferred classification solutions for the ‘Select beneficiary’ field.

Table 7.2: Performance metrics of best classifiers per method considered

	Method	Validation			Test		
		Accuracy	Sensitivity	Specificity	Accuracy	Sensitivity	Specificity
Field 'Say account' classifier	MLP ANN	**0.9844**	**0.9738**	**0.9951**	**0.9513**	**0.9269**	**0.9763**
	RBF ANN	0.9499	0.9202	0.9806	0.9376	0.9049	0.9715
	FIS	0.8068	0.7200	0.9943	0.8077	0.6615	0.9863
	SVM	0.9223	0.8942	0.9513	0.9191	0.8899	0.9493
	Ensemble	0.9679	0.9549	0.9812	0.9439	0.9205	0.9680
Field 'Say amount' classifier	MLP ANN	**0.9199**	**0.8803**	**0.9613**	**0.9796**	**0.9711**	**0.9883**
	RBF ANN	0.9143	0.8717	0.9590	0.9603	0.9404	0.9805
	FIS	0.8265	0.7022	0.9860	0.8254	0.6847	0.9951
	SVM	0.9246	0.8925	0.9579	0.9428	0.9178	0.9684
	Ensemble	**0.9261**	**0.8950**	**0.9583**	**0.9729**	**0.9664**	**0.9794**
Field 'Select beneficiary' Classifier	MLP ANN	0.9628	0.9394	0.9868	0.9722	0.9566	0.9880
	RBF ANN	0.9443	0.9132	0.9765	0.9573	0.9361	0.9789
	FIS	0.7843	0.6217	0.9900	0.7782	0.6273	0.9908
	SVM	**0.9641**	**0.9456**	**0.9831**	**0.9724**	**0.9608**	**0.9841**
	Ensemble	**0.9622**	**0.9418**	**0.9831**	**0.9723**	**0.9597**	**0.9850**
Field 'Say confirmation' Classifier	MLP ANN	0.9557	0.9206	0.9922	0.9021	0.8454	0.9625
	RBF ANN	0.9588	0.9428	0.9751	0.9048	0.8559	0.9565
	FIS	0.7951	0.7138	0.9958	0.7947	0.6576	0.9604
	SVM	**0.9637**	**0.9453**	**0.9824**	**0.9095**	**0.8597**	**0.9621**
	Ensemble	**0.9636**	**0.9450**	**0.9826**	**0.9097**	**0.8603**	**0.9619**
Call performance classifier	MLP ANN	0.9904	0.9845	0.9963	0.9918	0.9877	0.9960
	RBF ANN	0.9068	0.9784	0.8404	0.9019	0.9677	0.8406
	FIS	0.9152	0.9455	0.8858	0.9108	0.9333	0.8888
	SVM	0.9911	0.9857	0.9965	0.9923	0.9889	0.9957
	Ensemble	**0.9925**	**0.9902**	**0.9949**	**0.9929**	**0.9908**	**0.9949**

When presented with validation data, the field 'Say confirmation' SVM classifier yielded the best accuracy and sensitivity values. However, when presented with test data, the ensemble of 'Say confirmation' classifiers achieved the best accuracy and sensitivity values. Similar to the previous field classifiers findings, the field 'Say confirmation' FIS classifier yields the best specificity, but lowest accuracy and sensitivity values. This suggests that the FIS networks have a larger negative

classification rate, which results in lower accuracy. As a result, the ensemble of classifiers and the SVM classifier are the preferred classification solutions for the 'Say confirmation' field.

The ensemble of call performance classifiers are the most accurate on both validation and test data sets. When presented with validation and test data, this approach achieved the best sensitivity values. However, the call performance SVM classifier yielded the best specificity values on both the data sets. As a result, the SVM classifier has a larger negative classification rate. Therefore, the ensemble of call performance classifiers is the preferred classification solution for call performance.

Research that has been conducted in the utilization of computational intelligent methods to solve various classification and regression problems across different industries has yielded similar results. There are problems that are resolved most accurately by MLP ANN (Msiza et al., 2008), (Chen et al., 2007), (Shah and Salim, 2006), (Amendolia et al., 2003). Similarly, there are problems that are resolved most accurately by SVM classifiers (Ahmad et al., 2009), (Lau et al., 2009), (Nizam et al., 2009), (Radhika and Shashi, 2009), (Tolambiya and Kalra, 2008), (Angiulli et al., 2005), (Habtemariam et al., 2005), (Osowski et al., 2004), (Pal and Mather, 2004), (Bello and Dobeck, 2003), (Sadri et al., 2003), (Chan et al., 2002), (Tarantino et al., 2002), (Karras et al., 2001), (Ramaswamy et al., 2001), (Joachims, 1998). It can be concluded that the best solution for the problem is dependent on the application problem. The findings in this research as well as previous work conducted in computational intelligent techniques suggest that Empirical Risk Minimization (ERM) as well as SRM principles are appropriate techniques in resolving classification problems. However, in this research ERM principle is most appropriate in resolving the field 'Say account' classification problem. The SRM approach is suited in resolving the field 'Say amount', 'Select beneficiary', 'Say confirmation' classification problems as well as the call performance classification problem.

Chapter 8

Conclusions and further work

8.1. Conclusion

Contact centers experience operational challenges on a daily basis. The centers have to determine an optimal balance between reducing average call handling times and improve customer satisfaction rates. They have to reduce staffing expenses as well as decrease average call hold times. Interactive Voice Response (IVR) technology assist businesses in achieving this objective by providing contact centers with cost effective call management, automated call handling and informational messaging. IVR systems can also assist contact centers manage peak call volumes, enabling businesses to respond to a large number of customers. IVR technology provides customers twenty four hours, seven days a week services as well as privacy.

Speech technology such as Advanced Speech Recognition (ASR) can dramatically improve the IVR implementation efficiency as well as call experience. The benefits of speech recognition enabled self-service applications are shorter call durations, increased usage, natural conversation interactions and, therefore, increased customer

satisfaction. Due to the implementation of ASR IVR applications, businesses have reported an increase in IVR utilization from thirty five to seventy percent. Therefore, if implemented correctly, callers prefer speech-enabled IVR applications.

Today, customers interact with many businesses that provide excellent services. These interactions set customer expectations. As a result, in order to remain competitive in the current market, all customer-facing technologies should be scrutinized to ensure that these implementations support business service strategies and, therefore, deliver the expected, if possible, preferred customer experience.

Best practices have stated that businesses should evaluate the performance of the IVR applications as the business would increase Customer Service Agent (CSA) productivity, quality and call resolution. Businesses should have the capabilities to measure the IVR system performance from the perspective of the caller; the influence the automated application has on accomplishing the objective of the customer. Business analytics for IVR can provide contact centers with these essential capabilities.

This research entails the development of such a business analytics for IVR solution. The objectives of this research as stated in Chapter 1, section 1.4 were to:

1 Design a business analytics for IVR solution based on computational intelligence to assist contact centers in determining IVR application performance in relation to caller experience.

2 Develop a component within the proposed business analytics for IVR solution that assists contact centers to compute implementation detail performance indicators using computational intelligent techniques such as Artificial Neural Network (ANN), Fuzzy Inference System (FIS) and

Support Vector Machine (SVM). As a result, to provide IVR application developers with the capability to identify areas of improvement rapidly.

3 Utilizing ANN, FIS and SVM to develop a component within the proposed business analytics for IVR solution that provides contact centers with the capabilities of determining the complete call performance.

4 Create an ensemble of classifiers for each of the proposed components.

5 Compare the classifiers implemented to determine the superior approach for this application problem.

These objectives have been achieved. Chapter 2 examines the computational intelligence methodologies utilized in the development of the business analytics for IVR solution. Chapter 3 details the design of the call classification system developed for a pay beneficiary IVR application. Chapter 4, Chapter 5 and Chapter 6 detail the implementation of the call classification system components utilizing ANN as well as Genetic Algorithm (GA) solutions, FIS and SVM Artificial Intelligence (AI) techniques. Chapter 7 details the implementation of an ensemble of classifiers solution. The chapter also examines the results achieved by the methods considered to identify the superior techniques.

The proposed business analytics for IVR solution consisted of two components. These components were designed to provide the contact center as well as IVR application developers with business intelligence performance metric levels such as customer satisfaction, call containment, task completion, efficiency and usability.

The field classification component employed classifiers that were utilized to categorize caller behaviour at a field within the IVR automated applications into specific interaction classes. These interaction classes were designed to assist

developers to determine implementation detail performance indicators. Field transfer reason, hang-up reason, recognition level duration and difficulty attempt field performance interaction classes provide IVR application developers with the capabilities to measure efficiency as well as usability for each task or field within the self-service applications. The experienced caller and field performance interaction classes provide the capabilities to quantify caller satisfaction.

In order to facilitate rapid improvement of IVR applications with regards to caller experience, these field performance interaction classes can be used by developers to identify the modules that experienced unwanted caller behaviour. Utilizing the field interaction classes, IVR developers can determine the modules that experienced the largest number of transfers to Customer Service Agents (CSA) or the largest number of transfers to Dual Tone Multi Frequency (DTMF) due to difficulties. As a result, the field classification component provides developers with the capabilities to define specific performance indicators to assist in determining areas within the self-service application that requires improvement.

The objective of the call performance classification component is to evaluate the complete customer interaction with the self-service application in relation to caller behaviour. The component utilizes the field performance interaction classes to achieve this. Caller performance and experienced caller output classes provide the contact center with the capabilities to quantify customer satisfaction in relation to the complete call. Self-service level and caller disconnect transferred call performance output classes enable the ability to determine call containment, task completion as well as usability metrics. Speech-enabled level output class provides further implementation details for developers to utilize.

The computational intelligent techniques considered in this research yielded highly accurate field and call performance classifiers. Field ‘Say account’, ‘Say amount’,

‘Say confirmation’ and ‘Select beneficiary’ as well as call performance classifiers were developed.

The MLP ANN and RBF ANN implementation process entailed determining the optimal number of hidden nodes. Binary and real coded GA solutions that employed normalized geometric ranking as well as tournament selection functions were used to compute the optimal number of hidden nodes. Computational efficiency and quality of solution were used to evaluate the performance of the GA solutions. These GA solutions yielded accurate MLP and RBF ANN classifiers. However, the field and call performance MLP ANN classifiers resulted in more accurate solutions. In order to improve the call performance RBF ANN accuracies, the classification threshold has been optimized. This process did result in an improvement in accuracy of approximately eight percent.

The FIS classifiers were developed by initially identifying the cluster radius that resulted in the most accurate field and call performance classifications. Thereafter, the thresholds used to interpret these classifications were optimized. The accuracy of the field FIS classifiers did dramatically improve. However, the 0.5 classification threshold proved to be the optimal for the call performance FIS classifier.

The SVM development involved the identification of the SVM kernel function that yielded the most accurate results. The polynomial kernel function resulted in the most accurate field ‘Say account’ and ‘Say amount’ classifiers. However, the RBF kernel function provided to be most appropriate for the field ‘Select beneficiary’ and ‘Say confirmation’ classifiers. The linear kernel function is most appropriate for the classification of call performance.

Ensemble of field classifiers, consisting of the most accurate MLP ANN, RBF ANN and SVM classifiers has also been developed. Ensemble of call performance classifiers, consisting of the most accurate FIS, MLP ANN and SVM classifiers has

also be implemented. Accuracy, sensitivity and specificity performance metrics were computed and compared for the computational intelligent solutions.

MLP classifier solution is the preferred classification solution for the field 'Say account', achieving an accuracy of 95.13%. Due to the ensemble of field classifiers and the MLP classifier solutions yielding similar performance metrics, these are preferred classification approaches for the 'Say amount' field. The ensemble of field 'Say amount' classifiers and field 'Say amount' MLP classifier were 92.61% and 91.99% accurate, respectively. The ensemble of classifiers and SVM classifier are the preferred classification solutions for the 'Select beneficiary' and 'Say confirmation' field. These solutions were 96.22% and 96.44% accurate for field 'Select beneficiary' classification, respectively. However, in the classification of 'Say confirmation' caller field interaction, these solutions were 90.97% and 90.95% accurate, respectively. The ensemble of call performance classifiers is the preferred classification solution for call performance, yielding an accuracy of 99.25%.

The Empirical Risk Minimization (ERM) principle is employed by MLP ANN, RBF ANN and FIS solutions. This principle is concerned with reducing the training error. The Structural Risk Minimization (SRM) principle entails optimizing the training error as well as the classifier capacity. It can be concluded that both the Empirical Risk Minimization (ERM) and Structural Risk Minimization (SRM) principles are appropriate techniques in resolving the classification problems considered in this research. Specifically, the ERM principle is most appropriate for the field 'Say account' classification problem. The SRM principle is ideal in resolving the field 'Say amount', 'Select beneficiary', 'Say confirmation' classification problems as well as the call performance classification problem.

8.2. Further work and recommendations

This research entailed the development of a call classification system for a pay beneficiary self-service IVR application. The proposed system consisted of a field and call performance classification component. The field classification component comprised of field 'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' classifiers.

During this research ANN, specifically MLP and RBF feed-forward structured field as well as call performance classifiers were developed. FIS as well as SVM field and call performance classifiers were also implemented. Ensemble of classifier solutions were also developed for the field and call performance classification components.

In order to further this research, investigations into Artificial Intelligence (AI) techniques such as Gaussian Mixture Models (GMM) is proposed. These experiments would assist in identifying additional computational intelligent techniques that may prove to be appropriate in resolving these classification problems. Hybrid techniques that use a combination of AI techniques such as Fuzzy Support Vector Machines (FSVM) and hybrid MLP-SVM are also proposed. Furthermore, to improve the classification accuracy of the SVM classifiers, optimization techniques such as GA solutions should be utilized to optimize the SVM kernel parameters. It is anticipated that these investigations will result in improvements in the accuracy of field and call performance classifications.

In order to increase the capabilities of the call classification system, transcribed field utterances should be introduced to the field classification component. Field utterances contain recordings of actual caller responses to individual automated queries. When these utterances are transcribed, additional information that assists in capturing actual caller interaction such as coughing, sneezing and background noise is also included. Examples of background noise are people talking around the caller, music from a

radio that is in the vicinity of the caller and dogs barking within the surroundings of the caller.

An additional classifier should be implemented within this component to interpret the transcribed field utterances. The classifier would determine whether the response of the caller contained valid words that the IVR application accommodates. Furthermore, the classifier would also indicate, using the information captured in the transcription, whether or not the caller field interaction comprised of interference such as background noise that may adversely affect ASR. Figure 8.1 illustrates the field utterance classifier.

The transcribed field utterances would enable the system to determine the reason for the difficulties such as no matches and no inputs experienced within the call. The transcribed utterances would assist in determining whether the difficulties experienced within a call are due to the IVR application, the caller response or the caller environment. For example, if the transcriptions indicate that the caller sneezed after responding with a valid phrase, the reason for the difficulty experienced is due to caller sneezing and therefore due to the caller. However, if the caller did not sneeze and a difficulty such as a no match is experienced, the difficulty experienced is due to the IVR application. This would also improve the ability of the call classification systems to determine the reason for caller behaviour such as transfer to DTMF (Dual Tone Multi Frequency), transfer to Customer Service Agent (CSA) and caller disconnect. The outputs of the field utterance classifier would be utilized by the field classifier to compute the reason for difficulties.

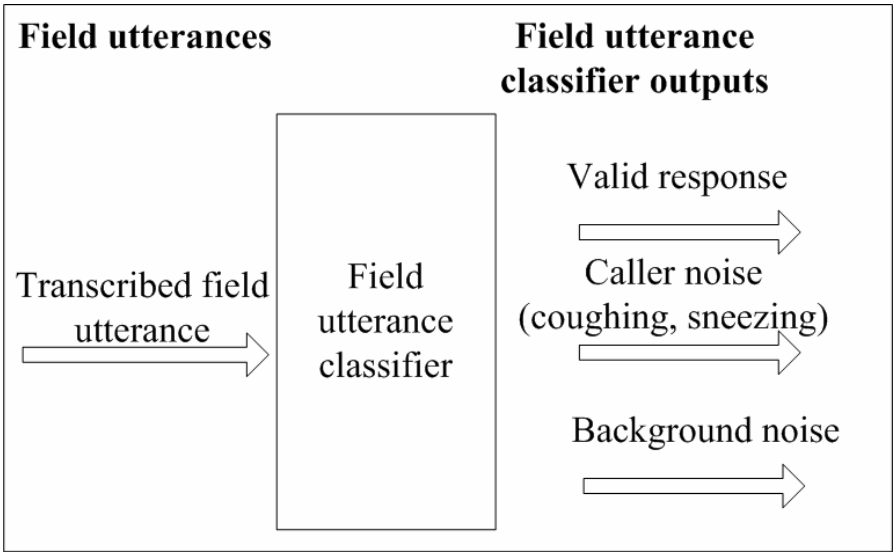

Figure 8.1: Field transcription classifier

In reality, there are callers that access several self-service applications within a single call to an IVR. As a result, it is proposed that the processes followed in this research to create the call classification system be employed to develop classification systems for self-service applications such as 'Inter-account transfers' and 'Balance enquiry'. Thereafter, to provide a complete multi-application call classification, a new component should be added to the system. This component should utilize the call performance output classes of each call classification system to provide complete multi-application call performance metrics. It is anticipated that this component will enable the contact centers to compare performance of various self-service processes within their IVR solutions. Figure 8.2 illustrates the proposed system.

It has been proposed in this research that the field interaction classes and call performance classes provide valuable insight into caller behaviour for IVR application developers as well as the contact center. In order to determine and identify further output classes, it is recommended that this system be deployed within an actual contact center for several months. Thereafter, an interview should be conducted involving the users of the system to determine future enhancements.

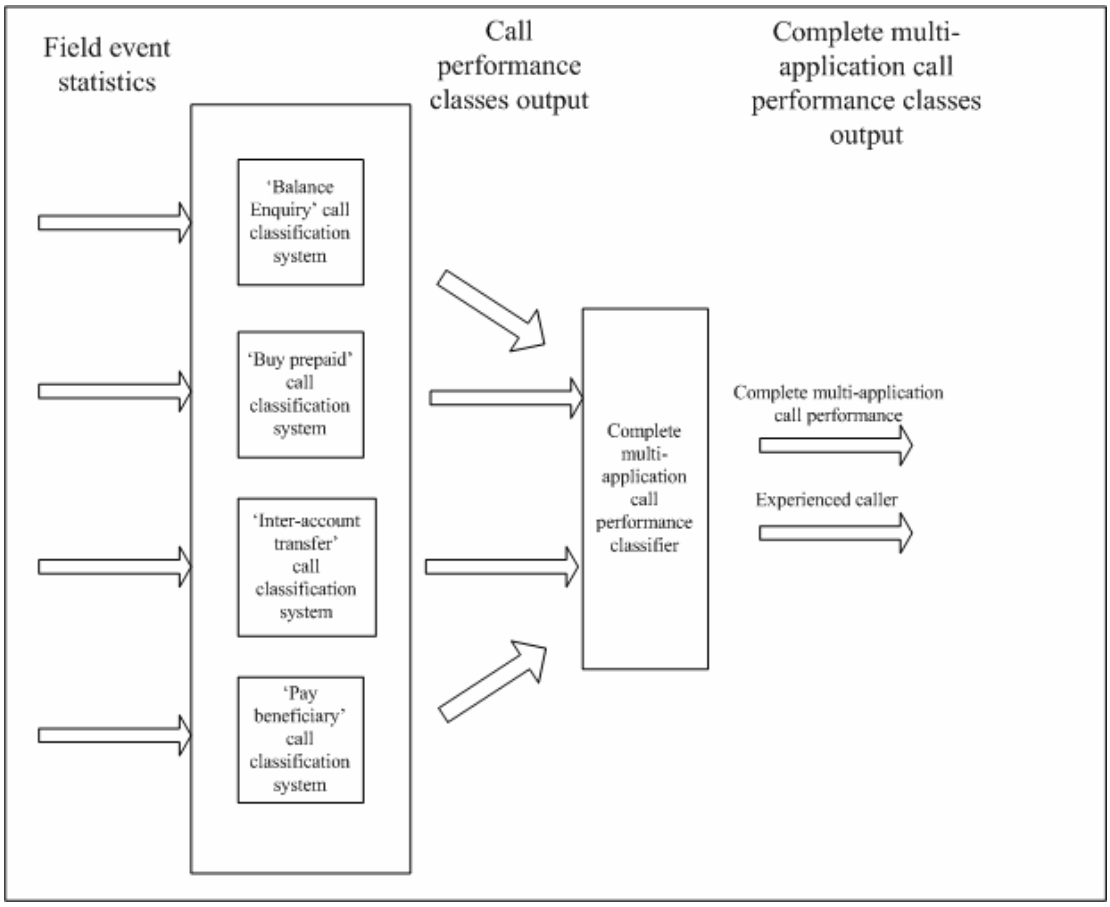

Figure 8.2: Proposed complete call classification system

It is also recommended that this system be developed for IVR applications deployed in various diverse industries such as entertainment, travel and logistics. Thereafter, as mentioned above, the call classification system should be deployed within these contact centers to determine the actual value the system provides to IVR application developers and contact center analysts.

References

Ahmad, R., Viard-Gaudin, C. and Khalid, M.: 2009, Lexicon-based Word Recognition Using Support Vector Machines and Hidden Markov Model, *10th International conference on Document Analysis and Recognition*, pp. 161-165.

Amendolia, S. R., Cossu, G., Ganadu, M. L., Golosio, B. Masala, G. L. and Mura, G. M.: 2003, A Comparative study of k-nearest neighbour, support vector machine and multi-layer perceptron for thalassemia screening, *Chemometrics and Intelligent Laboratory Systems*, vol. 69, no.1-2, pp. 13-20.

Angiulli, G., Barrile, V. and Cacciola, M: 2005, SAR Imagery Classification Using Multi-Class Support Vector Machines, *Journal of Electromagnetic Waves and Applications (JEWA), vol. 19, no. 14, pp. 1865-1872 and Proceedings of PIERS 2005*, pp. 218-222.

Ascent Group, Inc: 2008, *IVR improvement strategies 2008*, Ascent Group, Inc.

Baum, E.B. and Haussler, D.: 1989, What size net gives valid generalization?, *Neural Computation*, vol. 1, pp. 81-90.

Beale, R. and Jackson, T.: 1990: *Neural Computing-An introduction*, Taylor & Francis.

Bello, G. M. and Dobeck, G. J.: 2003, Comparison of support vector machines and multilayer perceptron networks in building mine classification models, *Proceedings of the SPIE*, vol. 5089, pp.77-94.

Bezdek, J. C.: 1981, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum.

Bishop, C.: 1995, *Neural Networks for Pattern Recognition*, Oxford University Press.

Booker, L. B., Goldberg, D. E. and Holland, J. H.: 1989, Classifier systems and genetic algorithm, *Artificial Intelligence*, vol. 40, no. 1-3, pp. 235–282.

Call Analytics, Inc: 2007, *VIA Improving the caller experience*, Call Analytics, Inc. http://www.call-analytics.com/media/VIA_Brochure.pdf

Chan, K., Lee, T. W., Sample, P. A., Goldbaum, M. H., Weinreb, R. N. and Sejnowski, T. J.: 2002, Comparison of Machine Learning and Traditional Classifiers in Glaucoma Diagnosis, *IEEE Transactions on Biomedical Engineering*, vol. 49, no. 9, pp. 963-974.

