USING A MACHINE LEARNING ALGORITHM TO DEVELOP AN INTELLIGENT AUTOMATED TELLER MACHINE (ATM)

WALTER KIRIKA NDEGWA

MASTER OF SCIENCE

(Software Engineering)

JOMO KENYATTA UNIVERSITY OF AGRICULTURE AND TECHNOLOGY

2010

Using a machine learning algorithm to develop an intelligent automated teller machine (ATM)

Walter Kirika Ndegwa

A thesis submitted in partial fulfillment for the Degree of Master of Science in Software Engineering in the Jomo Kenyatta University of Agriculture and Technology

2010

DECLARATION

This thesis is my original work and has not been presented for a degree in any other university.

Signature:..........……………………………… Date: …………………..

Walter Kirika Ndegwa

This thesis has been submitted for examination with my approval as University Supervisor.

Signature:..........……………………………… Date: …………………..

Dr. Waweru Mwangi

JKUAT, Kenya

DEDICATION

I dedicate this project to my Lord and Savior Jesus Christ. Thank you Lord, I owe it all to you.

ACKNOWLEDGEMENT

To my family especially Wanja, “you guys are God-sent”. D-Serve! Even when I didn’t have bus fare to make it to class, you guys still encouraged me. You are the best friends anybody could ever wish for

Dr. Waweru, I’m heavily indebted to you. Thanks for your support, encouragement and extremely brilliant input throughout this whole project. I hope one day I will be able to work with you, this time as a peer. Thank you so much Sir.

Dr. Kimani, Dr. Wafula, Thank you for your contribution and constructive criticism.

Class mates, thanks for your input.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
AQ	Algorithm Quasi-Optimal
ATM	Automated Teller Machine
CC	Cyclomatic complexity
CLS	Concept Learning System
DFD	Data Flow Diagram
ER	Entity Relationships
GLO	Generic Learning Outcome
ID	Identity Card
ID3	Itemized Dichotomozer 3
PIN	Personal Identification Number
STAR	Set of general descriptions of a particular event
V&V	Validation and verification

ABSTRACT

With major advancements having been made in information technology, computers can perform many operations exponentially much faster than a human being. Though the preceding statement is true there are many tasks where the computer falls much short of its human counterpart. An example of this is given two pictures a nursery school kid could easily tell the difference between a cow and a donkey. This simple task could confound today's computer.

This study established that the introduction of a learning component to the already existing framework would be acceptable and to demonstrate this, a sample prototype (learning component) was developed.

Majority of learning algorithms work well only with discrete values, i.e. (0 or 1, true or false). For a successful learning approach to be implemented a new method of learning had to be devised that supported continuous variables (multi-valued attributes). Question answer authentication was the method established to achieve this. The learning component was implemented on the premise of the AQ learning algorithm.

CHAPTER 1: INTRODUCTION

Machine learning refers to a system with the capability of the autonomous acquisition and integration of knowledge. This capacity to learn from experience, analytical observation, and other means, results in a system that can continuously self-improve and thereby offer increased efficiency and effectiveness.

Learning denotes changes in a system that ... enables a system to do the same task more efficiently the next time.

Learning is constructing or modifying representations of what is being experienced (Michalski *et al*, 1986).

Given a task we want the computer to do, the idea is to repeatedly demonstrate how the task is performed, and let the computer learn by example, i.e., generalize some rules about the task and turn these into a program.

Learning denotes changes in the system that are adaptive in the sense that they enable the system to do the same task or tasks drawn from the same population more efficiently and more effectively next time (Michalski *et al*, 1983).

The goal of machine learning is to program computers to use example data or past experience to solve a given problem (Alpaydin, 2004).

Machine learning is concerned with the question of how to construct computer programs that automatically improve with experience. In recent years many successful machine learning applications have been developed ranging from data mining programs that learn to detect fraudulent credit card transactions to autonomous vehicles that learn how to drive on public highways (Pomerleau, 1989; Mitchell, 1997).

Attempts to create self-improving software date to the 1960s. But 'machine learning,' as it's often called, has remained mostly the province of academic researchers, with only a few niche applications in the commercial world, such as speech recognition and credit card fraud detection. Now, researchers say, better algorithms, more powerful computers and a few clever tricks will move it further into the mainstream.

Machine learning is useful for the kinds of tasks that humans do easily -- speech and image recognition, for example -- but they have trouble explaining explicitly in software rules (Mitchell, 1997).

In machine-learning applications, software is 'trained' on test cases devised and labeled by humans, scored so it knows what it got right and wrong, and then sent out to solve real-world cases (Anthes, 2006).

Find a bug in a program, and fix it, and the program will work today. Show the program how to find and fix a bug, and the program will work forever (Selfridge, 2000).

A computer program is said to learn from experience **E** with respect to some class of tasks **T** and performance measure **P**, if its performance at tasks in **T**, as measured by **P**, improves with experience **E** (Mitchell, 1997).

An algorithm is a finite list of well-defined instructions for accomplishing some task that, given an initial state, will proceed through a well-defined series of successive states, possibly eventually terminating in an end-state. A machine learning algorithm is therefore a finite list of well defined instructions that will result in an end-product e.g. software that will have the capacity to learn from experience.

According to the encyclopedia of science and philosophy an adaptive system is a physical system that is capable of self-adapting in response to changing environments. An adaptive system is a system that is able to adapt its behavior according to changes in its environment or in parts of the system itself.

An adaptive system is one that has the capacity to perform a task in more than one way. It must make choices about the most appropriate course of action to take and is able to modify its choice making behavior should a choice turn out to be inappropriate.

Intelligence refers to the systems comparative level of performance in reaching its objectives. This implies having experiences where the system learned which actions best let it reach its objectives; an intelligent system learns how to act so it can reach its objectives.

The problem of inducing general functions from specific training examples is central to learning. Concept learning can be formulated as a pattern of searching through a predefined space of potential hypotheses for the best hypothesis that suits the training examples. Much of learning involves acquiring general concepts from specific training examples. People continually learn general concepts or categories, e.g. birds. Each such concept can be viewed as describing some subset of objects or events defined over a larger set (e.g., the subset of animals that constitute birds). Alternatively each concept can be thought of as a Boolean-valued function over this larger set, (e.g., a function defined over all animals, whose value is true for birds and false for other animals).

Concept learning is the inferring of a Boolean-valued function from training examples of its input and output (Mitchell, 1997).

1.1 Problem Statement

An Automated Teller Machine (ATM) is a computerized telecommunications device that provides the customers of a financial institution with access to financial transactions in a public space without the need for a human clerk or bank teller. ATMs are static, they don't learn from experience.

An intelligent system learns so as it can meet its objectives. The objective of an ATM is providing customers with access to financial services. Using machine learning an ATM can be trained using suitable training examples to offer better, efficient and effective services. The main aim of the plastic magnetic card or smart card is for customer identification. If the ATM is taken through a process of learning it can learn customers that use it and identify them in ways other than the card and pin code thus by learning it could still meet its objective of offering access to financial services though no physical card was swiped at the card reader.

1.2 Proposed Solution

Develop an appropriate machine learning algorithm that will facilitate the development (and training using training examples) of a learning component which through learning will train an ATM.

1.3 Objectives

The broad objective of this research is to facilitate and/or bring to light this concept of machine learning and