Chang, C-C and Lin, C-J: 2001, *LIBSVM : a library for support vector machines*, Department of Computer Science and Information Engineering, National Taiwan University. http://www.csie.ntu.edu.tw/~cjlin/libsvm

Chen, J., Xing, Y., Xi, G., Chen, J., Yi, J., Zhao, D., Wang, J.: 2007, A Comparison of Four Data Mining Models: Bayes, Neural Network, SVM and Decision Trees in Identifying Syndromes in Coronary Heart Disease, *Advances in Neural Networks – ISNN 2007*, pp. 1274-1279.

Chen, M-S.: 1991, *Analysis and Design of the Multi-layer Perceptron using Polynomial Basis Functions, Ph. D. Dissertation*, University of Texas.

Chiu, S.: 1994, Fuzzy Model Identification Based on Cluster Estimation, *Journal of Intelligent and Fuzzy System*, vol. 2, no. 3, pp. 267-278.

ClickFox, 2009, *The Leader in Customer Experience Analytics*, ClickFox. http://www.clickfox.com/solutions/business-solutions/

Cortes, C. and Vapnik, V. N.: 1995, Support Vector Networks, *Machine Learning*, vol. 20, no. 3, pp. 273-2

Datamonitor: 2008, *Hosted Speech and Outbound IVR Services (Strategic Focus)*, Datamonitor publishers.

DMG Consulting LLC: *2009, Hosted/Managed Service IVR Market Report*, DMG Consulting LLC.

Elmzabi, A., Bellafkih, M. and Ramdani, M.: 2007, An Adaptive Fuzzy Clustering Approach for the Network Management, *International Journal of Information Technology*, vol.3, no. 1, pp. 12-17.

Faulkner, A.: 2006, *How to right the wrongs of IVR*, callcentre helper. http://www.callcentrehelper.com/how-to-right-the-wrongs-of-ivr-98.htm

Freeman, E. A., Moisen, G. G.: 2008, A comparison of the performance of threshold criteria for binary classification in terms of predicted prevalence and kappa, *An International Journal on Ecological Modelling and systems ecology*, vol. 217, pp. 48-58.

Global Industry Analysts, *Speech Technology: 2008, A Global Strategic Business Report*, Electronics CA publications.

Goldberg, D. E.:1989, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley.

Gunn, S. R.: 1998, *Support Vector Machines for Classification and Regression, ISIS Technical Report*, University of Southampton.

GuruNet Corp: 2009, *Lofti Zadeh*, Answer.com, http://www.answers.com/topic/lotfi-zadeh

Habtemariam, E., Marwala, T. and Lagazio, M.:2005, Artificial intelligence for conflict management, *Proceedings of the IEEE International Joint Conference on Neural Networks*, vol. 4, pp. 2583-2588.

Hansen, L.K. and Salamon, P.: 1990, Neural Network Ensembles, *IEEE Transactions on Pattern Anal. Mach. Intell.*, vol.12, no.10, pp.993-1001.

Haykin, S: 1998, *Neural Networks: A Comprehensive Foundation*, Pearson Education India.

Holland, J.: 1975, *Adaptation in Natural and Artificial Systems*, University of Michigan Press.

Houck, C. R., Joines, J. A. and Kay, M. G.:1995, *A genetic algorithm for function optimization: a Matlab implementation, NCSU-IE Technical Report*, North Carolina State University.

Hsu, C.W., Chang, C.C and Lin, C. J.:2003, *A practical guide to support vector classification, Technical Report*, Department of Computer Science and Information Engineering, National Taiwan University. http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf

Hsu, C.W., and Lin, C.J.: 2002, A comparison of methods for multi-class support vector machines, *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415-425.

Huang, W. Y. and Lippmann, R.P.: 1988, Neural net and traditional classifier, Neural Information Processing Systems, *American Institute of Physics*, pp. 387-396.

Janikow, C. Z. and Michalewicz, Z.:1991, An Experimental Comparison of Binary and Floating Point Representations in Genetic Algorithms, *Proceedings of the 4th International Conference on Genetic Algorithms*, pp. 31–36.

Joachims, T.: 1998, Text categorization with Support Vector Machines: Learning with many relevant features, *Proceedings of the European Conference on Machine Learning*, pp. 137-142.

Jones, R.: 2008, *South African National 2007/2008 BPO & Call Centre Report*, C3 Africa Research and The MultiMedia Group.

Karras, D. A., Karkanis, S.A., Lakovidis, D. K., Maroulis, D. E. and Mertzios, B. G.: 2001, Support Vector Machines for Improved Defect Detection in Manufacturing using Novel Multidimensional Wavelet Feature Extraction Involving Vector Quantization and PCA Techniques, *NATO ASI-NIMIA*, pp. 139-143.

Kasabov, N. K.: 1996, *Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering*, MIT Press.

Keller, J. M., Gray, M. and Givens, J.: 1985, A fuzzy k-nearest neighbor algorithm, *IEEE Transaction on System, Man and Cybernetics*, vol.15, no. 4, pp. 580-585.

Kittler, J.: 1998, On Combining Classifiers, *IEEE Transactions on Pattern Anal. Mach. Intell.*, vol.20, no.3, pp. 226-238.

Lau, H.Y., Tong, K. Y., and Zhu, H.: 2009, Support vector machine for classification of walking conditions of persons after stroke with dropped foot, *Human Movement Science*, vol. 28, no. 4, pp. 504-514.

Marwala, T.: 2001, *Fault identification using neural networks and vibration data*, Cambridge University, Phd Thesis.

Michalewicz, Z.: 1996, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag.

Miller D.: 2007, *Analytics and Reporting for Phone-Based Self-Service*, Opus research network.

Msiza, I.S., Nelwamondo, F. V. and Marwala, T.: 2008, Water demand prediction using artificial neural networks and support vector regression, *Journal of Computers*, vol. 3, no. 11, pp. 1-8.

Nabney, I. T: 2002, *Netlab: Algorithms for Pattern Recognition*, Springer.

Neuro Solution Technologies: 2009, *What is a Neural Network?*, Neuro Solution Technologies. http://www.neurosolutions.com/products/ns/whatisNN.html.

Nichols, C.: 2006, *The Move from IVR to Speech – Why This is the Right Time to Make the Move to Speech Applications in Customer-Facing Operations*, Intervoice.

Nizam, M., Mohamed, A., Dabbagh, M. A. and Hussain A.: 2009, Using Support Vector Machine for Prediction Dynamic Voltage Collapse in an Actual Power System, *International Journal of Electrical Power and Energy Systems Engineering*, vol. 2, no. 2, pp. 81-86.

Nuance: 2002, *VoiceXML Reference:VXML properties*, Nuance Communications. http://community.voxeo.com/vxml/docs/nuance20/VXMLproperties.html

opinion-8: 2009, *IVR surveys for market research*, *white paper*, opinion-8. http://info.opinion-8.com/ivr-surveys-for-market-research.html

Orr, J.: 2006, *Neural Networks Slides*. http://www.willamette.edu/~gorr/classes/cs449/

Osowski, S., Siwek, K. and Markiewicz, T.: 2004, MLP and SVM Networks – a Comparative Study, *Proceedings of the 6th Nordic Signal Processing Symposium*, pp. 37-40.

Pal, M. and Mather, P. M.: 2004, Assessment of the Effectiveness of Support Vector Machines for Hyperspectral Data, *Future Generation Computer Systems*, vol. 20, no. 7, pp. 1215-1225.

Patel, P. B. and Marwala, T.: 2008, Interactive Voice Response field classifiers, *2008 IEEE International conference on Systems, Man and Cybernetics*, pp. 3425-3430.

Patel, P. B. and Marwala, T.: 2006, Neural Networks, Fuzzy Inference Systems and Adaptive-Neuro Fuzzy Inference Systems for Financial Decision Making, *International Conference on Neural Information Processing*, vol. 4234, no. 13. pp. 430-439.

pureXML: 2007, *Business Analytics for your IVR*, pureXML.. http://www.purexml.com/ivr_analytics.htm

Rababaah, H., Vrajitoru, D. and Wolfer, J.: 2005, Asphalt Pavement Crack Classification: A Comparison of GA, MLP, and SOM, *Proceeding of the Genetic and Evolutionary Computation Conference (GECCO'05 and SIGEVO 1)*, late breaking paper.

Radhika, Y. and Shashi, M.: 2009, Atmospheric Temperature Prediction using Support Vector Machines, *International Journal of Computer Theory and Engineering*, vol. 1, no.1 pp. 55-58.

Ramaswamy, S., Tamayo, P., Rifkin, R., Mukherjee, S., Yeang, C., Angelo, M., Ladd, C., Reich, M., Latulippe, E., Mesirov, J., Poggio, T., Gerald, W., Loda, M., Lander, E., and Golub, T.: 2001: Multi-Class Cancer Diagnosis Using Tumor Gene Expression Signatures, *Proceedings of National Academy of Sciences U.S.A*, vol. 98, no. 26, pp. 15149-15154.

Reyneri, L. M. and Sgarbi M.: 1997, Performance of Weighted Radial Basis Function Classifiers, *Proceedings of European Symposium on Artificial Neural Networks*, pp. 19-25.

Russo, M.: 1998, FuGeNeSys – A fuzzy genetic neural system for fuzzy modeling, *IEEE Transaction on Fuzzy Systems*, vol. 6, no. 3, pp. 373-388.

Sadri, J., Suen, C. Y., and Bui, T. D.: 2003, Application of Support Vector Machines for Recognition of Handwritten Arabic/Persian Digits, *Proceedings of the 2nd Conference on Machine Vision and Image Processing and Applications (MVIP2003)*, vol. 1, pp. 300-307.

Sasaki D., Morikawa, M., Obayashi, S. and Nakahashi, K.: 2001, Aerodynamic Shape Optimization of Supersonic Wings by Adaptive Range Multiobjective Genetic Algorithms, *Evolutionary Multi-Criterion Optimization: Lecture Notes in Computer Science*, vol. 1993, pp. 639-6521.

Selouani, S. A. and O'Shaughnessy, D.:2003, On the use of Evolutionary Algorithms to Improve the Robustness of Continuous Speech Recognition Systems in Adverse Conditions, *Special issue of the Journal on Applied Signal Processing*, no. 8, pp. 814-823.

Shah, J.Z. and Salim, N.B: 2006, Neural Networks and Support Vector Machines Based Bio-Activity Classification, *Proceedings of the 1st International conference on Natural Resources Engineering and Technology 2006*, pp. 484-49.

Smith, L.: 2003, *An Introduction to Neural Networks*, University of Sterling. http://www.cs.stir.ac.uk/~lss/NNIntro/InvSlides.html

Tarantino, Pasquariello, G., Ancona, N., Satalino, G., Blonda, P. and D'Addabbo, A.: 2002, Classifier Combination Techniques and Support Vector Machine: An Application to Character Recognition, *Proceedings of Conference on*

knowledge-based intelligent information engineering systems and applied technologies, KES 2002, pp.438-442.

Taylor, J. S and Cristianini, N.: 2000, *Support Vector Machines and other kernel-based learning methods*, Cambridge University Press.

The Mathworks, Inc.: 1995, *Fuzzy Logic Toolbox User's Guide Version 2*, The Mathworks, Inc.

Tolambiya and Kalra, P.K.: 2008, Contrast sensitive epsilon-SVR and its application in image compression, *2008 IEEE International conference on Systems, Man and Cybernetics*, pp. 359-264.

Uhrig, R. E. and Tsoukala, L. H.: 1997, *Fuzzy and Neural Approaches in Engineering*, John Wiley & Sons Ltd.

Vapnik, V. N.: 1995, *The Nature of Statistical Learning Theory*, Springer Verlag.

Vapnik, V. N.: 1998, *Statistical Learning Theory*, Wiley.

VoiceGenie Technologies Inc: 2009, *VoiceXML 2.0/VoiceXML 2.1 Reference*, VoiceGenie Technologies Inc. http://developer.voicegeniecom/voicexml2tagref.php?tag=st_field&display=standardtags

VoiceGenie Technologies Inc: 2005, *VoiceGenie 7 Tools User's Guide*, VoiceGenie Technologies Inc.

VoiceObjects: 2009, *Phone Application Server Solutions from VoiceObjects – VoiceObjects*, VoiceObjects. http://www.voiceobjects.com/en/products/index.html

VoiceXML Forum: 2009, *Welcome to the VoiceXML Forum*, VoiceXML Forum. http://www.voicexml.org/

Yen, J. and Wang, L.:1999, Constructing optimal fuzzy models using statistical information criteria, *Journal of Intelligent and Fuzzy Systems*, vol. 7, pp. 185-201

Zadrozny: 2002, Reducing multiclass to binary by coupling probability estimates, *Neural Information Processing Systems Foundation*, vol. 14, pp. 1041-1048.

Appendix: Papers published

Paper published in Advances in Computational Intelligence, Springer-Verlag

Caller Behaviour Classification: A Comparison of SVM and FIS Techniques

Pretesh B. Patel[1] and Tshilidzi Marwala[2]

[1] Researcher at the Faculty of Engineering and the Built Environment, University of Johannesburg, visiting Phd student from School of Electrical and Information Engineering, P.O. Box 524, Auckland Park, 2006, Johannesburg, South Africa
p.patel@ee.wits.ac.za
[2] Faculty of Engineering and the Built Environment, University of Johannesburg, P.O. Box 524, Auckland Park, 2006, Johannesburg, South Africa
tmarwala@uj.ac.za

Abstract. Accurate classification of caller interactions within Interactive Voice Response systems would assist corporations to determine caller behaviour within these telephony applications. This paper proposes a classification system with these capabilities. Fuzzy Inference Systems, Support Vector Machine and ensemble of field classifiers for a pay beneficiary application were developed. Accuracy, sensitivity and specificity performance metrics were computed and compared for these classification solutions. Ideally, a field classifier should have high sensitivity and high specificity. The Support Vector Machine field classifiers are the preferred models for the 'Say account', 'Select beneficiary' and 'Say confirmation' fields as these solutions yield the best performance results. However, the ensemble of field classifiers is the most accurate for the 'Say amount' field.

1 Introduction

Call centers experience operational challenges on a daily basis. The centers have to determine an optimal balance between reducing average call handling times and improve customer satisfaction rates. They have to reduce staffing expenses as well as decrease average call hold times.

Interactive Voice Response (IVR) systems can assist in resolving these challenges by providing a convenient, reliable as well as repeatable caller experience. An IVR system is an automated telephony system that interacts with callers, gathers relevant information and routes calls to the appropriate destinations [1]. The inputs to the IVR system can be voice, Dual Tone Multi-Frequency (DTMF) key-pad selection or a combination of the 2. IVR systems can provide appropriate responses in the form of voice, fax, callback, e-mails and other media [1]. An IVR system solution may consist of telephony equipment, software applications, databases and supporting infrastructure.

However, there are many IVR systems that have application design problems or are configured poorly that result in caller frustration when calling the system. An

W. Yu and E.N. Sanchez (Eds.): Advances in Computational Intell., AISC 61, pp. 199–208.
springerlink.com

example of a cause of caller frustration would be an IVR application that does not provide the caller sufficient time to respond to a prompt. Due to this, the system does not interpret the caller responses correctly. The caller experience is poor and this would probably result in a caller disconnect.

The aim of this research is to develop a field classification application, using computational intelligent methods, which could assist companies in quantifying caller behaviour within their IVR systems. It is anticipated that this application would be used in conjunction with other customer behaviour analysis techniques such as listening to recorded calls. As a result, this application should be used to confirm the system performance in relation to customer interaction.

IVR applications are developed in Voice Extensible Markup Language (VXML). VXML applications are voice-based dialog scripts that consist of form or dialog elements. The form or dialog elements are used to group input and output sections together. A field element is used to obtain and interpret user input information. As a result, the form or dialog elements contain field elements [2].

The classification system developed categorizes caller behaviour at a field within the IVR applications into specific interaction classes. As a result, these interaction classes can assist in determining trends of caller behaviour within the self service systems. For example, the field classification application can identify calls where the automated speech recognition at a particular field is low. Thereafter, analysts can listen to a sample of these calls and determine the reason for this. The field classification system can also identify the fields that resulted in the majority of the callers transferring to a Customer Service Agent (CSA) or caller disconnecting due to difficulties experienced.

In order to develop such an application, the classification of data must be accurate. This paper compares field classifiers that were developed utilizing Support Vector Machine (SVM) and Fuzzy Inference System (FIS) techniques. Ensembles of classifiers were also developed.

Support Vector Machines (SVMs) perform well for modeling challenging high-dimensional data. SVMs have been used successfully in text mining [3], image mining [4], bioinformatics [5] and information fusion [6]. SVM performance has been demonstrated to be superior to the performance of decision trees, neural networks and Bayesian techniques [3][5][6].

A fundamental method in data mining and pattern recognition is clustering of data. Fuzzy clustering involves the natural grouping of data in a large data set and provides a basis for constructing rule-based fuzzy model [7]. Fuzzy c-means, mountain clustering, subtractive clustering and entropy-based fuzzy clustering are among the fuzzy clustering algorithms used. In this paper we are interested in subtractive clustering.

The classification of data into various classes has been an important research area for many years. ANNs have been applied to pattern classification [8]. Research has also been conducted on fuzzy classification. This resulted in many algorithms, such as fuzzy K-nearest neighbour [9] and fuzzy c-means [10], being applied to classification problems. Fuzzy systems constructed using genetic algorithms have been utilized [11]. Fuzzy neural networks have also been employed in pattern classification applications [12].

SVMs have been applied to multi-category classification problems [13]. These classification tasks have also been implemented by combining multiple simpler specialized classifiers [14].

The sections to follow examine the caller behaviour classification system as well as its implementation methodology. The paper ends with the comparison of the various field classifiers developed and the selection of the superior networks.

2 The Developed System

As the developed system is to be used to identify trends of caller behaviour at a field within the IVR VXML applications, the system is trained based on data extracted from IVR log event files. These files are generated by the IVR platform as specific events occur during a call to the system. Events such as call begin, form enter, form select, automatic speech recognition events, transfer events and call end events are written to the logs [15].

Table 1 shows the inputs and outputs of the field classification system. These specific inputs have been selected to characterize the caller experience at a field within a VXML application. The outputs of the classifiers summarize the caller field behaviour through the use of interaction classes.

The confidence input illustrates the IVR speech recognition probability. The value is a percentage. The larger the percentage, the greater the probability the system interpreted the caller successfully.

A caller may answer a question the VXML application prompts with a response the application does not accommodate. These events are represented by the no match inputs. In general, most VXML applications accommodate 3 no match events per field. On a third no match event, the call is transferred to a DTMF field. If the caller fails to complete the DTMF field successfully on attempt 1, the call is transferred to a CSA. The same process is used for the third no input and maximum speech time out events. The no match field classifier inputs assist in identifying callers that misunderstood the VXML prompt as well as unique responses that the VXML application can use to improve field recognition coverage.

In response to a prompt, a caller may remain silent. These events are represented by the no input parameters. VXML applications normally accommodate 3 no input events on each field. These input parameters assist in identifying callers that were confused when prompted with the automated application question. As a result, the caller remained silent.

Callers may reply to VXML applications by talking beyond the allocated timeout period of the field. These events are represented by the maximum speech timeout input parameters of the field classifiers [16]. Maximum speech timeout input parameters are important as they assist in determining whether the timeout periods are adequate for callers to complete their responses.

Barge-in input parameters illustrate whether or not a caller interrupted the application while the automated question prompt played. Caller disconnects, transfer to DTMF, transfers to Customer Service Agents (CSAs) and System errors are represented by the hang-up, DTMF transfer, transfer to service agent and system error input parameters, respectively. These inputs can also assist in determining the level of

difficulty the caller experienced in the field. The duration input parameter illustrates the time the caller spent completing the field. Confirmation of transaction represents whether or not the caller verified the application recognition as being true.

Table 1. The inputs and outputs of the field classifier

Inputs	Outputs	Output interaction class
Confidence	Field performance	Good, acceptable, investigate, bad
No matches	Field transfer reason	Unknown, difficulty
No inputs	Field hang-up reason	Unknown, difficulty
Max speech timeouts	Field difficulty attempt	Attempt 1, attempt 2, attempt 3
Barge-ins	Field duration	High, medium, low
Hang-up	Field recognition level	High, medium, low
Transfer to Service Agent	Experienced caller	True, false
DTMF transfer		
Duration		
System error		
Confirmation of transaction		

The field performance output interaction class of the classifier will illustrate whether the caller behaviour is good, acceptable, investigate or bad. The field transfer reason and field hang-up reason interaction classes attempt to identify the motivation for the transfer to CSA or caller disconnect, respectively. Field difficulty attempt interaction class computes the number of difficulty events that occurred during the field interaction. The field duration as well as field recognition level classes illustrate 3 categories of performance, low, medium and high. As a result, these output parameters will assist in characterizing the caller experience at a VXML field.

Experienced caller output parameter categorizes whether or not the caller is a regular user of the application. In determining the number of experienced callers, the contact center can determine the usage of the application.

3 Selection and Preprocessing of Data

The data utilized in developing the classifiers is based on data extracted from IVR log event files. A business intelligence solution that involved Extract, Transform and Load (ETL) processes was created to extract and compute information such as recognition confidence values, duration values and call completion information. This information was stored within a database and was then manipulated utilizing specific rules to create the data sets. Rules such as if no hang-up, transfer to CSA, DTMF transfer, system error, no inputs, no matches or maximum speech timeouts occur, but the confidence level at the field is greater than 80%, the duration to complete the field is less than the average field duration and the field confirmation is true, the field performance interaction class would be computed as ‘good’, were followed.

No match, no input and maximum speech timeout information is presented to the field classifiers, using a binary notation. These inputs are presented by 3 digit binary words. For example, if a no match 1 and a no match 2 occur at a field, the binary notation will be '011'. A similar binary notation is employed for the no input and maximum speech timeout classifier inputs. The barge-in, hang-up, transfer to CSA, DTMF transfer, system error and confirmation of transaction input information were represented by bit binary words. A similar binary notation scheme has also been utilized to interpret the interaction classes outputted.

The confidence and duration input parameters of the classifiers were preconditioned by normalizing the data. Normalizing the data entails manipulating the data sets such that the values within the sets are between 0 and 1. Normalization is accomplished by acquiring the minimum and maximum values within the data sets. These values are then utilized to compute the normalized values.

The research conducted entailed the creation of 'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' field classifiers. Caller behaviour per field is unique. For example, at a 'Say confirmation' field the caller is required to say 'yes' or 'no'. However, the caller is requested to say the account name at the 'Say account' field. As a result, the duration to complete the VXML application field is much shorter at the confirmation field. Therefore, each classifier is trained with data relevant to the field.

In order to ensure that over-fitting and under-fitting were avoided, the data has been divided into 3 sets. The data is divided into training, validation and test sets. The training data set is used to train the algorithms to find the general classification groups within the data. The validation data set is used to assess the classifier and the test data is used to confirm the classification capability of the developed models.

4 Support Vector Machine Field Classifiers

SVM is a reputable computational intelligent technique for resolving classification problems. SVMs have many advantages in solving small sample size, nonlinear and high-dimensional pattern recognition problems [17]. SVM utilizes support vector (SV) kernel functions to map the data in the input space to a higher dimensional feature space where the problem can be processed in a linear form [17]. As a result the kernel function is a key technology of SVM. The type of kernel function will affect the learning ability and generalization ability. Different kernel functions will construct different SVM classifiers.

This research considers the linear, polynomial, Radial Basis Function (RBF) and sigmoid kernel functions. Linear kernel function is suitable to problems where the number of training instances is less than the number of features within the data [18]. RBF kernel function has the ability to accommodate non-linear relationships between input instances and output classes. The sigmoid kernel function behaves similar to the RBF kernel functions for certain parameters. The RBF kernel function has less hyperparameters than the polynomial kernel function [18]. For detailed information on these kernel functions refer to [19].

SVM implementation process involved creating field classifiers that employed the kernel functions mentioned above. The validation and test data set accuracies of the resulting SVM classifiers were then compared to determine the kernel function most suitable for this application. Since this is a classification implementation, a confusion

matrix is employed to identify the number of true and false classifications that are generated by the model developed. This is then utilized to compute the true accuracy of the classifiers, using the accuracy equation stated in [8].

Good results were obtained that yielded field classifiers with excellent generalization capabilities. Table 2 illustrates the results of the SVM implementation. It is evident that the polynomial kernel function resulted in the most accurate 'Say account' field classifier. The linear and RBF kernel function classifiers were only 2% less accurate on unseen 'Say account' data. Similarly, the sigmoid kernel function 'Say amount' classifiers were most accurate. The linear kernel function proved to be most appropriate for the 'Select beneficiary' and 'Say confirmation' field classifiers.

The SVM field classifiers created employed a classification threshold value of 0.5. This threshold value of 0.5 proved to be adequate for the implementations, resulting in 90% accurate classifications on the training, validation and test data sets.

Table 2. Results of support vector machine implementation

Kernel function	Field classifier	Accuracy (Validation)	Accuracy (Test)
Linear	'Say account'	0.9688	0.8814
	'Say amount'	0.9068	0.9691
	'Select beneficiary'	0.9447	0.9453
	'Say confirmation'	0.9630	0.9029
Polynomial	'Say account'	0.9047	0.9052
	'Say amount'	0.8521	0.8473
	'Select beneficiary'	0.8756	0.8418
	'Say confirmation'	0.9005	0.8583
RBF	'Say account'	0.9627	0.8832
	'Say amount'	0.9085	0.9401
	'Select beneficiary'	0.8950	0.8594
	'Say confirmation'	0.9257	0.8994
Sigmoid	'Say account'	0.9519	0.8776
	'Say amount'	0.9093	0.9263
	'Select beneficiary'	0.8939	0.8571
	'Say confirmation'	0.9064	0.8703

5 Fuzzy Inference System Field Classifiers

The FIS utilized in the development of the field classifiers, employed subtractive clustering to generate the required membership functions and set of fuzzy inference rules. The objective of clustering is to locate "natural classes" in a set of given inputs such that similar inputs are grouped together in the same class [20].

The cluster radius indicates the range of influence of a cluster. A small cluster radius results in small clusters in the data and, therefore, many fuzzy rules. Large cluster radii yield few large clusters in the data and, hence, fewer fuzzy rules [20].

The cluster radius has been optimized by minimizing an error function that mapped the radius to the accuracy of the developed inference systems. This process was performed on the validation data sets.

The optimization process followed entailed the construction of various inference systems with the cluster radius ranging from 0.01 to 1. During the cluster radius optimization, classification threshold of 0.5 has been employed. Once the optimal cluster radii have been identified, the classification threshold is optimized.

Table 3 illustrates the cluster radii that resulted in the most accurate field classifiers. FIS 'Say account' field classifier proved to the most accurate, yielding an accuracy of 78.00% on validation data set. However, the FIS 'Say amount' classifier is the least accurate, producing an accuracy of 63.11% on validation data set.

In order to improve the accuracy of the FIS field classifiers, the classification threshold is optimized. The classification threshold is optimized by minimizing an error function that mapped the classification thresholds to the accuracy of the developed classifiers. The process is performed on the validation and test data sets.

This optimization process involved varying the classification threshold from 0.1 to 0.5 in iterations of 0.01. During this process, the optimized cluster radii identified above has been used. For each of the threshold values the accuracy of the FIS is calculated using the accuracy equation mentioned in [8].

Table 3 illustrates the threshold values that resulted in the largest accuracy value for the validation and test data sets. It is evident that the validation data set accuracy of the field classifiers has improved. The FIS 'Say amount' classifier has become the most accurate with an accuracy of 82.54% on test data. The least accurate is the FIS 'Select beneficiary' classifier, yielding an accuracy of 77.82% on test data.

Table 3. Results of FIS optimization

Radius	Threshold	Field classifier	Accuracy (Validation)	Accuracy (Test)
Cluster radius optimization				
0.16	0.50	'Say account'	0.7800	0.8723
0.26	0.50	'Say amount'	0.6311	0.9566
0.40	0.50	'Select beneficiary'	0.7288	0.9339
0.78	0.50	'Say confirmation'	0.7074	0.8674
Classification threshold optimization				
0.16	0.16	'Say account'	0.8068	0.8077
0.26	0.15	'Say amount'	0.8265	0.8254
0.40	0.11	'Select beneficiary'	0.7843	0.7782
0.78	0.21	'Say confirmation'	0.7951	0.7947

6 Comparison of the Support Vector Machine and Fuzzy Inference System Field Classifiers

It is evident, from the investigations conducted, that the SVM 'Say account', SVM 'Say amount', SVM 'Select beneficiary' and SVM 'Say confirmation' field

classifiers are more accurate than the corresponding FIS classifiers by approximately 10%, 8%, 16% and 11%, respectively. The 3 most accurate SVM field classifiers were used in ensembles of classifiers. The outputs of classifiers that employed these kernel functions were fed into a voting system. The voting system determined the final output of the ensemble. If the majority of the classifiers within the ensemble categorized an output into a certain class, the voting system would generate an output as the class. If all of the models within the ensemble classified an output into different classes, the voting system would classify the output of the ensemble as undecided.

In relation to this application, the ensemble does not always result in a more accurate solution. The 'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' ensemble of field classifiers produced validation data accuracy values of 96.16%, 90.88%, 89.66% and 92.68%, respectively. These ensembles achieved test data accuracy values of 88.04%, 94.45%, 86.09% and 89.84%, respectively.

The most accurate SVM classifier outperforms the ensemble on validation data. This is true for all fields. The 'Say account' ensemble of field classifiers is 6% more accurate on validation data. However, the SVM 'Say account' field classifier produces similar accuracy values on both data sets. This demonstrates good generalization capabilities.

The 'Say amount' ensemble of classifiers is only 0.05% more accurate on validation data. The 'Say amount' field classifier is almost 2% less accurate on test data. As a result, the 'Say amount' ensemble of classifiers is appropriate for this application. The FIS classifiers are outperformed by the ensemble of SVM models.

In order to confirm the accuracy of the various classifiers developed, sensitivity and specificity were calculated using the equations in [21]. In this research, sensitivity is defined as the probability that the field classifier categorizes a set of caller behaviour inputs to the correct specific interaction classes. Specificity is defined as the probability that the classifier indicates that a set of caller behaviour inputs does not correctly belong to specific interaction classes. The former measure describes the effectiveness of the classifier at categorizing interaction classes correctly, while the latter characterizes the performance of the classifier at discarding the other interaction classes.

Table 4 illustrates these performance metrics for the most accurate FIS, SVM and ensemble of SVM field classifiers. Ideally, a field classifier should have high sensitivity as well as high specificity. This is evident in the SVM and ensemble of SVM field classifiers performance metrics.

The SVM 'Say account' field classifier yields similar performance results on both validation and test data. This is not the case for the ensemble. As a result, the SVM 'Say account' field classifier has good generalization capabilities and is the preferred model for this field. The ensemble is more accurate, has greater sensitivity and specificity values for the 'Say amount' test data. Therefore, the ensemble of SVM 'Say amount' field classifiers is the preferred model. The SVM 'Select beneficiary' and 'Say confirmation' field classifiers are the preferred models for these fields due to the best performance metrics achieved.

Table 4. Performance metrics of field classifiers

Field classifier	Method	Sensitivity	Specificity	Sensitivity	Specificity
		Validation		Test	
'Say account'	SVM (Polynomial)	0.8655	0.9457	0.8766	0.9347
	FIS	0.7200	0.9943	0.6615	0.9863
	Ensemble	0.9439	0.9798	0.8042	0.9638
'Say amount'	SVM(Sigmoid)	0.8681	0.9524	0.8948	0.9589
	FIS	0.7022	0.9860	0.6847	0.9951
	Ensemble	0.8623	0.9577	0.9180	0.9717
'Select beneficiary'	SVM (Linear)	0.9151	0.9754	0.9159	0.9756
	FIS	0.6129	0.9900	0.6097	0.9931
	Ensemble	0.8468	0.9493	0.7795	0.9509
'Say confirmation'	SVM (Linear)	0.9498	0.9764	0.8523	0.9564
	FIS	0.7138	0.9958	0.6576	0.9604
	Ensemble	0.8944	0.9603	0.8439	0.9564

7 Conclusion

This research entailed the development of a field classification application. 'Say account', 'Say amount', 'Say confirmation' and 'Select beneficiary' field classifiers were created using FIS, SVM and ensemble of classifiers.

The implementation process entailed identifying the SVM kernel function that yielded the most accurate results. The polynomial and sigmoid kernel function resulted in the most accurate 'Say account' and 'Say amount' field classifiers, respectively. However, the linear kernel function provided to be most appropriate for the 'Select beneficiary' and 'Say confirmation' field classifiers.

The FIS classifiers were developed by initially identifying the cluster radius that resulted in the most accurate model. Thereafter, the thresholds used to interpret the classification were optimized. The accuracy of the FIS classifiers did improve, but the SVM approach has been found to be more accurate.

Ensemble of field classifiers, consisting of the 3 most accurate SVM classifiers, has also been developed. Performance metrics were computed and compared for the computational intelligent solutions. The SVM and ensemble of field classifiers achieved high accuracy, sensitivity and specificity. The SVM field classifiers are the preferred models for the 'Say account', 'Select beneficiary' and 'Say confirmation' fields as they yield the best performance results. It has also been determined that the ensemble of field classifiers is the most accurate for the 'Say amount' field.

References

[1] Nichols, C.: The Move from IVR to Speech – Why This is the Right Time to Make the Move to Speech Applications in Customer-Facing Operations, Intervoice (2006)

[2] VoiceXML 2.0/VoiceXML 2.1 Reference, `http://developer..voicegeniecom/voicexml2tagref.php?tag=st&display=standardtags` (last accessed: 8 April 2009)

[3] Joachims, T.: Text categorization with Support Vector Machines: Learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998)
[4] Tolambiya, A., Kalra, P.K.: Contrast sensitive epsilon-SVR and its application in image compression. In: 2008 IEEE International conference on Systems, Man and Cybernetics (SMC 2008), pp. 359–364 (2008)
[5] Ramaswamy, S., Tamayo, P., Rifkin, R., Mukherjee, S., Yeang, C., Angelo, M., Ladd, C., Reich, M., Latulippe, E., Mesirov, J., Poggio, T., Gerald, W., Loda, M., Lander, E., Golub, T.: Multi-Class Cancer Diagnosis Using Tumor Gene Expression Signatures. Proc. National Academy of Sciences U.S.A. 98(26), 15149–15154 (2001)
[6] Pal, M., Mather, P.M.: Assessment of the Effectiveness of Support Vector Machines for Hyperspectral Data. Future Generation Computer Systems 20(7), 1215–1225 (2004)
[7] Elmzabi, A., Bellafkih, M., Ramdani, M.: An Adaptive Fuzzy Clustering Approach for the Network Management. International Journal of Information Technology 3(1), 12–17 (2007)
[8] Patel, P.B., Marwala, T.: Interactive Voice Response field classifiers. In: 2008 IEEE Interna-tional conference on Systems, Man and Cybernetics (SMC 2008), pp. 3425–3430 (2008)
[9] Keller, J.M., Gray, M., Givens, J.: A fuzzy k-nearest neighbor algorithm. IEEE Trans-action on System, Man and Cybernetics 15(4), 580–585 (1985)
[10] Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum (1981)
[11] Russo, M.: FuGeNeSys – A fuzzy genetic neural system for fuzzy modeling. IEEE Transaction on Fuzzy Systems 6(3), 373–388 (1998)
[12] Patel, P.B., Marwala, T.: Neural Networks, Fuzzy Inference Systems and Adaptive-Neuro Fuzzy Inference Systems for Financial Decision Making. In: International Conference on Neural Information Processing, vol. 4234(13), pp. 430–439 (2006)
[13] Hsu, C.-W., Lin, C.-J.: A comparison of methods for multi-class support vector machines. IEEE Transactions on Neural Neworks 13(2), 415–425 (2002)
[14] Zadrozny, B.: Reducing multiclass to binary by coupling probability estimates. In: Neural Information Processing Systems Foundation, vol. 14, pp. 1041–1048 (2002)
[15] VoiceGenie Technologies Inc., VoiceGenie 7 Tools User's Guide, VoiceGenie Technologies Inc. (2005)
[16] VoiceXML properties, http://community.voxeo.com/vxml/docs/nuance20/VXMLproperties.html (last accessed: April 8, 2009)
[17] Taylor, J.S., Cristianini, N.: Support Vector Machines and other kernel-based learning methods. Cambridge University Press, Cambridge (2000)
[18] Hsu, C.W., Chang, C.C., Lin, C.J.: A practical guide to support vector classification, Taipei, Tech. Rep. (2003), http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf
[19] LIBSVM - A library for Support Vector Machines, http://www.csie.ntu.edu.tw/~cjlin/libsvm (last accessed: April 8, 2009)
[20] Chiu, S.: Fuzzy Model Identification Based on Cluster Estimation. Journal of Intelligent and Fuzzy Systems 2(3), 267–278 (1994)
[21] Salzberg, S.L., Searls, D.B., Kasif, S.: Computational methods in molecular biology. Elsevier, Amsterdam (1998)

Paper published in Advances in Neuro-Information Processing: Lecture Notes in Computer Science

Caller Interaction Classification: A Comparison of Real and Binary Coded GA-MLP Techniques

Pretesh B. Patel and Tshilidzi Marwala

School of Electrical and Information Engineering, University of the Witwatersrand, Private Bag 3, 2050, Johannesburg, South Africa
p.patel@ee.wits.ac.za, t.marwala@ee.wits.ac.za
http://dept.ee.wits.ac.za/~marwala/

Abstract. This paper employs pattern classification methods for assisting contact centers in determining caller interaction at a 'Say account' field within an Interactive Voice Response application. Binary and real coded genetic algorithms (GAs) that employed normalized geometric ranking as well as tournament selection functions were utilized to optimize the Multi-Layer Perceptron neural network architecture. The binary coded genetic algorithm (GA) that used tournament selection function yielded the most optimal solution. However, this algorithm was not the most computationally efficient. This algorithm demonstrated acceptable repeatability abilities. The binary coded GA that used normalized geometric selection function yielded poor repeatability capabilities. GAs that employed normalized geometric ranking selection function were computationally efficient, but yielded solutions that were approximately equal. The real coded tournament selection function GA produced classifiers that were approximately 3% less accurate than the binary coded tournament selection function GA.

1 Introduction

This research focuses on a pattern classification problem utilized within an application that could assist contact centers in determining customer activities within their Interactive Voice Response (IVR) systems. During the last 5 years, the South African contact centre industry has experienced exceptional growth [1]. In order to gain a competitive advantage, contact centers are to fulfill customer expectations efficiently with informed responses and actions while discovering techniques to reduce overall cost of providing such a service [2]. It is therefore essential that contact centers evaluate the performance of their solutions in relation to customer interaction. This will assist in quantifying the self-service customer perception.

Customers want to resolve problems on their first call. They want convenient and reliable information fast. IVR systems can provide this. An IVR system is an automated telephony system that interacts with callers, gathers relevant information and routes calls to the appropriate destinations [2]. The inputs to the IVR system can be voice, Dual Tone Multi-Frequency (DTMF) keypad selection or a combination of the 2.

M. Köppen et al. (Eds.): ICONIP 2008, Part II, LNCS 5507, pp. 728–735, 2009.

The aim of this research is to develop a field classification application, using computational intelligent methods, which could assist companies in quantifying customer activities within their IVR systems.

IVR applications are developed in Voice Extensible Markup Language (VXML). VXML applications are voice-based dialog scripts that consist of form or dialog elements. The form or dialog elements are used to group input and output sections together. A field element is used to obtain and interpret user input information. As a result, the form or dialog elements contain field elements [3].

The classification system developed categorizes caller behaviour at a field within the IVR applications into specific interaction classes. As a result, these interaction classes can assist in determining trends of caller behaviour within the self-service systems. For example, the field classification application can identify areas within the self-service applications that experienced the most caller disconnects. Thereafter, analysts can listen to a sample of these calls and determine the reason for this. The field classification system can also identify the fields that resulted in the majority of the callers transferring to a Customer Service Agent (CSA) due to difficulties experienced.

In order to develop such an application, the classification of data must be accurate. This paper details the development of artificial neural network (ANN) field classifiers using Multi- Layer Perceptron (MLP) neural network and genetic algorithms (GAs). GAs employing floating-point and binary representations are considered. Detailed explanations on these ANN architectures can be found in [4].

GAs are known to be robust optimization procedures based on the mechanism of the natural evolution. GAs have the capability of locating a global optimum as these procedures do not use any derivative information and GAs search from multiple points. In traditional GAs, binary representation has been used for chromosomes. Floating-point representation, real-coded GAs, of parameters as a chromosome has also been used [5].

The classification of data into various classes has been an important research area for many years. Artificial neural networks (ANNs) have been applied to pattern classification [6][7][8]. Fuzzy systems [9] and neural networks constructed using GAs have been utilized [10].

The section to follow provides a brief explanation of the field classification system. Thereafter, the implementation methodology is described. The paper ends with the comparison of the various field classifiers developed and the selection of the superior networks.

2 The Developed System

As the developed system is to be used to identify trends of caller behaviour at a field within the IVR VXML applications, the system is trained based on data extracted from IVR log event files. These files are generated by the IVR platform as specific events occur during a call to the system. Events such as call begin, form enter, form select, automatic speech recognition events, transfer events and call end events are captured in the logs [11].

Table 1. The inputs and outputs of the field classifier

Inputs	Outputs	Output interaction class
Confidence	Field performance	Good, acceptable, bad
No matches	Field transfer reason	Unknown, difficult
No inputs	Field hang-up reason	Unknown, difficult
Max speech timeouts	Field duration	High, medium, low
Barge-ins	Field recognition level	High, medium, low
Hang-ups		
Transfer to Customer Service Agent		
Duration		

Table 1 shows the inputs and outputs of the field classification system. These specific inputs have been selected to characterize the caller difficulty experienced at a field within a VXML application. The outputs of the classifiers summarize the caller field behaviour through the use of interaction classes. The confidence input illustrates the IVR speech recognition probability. The value is a percentage. A caller may answer a question the VXML application poses with a response the application did not anticipate. These events are represented by the no match inputs. Callers may reply to VXML applications by talking beyond the allocated time-out period of the field. These events correspond to the maximum speech timeout input parameters of the field classifiers [12]. When a question is presented by the automated application, a caller may remain silent. No input events represent these occurrences. In general, most self-service applications accommodate 3 of the above events per field. On the third attempt, if the caller is unsuccessful in completing the field, the caller is usually transferred to a CSA. These inputs are important as they assist in identifying difficulties experienced at the field.

The duration input parameter illustrates the time the caller consumed completing the VXML application field. The barge-in input parameter illustrates whether or not a caller interrupted the application while the automated question prompted played. Caller disconnects and transfers to Customer Service Agents (CSAs) are represented by the hang-up and transfer to agent input parameters, respectively. These inputs can also assist in determining the level of difficulty the caller experienced in the field.

The field performance output interaction class of the classifier will illustrate whether the caller behaviour is good, acceptable or bad. The field transfer reason and field hang-up reason interaction classes attempt to identify the motivation for the transfer to CSA or caller disconnect, respectively. The field duration as well as field recognition level classes illustrate 3 categories of performance, low, medium and high. As a result, these output parameters will assist in characterizing the caller experience at a VXML field.

Once the field classification system has been employed to assess the performance of various interaction areas within the VXML applications, the caller perception of the solution can be determined. For example, after analyzing the

outputs of the classifiers, it is determined many callers are disconnecting due to difficulties experienced during the preliminary fields of an application, it can be concluded that the callers are frustrated with the system. In order to reduce the caller dissatisfaction, it would be essential to optimize the various grammars that interpret the caller responses.

3 Implementation Methodology

The development process was divided into various stages. The remainder of this section will elaborate on these stages of implementation The following procedure has been pursued in the creation of the various classifiers employed:
1. Selection and pre-processing of data to be used by the classifiers.
2. Optimization of the classifier architectures using GAs.
3. Comparison of the various GAs developed and the selection of the superior network.

3.1 Selection and Pre-processing of Data

The data utilized in developing the ANNs is based on data extracted from IVR log event files. A parsing application that extracted information such as recognition confidence values, caller barge-in information has been utilized to generate the data sets. This application also executed specific instructions in the creation of the data sets. Rules such as if 3 occurrences of no inputs, no matches or maximum speech timeouts occur within the caller interaction to a particular field and a transfer occurs thereafter, the field transfer reason computed will be 'difficulties' were followed.

In order to present the no match, no input and maximum speech timeout information to the field classifiers, a binary notation has been employed. These inputs are presented by 3 digit binary words. For example, if a no match 1 and a no match 2 occur at a field, the binary notation will be '011'. A similar binary notation is employed for the no input and maximum speech timeout classifier inputs. The barge-in, hang-up and transfer to CSA input information were represented by bit binary words. A similar binary notation scheme has also been utilized to interpret the interaction classes outputted. The data is divided into training, validation and test sets. The validation data set is used to assess the network and the test data is used to confirm the classification capability of the developed networks.

The confidence and duration input parameters of the classifiers were preconditioned by normalizing the data. Due to the binary word representation utilized to present the remaining inputs, normalization of these input parameters is not necessary.

3.2 Optimization of Classifier Architecture Using Genetic Algorithms

This stage of implementation involved the optimization of the ANN architectures. As a result, this step of development involved the identification of the correct number of hidden neurons that would yield the most accurate results.

Binary and real coded GAs were employed to optimize the field classifier architecture. Populations of MLP ANNs were generated by the GAs. Due to the MLP ANN non-linear capabilities, they are said to be excellent universal approximators that provide highly accurate solutions. As a result, these networks produce very practical tools for classification and inversion problems [4].

It has been stated that a network with 1 hidden layer, provided with sufficient data, can be used to model any function [4]. Therefore, the MLP ANNs employed consisted of only 1 hidden layer. The MLP ANN hidden layer consists of non-linear activation functions. The choice of the activation function is largely dependent on the application of the model [4]. However, it has been found that the hyperbolic tangent activation function offers a practical advantage of faster convergence during training [6]. As a result, this function has been employed within the MLP network. The most appropriate selection of the output layer activation function for a classification problem is the logistic sigmoidal function [6]. Therefore, this function has been employed within the output layer of the MLP network.

An error function that mapped the number of hidden nodes to the accuracy of the developed network was used as the evaluation function for the GAs. The fitness of the individuals within a population was determined by calculating the accuracy of the ANNs when presented with validation and test data sets. The minimum value of these accuracies determined the fitness of the individual.The outputs of the ANNs were interpreted by utilizing a classification threshold value of 0.5. This value proved to be adequate for the MLP ANN implementations. A confusion matrix is utilized to identify the number of true and false classifications that are generated by the models developed. This is then used to calculate the true accuracy of the classifiers, using the following equation:

$$Accuracy = \sqrt{\frac{TP * TN}{(TP + FN) * (FP + TN)}} \tag{1}$$

where
TP is the true positive (1 classified as a 1),
TN is the true negative (0 classified as a 0),
FN is the false negative (1 classified as a 0),
FP is the false positive (0 classified as a 1).

The GAs produced 25 generations of 10 MLP ANN individuals within the population. The GAs were limited to produce MLP ANN individuals with the number of hidden nodes between 5 and 100. Networks with hidden nodes greater than 100 were not developed due to the generalization capabilities reducing as the number of intermediate units increase [13].

In order to produce successive generations, the selection function determines which of the individuals will survive to the next generation. Roulette wheel selection, scaling techniques, tournament, normal geometric, elitist models and ranking methods are examples of selection functions used [5]. The selection approach assigns a probability of selection to each individuals based on its fitness

value. This research compares solutions produced by GAs that employ normalized geometric ranking and tournament selection functions.

3.3 Comparison of the Various Genetic Algorithms and Selection of the Superior Model

The binary and real coded GAs were compared in terms of their repeatability, computational efficiency as well as quality of the solution.

Repeatability is defined as the number of repetitions that returned the same optimal number of hidden nodes. Repeatability has been demonstrated by executing 8 repetitions of the binary and real coded GAs. This investigation revealed that the binary coded ranking selection GA returned different number of hidden nodes with different fitness values on each execution of the algorithm. Figure 1 also illustrates that the remaining GAs returned the same number of hidden nodes with the same fitness value for a number of the repetitions.

Computational efficiency, in this context, is defined as the number of generations the GA utilized to converge to the most optimal number of hidden nodes. Figure 1 illustrates that, the real coded tournament selection GA is most efficient. In the majority of the repetitions, this GA converged to an optimal solution before 9 generations. In the majority of the investigations, the binary coded ranking and the binary coded tournament selection GA converged to an optimal solution before 10 generations. However, the real coded ranking selection GA, in majority of the repetitions, converged before generation 13. Figure 1 also illustrates that the binary coded tournament selection GA resulted in the most accurate MLP ANNs with accuracies of approximately 96.45%. This algorithm returned the same fitness value for 2 of the 8 repetitions. The number of hidden nodes corresponding to this accuracy is 5. However, the algorithm converged to this solution at generation 23.

Quality of the GA solution is the verification that the number of hidden nodes returned by the GA is really the most optimal value. It has been determined that 5, 6, 7 and 9 number of hidden nodes resulted in the most accurate MLP classifiers. This has been determined by identifying the most accurate MLP classifiers generated from the 8 GA repetitions executed. These number of hidden nodes were identified by the binary coded tournament, real coded ranking, binary coded ranking and real coded tournament selection GAs, respectively. In order to determine the quality of the GA solutions, MLP classifiers were created containing these number of hidden nodes. The accuracy of these networks was determined by using equation 1. The sensitivity and false positive ratio were also calculated to measure the true performance of the networks. Table 2 illustrates the results of the quality of solution investigation. As illustrated, the number of hidden nodes that resulted in the most accurate MLP classifier is 5. This number of hidden nodes creates a network that performs accurately on both the validation and test data sets. As a result, the classifier has good generalization capabilities, as compared to the other MLP classifiers that perform approximately 4% more accurately on the test data set. The sensitivity and false positive ratio values also support these findings.

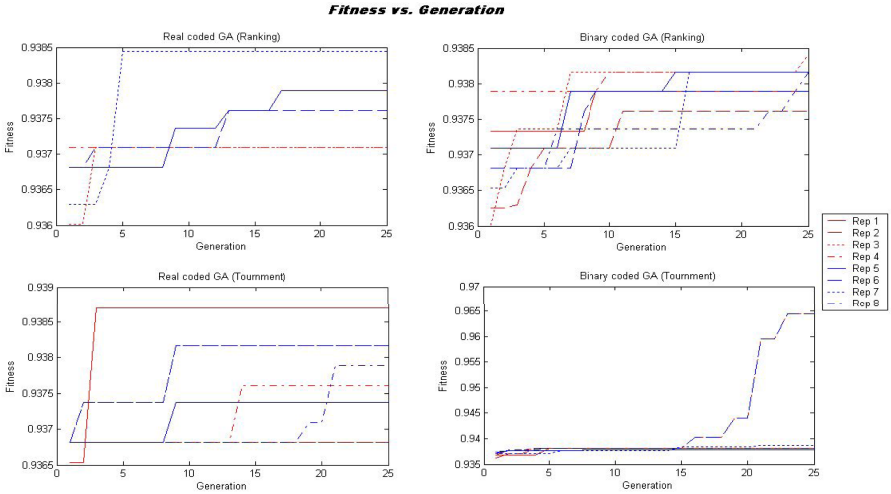

Fig. 1. This figure shows the Fitness vs. Generation for all 8 repetitions

Table 2. This table illustrates the various models that were created. Accuracies are presented as percentages.

Hidden Nodes	Accuracy (Validation)	Accuracy (Test)	Hit Rate (Validation)	Hit Rate (Test)	False Alarm Rate (Validation)	False Alarm Rate (Test)
5	96.36	96.80	95.02	95.12	0.023	0.015
6	93.73	98.17	90.12	97.08	0.025	0.007
7	93.68	98.00	89.98	96.75	0.025	0.007
9	93.60	97.98	89.83	96.74	0.025	0.008

4 Conclusion

This research entailed the development of a 'Say account' field classification system. MLP networks were used to classify caller interactions. Binary coded and real coded GAs that utilized ranking as well as tournament selection functions were also employed to optimize the classifier architecture.

The development methodology utilized for creating all the networks involved, initially, pre-processing the data sets. This ensured that the classifiers would interpret the inputs proficiently. Thereafter, the numbers of hidden nodes were optimized utilizing the GA algorithm. This resulted in creating acceptable network architecture. GA results were compared in terms of computational efficient, repeatability and the quality of the solution. These analyses assisted in determining the algorithm that was most suited to this application and the algorithm that yielded the most accurate classifier. Acceptable classification accuracies were achieved. The most accurate network that illustrated excellent generalization capabilities yielded accuracies of approximately 96%. This illustrates that MLP ANNs are proficient in classifying field caller interaction.

The binary coded GA that utilized tournament selection yielded the most accurate MLP classifier. The algorithm yielded the same result on 2 of the 8 repetitions. The number of hidden nodes of 5 returned by the algorithm was confirmed to be the optimal in the quality of the solution investigation. However, the algorithm converged at this solution at generation 23 of 25 generations. As a result, it can be concluded that this GA is most suited to this application in terms of optimal solution, but it is not the most computational efficient algorithm.

References

1. The first major South African Contact Centre and BPO Market Quantification study in over 5 years, `http://www.contactindustryhub.co.za/research.php` (last accessed August 17, 2008)
2. Nichols, C.: The Move from IVR to Speech – Why This is the Right Time to Make the Move to Speech Applications in Customer-Facing Operations. Intervoice (2006)
3. VoiceXML 2.0/VoiceXML 2.1 Reference, `http://developer.voicegenie.com/voicexml2tagref.php` (last accessed August 17, 2008)
4. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press, Oxford (1995)
5. Houck, C.R., Joines, J.A., Kay, M.G.: A genetic algorithm for function optimization: a Matlab implementation. NCSU-IE Technical Report. North Carolina State University (1995)
6. Nabney, I.T.: Netlab: Algorithms for Pattern Recognition. Springer, Heidelberg (2002)
7. Patel, P.B., Marwala, T.: Neural Networks, Fuzzy Inference Systems and Adaptive-Neuro Fuzzy Inference Systems for Financial Decision Making. In: King, I., Wang, J., Chan, L.-W., Wang, D. (eds.) ICONIP 2006. LNCS, vol. 4234, pp. 430–439. Springer, Heidelberg (2006)
8. Marwala, T.: Fault classification using pseudo-model energies and neural networks. American Institute of Aeronautics and Astronautics 41, 82–89 (2003)
9. Russo, M.: FuGeNeSys A fuzzy genetic neural system for fuzzy modeling. IEEE Transaction on Fuzzy Systems 6, 373–388 (1998)
10. Abdella, M., Marwala, T.: The use of genetic algorithms and neural networks to approximate missing data in database. Computing and Informatics 24, 1001–1013 (2006)
11. VoiceGenie Technologies Inc. VoiceGenie 7 Tools Users Guide. VoiceGenie Technologies Inc. (2005)
12. VoiceXML properties, `http://community.voxeo.com/vxml/docs/nuance20/VXMLproperties.html` (last accessed August 25, 2008)
13. Baum, B.E., Haussler, D.: What size net gives valid generalization? Neural Computation 1, 81–90 (1989)

Paper published in Conference Proceedings of 2008 IEEE International conference on Systems, Man and Cybernetics

Interactive Voice Response Field Classifiers.

P. B. Patel
School of Electrical and Information Engineering
University of the Witwatersrand
Johannesburg, South Africa
p.patel@ee.wits.ac.za

T. Marwala
School of Electrical and Information Engineering
University of the Witwatersrand
Johannesburg, South Africa
t.marwala@ee.wits.ac.za

Abstract— **Accurate classification of caller interaction within Interactive Voice Response systems would assist corporations to determine caller behaviour within these solutions. This paper proposes an application, which employs artificial neural networks that could assist contact centers to determine caller activity within their automated systems. Multi-layer perceptron and Radial Basis Function neural network architectures are implemented as classifiers to determine caller interaction. Field classifiers for a pay beneficiary application were developed. 'Say account' networks were created utilizing 'generated' and 'live' data sources. Multi-layer perceptron networks proved appropriate for this application. The most accurate network created, 99.99%, is the 'Say account' classifier. The difference in accuracy between the 'generated' and 'live' classifiers is approximately 2%. However, greater development effort is required to implement the former. As a result, the 'live' data source methodology is preferred.**

Keywords— **neural networks, classification of data, contact center analytics, Interactive Voice Response, Voice Extensible Markup Language**

I. INTRODUCTION

Research conducted for NCR Corporation, an international company specializing in self-service technologies, by Opinion Research Corporation in the United States found that customers spend approximately 2 days each year waiting in line for service [1]. Approximately 40% of the participants of the study were willing to utilize self-service kiosks or other self-service devices to reduce time consumed waiting for service [1].

Customers want to resolve problems on their first call to the contact center. They want convenient and reliable information fast. Interactive Voice Response (IVR) systems can provide this. An IVR system is an automated telephony system that interacts with callers, gathers relevant information and routes calls to the appropriate destinations [2]. The inputs to the IVR system can be voice, Dual Tone Multi-Frequency (DTMF) keypad selection or a combination of the 2. IVR systems can provide appropriate responses in the form of voice, fax, callback, e-mails and other media [2]. An IVR system solution may consist of telephony equipment, software applications, databases and supporting infrastructure.

However, there are many companies that have IVR systems, which are poorly designed or implemented that result in caller frustration when utilizing the automated solutions [3]. An example of a cause of caller frustration would be a voice prompt that plays a long list of menu options until the caller is confused on what to do. As a result, the customer would probably end the call.

The aim of this research is to develop a field classification application, using computational intelligent methods, which could assist companies in determining customer activities within their IVR systems. It is anticipated that this application would be used in conjunction with other customer behaviour analysis techniques such as listening to recorded calls. As a result, this application should be used to confirm the system performance in relation to customer interaction.

IVR applications are developed in Voice Extensible Markup Language (VXML). VXML applications are voice-based dialog scripts that consist of form or dialog elements. The form or dialog elements are used to group input and output sections together. A field element is used to obtain and interpret user input information. As a result, the form or dialog elements contain field elements [4].

The classification system developed categorizes caller behaviour at a field within the IVR applications into specific interaction classes. As a result, these interaction classes can assist in determining trends of caller behaviour within the self-service systems. For example, the field classification application can identify calls where the automated speech recognition at a particular field is low. Thereafter, analysts can listen to a sample of these calls and determine the reason for this. The field classification system can also identify the fields that resulted in the majority of the callers transferring to a Customer Service Agent (CSA).

In order to develop such an application, the classification of data must be accurate. This paper compares field classifiers that were developed utilizing different data sources. However, the trained classifiers were tested with the same 'live' input data source.

The classification of data into various classes has been an important research area for many years. ANNs have been applied to pattern classification [5]. Research has also been conducted on fuzzy classification. This resulted in many algorithms, such as fuzzy K-nearest neighbour [6] and fuzzy c-means [7], being applied to classification problems. Fuzzy systems constructed using genetic algorithms have been utilized [8] [9] [10]. Fuzzy neural networks have also been employed in pattern classification applications [11] [12] [13].

1-4244-2384-2/08/$20.00 ©2008 IEEE

Support Vector Machines have been applied to multi-category classification problems [14]. These classification tasks have also been implemented by combining multiple simpler specialized classifiers [15] [16] [17].

The section to follow examines the field classification system as well as its implementation methodology. The paper ends with the comparison of the various field classifiers developed and the selection of the superior networks.

II. THE DEVELOPED SYSTEM.

As the developed system is to be used to identify trends of caller behaviour at a field within the IVR VXML applications, the system is trained based on data extracted from IVR log event files. These files are generated by the IVR platform as specific events occur during a call to the system. Events such as call begin, form enter, form select, automatic speech recognition events, transfer events and call end events are captured in the logs [18].

Table I shows the inputs and outputs of the field classification system. These specific inputs have been selected to characterize the caller difficulty experienced at a field within a VXML application. The outputs of the classifiers summarize the caller field behaviour through the use of interaction classes.

The confidence input illustrates the IVR speech recognition probability. The value is a percentage. The larger the percentage, the greater the probability the system interpreted the caller successfully.

A caller may answer a question the VXML application prompts with a response the application did not anticipate. These events are represented by the no match inputs. In general, most VXML applications accommodate 3 no match events per field. On the third attempt, if the caller replies with another response that the application does not accommodate, the caller is usually transferred to a CSA. The no match input parameters to the field classification system is important as they can be utilized to assist in determining the number of callers that understood the VXML prompt correctly as well as the number of callers that misinterpreted the question prompted.

A caller may not answer a question presented by the automated application. When the application anticipates a response, the caller remains silent. These events are represented by the no input parameters. VXML applications normally accommodate 3 no input events on each field. A transfer to CSA usually occurs after the third no input event. These input parameters are also important as they assist in identifying callers that were confused when prompted with the automated application question. As a result, the caller remained silent.

Callers may reply to VXML applications by talking beyond the allocated timeout period of the field. These events are represented by the maximum speech timeout input parameters of the field classifiers [19]. In general, VXML applications accommodate 3 of these events. Similar to the no match and no input events, on the third attempt, if a maximum speech timeout occurs, the caller is transferred to a CSA. These inputs are important as they assist in determining whether the timeout periods are adequate for callers to complete their responses.

TABLE I. THE INPUTS AND OUTPUTS OF THE FIELD CLASSIFIER

Inputs	Outputs	Output interaction class
Confidence.	Field performance.	Good, acceptable, bad.
No matches.	Field transfer reason.	Unknown, difficult.
No inputs	Field hang-up reason.	Unknown, difficult.
Max speech timeouts.	Field duration.	High, medium, low.
Barge-ins.	Field recognition level.	High, medium, low.
Hang-ups.		
Transfer to Customer Service Agent.		
Duration.		

The duration input parameter illustrates the time the caller consumed completing the VXML application field. The barge-in input parameter illustrates whether or not a caller interrupted the application while the automated question prompted played. Caller disconnects and transfers to Customer Service Agents (CSAs) are represented by the hang-up and transfer to agent input parameters, respectively. These inputs can also assist in determining the level of difficulty the caller experienced in the field.

The field performance output interaction class of the classifier will illustrate whether the caller behaviour is good, acceptable or bad. The field transfer reason and field hang-up reason interaction classes attempt to identify the motivation for the transfer to CSA or caller disconnect, respectively. The field duration as well as field recognition level classes illustrate 3 categories of performance, low, medium and high. As a result, these output parameters will assist in characterizing the caller experience at a VXML field.

III. IMPLEMENTATION METHODOLOGY.

The development process was divided into various stages. The following procedure has been pursued in the creation of the various ANN architectures employed:

- Selection and processing of data to be used by ANNs during training, validation and testing.
- Optimization of the number of iterations used to train the ANNs.
- Optimization of the number of hidden neurons or nodes within the ANN.
- Comparison of the various networks developed and the selection of the superior network.

The remainder of this section will elaborate on the various stages of the procedure stated above.

A. Selection and processing of data.

The data utilized in developing the ANNs is based on data extracted from IVR log event files. 2 Distinct data sources were utilized in the development of the classifiers. The 'generated' data source included data computed by a rule based application. The 'live' data source consisted of caller information extracted from actual IVR log event files.

The rule based application executed specific instructions in the creation of the data to be used in the development of the ANNs. Rules such as if 3 occurrences of no inputs, no matches or maximum speech timeouts occur within the caller interaction to a particular field and a transfer occurs thereafter, the field transfer reason computed will be 'difficulties' were followed. Various combinations of input parameters that were based on speech analysis experience were generated and propagated through the rule based application to create the 'generated' input output data sets.

A parsing application that extracted information such as recognition confidence values, caller barge-in information has been utilized to develop the 'live' data source. This information is also propagated through the rule based application to produce the 'live' output data sets.

The research conducted entailed the creation of 'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' field classifiers. Caller behaviour per field is unique. For example, at a 'Say confirmation' field the caller is required to say 'yes' or 'no'. However, the caller is requested to say the account name at the 'Say account' field. As a result, the duration to complete the VXML application field is much shorter at the confirmation field. Therefore, each classifier is trained with data relevant to the field.

Networks developed utilizing the 'generated' data source were only created for the 'Say account' field classifier. These networks were compared in accuracy with the corresponding classifiers developed using the 'live' data source. The remaining field classifiers developed employed 'live' data sources.

In order to present the no match, no input and maximum speech timeout information to the field classifiers, a binary notation has been employed. These inputs are presented by 3 digit binary words. For example, if a no match 1 and a no match 2 occur at a field, the binary notation will be '011'. A similar binary notation is employed for the no input and maximum speech timeout classifier inputs. The barge-in, hang-up and transfer to CSA input information were represented by bit binary words. Table II shows the input representation format.

A similar binary notation scheme has also been utilized to interpret the interaction classes outputted. Table III illustrates the manner in which the outputs are to be translated.

In order to ensure that over-fitting and under-fitting were avoided, the data has been divided into 3 sets. Over-fitting occurs when the network does not generalize but rather tends to memorize the training data. Under-fitting occurs when the network does not follow the data at all [20]. The data is divided into training, validation and test sets. The training data set is used to train the ANN to find the general pattern between its inputs and outputs. The validation data set is used to assess the network and the test data is used to confirm the classification capability of the developed networks.

TABLE II. THE INPUT REPRESENTATION FORMAT

Input	Input value	Binary notation		
No matches	No match 1	0	0	1
No matches	No match 2	0	1	0
No matches	No match 3	1	0	0
No matches	No match 1 and no match 2	0	1	1
No matches	No match 1, no match 2 and no match 3	1	1	1
No inputs	No input 1	0	0	1
No inputs	No input 2	0	1	0
No inputs	No input 3	1	0	0
No inputs	No input 1 and no input 2	0	1	1
No inputs	No input 1, no input 2and no input 3	1	1	1
Max speech timeout	Max speech timeout 1	0	0	1
Max speech timeout	Max speech timeout 2	0	1	0
Max speech timeout	Max speech timeout 3	1	0	0
Max speech timeout	Max speech timeout 1 and max speech timeout 2	0	1	1
Max speech timeout	Max speech timeout 1, max speech timeout 2 and max speech timeout 3	1	1	1
Barge-in	Occurred	-	-	1
Hang-up	Occurred	-	-	1
Transfer to CSA	Occurred	-	-	1

TABLE III. THE OUTPUT REPRESENTATION FORMAT

Output	Output interaction class	Binary notation		
Field performance	Good	0	0	1
Field performance	Acceptable	0	1	0
Field performance	Bad	1	0	0
Field transfer reason	Difficulties	-	0	1
Field transfer reason	Unknown	-	1	0
Field hang-up reason	Difficulties	-	0	1
Field hang-up reason	Unknown	-	1	0
Field duration	High	1	0	0
Field duration	Medium	0	1	0
Field duration	Low	0	0	1
Field recognition level	High	1	0	0
Field recognition level	Medium	0	1	0
Field recognition level	Low	0	0	1

The confidence and duration input parameters of the classifiers were preconditioned by normalizing the data. Normalizing the data entails manipulating the data sets such that the values within the sets are between 0 and 1. The networks developed were trained utilizing the normalized data sets. Due to the binary word representation utilized to present the remaining inputs, normalization of these input parameters is not necessary.

Normalization is accomplished by acquiring the minimum and maximum values within the data sets. These values are then utilized to compute the normalized values.

The purpose of normalizing the data sets is to modify the variable levels to a reasonable value. If such a transformation is not employed, the value of the variable could be too large for the network to process, especially when several layers of nodes within the ANN are involved [20]. Normalizing the data sets also reduces the fluctuation and noise within the data [20].

There are a variety of practical reasons that illustrate normalizing the data sets can result in faster training and reduce the chances of obtaining local optima. Some of these reasons include better numerical conditioning (Hessian matrices), better weight initialization values and better weight decay estimates [20].

B. Optimization of the number of iterations used to train the neural network.

Multi-layer Perceptron (MLP) and the Radial Basis Function (RBF) neural network architectures were utilized in the classification of call behaviour. The MLP and RBF neural network architectures are possibly the most extensively employed ANNs in pattern classification [5]. Due to the non-linear capabilities of these networks, they are said to be excellent universal approximators that provide highly accurate solutions. As a result, these networks produce very practical tools for classification and inversion problems [20].

It has been stated that a network with 1 hidden layer, provided with sufficient data, can be used to model any function [20]. As a result, the MLP and RBF neural network architectures employed consisted of only 1 hidden layer.

The MLP network hidden layer consists of non-linear activation functions. There are 3 major forms of the function that should be considered. These are the linear, logistic sigmoidal and softmax activation functions [5].The choice of the activation function is mainly dependant on the application of the network [20]. However, it has been found that the hyperbolic tangent activation function offers a practical advantage of giving rise to faster convergence during training [5]. As a result, this function has been utilized within the MLP networks.

The MLP network output layer also consists of activation functions. There are 3 major forms of functions that should be considered. These are the linear, logistic sigmoidal and softmax activation functions [5]. It has been stated that the appropriate selection of the output-unit activation function for a classification problem is the logistic sigmoidal function [5]. As a result, this function has been employed within the output layer of the MLP network.

The RBF network that has been developed contained a Gaussian activation function within its hidden layer and a linear activation function within its output layer.

During this stage of implementation, the number of hidden nodes within the ANNs was assumed to be arbitrary. This will be optimized at a later stage of development. During this stage of development, the number of hidden nodes within the ANNs was 8. This stage of implementation involved the optimization of the number of iterations utilized to train the classifiers. As a result, this involved the selection of an appropriate number of iterations that would result in classifiers with excellent generalization capabilities.

The number of iterations has been optimized by minimizing an error function that mapped the number of iterations to the accuracy of the developed classifiers. The process has been performed on the validation and test data sets.

Since this is a classification implementation, a confusion matrix is employed to identify the number of true and false classifications that are generated by the ANN developed. This is then utilized to calculate the true accuracy of the ANN classifiers. The sensitivity and false positive ratio were also calculated to assist in measuring true performance. The following equations were used:

$$Accuracy = \sqrt{\frac{TP \times TN}{(TP + FN) \times (FP + TN)}} \quad (1)$$

$$Sensitivity = \frac{TP}{TP + FN} \quad (2)$$

$$False\ \ Positive\ \ Ratio = \frac{FP}{FP + TN} \quad (3)$$

where

TP is the true positive (1 classified as a 1),

TN is the true negative (0 classified as a 0),

FN is the false negative (1 classified as a 0),

FP is the false positive (0 classified as a 1).

The outputs of the ANNs were interpreted by utilizing a classification threshold value of 0.5. This implies that if the classifier outputs a value less than 0.5, the output will be regarded as a 0. Similarly, if the output value is larger than or equal to 0.5, the output will be interpreted as a 1. This threshold value of 0.5 proved to be adequate for the MLP and RBF ANN implementations.

The number of iterations has been optimized by constructing various MLP and RBF networks with the number of iterations ranging from 50 to 1000.

Utilizing the training data set, these networks were trained. Thereafter, the validation and test data set was presented to the networks. The accuracy of the developed networks was calculated for the training, validation and test data sets. MLP and RBF networks trained with the number of iterations that resulted in the largest accuracy value, when presented with the validation and test data sets, were analyzed.

Good results were obtained that yielded both MLP and RBF networks with excellent generalization capabilities. Table IV and Table V illustrates the results of this stage of implementation.

Networks consisting of these optimized iteration values were developed in the next stage of implementation.

C. Optimization of the number of hidden nodes within the neural network.

This stage of implementation involved the optimization of the ANN architecture. As a result, this step of development entailed selecting the correct number of hidden neurons that would yield the most accurate results.

TABLE IV. RESULTS OF VARIED NUMBER OF ITERATIONS.

Network architecture	Iterations	Hidden nodes	Accuracy (Validation)	Accuracy (Test)
'Say account' classifier ('Live' data source).				
MLP	400	8	0.9471	0.9113
MLP	500	8	0.9445	0.9121
RBF	600	8	0.7887	0.8245
RBF	700	8	0.8679	0.8971
'Say account' classifier. ('Generated' data source).				
MLP	450	8	0.9999	0.9296
MLP	550	8	0.9999	0.9488
RBF	450	8	0.7942	0.7849
RBF	550	8	0.8384	0.8373
'Say amount' classifier.				
MLP	450	8	0.9872	0.9715
MLP	550	8	0.9899	0.9744
RBF	800	8	0.8391	0.8076
RBF	900	8	0.9195	0.9072
'Say confirmation' classifier.				
MLP	20	8	0.9662	0.9817
MLP	40	8	0.9881	0.9952
RBF	520	8	0.9338	0.9500
RBF	540	8	0.9826	0.9893
'Select beneficiary' classifier.				
MLP	850	8	0.9518	0.9853
MLP	950	8	0.9589	0.9869
RBF	200	8	0.9316	0.9030
RBF	300	8	0.9601	0.9560

TABLE V. RESULTS OF VARIED NUMBER OF ITERATIONS (ROC CURVE).

	Iterations	Nodes	Validation		Test	
			FA	HR	FA	HR
'Say account' classifier ('Live' data source).						
MLP	500	8				
RBF	700	8				
'Say account' classifier. ('Generated' data source).						
MLP	550	8				
RBF	550	8				
'Say amount' classifier.						
MLP	550	8				
RBF	900	8				
'Say confirmation' classifier.						
MLP	40	8				
RBF	540	8				
'Select beneficiary' classifier.						
MLP	950	8				
RBF	300	8				

The number of hidden neurons or nodes has been optimized by minimizing an error function that mapped the number of hidden nodes to the accuracy of the developed networks. The process has been performed on the validation and test data sets.

The hidden neurons or intermediate units were optimized by creating various MLP and RBF ANNs with hidden nodes of 5 to 25. Networks with hidden nodes greater than 25 were not developed due to the predictive capabilities or generalization capabilities reducing as the number of intermediate units increase. More hidden nodes increases the dimensionality of the function being fitted, enabling easier training which results from higher training capacity. However, this detrimentally affects the generalization capabilities of the network. A major consideration when developing a suitable ANN for a classification application is to make a trade-off between convergence and generalization [21].

Utilizing the training data set, these networks were trained. The validation and test data set was then presented to the networks. Thereafter, the accuracy of the developed networks was calculated for the training, validation and test data sets. MLP and RBF networks with the number of hidden nodes that resulted in the largest accuracy value, when presented with the validation and test data sets, were analyzed.

Table VI illustrates the hidden nodes that resulted in the largest accuracy value for the validation data set. The networks also employed the same activation functions mentioned in the previous section.

D. Comparison of the various networks developed and the selection of the superior network.

This stage of development entailed the comparison of the various ANNs that were created. It also involves the selection of the best networks to classify caller field behaviour into interaction classes.

It is evident, from the investigations conducted, that the ANNs illustrated in Table VII and Table VIII resulted in the most accurate classifiers.

It is also evident that the difference in accuracy between the field classifiers created utilizing the 'generated' and 'live' data source is small. The difference between the networks is approximately 2%. However, the development effort required to implement the field classifier using the 'generated' data source is much greater. It requires a training data set to be created based on expert knowledge of caller behaviour within VXML applications. This is required to create a training data set that represents input parameter combinations that reflect actual caller behaviour. However, the 'live' data source development method extracts information from actual log files that contain real caller behaviour. As a result, less effort is required and is therefore the preferred method.

It has also been determined that the MLP network architecture is particularly suited for this application. This is relevant for all the networks except the 'Select beneficiary' field classifier. The difference in accuracy between the 2 network architectures is on average 1%.

TABLE VI. RESULTS OF VARIED HIDDEN NODES.

Network architecture	Number of hidden nodes
'Say account' classifier ('Live' data source).	
MLP	10, 15, 17.
RBF	18, 20, 22.
'Say account' classifier. ('Generated' data source).	
MLP	13, 19, 21.
RBF	19, 22, 23.
'Say amount' classifier.	
MLP	7, 13, 14.
RBF	19, 24, 25.
'Say confirmation' classifier.	
MLP	8, 22, 24.
RBF	21, 23, 24.
'Select beneficiary' classifier.	
MLP	6, 20, 25.
RBF	15, 18, 20.

TABLE VII. NETWORKS SELECTED.

Network architecture	Iterations	Hidden nodes	Accuracy (Validation)	Accuracy (Test)
'Say account' classifier ('Live' data source).				
MLP	500	15	0.9989	0.9629
RBF	700	22	0.9425	0.9355
'Say account' classifier. ('Generated' data source).				
MLP	550	21	1	0.9999
RBF	550	22	0.9817	0.9816
'Say amount' classifier.				
MLP	550	13	0.9997	0.9922
RBF	900	25	0.9751	0.9599
'Say confirmation' classifier.				
MLP	40	22	0.9962	0.9921
RBF	540	24	0.9931	0.987
'Select beneficiary' classifier.				
MLP	950	20	1	0.9420
RBF	300	18	0.9871	0.9752

TABLE VIII. RESULTS OF VARIED NUMBER OF ITERATIONS (ROC CURVE).

	Iterations	Nodes	Validation		Test	
			FA	HR	FA	HR
'Say account' classifier ('Live' data source).						
MLP	500	8				
RBF	700	8				
'Say account' classifier. ('Generated' data source).						
MLP	550	8				
RBF	550	8				
'Say amount' classifier.						
MLP	550	8				
RBF	900	8				
'Say confirmation' classifier.						
MLP	40	8				
RBF	540	8				
'Select beneficiary' classifier.						
MLP	950	8				
RBF	300	8				

IV. CONCLUSION

This research entailed the development of a field classification system. 'Say account', 'Say amount', 'Say confirmation' and 'Select beneficiary' field classifiers were created. The 'Say account' field classifier was created utilizing 'generated' data sources and 'live' data sources. The networks were then compared to determine the technique that required least development effort and yield the most accurate classifier.

The development methodology utilized for creating all the networks involved, initially, varying the number of iterations employed to train the classifiers. This ensured that the classifiers were not over-trained or under-trained. Thereafter, the numbers of hidden nodes were varied. This resulted in creating an acceptable network architecture.

Acceptable classification accuracies were achieved. It has been determined that the MLP network architecture was exceptionally proficient for this application. The best and worst accuracy levels obtained were 99.99% and 94.20%, respectively. These accuracy levels were attained for the 'Say account' and the 'Select beneficiary' field classifiers, respectively. This 'Say account' classifier was created utilizing the 'generated' data source. As a result, it can be concluded that the 'generated' data source approach is relevant and can result in highly accurate solutions.

However, the 'live' data source development effort required less effort than the 'generated' data source approach. The difference in accuracy of these networks was approximately 2%. This is small. As a result, it can be concluded that the 'live' data source development methodology is the preferred technique.

REFERENCES

[1] Self-Service becoming a way of life for South African consumers, last accessed: 3 March 2007. [Online]. Available: http://www.contactindustryhub.co.za/news_item.php?news_id=265.

[2] C. Nichols, The Move from IVR to Speech – Why This is the Right Time to Make the Move to Speech Applications in Customer-Facing Operations, Intervoice, 2006.

[3] How to right the wrongs of IVR, last accessed : 3 March 2007. [Online]. Available : http://www.callcentrehelper.com/wrongs-ivr-4684.html

[4] VoiceXML 2.0/VoiceXML 2.1 Reference, last accessed: 3 March 2007 [Online]. Available: http://developer.voicegenie.com/voicexml2tagref.php?tag=st_field&display=standardtags.

[5] T. Nabney, Netlab: Algorithms for Pattern Recognition., Springer, 2002.

[6] J. M. Keller, M. Gray and J. Givens, "A fuzzy k-nearest neighbor algorithm.", IEEE Transaction on System, Man and Cybernetics., vol. 15, no. 4, pp. 580-585,1985.

[7] J. C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms., Plenum, 1981.

[8] H. Ishibuchi, K. Nozaki, N. Yamamoto and H. Tanaka, "Construction of fuzzy classification systems with rectangular fuzzy rules using genetic algorithms.", Fuzzy Sets and Systems., vol. 65, no. 2-3, pp.237-253, 1994.

[9] H. Ishibuchi, K. Nozaki, N. Yamamoto and H. Tanaka, "Selecting fuzzy if-then rules for classification problems using genetic algorithms.",*IEEE Transactions on Fuzzy Systems.*, vol. 3, no. 3, pp. 260-270, 1995.

[10] M. Russo, "FuGeNeSys – A fuzzy genetic neural system for fuzzy modeling.", *IEEE Transaction on Fuzzy Systems.*, vol. 6, no. 3, pp.373-388, 1998..

[11] F. Masulli, F. Casalino and F. Vannucci, "Bayesian properties and performances of adaptive fuzzy systems in pattern recognition problems.", *Proceedings of the European Conference on Artificial Neural Networks*., *ICANN-94*., pp. 189-192, 1994.

[12] D. Nauck and R. Kruse, "A neuro-fuzzy method to learn fuzzy classification rules from data.", *Fuzzy Sets and Systems.*, vol. 89, no. 3, pp. 277-288, 1997.

[13] C-T. Sun and J-S. R. Jang, "A neuro-fuzzy classifier and its applications.", *Proceedings of the IEEE International Conference on Fuzzy System.*, vol. 1, pp. 94-98, 1993.

[14] C-W. Hsu and C-J. Lin, "A comparison of methods for multi-class support vector machines", *IEEE Transactions on Neural Neworks.*, vol. 13, no. 2, pp. 415-425, 2002.

[15] J. H. Friedman, "Another approach to polychotomous classification.", *Department of Statistics, Stanford University*., Technical report, 1996.

[16] J. Schurmann, *Pattern Classification. A Unified View of Statistical and Neural Principles.*, John Wiley & Sons Inc, 1996.

[17] B. Zadrozny, "Reducing multiclass to binary by coupling probability estimates.", *Neural Information Processing Systems Foundation 14*, pp.1041-1048, 2002.

[18] VoiceGenie Technologies Inc., VoiceGenie 7 Tools User's Guide, VoiceGenie Technologies Inc, 2005.

[19] VoiceXML properties, last accessed: 3 March 2007. [Online]. Available: http://community.voxeo.com/vxml/docs/nuance20/VXMLproperties.html.

[20] C. M. Bishop, *Neural Networks for Pattern Recognition.*, Oxford University Press, 1995.

[21] E. B. Baum, D. Haussler., "What size net gives valid generalization?", *Neural Computation*, vol. 1, pp. 81-90, 1989.

Paper to be published in Conference Proceedings of IEEE 2009 International Conference on Machine Learning and Applications

Genetic Algorithms, Neural Networks, Fuzzy Inference System, Support Vector Machines for Call performance classification.

Pretesh B Patel
Faculty of Engineering and the Built Environment
University of Johannesburg
Johannesburg, South Africa
e-mail: ppatel@uj.ac.za

Tshilidzi Marwala
Faculty of Engineering and the Built Environment
University of Johannesburg
Johannesburg, South Africa
e-mail: tmarwala@uj.ac.za

Abstract—**Accurate classification of caller interactions within Interactive Voice Response systems would assist corporations to determine caller behavior within these telephony applications. This paper details the development of such a classification system for a pay beneficiary application. Fuzzy Inference Systems, Multi-Layer Perceptron, Support Vector Machine and ensemble of classifiers were developed. Accuracy, sensitivity and specificity performance metrics were computed as well as compared for these classification solutions. Ideally, a classifier should have high sensitivity and high specificity. Exceptional results were achieved. The ensemble of classifiers is the preferred solution, yielding an accuracy of 99.17%.**

Keywords-Artificial Neural Networks, Genetic Algorithms, Fuzzy Inference Systems, Support Vector Machines, Ensemble of classifiers, Interactive Voice Response, Caller experience performance classification.

I. INTRODUCTION

Customer satisfaction fosters loyalty, increases the probability of selling additional products and services as well as reduces the chances of competitive replacement. However, customer dissatisfaction results into direct revenue losses due to customer churn and indirect losses such as damage to reputation. Improving the customer experience is a vital priority for contact centers across different industries.

In order to provide customers with access to convenient and reliable information fast, Interactive Voice Response (IVR) systems have been adopted by businesses. If implemented correctly, these systems can assist in improving the customer experience [1]. An IVR system is an automated telephony system that interacts with callers, gathers relevant information and routes calls to the appropriate destinations [1]. The inputs to the IVR system can be voice, Dual Tone Multi-Frequency (DTMF) keypad selection or a combination of the 2. IVR systems can provide appropriate responses in the form of voice, fax, callback, e-mails and other media [1]. An IVR system solution may consist of telephony equipment, software applications, databases and supporting infrastructure.

The aim of this research is to develop a call performance classification application, using computational intelligent methods, which could assist companies in determining customer activities within their IVR systems. IVR applications are developed in Voice Extensible Markup Language (VXML). These applications are voice-based dialog scripts that consist of form or dialog elements. Input and output sections are grouped together using these form or dialog elements. In order to acquire and interpret caller input information, a field element is used. As a result, the form or dialog elements contain field elements [2].

The classification system developed consists of field categorization and call performance classification processes. The field categorization process computes statistics that are used to classify caller behavior experienced at various fields within IVR applications into specific interaction classes. Utilizing these interaction classes, the call performance classification process, using computational intelligent methods, evaluates the performance of the customer call in relation to caller behavior experienced.

Therefore, the output call performance classification application classes can assist in determining trends of caller behavior within the self service systems. For example, the developed application can identify calls that transferred or disconnected at the final step of the automated process as well as calls where the automated speech recognition performed poorly. Thereafter, analysts can listen to a sample of these calls and determine the reason for this.

In order to develop such an application, the classification of data must be accurate. This paper compares classifiers that were developed utilizing Genetic Algorithms (GAs), Artificial Neural Network (ANN), Support Vector Machine (SVM) and Fuzzy Inference System (FIS) techniques. Ensembles of classifiers were also considered.

The classification of data into various classes has been an important research area for many years. ANNs have been applied to pattern classification [3]. Research has also been conducted on fuzzy classification. This resulted in many algorithms, such as fuzzy K-nearest neighbor [4] and fuzzy c-means [5], being applied to classification problems. Fuzzy

systems constructed using genetic algorithms have been utilized [6]. Fuzzy neural networks have also been employed in pattern classification applications [7].

Support Vector Machines have been applied to multi-category classification problems [8]. These classification tasks have also been implemented by combining multiple simpler specialized classifiers [9].

The section to follow examines the classification system as well as its implementation methodology. The paper ends with the comparison of the various call performance classifiers developed and the selection of the superior networks.

II. THE DEVELOPED SYSTEM

The research conducted entailed the creation of a call classification system for a beneficiary payment IVR application. Field categorization process extracts data from IVR log event files that is generated by the IVR platform as specific events occur during a call to the system. Events such as call begin, form enter, form select, automatic speech recognition events, transfer events and call end events are written to the logs [10].

Fig. 1 shows the field categorization process computed caller field statistics and outputs as well as corresponding interaction classes. Call begin, automatic speech recognition, form select, no match, no input, maximum speech timeout, prompt barge-in, system error, filling, call end events are extracted from log files during this process to calculate the caller field statistics. These events characterize the caller experience at a field within the VXML application. For detailed explanations of these events, refer to [2].

The IVR application consists of 'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' fields. This application accommodates 3 no match events per field. On a third no match event, the call is transferred to a DTMF field. If the caller fails to complete the DTMF field successfully on attempt 1, the call is transferred to a service agent. The same process is used for the third no input and maximum speech timeout events. As a result, the field categorization process caters for only 3 no match, no input and maximum speech timeout events.

The field performance output interaction class illustrates whether the caller behavior is good, acceptable, investigate or bad. The field transfer reason and field hang-up reason interaction classes attempt to identify the motivation for the transfer to agent or caller disconnect, respectively. Field difficulty attempt interaction class computes the number of difficulty events that occurred during the caller interaction. The field duration as well as field recognition level classes calculate 3 categories of performance; low, medium and high. As a result, these outputs will assist in characterizing the caller experience at a VXML field.

Experienced caller output indicates whether or not the caller is a regular user of the application and is therefore comfortable with the application call flow. In determining the number of experienced callers, the contact center can determine the usage of the application.

The call performance classifier utilizes these interaction classes as inputs. The function of the call performance process is to provide a summarized performance evaluation of the complete call based on all fields accessed during the call. Fig. 1 also illustrates the inputs and outputs of the component.

The call performance output class provides a measure of the call performance based on field performance interaction classes. Experienced caller output parameter assists in determining whether the caller is familiar with the IVR application as the caller has used the application previously. Self-service level illustrates a measure of the extent of the IVR application usage before the application ended the call, caller disconnects or transfer to service agent event occurs. Speech-enabled level output parameter illustrates 3 categories of performance; good, acceptable or investigate. This provides a measure of the number of fields the caller completed successfully using speech recognition. The caller disconnect transferred call performance output parameter identifies the field a caller disconnect or transfer event occurred utilizing the field classifier disconnect and transfer interaction classes.

III. SELECTION AND PREPROCESSING OF DATA

The field categorization process utilized a business intelligence solution that extracted and computed information such as recognition confidence values, duration values and call completion information. This information is stored within a database and is then manipulated utilizing specific rules to generate the data sets. Rules such as if no hang-up, transfer to agent, DTMF transfer, system error, no inputs, no matches or maximum speech timeouts occur, but the confidence level at the field is greater than 80%, the duration to complete the field is less than the average field duration and the field confirmation is true, the field performance interaction class would be computed as 'good', were followed.

The call performance classifier input data set values were calculated using the interaction classes computed based on

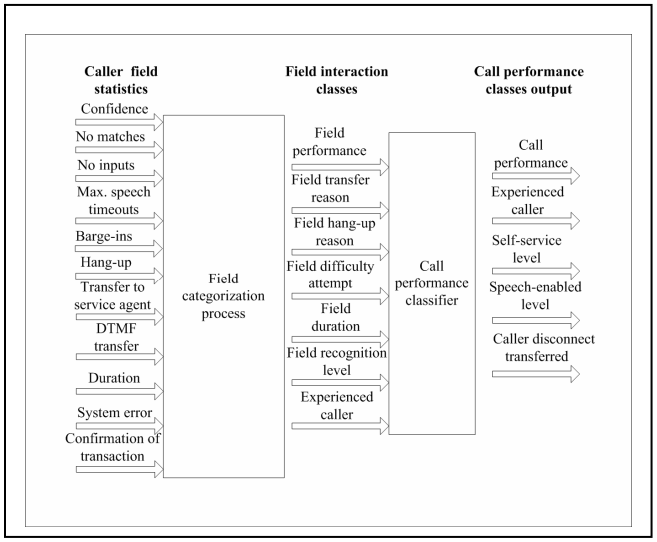

Figure 1. Call performance classification system.

the rules mentioned above. Similar rules such as when the field performance interaction class is good, acceptable, investigate or bad for all the fields, the call performance level would be good, acceptable, investigate or bad, respectively, were used to create the call performance classification output data set.

The experienced call output class used the experienced caller interaction class to generate an output value. If 2 or more of the experienced caller interaction classes is set to true, the experienced call output class would also be true.

The speech-enabled level, self-service level and caller disconnect transferred output parameters utilized the caller disconnect and transfer interaction classes to compute an output value. However, the speech enabled level output value calculation is also based on DTMF transfer information.

A binary notation has been used to present the interaction classes. The field performance interaction classes are presented by 4 digit binary words. For example, if the field performance is categorized as investigate; the binary notation will be '0010'. A similar 3 digit binary notation is employed for field difficulty attempt, field duration and field recognition level classifier inputs. Field transfer reason and field hang-up reason input information were represented by 2 digit binary words. However, the experienced caller interaction class is presented to the classifier using bit binary word. This data presentation method has also been employed in interpreting the call performance classification process output values.

In order to ensure that over-fitting and under-fitting were avoided, the data has been divided into training, validation and test sets. The training data set is used to train the algorithms to identify the general classification groups within the data. The validation data set is used to assess the classifier and the test data is used to confirm the classification capability of the developed models.

IV. Multi-layer Perceptron Neural Network Architectures

The MLP ANN architectures are possibly the most extensively employed ANNs in pattern classification [11]. Due to the non-linear capabilities of these networks, they are said to be excellent universal approximators that provide highly accurate solutions. As a result, these networks produce very practical tools for classification and inversion problems [12]. Detailed explanations of these ANN architectures can be found in [11].

The MLP neural network architectures employed consisted of only 1 hidden layer. Hyperbolic tangent hidden layer activation function is used due to its practical advantage of faster convergence during training [11]. Similarly, the logistic sigmoidal output-unit activation function is utilized as it is most appropriate for classification problems [11]

The MLP ANN implementation process involved the optimization of the network architectures. As a result, this entailed the identification of the correct number of hidden neurons that would yield the most accurate results.

Binary and real coded GAs that used normalized geometric ranking and tournament selection functions were employed to optimize the call performance ANN classifier architectures. The binary coded GAs utilized binary mutation and the real coded GAs used non-uniform mutation genetic operators. Arithmetic and simple cross over operators have been used within the binary and real coded GAs, respectively. Refer to [13] for further information about these GAs.

Populations of MLP ANN individuals were generated by the GAs. An error function that mapped the number of hidden nodes to the accuracy of the developed network has been used as the evaluation function for the GAs. The fitness of the individuals within a population is determined by calculating the accuracy of the ANNs when presented with validation and test data sets. The minimum value of these accuracies determined the fitness of the individual.

Since this is a classification implementation, a confusion matrix is employed to identify the number of true and false classifications. This is then utilized to calculate the true accuracy of the ANN classifiers, using the following equation:

$$Accuracy = \sqrt{\frac{TP \times TN}{(TP + FN) \times (FP + TN)}}, \qquad (1)$$

where
TP is the true positive (1 classified as a 1),
TN is the true negative (0 classified as a 0),
FN is the false negative (1 classified as a 0),
FP is the false positive (0 classified as a 1).

The GAs produced 25 generations of 10 ANN individuals within the population. The GAs were limited to produce ANN individuals with the number of hidden nodes between 5 and 100. Networks with hidden nodes greater than 100 were not developed due to the predictive capabilities or generalization capabilities reducing as the number of intermediate units increase. A major consideration when developing a suitable ANN for a classification application is to make a trade-off between convergence and generalization [14].

Binary and real coded GA solutions were compared in terms of computational efficiency and quality of the GA solution. Computational efficiency, in this context, is defined as the number of generations the GA utilized to converge to the most optimal number of hidden nodes. Quality of the GA solution is the verification that the number of hidden nodes returned by the GA is really the most optimal value.

Table I illustrates the results of the ANN implementation. It is evident, in terms of computational efficiency; the binary coded normalized geometric ranking selection GA outperformed all GA solutions considered in optimizing the MLP number of hidden nodes. This algorithm converged at a solution at the generation 1.

In order to determine the quality of the GA solutions, MLP classifiers were created containing the optimal number of hidden nodes returned by the algorithms.

As illustrated in Table I, the number of hidden nodes that resulted in the most accurate MLP ANN classifier is 10. This number of hidden nodes creates a network that performs accurately on both the unseen data sets. As a result, the classifier has good generalization capabilities.

Therefore, in terms of quality of GA solution, the real coded normalized geometric ranking selection GA is most suited in optimizing the MLP network architecture. However, this algorithm converged to this solution at generation 21 of 25.

TABLE I. MLP IMPLEMENTATION RESULTS

GA	Hidden nodes	Gen[a]	Accuracy (Validation)	Accuracy (Test)
Binary coded normalized geometric ranking selection	75	1	0.9891	0.9896
Binary coded tournament selection	47	16	0.989	0.9901
Real coded normalized geometric ranking selection	10	21	0.9904	0.9918
Real coded tournament selection	69	6	0.9888	0.9893

a. Gen represents Generation

V. FUZZY INFERENCE SYSTEM

FISs utilize fuzzy inference rules that enable the categorizing of data [15]. Fuzzy inference methods are algorithms that deduce results from the fuzzy inference rules and present inputs. Fuzzy inference methods are based on fuzzy logic. FISs are universal approximators [15].

The FIS utilized in the development of the call performance classifiers, employed subtractive clustering to generate the required membership functions and set of fuzzy inference rules. The objective of clustering is to locate "natural classes" in a set of given inputs such that similar inputs are grouped together in the same class [16].

The cluster radius indicates the range of influence of a cluster. A small cluster radius results in small clusters in the data and, therefore, many fuzzy rules. Large cluster radii yield few large clusters in the data and, hence, fewer fuzzy rules [17]. The cluster radius has been optimized by minimizing an error function that mapped the radius to the accuracy of the developed inference systems. This process has been performed on the validation data sets.

The optimization process followed entailed the construction of various inference systems with the cluster radius ranging from 0.01 to 1. Fig. 2 illustrates the results of the FIS cluster radius optimization. It is evident that cluster radii of 0.13 and 0.14 yield the same validation and test data set accuracies. These cluster radii also achieve the best accuracies, 92% and 91% on validation and test data sets, respectively.

VI. SUPPORT VECTOR MACHINES

The type of kernel function utilized within a SVM will affect the learning ability and generalization ability. Different kernel functions will construct different SVM classifiers. This research considers the linear, polynomial, Radial Basis Function (RBF) and sigmoid kernel functions. For detailed information on these kernel functions refer to [18].

SVM implementation process involved creating call performance classifiers that employed the kernel functions mentioned above. The validation and test data set accuracies of the resulting SVM classifiers were then compared to determine the kernel function most suitable for this application. As a result, this involved the selection of an appropriate kernel function that would result in classifiers with excellent generalization capabilities.

Good results were obtained that yielded call performance classifiers with excellent generalization capabilities. The kernel functions used created classifiers with accuracies larger than 95% on validation and test data sets. Table II illustrates the results of the SVM implementation. It is evident that the linear kernel function resulted in the most accurate call performance classifier.

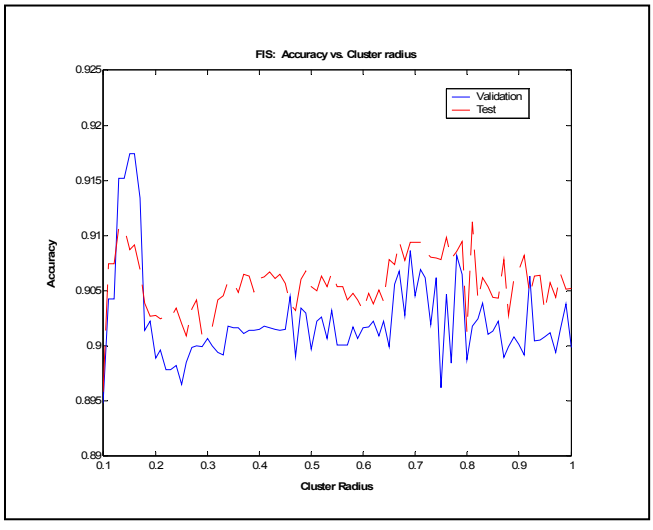

Figure 2. FIS Accuracy versus cluster radius.

TABLE II. SVM IMPLEMENTATION RESULTS

Kernel function	Accuracy (Validation)	Accuracy (Test)
Linear	0.9896	0.9919
Polynomial	0.9609	0.963
Radial Basis Function	0.9756	0.9736
Sigmoid	0.9713	0.9696

VII. COMPARISON OF THE VARIOUS METHODS CONSIDERED

The computational intelligent methods considered in this research produced highly accurate call performance classifiers. As illustrated in Table III, these classifiers achieved accuracy values larger than 90% with the SVM classifier achieving an accuracy value of 99.19% on test data set. However, the MLP ANN performed the best on validation data set. In order to determine if this accuracy value could be improved, an ensemble of the classifiers consisting of MLP ANN, FIS and SVM call performance classifiers has been created.

The outputs of classifiers were fed into a voting system that determined the final output of the ensemble

. If the majority of the classifiers within the ensemble categorized an output into a certain class, the voting system would classify the output of the ensemble as the class. If all of the models within the ensemble classified an output into different classes, the voting system would classify the output of the ensemble as undecided.

As shown in Table III, the ensemble of classifiers proved to be an accurate solution. The models yielded large accuracy values on both validation and test data sets. When comparing the call performance classifiers developed, the ensemble of classifiers has the best generalization capabilities and is the preferred solution.

In order to confirm the performance of the classifiers created, the sensitivity and specificity values were also compared. Sensitivity is defined as the probability that the classifier categorizes a set of interaction class inputs to the correct specific call performance classes. Specificity is defined as the probability that the classifier indicates that a set of interaction class inputs does not correctly belong to specific output classes. The former measure describes the effectiveness of the classifier at categorizing interaction classes correctly, while the latter characterizes the performance of the classifier at discarding the other interaction classes. Equation (2) and (3) below illustrate the sensitivity and specificity formula utilized, respectively.

$$Sensitivity = \frac{TP}{TP + FN}, \qquad (2)$$

$$Specificity = \frac{TN}{TN + FP}, \qquad (3)$$

where
TP is the true positive (1 classified as a 1),
TN is the true negative (0 classified as a 0),
FN is the false negative (1 classified as a 0),
FP is the false positive (0 classified as a 1).

These metrics confirmed that the ensemble of call performance classifiers outperform the MLP ANN, FIS and SVM classifiers. These classifiers produced large sensitivity and specificity values. As a result, the ensemble has a high positive as well as negative classification rate on both the validation and test data sets.

VIII. CONCLUSION

The system detailed in this paper consisted of field categorization and call performance classification processes. The field categorization component computed caller field statistics that were used to determine caller interaction classes. Call performance classifier used these interaction classes as inputs. The function of the call performance classifier is to provide a summarized performance evaluation of the complete call based on all fields accessed during the call.

Call performance classifiers were developed using MLP ANN, FIS, SVM and ensemble of classifiers.

The MLP ANN implementation process entailed determining the optimal number of hidden nodes. Binary and real coded GAs that employed normalized geometric ranking and tournament selection functions were used to compute the optimal number of hidden nodes. Computational efficiency and quality of solution were used to evaluate the performance of the GA solutions. These GA solutions yielded accurate MLP ANN classifiers. Binary coded normalized geometric ranking selection GA outperformed all GA solutions considered in terms of computational efficiency. This GA solution converged to the optimal number of hidden nodes at generation 1. However, the real coded normalized geometric ranking selection GA is most suited in optimizing the MLP network architecture, in regard to quality of solution. However, this algorithm converged to this solution at generation 21 of 25.

TABLE III. CALL PERFORMANCE CLASSIFIERS DEVELOPED

Method	Accuracy	Sensitivity	Specificity
Validation			
MLP ANN	0.9904	0.9845	0.9963
FIS	0.9152	0.9455	0.8858
SVM	0.9896	0.9844	0.9949
Ensemble	0.9917	0.9901	0.9933
Test			
MLP ANN	0.9918	0.9877	0.9960
FIS	0.9108	0.9333	0.8888
SVM	0.9919	0.9889	0.9949
Ensemble	0.9925	0.9908	0.9941

The SVM and FIS development involved the identification of the SVM kernel function and cluster radius that yielded the most accurate results, respectively. Linear kernel function resulted in the most accurate call performance SVM classifier. Custer radii of 0.13 and 0.14 yielded the most accurate classification systems.

Ensemble of call performance classifiers, consisting of the most accurate FIS, MLP ANN and SVM classifiers has also been developed. Accuracy, sensitivity and specificity performance metrics were computed and compared for the computational intelligent solutions. The MLP and ensemble of classifiers achieved high sensitivity and specificity. These classifiers were 99.04% and 99.17% accurate on unseen data, respectively. Therefore, the ensemble of call performance classifiers is the preferred computational intelligent method.

REFERENCES

[1] C. Nichols, The Move from IVR to Speech – Why This is the Right Time to Make the Move to Speech Applications in Customer-Facing Operations, Intervoice, 2006.

[2] VoiceXML 2.0/VoiceXML 2.1 Reference, last accessed: 5 July 2009 [Online]. Available: http://developer.voicegenie.com/voicexml2tagref.php?tag=st_field&display=standardtags.

[3] P. B. Patel and T. Marwala, "Forecasting closing price indices using neural networks", IEEE International Conference on Systems, Man and Cybernetics, 2006, pp. 2351-2356.

[4] J. M. Keller, M. Gray and J. Givens, "A fuzzy k-nearest neighbor algorithm", IEEE Transaction on System, Man and Cybernetics, vol. 15, no. 4, 1985, pp. 580-585.

[5] J. C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum, 1981.

[6] M. Russo, "FuGeNeSys – A fuzzy genetic neural system for fuzzy modeling", IEEE Transaction on Fuzzy Systems, vol. 6, no. 3, 1998, pp. 373-388.

[7] P. B. Patel and T. Marwala, "Neural Networks, Fuzzy Inference Systems and Adaptive-Neuro Fuzzy Inference Systems for Financial Decision Making", International Conference on Neural Information Processing, vol. 4234, no. 13, 2006, pp. 430-439.

[8] C-W. Hsu and C-J. Lin, "A comparison of methods for multi-class support vector machines", IEEE Transaction on Neural Networks, vol. 13, no. 2, 2002, pp. 415-425.

[9] B. Zadrozny, "Reducing multiclass to binary by coupling probability estimates", Neural Information Processing Systems Foundation, vol. 14, 2002, pp. 1041-1048.

[10] VoiceGenie Technologies Inc., VoiceGenie 7 Tools User's Guide, VoiceGenie Technologies Inc, 2005.

[11] C. M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.

[12] T. Nabney, Netlab: Algorithms for Pattern Recognition, Springer, 2002

[13] C. R. Houck, J. A. Joines, M. G. Kay, A genetic algorithm for function optimization: a Matlab implementation, NCSU-IE Technical Report, North Carolina State University, 1995.

[14] E. B. Baum, D. Haussler,"What size net gives valid generalization?", Neural Computation, vol. 1, 1989, pp. 81–90.

[15] F. Lotte, A. Lecuyer, F. Lamarche and B. Arnaldi, "Studying the use of Fuzzy Inference Systems for motor imagery classification.", IEEE transactions on Neural Systems and Rehabilitation Engineering, vol. 15, no. 2, 2007, pp. 322-324,.

[16] The Mathworks, Inc., Fuzzy Logic Toolbox User's Guide Version 2, The Mathworks, Inc, 1995.

[17] S. Chiu, "Fuzzy Model Identification Based on Cluster Estimation", Journal of Intelligent and Fuzzy Systems, vol. 2, no. 3, 1994, pp. 267-278.

[18] LIBSVM - A library for Support Vector Machines, last accessed: 4 July 2009. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm

Paper under review for publication in International Journal of Neural Systems (IJNS)

International Journal of Neural Systems

CALLER BEHAVIOUR CLASSIFICATION USING COMPUTATIONAL INTELLIGENT METHODS

PRETESH B PATEL
Faculty of Engineering and the Built Environment, University of Johannesburg,
P O Box 524, Auckland Park, 2006, Johannesburg, South Africa
E-mail:p.patel@uj.ac.za
www.uj.ac.za

TSHILIDZI MARWALA, *
Faculty of Engineering and the Built Environment, University of Johannesburg,
P O Box 524, Auckland Park, 2006, Johannesburg, South Africa
E-mail: tmarwala@uj.ac.za
www.tshilidzimarwala.com

A classification system that accurately categorizes caller interaction within Interactive Voice Response systems is essential in determining caller behaviour. Field and call performance classifiers for pay beneficiary application are developed. Exceptional results were achieved. Classifiers with accuracy values greater than 90% were developed. The preferred models for field 'Say amount', 'Say confirmation' and call performance classification are the ensemble of classifiers. However, the Multi-Layer Perceptron classifiers performed the best in field 'Say account' and 'Select beneficiary' classification.

Keywords: multi-layer perceptron, radial basis function, support vector machines, fuzzy inference system, interactive voice response, caller behaviour.

1. Introduction

Businesses may have the best products, lowest prices and most intelligent employees, but if potential customers perceive that the business does not have the capacity to complete the project, the companies will lose these customers. In the current economic condition, businesses are required to be aggressive in increasing the profile of their brand, establish a good reputation as well as presence in their market sector.

Although, businesses may spend precious capital into these ventures, there are more efficient and inexpensive ways to achieve the same results through the use of affordable latest technologies such as Interactive Voice Response (IVR) systems. A telephone is more than a phone. It is a major interaction point to the customer for the business. It is the "front door" of the company. The manner in which calls from customers are answered and the manner in which the calls are processed is an instant measure of the business efficiency and customer relations attitude.

By using the correct combination of new call handling tools, a small business can also project professionalism and competence from the first crucial customer interaction. Intelligent phone systems incorporate, accommodate and integrate a wide range of key business processes that have resulted in significant increase in IVR system implementations[1].

An IVR system is an automated telephony system that interacts with callers, gathers relevant information and routes calls to the appropriate destinations[2]. The inputs to the IVR system can be voice, Dual Tone Multi-Frequency (DTMF) keypad selection or a combination of the two. IVR systems can provide appropriate responses in the form of voice, fax, callback, e-mails and other media[2]. An IVR system solution may consist of telephony equipment, software applications, databases and supporting infrastructure.

* To whom all correspondence should be addressed.

However, there are many businesses that provide IVR systems to customers, which are poorly designed or implemented that result in caller frustration when utilizing the automated solutions[3]. An example of a cause of caller frustration would be a voice prompt that plays a detailed description of the options available. The descriptions are long, thus resulting in caller confusion as the caller cannot remember all the options presented. Another example is an IVR application that does not provide the caller sufficient time to respond to a prompt. As a result, these customers would probably end the call or request for a transfer to a Customer Service Agent (CSA).

The aim of this paper is to develop a call classification application, using computational intelligent methods, which could assist companies in quantifying caller behaviour within their IVR systems. It is anticipated that this application would be used in conjunction with other customer behaviour analysis techniques such as listening to recorded calls. As a result, this application should be used to confirm the IVR system performance in relation to customer interaction.

IVR applications are developed in Voice Extensible Markup Language (VXML). VXML applications are voice-based dialog scripts that consist of form or dialog elements. The form or dialog elements are used to group input and output sections together. A field element is used to obtain and interpret user input information. As a result, the form or dialog elements contain field elements[4].

The classification system developed consists of 2 components, the field and the call performance classification components. The field classification component consists of classifiers that categorize caller behaviour at a field within the IVR applications into specific interaction classes. A call performance classifier utilizes these interaction classes to evaluate the performance of the customer call in relation to caller behaviour.

As a result, the call performance classes can assist in determining trends of caller behaviour within the self service systems. For example, the caller behaviour classification application can identify calls that transferred or disconnected at the final step of the automated process as well as calls where the automated speech recognition performed poorly. Thereafter, analysts can listen to a sample of these calls and determine the reason for this. The caller behaviour classification system can also identify the field that resulted in the majority of the callers transferring to a CSA or caller disconnecting. The field interaction classes can elaborate on the reason for the caller behaviour experienced.

In order to develop such an application, the classification of data must be accurate. This paper compares field classifiers as well as call performance classifiers that were developed utilizing Artificial Neural Network (ANN), Fuzzy Inference System (FIS), Genetic Algorithms (GAs) and Support Vector Machine (SVM) techniques. Ensembles of classifiers were also considered.

Multi-Layer Perceptron (MLP) and Radial Basis Function (RBF) neural network architectures considered, are feed-forward structures whereby each unit receives inputs only from lower layer units. In the majority of implementations, the network consists of 2 layers of adaptive weights with full connectivity between inputs and hidden units as well as between hidden units and outputs[5].

The training of these networks is accomplished through backpropagation and a complex nonlinear optimization of the network hidden and output weights. At iterations, the error of the network is assessed and the derivative of this error is calculated with respect to each weight within the network.

The error function generally used in ANN computation is the squared difference between the actual and desired outputs. Optimization methods are then used to minimize the error function by altering the weights, initially in the output layer and then the hidden layer. Essentially, the error is backpropagated from the output of the network, through the output weights and to the hidden weights[5]. Detailed explanations on these ANN architectures can be found in Ref. 5.

GAs are known to be robust optimization procedures based on the mechanism of the natural evolution. GAs have the capability of locating a global optimum as these procedures do not use any derivative information and GAs search from multiple points.

In traditional GAs, binary representation has been used for chromosomes[6]. This results in an even discrete depiction of the real optimization problem. Within these binary-coded GAs, binary substrings representing each parameter with a desired precision are concatenated to form a chromosome. Therefore, a large number of variables in a real-world problem would result in

chromosomes encoded in long strings. Also, there is a discrepancy between the binary representation space and the actual problem space. For example, 2 points close to each other in the real space might be far away in the binary represented space.

In order to resolve these problems, floating-point representation of parameters as a chromosome is utilized[7]. In these real-coded GAs, a chromosome is coded as a finite-length string of the real numbers corresponding to the real-world problem variables. Real-coded GAs are robust, accurate as well as efficient because they are conceptually closest to the real-world problem and moreover, the string length reduces to the number of variables. It has been reported that the real-coded GAs outperformed binary-coded GAs in many design problems[8].

This research will determine if this is true in relation to the field and call performance classification application.

Support Vector Machines (SVMs) perform well for modeling challenging high-dimensional data. SVMs have been used successfully in text mining[9], image mining[10], bioinformatics[11] and information fusion[12]. SVM performance has been demonstrated to be superior to the performance of decision trees, neural networks and Bayesian techniques[9, 11, 12].

A fundamental method in data mining and pattern recognition is clustering of data. Fuzzy clustering involves the natural grouping of data in a large data set and provides a basis for constructing rule-based fuzzy model[13]. Fuzzy c-means, mountain clustering, subtractive clustering and entropy-based fuzzy clustering are among the fuzzy clustering algorithms used. In this paper we are interested in subtractive clustering.

The classification of data into various classes has been an important research area for many years. ANNs have been applied to pattern classification[14]. Research has also been conducted on fuzzy classification. This resulted in many algorithms, such as fuzzy K-nearest neighbour[15] and fuzzy c-means[16], being applied to classification problems. Fuzzy systems constructed using genetic algorithms have been utilized[17]. Fuzzy neural networks have also been employed in pattern classification applications[18].

Support Vector Machines have been applied to multi-category classification problems[19]. These classification tasks have also been implemented by combining multiple simpler specialized classifiers[20].

The sections to follow examine the caller behaviour classification system as well as its implementation methodology. The paper ends with the comparison of the various classifiers developed and the selection of the superior networks.

2. The developed system

This section examines the caller behaviour classification system illustrated in Fig. 1. As the developed system is to be used to identify trends of caller behaviour within the IVR VXML applications, the field classifiers are trained based on data extracted from IVR log event files. These files are generated by the IVR platform as specific events occur during a call to the system. Events such as call begin, form enter, form select, automatic speech recognition events, transfer events and call end events are written to the logs[21].

Table 1 shows the inputs and outputs of the field classification component. These specific inputs have been selected to characterize the caller experience at a field within a VXML application. The outputs of the classifiers summarize the caller field behaviour through the use of interaction classes.

The confidence input illustrates the IVR speech recognition probability. The value is a percentage. The larger the percentage, the greater the probability the system interpreted the caller successfully.

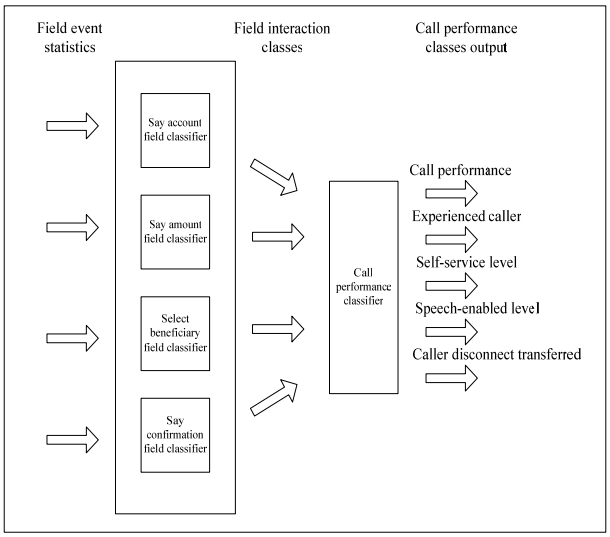

Fig. 1. The caller behaviour classification system

Table 1. The inputs and outputs of the field classifier.

Inputs	Outputs	Output interaction class
Confidence	Field performance	Good, acceptable, investigate, bad
No matches	Field transfer reason	Unknown, difficulty
No inputs	Field hang-up reason	Unknown, difficulty
Max speech timeouts	Field difficulty attempt	Attempt 1, attempt 2, attempt 3
Barge-ins	Field duration	High, medium, low
Hang-up	Field recognition level	High, medium, low
Transfer to Service Agent	Experienced caller	True, false
DTMF transfer		
Duration		
System error		
Confirmation of transaction		

A caller may answer a question the VXML application prompts with a response the application does not accommodate. These events are represented by the no match inputs. In general, most VXML applications accommodate 3 no match events per field. On a third no match event, the call is transferred to a DTMF field. If the caller fails to complete the DTMF field successfully on attempt 1, the call is transferred to a CSA. The same process is used for the third no input and maximum speech timeout events. The no match field classifier inputs assist in identifying callers that misunderstood the VXML prompt as well as unique responses that the VXML application can use to improve field recognition coverage.

In response to a prompt, a caller may remain silent. These events are represented by the no input parameters. VXML applications normally accommodate 3 no input events on each field. These input parameters assist in identifying callers that were confused when prompted with the automated application question. As a result, the caller remained silent.

Callers may reply to VXML applications by talking beyond the allocated timeout period of the field. These events are represented by the maximum speech timeout input parameters of the field classifiers[22]. Maximum speech timeout input parameters are important as they assist in determining whether the timeout periods are adequate for callers to complete their responses.

Barge-in input parameters illustrate whether or not a caller interrupted the application while the automated question prompt played. Caller disconnects, transfer to DTMF, transfers to Customer Service Agents (CSAs) and system errors are represented by the hang-up, DTMF transfer, transfer to service agent and system error input parameters, respectively. These inputs can also assist in determining the level of difficulty the caller experienced in the field. The duration input parameter illustrates the time the caller spent completing the field. Confirmation of transaction represents whether or not the caller verified the application recognition as being true.

The field performance output interaction class of the classifier will illustrate whether the caller behaviour is good, acceptable, investigate or bad. The field transfer reason and field hang-up reason interaction classes attempt to identify the motivation for the transfer to CSA or caller disconnect, respectively. Field difficulty attempt interaction class computes the number of difficulty events that occurred during the field interaction. The field duration as well as field recognition level classes illustrate 3 categories of performance; low, medium and high. As a result, these output parameters will assist in characterizing the caller experience at a VXML field.

Experienced caller output parameter categorizes whether or not the caller is a regular user of the application and is therefore comfortable with the application call flow. In determining the number of experienced callers, the contact center can determine the usage of the application.

The function of the call performance classifier is to provide a summarized performance evaluation of the complete call based on all fields accessed during the call. Fig.1 illustrates the inputs and outputs of the component. The call performance output class provides a measure of the call performance based on field performance interaction classes. Experienced caller output parameter assists in determining whether the caller is familiar with the IVR application as the caller has used the application previously. Self-service level illustrates a measure of the extent of the IVR application usage before the application ended the call, caller disconnects or transfer to CSA event occurs. Speech-enabled level output parameter illustrates 3 categories of

performance; good, acceptable or investigate. This provides a measure of the number of fields the caller completed successfully using speech recognition. The caller disconnect transferred call performance output parameter identifies the field a caller disconnect or transfer event occurred utilizing the field classifier disconnect and transfer interaction classes.

3. Selection and preprocessing of data

This section examines the data used in the creation of the field and call performance classification components. The data utilized in implementing the field classification component classifiers has been based on data extracted from IVR log event files. A business intelligence solution that involved Extract, Transform and Load (ETL) processes has been developed to extract and compute information such as recognition confidence values, duration values and call completion information. This information is stored within a database and is then manipulated utilizing specific rules to generate the data sets. Rules such as if no hang-up, transfer to CSA, DTMF transfer, system error, no inputs, no matches or maximum speech timeouts occur, but the confidence level at the field is greater than 80%, the duration to complete the field is less than the average field duration and the field confirmation is true, the field performance interaction class would be computed as 'good', were followed.

The call performance classification component input data set values were calculated using the interaction classes computed based on the rules mentioned above. Similar rules such as when all the field classifiers compute the field performance interaction class as good, acceptable, investigate or bad the call performance level would be good, acceptable, investigate or bad, respectively, were used to create the call performance classification output data set.

The call performance output class provides a measure of the call performance based on field performance interaction classes. The experienced call output class used the field classifier experienced caller outputs to generate an output value. If 2 or more of the field classifiers experienced caller output is set to true, the experienced call output class would also be true.

The self-service level, speech enabled level and caller disconnect transferred output parameters utilized the field classifier disconnect and transfer interaction classes to compute an output value. However, the speech enabled level output value calculation is also based on DTMF transfer information.

No match, no input and maximum speech timeout information has been presented to the field classifiers, using a binary notation. These inputs are presented by 3 digit binary words. For example, if a no match 1 and a no match 2 occur at a field, the binary notation will be '011'. A similar binary notation is employed for the no input and maximum speech timeout classifier inputs. The barge-in, hang-up, transfer to CSA, DTMF transfer, system error and confirmation of transaction input information were represented by bit binary words. A similar binary notation scheme has also been utilized to interpret and present the interaction classes to the call performance component. This data presentation method has also been employed in interpreting the call performance classification component output values.

The confidence and duration input parameters of the field classifiers were preconditioned by normalizing the data. Normalizing the data entails manipulating the data sets such that the values within the sets are between 0 and 1. The field classification component classifiers developed were trained utilizing the normalized data sets. Due to the binary word representation utilized to present the remaining field classifier inputs and the interaction classes computed, normalization of these values is not necessary.

Normalization is accomplished by acquiring the minimum and maximum values within the data sets. These values are then utilized to compute the normalized values.

The research conducted entailed the creation of 'Say account', 'Say amount', 'Select beneficiary' and 'Say confirmation' field classifiers. Caller behaviour per field is unique. For example, at a 'Say confirmation' field the caller is required to say 'yes' or 'no'. However, the caller is requested to say the currency value at the 'Say amount' field. As a result, the duration to complete the VXML application field is much shorter at the confirmation field. Therefore, each classifier is trained with data relevant to the field.

In order to ensure that over-fitting and under-fitting were avoided, the data has been divided into 3 sets. The data is divided into training, validation and test sets. The training data set is used to train the algorithms to identify the general classification groups within the data. The validation data set is used to assess the classifier and the test data is used to confirm the

classification capability of the developed models. This method is employed in the implementation of the field and call performance classification components.

4. Artificial neural networks

MLP and the RBF ANN architectures were utilized in the development of both the field and call performance classification components.

The MLP and RBF ANN architectures are possibly the most extensively employed ANNs in pattern classification[23]. Due to the non-linear capabilities of these computational intelligent methods, they are said to be excellent universal approximators that provide highly accurate solutions. As a result, these networks produce very practical tools for classification and inversion problems[5].

It has been stated that a network with 1 hidden layer, provided with sufficient data, can be used to model any function[24]. As a result, the MLP and RBF ANN architectures employed consisted of only 1 hidden layer.

The MLP network hidden layer consists of non-linear activation functions. The choice of the activation function is mainly dependant on the application of the ANN[5]. However, it has been found that the hyperbolic tangent activation function offers a practical advantage of giving rise to faster convergence during training[23]. As a result, this function has been utilized within the MLP networks.

The MLP network output layer also consists of activation functions. There are 3 major forms of the function that should be considered. These are the linear, logistic sigmoidal and softmax activation functions[23]. It has been stated that the appropriate selection of the output-unit activation function for a classification problem is the logistic sigmoidal function[23]. As a result, this function has been employed within the output layer of the MLP network.

The RBF network that has been developed contained a Gaussian activation function within its hidden layer and a linear activation function within its output layer.

The MLP and RBF ANN implementation process involved the optimization of the network architectures. As a result, this entailed the identification of the correct number of hidden neurons that would yield the most accurate results.

Binary and real coded GAs were employed to optimize the field and call performance ANN classifier architectures. Populations of MLP and RBF ANN individuals were generated by the GAs.

GAs are known to be robust optimization procedures based on the mechanism of the natural evolution. GAs have the capability of locating a global optimum as these procedures search from multiple points. In traditional GAs, binary representation has been used for chromosomes. Floating-point representation, real-coded GAs, of parameters as a chromosome has also been used[6].

An error function that mapped the number of hidden nodes to the accuracy of the developed network is used as the evaluation function for the GAs. The fitness of the individuals within a population is determined by calculating the accuracy of the ANNs when presented with validation and test data sets. The minimum value of these accuracies determined the fitness of the individual. The ANNs were trained using the training data set. The outputs of the ANNs were interpreted by utilizing a classification threshold value of 0.5. This implies that if the classifier outputs a value less than 0.5, the output will be regarded as a 0. Similarly, if the output value is larger than or equal to 0.5, the output will be interpreted as a 1.

Since this is a classification implementation, the accuracy of the networks developed can no longer be calculated utilizing the sum of square error of the difference between the target and the network output values. Instead, a confusion matrix is employed to identify the number of true and false classifications that are generated by the ANN developed. This is then utilized to calculate the true accuracy of the ANN classifiers, using the following equation:

$$Accuracy = \sqrt{\frac{TP \times TN}{(TP + FN) \times (FP + TN)}}, \qquad (1)$$

where

TP is the true positive (1 classified as a 1),
TN is the true negative (0 classified as a 0),
FN is the false negative (1 classified as a 0),
FP is the false positive (0 classified as a 1).

The GAs produced 25 generations of 10 ANN individuals within the population. The GAs were limited to produce ANN individuals with the number of hidden nodes between 5 and 100. Due to the predictive capabilities or generalization capabilities reducing as the number of intermediate units increase, ANNs with

hidden nodes greater than 100 were not considered. More hidden nodes increases the dimensionality of the function being fitted, enabling easier training which results from higher training capacity. However, this detrimentally affects the generalization capabilities of the network. A major consideration when developing a suitable ANN for a classification application is to make a trade-off between convergence and generalization[25].

In order to produce successive generations, selection of individuals is important in a GA. The selection function determines which of the individuals will survive and move on to the next generation. A probabilistic selection is performed based upon the fitness of an individual such that the superior individuals have a higher chance of being selected. There are several schemes for the selection process. Roulette wheel selection and its extensions, scaling techniques, tournament, normal geometric ranking and elitist models are examples of selection functions used[6]. The selection approach assigns a probability of selection to each individuals based on its fitness value. GA solutions that used normalized geometric ranking and tournament selection functions were compared in this research.

Crossover and mutation operators provide basic search mechanism of the GA. Crossover operators transform 2 individuals into 2 new individuals, while mutation operators alter 1 individual to produce a single solution. In this research, binary coded GAs utilized binary mutation and the real coded GAs used non-uniform mutation genetic operators. Arithmetic and simple cross over operators have been used within the binary and real coded GAs, respectively. For further information on these crossover and mutation operators, refer to Ref. 6.

Binary and real coded GAs that used normalized geometric ranking as well as tournament selection functions were compared in terms of computational efficiency and quality of the GA solution.

Computational efficiency, in this context, is defined as the number of generations the GA utilized to converge to the most optimal number of hidden nodes.

Table 2 illustrates the results of the MLP and RBF ANN hidden nodes optimization using GAs. During the field 'Say account MLP classifier optimization, it is evident that the binary coded GAs and the real coded GAs converged to a solution by generation 22 and 9, respectively. As a result, in regards to the field 'Say account' MLP classifier, the real coded GAs outperformed the binary coded GAs. However, during the field 'Say amount' MLP classifier optimization, the GAs using tournament selection function outperformed the GAs that employed normalized ranking selection function. The real coded GA that used normalized ranking selection function converged the fastest to an optimal number of field 'Select beneficiary' and 'Say confirmation' MLP classifier hidden nodes solution. However, the GA that employed binary coded tournament selection function also yielded the optimal field 'Say confirmation' MLP classifier number of hidden nodes, utilizing the same number of generations.

In terms of computational efficiency, the binary coded GA and the real coded GA that used normalized geometric ranking selection function outperformed all the GA solutions considered in optimizing the field 'Say account' and 'Say amount' RBF classifier architectures, respectively. However, the real coded GA that employed tournament selection function performed the best in optimizing the field 'Select beneficiary' and 'Say confirmation' RBF classifier architectures.

During the call performance classification RBF ANN architecture optimization, it has been determined that the binary and real coded GAs converged to a solution at generation 1. However, the call performance classification MLP ANN architecture optimization achieved different results. Table 2 illustrates the results of these network implementations.

As a result, in terms of computational efficiency, all GA solutions considered in developing the call performance classification component to optimize the RBF architecture performed equally well. However, when optimizing the MLP number of hidden nodes in relation to computational efficiency, the binary coded GA that employed normalized geometric ranking selection outperformed all GA solutions considered. This algorithm converged to a solution at generation 1.

Quality of the GA solution is the confirmation that the optimal number of hidden nodes computed by the GA is truly the most suitable value. In order to determine the quality of the GA solutions, field MLP and RBF as well as call performance classifiers were created containing the optimal number of hidden nodes calculated by the algorithms.

The real coded GA utilizing tournament selection function is best suited in optimizing the field classifier architectures as this algorithm computed the optimal number of hidden nodes that produced the most accurate classifiers. This is true for the field 'Say account', 'Say

amount' and 'Say confirmation' MLP classifiers. This is also valid for the field 'Say account', 'Select beneficiary' and 'Say confirmation' RBF classifiers. However, the binary coded GA that used the tournament selection function returned the optimal number of hidden nodes that resulted in the most accurate field 'Select beneficiary' MLP classifier. This GA also computed the same optimal number of 'Say confirmation' MLP hidden nodes as the real coded GA that employed tournament selection function, which produced the best field 'Say confirmation' classifier. Similarly the real coded GA that used normalized geometric ranking selection function yielded the same number of field 'Say amount' MLP hidden nodes as the real coded GA that utilized tournament selection function. However, the binary coded GA that used normalized geometric ranking selection function returned the optimal number of hidden nodes that produced the most accurate field 'Say amount' RBF classifier.

It is evident that, during the call performance RBF ANN number of hidden nodes optimization, the binary coded and real coded GAs computed different optimal solutions. The binary coded GAs yielded 40 number of hidden nodes and the real coded GAs calculated 25 number of hidden nodes as optimal. During this

Table 2. The results of the ANN implementation of the field and call performance classifiers. In the table below, the Gen column represents the number of generations the GA used to compute the optimal solution.

GA	Field	MLP				RBF			
		Hidden nodes	Gen	Accuracy Validation	Accuracy Test	Hidden nodes	Gen	Accuracy Validation	Accuracy Test
Field classification component									
Binary coded normalized geometric ranking selection	'Say account'	7	22	0.9833	0.9507	66	13	0.9478	0.9363
	'Say amount'	96	21	0.9095	0.9694	94	16	0.9143	0.9603
	'Select beneficiary'	9	22	0.9537	0.9654	61	18	0.9401	0.9529
	'Say confirmation'	34	19	0.9556	0.9018	73	25	0.9641	0.9033
Binary coded tournament selection	'Say account'	13	22	0.9782	0.9431	96	18	0.9492	0.9358
	'Say amount'	95	3	0.9092	0.9699	53	5	0.9123	0.9600
	'Select beneficiary'	6	7	0.9628	0.9722	83	16	0.9426	0.9534
	'Say confirmation'	50	6	0.9557	0.9021	77	16	0.9517	0.9025
Real coded normalized geometric ranking selection	'Say account'	8	9	0.9844	0.9513	70	24	0.9493	0.9371
	'Say amount'	9	18	0.9199	0.9796	31	4	0.9125	0.9595
	'Select beneficiary'	40	6	0.9499	0.9635	26	19	0.9391	0.9552
	'Say confirmation'	54	6	0.9567	0.9020	88	24	0.9632	0.9041
Real coded tournament selection	'Say account'	8	9	0.9844	0.9513	94	15	0.9499	0.9376
	'Say amount'	9	1	0.9199	0.9796	68	9	0.9117	0.9588
	'Select beneficiary'	44	23	0.9494	0.9670	68	8	0.9443	0.9573
	'Say confirmation'	50	21	0.9557	0.9021	60	12	0.9588	0.9048
Call performance classification component									
Binary coded normalized geometric ranking selection	-	75	1	0.9891	0.9896	40	1	0.8281	0.8186
Binary coded tournament selection	-	47	16	0.9890	0.9901	40	1	0.8281	0.8186
Real coded normalized geometric ranking selection	-	10	21	0.9904	0.9918	25	1	0.8281	0.8186
Binary coded tournament selection	-	69	6	0.9888	0.9893	25	1	0.8281	0.8186

optimization process, it has been determined that both numbers of hidden nodes achieve the same validation and test data set accuracies. As a result, in relation to quality of the GA solution, all GA solutions perform equally when optimizing the call performance RBF classifier architectures.

Table 2 shows the number of hidden nodes that resulted in the most accurate call performance MLP ANN classifier is 10. This number of hidden nodes creates a network that performs accurately on both the validation and test data sets. As a result, this classifier has good generalization capabilities.

Therefore, in terms of quality of GA solution, the real coded normalized geometric ranking selection GA is most suited in optimizing the call performance MLP network architecture. However, this algorithm converged to this solution at generation 21 of 25.

In order to improvement the classification accuracy of the call performance RBF ANN implementations, investigations were conducted to optimize the classification threshold of the RBF ANN. These experiments resulted in a significant improvement in performance on both the validation and test data sets.

When utilizing a classification threshold of 0.5, the field MLP ANN and call performance classifier implementations achieved accuracy values larger than 85%. Similarly, the field RBF ANN classifiers yielded accuracy values greater than 85%. As a result, the classification threshold of 0.5 is appropriate for the field and call performance MLP ANN classifiers as well as field RBF ANN classifier.

Classification threshold has been optimized by minimizing an error function that mapped the classification thresholds to the sum of the sensitivity and specificity of the developed call performance classifiers.

In this research, sensitivity is defined as the probability that the classifier categorizes a set of inputs to the correct specific interaction or call performance classes. Specificity is defined as the probability that the classifier indicates that a set of inputs does not correctly belong to specific interaction or call performance classes. The former measure describes the effectiveness of the classifier at categorizing output classes correctly, while the latter characterizes the performance of the classifier at discarding the other output classes.

This optimization process involved varying the classification threshold from 0.1 to 0.7 in iterations of 0.01 for both validation and test data sets. During this process, the optimized hidden nodes of 25 have been used. For each of the threshold values the accuracy, sensitivity and specificity of the call performance RBF ANN are calculated. The accuracy of the classifier is calculated using, Eq. (1) above. The sensitivity and specificity of the RBF ANN is calculated using Eq. (2) and Eq. (3) below, respectively.

$$Sensitivity = \frac{TP}{TP + FN} \tag{2}$$

$$. \; Specificity = \frac{TN}{TN + FP} \tag{3}$$

where

TP is the true positive (1 classified as a 1),
TN is the true negative (0 classified as a 0),
FN is the false negative (1 classified as a 0),
FP is the false positive (0 classified as a 1).

The optimization criteria employed in this investigation required the identification of the threshold value that yielded the maximum of the sum of sensitivity and specificity. This minimizes the mean of the error rate for positive classifications and the error rate for negative classifications. This optimization criteria is equivalent to determining the point on the Receiver Operating Characteristic (ROC) where the tangent has a slope of one[26]. It is also equivalent to maximizing the Youden's index or the true skill statistic.

Table 3 illustrates the call performance RBF ANN implementation threshold values that resulted in the largest sum of sensitivity and specificity value for the validation as well as test data sets. As illustrated in the table, the optimal threshold value of 0.65 is an appropriate value for this classification problem. As a result, the optimal RBF call performance classifier achieved an accuracy of 91% and 90% on validation and test data sets, respectively. This demonstrates that this classifier has good generalization capabilities. However, the call performance MLP ANN with 10 number of hidden nodes is approximately 9% more accurate than the RBF ANN classifier.

Table 3. The RBF ANN threshold optimization results. In the table below, sum refers to the addition of sensitivity and specificity.

Threshold	Accuracy	Sensitivity	Specificity	Sum
Validation				
0.10-0.11	0.6583	0.4408	0.9831	1.4240
0.12-0.18	0.6869	0.4831	0.9765	1.4596
0.19	0.7170	0.5313	0.9678	1.4990
0.20-0.31	0.7530	0.5914	0.9586	1.5500
0.32-0.35	0.7853	0.6535	0.9436	1.5971
0.36-0.57	0.8281	0.7385	0.9285	1.6670
0.58-0.64	0.8394	0.7838	0.8990	1.6827
0.65-0.70	0.9068	0.9784	0.8404	1.8188
Test				
0.10-0.11	0.6507	0.4328	0.9783	1.4111
0.12-0.18	0.6789	0.4741	0.9721	1.4462
0.19	0.7097	0.5223	0.9643	1.4866
0.20-0.31	0.7469	0.5833	0.9563	1.5396
0.32-0.35	0.7795	0.6452	0.9419	1.5870
0.36-0.57	0.8186	0.7238	0.9258	1.6496
0.58-0.64	0.8306	0.7688	0.8973	1.6661
0.65-0.70	0.9019	0.9677	0.8406	1.8083

5. Fuzzy Inference Systems

Fuzzy Inference Systems (FISs) were also considered in the development of both the field and call performance classification components.

FISs are computational intelligent procedures that employ fuzzy logic to formulate a mapping from a given input to an output[27]. These mappings provide a foundation that is used to make decisions. FISs have been successfully applied in fields such as automatic control, data classification, decision analysis and expert systems. Due to its multidisciplinary nature, these FISs are also known as fuzzy-rule-based systems, fuzzy expert systems, fuzzy modeling, fuzzy logic controllers as well as fuzzy systems[27].

Categorizing of data within FISs is accomplished through the use of fuzzy inference rules[28]. Each rule comprises of a premise that is described by a fuzzy proposition and a consequence, which can be a fuzzy conclusion. Fuzzy inference methods are algorithms that compute results based on the fuzzy inference rules and presented inputs. Fuzzy inference methods are based on fuzzy logic. A FIS comprises of fuzzification, inference and defuzzification processes. Fuzzification process is a mapping from the presented input to the fuzzy sets defined in the corresponding universe. Inference process involves a decision making logic, which employs the fuzzy inference rules to determine fuzzy outputs corresponding to fuzzified inputs. Defuzzification yield nonfuzzy outputs[27]. FISs are also known as universal approximators[28].

Clustering of numerical data establishes the basis of many classification and system modeling applications. The primary intention of clustering is to locate natural groupings in a set of presented inputs with the purpose of congregated similar inputs in the same class[27]. Due to the utilization of data clustering to compute fuzzy inference rules, the resultant rules are specifically tailored to the data. As a result, this is an advantage when compared to a FIS developed without clustering[27].

The fuzzy c-means and subtractive data clustering techniques are the 2 most popular methods used[27]. Fuzzy c-means technique entails each data point belonging to a cluster to some degree, which is specified by a membership grade[27]. Fuzzy c-means clustering technique has the ability to group data points that populate a multidimensional space into a specific number of unique clusters. This data clustering method requires 2 predefined clusters that indicate the mean location of each cluster[27]. Every data point is assigned a membership grade for each cluster. The cluster centers and the membership grades for each data point are updated iteratively. As a result, this process relocates the cluster centers to the correct co-ordinates within the data set. The iterative process entails minimizing a function that represents the distance from any given data point to a cluster center weighted by the membership grade of that data point[27].

Subtractive data clustering technique is a modified form of the Mountain Method for cluster estimation[29]. In this data clustering method, each data point is considered as a potential cluster center and defines a measure of the potential of a data point[30]. The measure of potential for a given point is a function of its distances to all other data points. As a result, a point with many neighbouring points will have a high potential value. Once the potential of every data point has been computed, the point with the largest potential value is selected as the first cluster center. Thereafter, in order to determine the next cluster and its center, all the data points in the vicinity of the first cluster center that is determined by a radius of influence or cluster radius, is removed. This process is iterated until all the input data are within a cluster radius of a cluster center[27]. Specifying a small cluster radius will usually yield many clusters in the data. However, specifying a large cluster radius will result in few cluster centers in the data[27].

The quality of the fuzzy c-means method depends strongly on the choice of the number of centers and the

initial cluster positions[30]. This method is also known to possess the "curse of dimensionality"[30]. This implies that the number of rules increases exponentially as the input data increases in size. As a result, due to these problems, the FISs utilized in the development of the field and call performance classifiers, used subtractive clustering to generate the required membership functions and set of fuzzy inference rules.

The development of the field and call performance FIS classifiers involved the optimization of the cluster radius used within these components. The cluster radius has been optimized by minimizing an error function that mapped the radius to the accuracy of the developed inference systems. This process used the validation data sets.

The optimization process followed entailed the construction of various field and call performance inference systems with the cluster radius ranging from 0.01 to 1. During the cluster radius optimization, classification threshold of 0.5 has been employed. Thereafter, when the optimal cluster radii have been identified, the classification threshold is optimized.

Table 4 illustrates the cluster radii that resulted in the most accurate field classifiers. As shown in the table, field 'Say account' FIS classifier proved to be the most accurate, yielding an accuracy of 78.00% on validation data set. However, the field 'Say amount' FIS classifier is the least accurate, producing an accuracy of 63.11% on validation data set.

Table 4. The Results of FIS cluster radius optimization.

Radius	Field	Accuracy Validation	Accuracy Test
Field classification component			
0.16	'Say account'	0.7800	0.8723
0.26	'Say amount'	0.6311	0.9566
0.40	'Select beneficiary'	0.7288	0.9339
0.78	'Say confirmation'	0.7074	0.8674
Call performance classification component			
0.13	-	0.9152	0.9108
0.14	-	0.9152	0.9108

Table 4 also illustrates the results of the call performance FIS classifier cluster radius optimization. It has been determined that cluster radii of 0.13 and 0.14 achieved the same validation and test data set accuracies. These cluster radii also achieve the best accuracies of 92% and 91% on validation and test data sets, respectively.

In order to attempt to improve the accuracy of the FIS field and call performance classifiers, the classification threshold is optimized. Similar to the ANN threshold optimization procedure followed previously, the classification threshold is optimized by minimizing an error function that mapped the classification thresholds to the sum of the sensitivity and specificity of the developed FIS classifiers. The process is performed on the validation and test data sets.

This classification threshold optimization process entailed varying the classification threshold from 0.1 to 0.7 in iterations of 0.01. The procedure used the optimized field FIS classifier cluster radii illustrated in Table 3. However, the process employed to optimize the call performance FIS classifier classification threshold used a cluster radius of 0.14. For each of the threshold values the accuracy, the sensitivity and specificity of the FIS is calculated using, Eq. (1), Eq. (2) and Eq. (3) above, respectively.

Table 5 illustrates the threshold values that resulted in the largest sum of sensitivity and specificity value for the validation and test data sets. It is evident that by identifying suitable thresholds values, the validation data set accuracy of the field classifiers has improved. The field 'Say amount' FIS classifier has become the most accurate, producing an accuracy of 82.54% on test data. The least accurate is the field 'Select beneficiary' FIS classifier, achieving an accuracy of 77.82% on test data.

It is also evident that the original threshold value of 0.5 is an appropriate value for the call performance classification problem. As a result, the optimal FIS call performance classifier achieved an accuracy of 92% and 91% on validation and test data sets, respectively. Due to these results, it can be concluded that this classifier has good generalization capabilities.

6. Support vector machines

During the development of the field and call performance components, SVM classifiers were also implemented.

Table 5. The FIS threshold optimization results.

Radius	Threshold	Field	Validation				Test			
			Accuracy	Sensitivity	Specificity	Sum	Accuracy	Sensitivity	Specificity	Sum
Field classification component										
0.16	0.16	'Say account'	0.8068	0.7200	0.9943	1.7143	0.8077	0.6615	0.9863	1.6478
0.26	0.15	'Say amount'	0.8265	0.7022	0.9860	1.6882	0.8254	0.6847	0.9951	1.6798
0.40	0.11	'Select beneficiary'	0.7843	0.6217	0.9900	1.6117	0.7782	0.6273	0.9908	1.6181
0.78	0.21	'Say confirmation'	0.7951	0.7138	0.9958	1.7096	0.7947	0.6576	0.9604	1.6180
Call performance classification component										
	0.10-0.24	-	0.6846	0.4853	0.9658	1.4511	0.6846	0.4826	0.9710	1.4536
	0.25-0.33		0.6849	0.4857	0.9658	1.4515	0.6847	0.4828	0.9710	1.4538
	0.34-0.49	-	0.6827	0.4838	0.9633	1.4471	0.6831	0.4814	0.9695	1.4509
0.14	0.5	-	0.9152	0.9455	0.8858	1.8313	0.9108	0.9333	0.8888	1.8221
	0.51-0.60	-	0.9148	0.9452	0.8853	1.8305	0.9106	0.9331	0.8886	1.8217
	0.61-0.66	-	0.9147	0.9452	0.8852	1.8304	0.9106	0.9331	0.8886	1.8217
	0.67-0.70	-	0.9145	0.9458	0.8842	1.8300	0.9099	0.9333	0.8871	1.8204

SVM is a reputable computational intelligent method for resolving classification problems. Support Vector Machines (SVMs) have many advantages in solving small sample size, nonlinear and high dimensional pattern recognition problems[31]. SVM utilizes support vector (SV) kernel functions to map the data in the input space to a higher dimensional feature space where the problem can be processed in a linear form[31]. As a result the kernel function is a key technology of SVM. The type of kernel function will affect the classifier learning and generalization capabilities. Different kernel functions will construct different SVM classifiers.

This research considers the linear, polynomial, Radial Basis Function (RBF) and sigmoid kernel functions.

When the number of training instances is less than the number of features within the data, the linear kernel function is most appropriate[32]. However, the RBF kernel function has the ability to accommodate non-linear relationships between input instances and output classes. The sigmoid kernel function behaves similar to the RBF kernel functions for certain parameters. The RBF kernel function has less hyperparameters than the polynomial kernel function[32]. For detailed information on these kernel functions refer to Ref. 32.

SVM implementation process involved creating field and call performance classifiers that employed the kernel functions mentioned above. The validation and test data set performance metrics of the resulting SVM classifiers were then compared to determine the kernel function most suitable for this application. As a result, this involved the selection of an appropriate kernel function that would result in classifiers with excellent generalization capabilities.

Utilizing the training data set, these SVM classifiers were trained. Thereafter, the validation and test data set were presented to the models. The accuracy of the developed classifiers has been calculated for the validation and test data sets using Eq. (1). SVMs that resulted in the largest accuracy value, when presented with the validation and test data sets, were analyzed.

Exceptional results were obtained that yielded field and call performance classifiers with good generalization capabilities. These results are shown in Table 6. It is evident that the polynomial kernel function resulted in the most accurate field 'Say account' SVM classifier. The linear and RBF kernel function classifiers were only 3% less accurate on unseen 'Say account' data. Similarly, the sigmoid kernel function 'Say amount' classifiers were most accurate. The linear kernel function provided to be most appropriate for the field 'Select beneficiary' and 'Say confirmation' classifiers.

The kernel functions used created call performance classifiers with accuracies larger than 95% on validation and test data sets. It is evident that the linear kernel function resulted in the most accurate call performance classifier. The RBF and sigmoid kernel function call performance classifiers were only approximately 1.5% and 2% less accurate on the test data set, respectively. However, the polynomial kernel function call performance classifier is approximately 3% less accurate than the linear kernel function classifier.

Table 6. The Results of SVM implementation.

Kernel function	Field	Accuracy Validation	Accuracy Test
Field classification component			
Linear	'Say account'	0.9688	0.8814
	'Say amount'	0.9068	0.9691
	'Select beneficiary'	0.9447	0.9453
	'Say confirmation'	0.9630	0.9029
Polynomial	'Say account'	0.9047	0.9052
	'Say amount'	0.8521	0.8473
	'Select beneficiary'	0.8756	0.8418
	'Say confirmation'	0.9005	0.8583
Radial Basis Function	'Say account'	0.9627	0.8832
	'Say amount'	0.9085	0.9401
	'Select beneficiary'	0.8950	0.8594
	'Say confirmation'	0.9257	0.8994
Sigmoid	'Say account'	0.9519	0.8776
	'Say amount'	0.9093	0.9263
	'Select beneficiary'	0.8939	0.8571
	'Say confirmation'	0.9064	0.8703
Call performance classification component			
Linear	-	0.9896	0.9919
Polynomial	-	0.9609	0.9630
Radial Basis Function		0.9756	0.9736
Sigmoid		0.9713	0.9696

The field and call performance SVM classifiers created employed a classification threshold value of 0.5. This threshold value resulted in above 90% accurate classifications. This has been demonstrated on the training, validation and test data sets. As result, this threshold value of 0.5 proved to be adequate for the SVM implementations.

7. Comparison of the computational intelligent methods considered and the selection of the superior classifiers

The computational intelligent techniques considered produced highly accurate field and call performance classifiers. The MLP ANN, RBF ANN and the SVM methods generated solutions that achieved accuracy values larger than 90% on unseen validation as well as test data. However, the field FIS classifier yielded accuracies less than 85%.

The MLP field classifiers produced the most accurate solutions, outperforming the RBF and SVM field classifiers on both validation and test data sets. This is true for the 'Say account', 'Say amount' and 'Select beneficiary' fields. However, the 'Say confirmation' field SVM classifier achieved the best results on both validation and test data sets.

The call performance SVM classifiers proved to be the most accurate, achieving an accuracy of 99.19% on test data. However, the call performance MLP ANN approach performed the best on validation data.

In order to improve the accuracy of the field and call performance classifiers, an ensemble of networks has been considered. Ensembles of field 'Say account ','Say amount ','Select beneficiary' and 'Say confirmation' classifiers, consisting of the most accurate MLP ANN, RBF ANN as well as SVM networks, were developed. The call performance ensemble of classifiers considered consisted of the best MLP ANN, FIS and SVM models. Table 7 details the classifiers used within the ensembles.

Table 7. Ensemble of classifiers.

Ensemble	Method	Accuracy Validation	Accuracy Test
'Say account' field Ensemble of classifiers	MLP ANN (8 hidden nodes)	0.9844	0.9513
	RBF ANN (94 hidden nodes)	0.9499	0.9376
	SVM (Polynomial)	0.9047	0.9052
'Say amount' field Ensemble of classifiers	MLP ANN (9 hidden nodes)	0.9199	0.9796
	RBF ANN (94 hidden nodes)	0.9143	0.9603
	SVM (Sigmoid)	0.9093	0.9263
'Select beneficiary' field Ensemble of classifiers	MLP ANN (6 hidden nodes)	0.9628	0.9722
	RBF ANN (68 hidden nodes)	0.9443	0.9573
	SVM (Linear)	0.9447	0.9453
'Say confirmation' field Ensemble of classifiers	MLP ANN (50 hidden nodes)	0.9557	0.9021
	RBF ANN (60 hidden nodes)	0.9588	0.9048
	SVM (Linear)	0.9630	0.9029
Call performance Ensemble of classifiers	MLP ANN (10 hidden nodes)	0.9904	0.9918
	FIS (0.14 cluster radius)	0.9152	0.9108
	SVM (Linear)	0.9896	0.9919

It has been stated that classifiers utilized simultaneously as committees or ensemble, will provide an average error that is lower than any individual classifier[5]. Therefore, a combination of networks as a classifier should outperform a single network classifier.

The outputs of classifiers were fed into a voting system. The voting system determined the final output of the ensemble. If the majority of the classifiers within the ensemble categorized an output into a certain class, the voting system would classify the output of the ensemble as the class. If all of the models within the ensemble classified an output into different classes, the voting system would classify the output of the ensemble as undecided.

As shown in Table 8, the ensemble of classifiers proved to be an accurate solution. These committees yielded large accuracy values on both validation and test data sets. In order to confirm the performance of the classifiers created, the sensitivity and specificity values were also compared. These metrics confirmed that the ensemble of field 'Say amount' classifiers and the ensemble of field 'Say confirmation' classifiers outperform the MLP ANN, RBF ANN, FIS as well as SVM field performance classification solutions. However, the MLP ANN classifiers for these fields had a larger specificity value for both the validation and test data sets. This indicates that the MLP ANN classifiers for these fields have a larger negative classification rate on both the data sets. Similarly, the ensemble of call performance classifiers also outperformed the single

Table 8. The performance metrics of best classifiers per method considered.

	Method	Validation			Test		
		Accuracy	Sensitivity	Specificity	Accuracy	Sensitivity	Specificity
Field 'Say account' classifier	MLP ANN	**0.9844**	**0.9738**	**0.9951**	**0.9513**	**0.9269**	**0.9763**
	RBF ANN	0.9499	0.9202	0.9806	0.9376	0.9049	0.9715
	FIS	0.8068	0.7200	0.9943	0.8077	0.6615	0.9863
	SVM	0.9047	0.8655	0.9457	0.9052	0.8766	0.9347
	Ensemble	0.9685	0.9559	0.9813	0.9429	0.9191	0.9673
Field 'Say amount' classifier	MLP ANN	0.9199	0.8803	0.9613	0.9796	0.9711	0.9883
	RBF ANN	0.9143	0.8717	0.9590	0.9603	0.9404	0.9805
	FIS	0.8265	0.7022	0.9860	0.8254	0.6847	0.9951
	SVM	0.9093	0.8681	0.9524	0.9263	0.8948	0.9589
	Ensemble	**0.9229**	**0.8891**	**0.9581**	**0.9683**	**0.9575**	**0.9792**
Field 'Select beneficiary' classifier	MLP ANN	**0.9628**	**0.9394**	**0.9868**	**0.9722**	**0.9566**	**0.9880**
	RBF ANN	0.9443	0.9132	0.9765	0.9573	0.9361	0.9789
	FIS	0.7843	0.6217	0.9900	0.7782	0.6273	0.9908
	SVM	0.9447	0.9151	0.9754	0.9453	0.9159	0.9756
	Ensemble	0.9574	0.9384	0.9768	0.9658	0.9519	0.9799
Field 'Say confirmation' classifier	MLP ANN	0.9557	0.9206	0.9922	0.9021	0.8454	0.9625
	RBF ANN	0.9588	0.9428	0.9751	0.9048	0.8559	0.9565
	FIS	0.7951	0.7138	0.9958	0.7947	0.6576	0.9604
	SVM	0.9630	0.9498	0.9764	0.9029	0.8523	0.9564
	Ensemble	**0.9649**	**0.9526**	**0.9773**	**0.9068**	**0.8590**	**0.9573**
Call performance classifier	MLP ANN	0.9904	0.9845	0.9963	0.9918	0.9877	0.9960
	RBF ANN	0.9068	0.9784	0.8404	0.9019	0.9677	0.8406
	FIS	0.9152	0.9455	0.8858	0.9108	0.9333	0.8888
	SVM	0.9896	0.9844	0.9949	0.9919	0.9889	0.9949
	Ensemble	**0.9917**	**0.9901**	**0.9933**	**0.9925**	**0.9908**	**0.9941**

classifier solutions. As a result, the ensemble of classifiers is the preferred solution for 'Say amount' and 'Say confirmation' field classification. When comparing the call performance classifiers developed, the ensemble of classifiers has the best generalization capabilities. Therefore, the ensemble of classifiers is also the preferred solution for call performance classification.

However, the field 'Say account' and 'Select beneficiary' MLP classifiers outperformed the ensemble of classifiers solution. These classifiers produced large sensitivity and specificity values. As a result, these MLP classifier solutions have high positive as well as negative classification rates on both the validation and test data sets and are therefore, the preferred classification solutions.

8. Conclusion

The computational intelligent methods considered in this research yielded highly accurate field and call performance classifiers. Field 'Say account', 'Say amount', 'Say confirmation' and 'Select beneficiary' as well as call performance classifiers were developed using MLP ANN, RBF ANN, FIS, SVM and ensemble of classifiers.

The MLP ANN and RBF ANN implementation process entailed determining the optimal number of hidden nodes. Binary and real coded GAs that employed normalized geometric ranking and tournament selection functions were used to compute the optimal number of hidden nodes. Computational efficiency and quality of solution were used to evaluate the performance of the GA solutions. These GA solutions yielded accurate MLP and RBF ANN classifiers. However, the field and call performance MLP ANN classifiers resulted in more accurate solutions. In order to improve the call performance RBF ANN accuracies, the classification threshold has been optimized. This process did result in an improvement in accuracy.

The SVM development involved the identification of the SVM kernel function that yielded the most accurate results. The polynomial and sigmoid kernel function resulted in the most accurate 'Say account' and 'Say amount' field classifiers, respectively. However, the linear kernel function provided to be most appropriate for the 'Select beneficiary' and 'Say confirmation' field classifiers.

The FIS classifiers were developed by initially identifying the cluster radius that resulted in the most accurate field and call performance classifications. Thereafter, the thresholds used to interpret these classifications were optimized. The accuracy of the field FIS classifiers did improve. However, the 0.5 classification threshold proved to be the optimal for the call FIS performance classifier.

Ensemble of field classifiers, consisting of the most accurate MLP ANN, RBF ANN and SVM classifiers, has also been developed. Accuracy, sensitivity and specificity performance metrics were computed and compared for the computational intelligent solutions. The MLP and ensemble of field classifiers achieved high sensitivity and specificity. The field MLP classifiers are the preferred models for the 'Say account' and 'Select beneficiary' fields as they yield the best performance results. These classifiers were 95.13% and 96.28% accurate on unseen data, respectively. It has also been determined that the ensemble of field classifiers is the most accurate for the field 'Say amount' and 'Say confirmation' MLP classifiers. These classifiers achieved accuracy values of 92.29% and 90.68%, respectively.

The most accurate call performance classification method is the ensemble of classifiers, yielding an accuracy of 99.17%.

References

1. R. Jones, *South African National 2007/2008 BPO & Call Centre Report*, (C3 Africa Research and The MultiMedia Group, Sandton, 2008).
2. C. Nichols, *The Move from IVR to Speech – Why This is the Right Time to Make the Move to Speech Applications in Customer-Facing Operations*, (Intervoice, 2006).
3. How to right the wrongs of IVR last accessed: 20 June 2009. [Online]. Available: http://www.callcentrehelper.com/wrongs-ivr-4684.html
4. VoiceXML 2.0/VoiceXML 2.1 Reference, last accessed 20 June 2009 [Online]. Available http://developer.voicegeniecom/voicexml2tagref.php?tag=st_field&display=standardtags.
5. C. M. Bishop, *Neural Networks for Pattern Recognition* (Oxford University Press, 1995).
6. C. R. Houck, J. A. Joines, M. G. Kay, *A genetic algorithm for function optimization: a Matlab implementation*, (NCSU-IE Technical Report, North Carolina State University, 1995).
7. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd revised edn. (Springer–Verlag, New York 1996).
8. C. Z. Janikow and Z. Michalewicz, An Experimental Comparison of Binary and Floating Point

Representations in Genetic Algorithms, in *Proc. of the 4th Int. Conf. on Genetic Algorithms* (1991), 31–36.
9. T. Joachims, Text categorization with Support Vector Machines: Learning with many relevant features, in *Proc. of the European Conf. on Machine Learning* (1998) 137-142.
10. A. Tolambiya and P.K. Kalra, Contrast sensitive epsilon-SVR and its application in image compression, *2008 IEEE Int. conf. on Systems, Man and Cybernetics* (2008) 359-364.
11. S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. Mesirov, T. Poggio, W. Gerald, M. Loda, E. Lander, and T. Golub, Multi-Class Cancer Diagnosis Using Tumor Gene Expression Signatures, in *Proc. National Academy of Sciences U.S.A.* **98**(26) (2001) 15149-15154.
12. M. Pal and P. M. Mather, Assessment of the Effectiveness of Support Vector Machines for Hyperspectral Data, *Future Generation Computer Systems.* **20**(7) (2004) 1215-1225.
13. A. Elmzabi, M. Bellafkih, and M. Ramdani, An Adaptive Fuzzy Clustering Approach for the Network Management, *Int. Journal of Information Technology*. **3**(1) (2007) 12-17.
14. P. B. Patel, T. Marwala, Interactive Voice Response field classifiers, *2008 IEEE Int. conf. on Systems, Man and Cybernetics* (2008) 3425-3430.
15. J. M. Keller, M. Gray and J. Givens, A fuzzy k-nearest neighbor algorithm, *IEEE Transaction on System, Man and Cybernetics*. **15**(4) (1985) 580-585.
16. J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms* (Plenum, New York, 1981).
17. M. Russo, FuGeNeSys – A fuzzy genetic neural system for fuzzy modeling, *IEEE Transaction on Fuzzy Systems.* **6**(3) (1998) 373-388.
18. P. B. Patel and T. Marwala, Neural Networks, Fuzzy Inference Systems and Adaptive-Neuro Fuzzy Inference Systems for Financial Decision Making, *Int. Conf. on Neural Information Processing*. **4234**(13) (2006) 430-439.
19. C-W. Hsu and C-J. Lin, A comparison of methods for multi-class support vector machines, *IEEE Transactions on Neural Networks*. **13**(2) (2002) 415-425.
20. B. Zadrozny, Reducing multiclass to binary by coupling probability estimates, *Neural Information Processing Systems Foundation 14*, (2002) 1041-1048.
21. VoiceGenie Technologies Inc., *VoiceGenie 7 Tools User's Guide*. (VoiceGenie Technologies Inc, 2005).
22. VoiceXML properties, last accessed: 20 June 2009. [Online]. Available: http://community.voxeo.com/vxml/docs/nuance20/VXMLproperties.html.
23. T. Nabney, *Netlab: Algorithms for Pattern Recognition.* (Springer, 2002).
24. R. Beale, T. Jackson, Neural Computing-An introduction. (Taylor & Francis, 1990).
25. E. B. Baum, D. Haussler, What size net gives valid generalization? *Neural Computation*, **1** (1989) 81–90.
26. E. A. Freeman, G. G. Moisen, A comparison of the performance of threshold criteria for binary classification in terms of predicted prevalence and kappa, *An Int. Journal on Ecological Modelling and systems ecology*. **217** (2008) 48-58.
27. The Mathworks, Inc., Fuzzy Logic Toolbox User's Guide Version 2. (The Mathworks, Inc, 1995).
28. F. Lotte, A. Lecuyer, F. Lamarche and B. Arnaldi, Studying the use of Fuzzy Inference Systems for motor imagery classification. *IEEE transactions on Neural Systems and Rehabilitation Engineering*. **15**(2) (2007) 322-324.
29. J. Yen, L. Wang, Constructing optimal fuzzy models using statistical information criteria. *Journal of Intelligent and Fuzzy Systems*. **7**(1999) 185-201.
30. S. Chiu, Fuzzy Model Identification Based on Cluster Estimation. *Journal of Intelligent and Fuzzy Systems*. **2**(3) (1994) 267-278.
31. J.S. Taylor and N. Cristianini, *Support Vector Machines and other kernel-based learning methods*. (Cambridge University Press, 2000).
32. C. W. Hsu, C. C. Chang and C. J. Lin, *A practical guide to support vector classification.* (Technical Report, Taipei, 2003). [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf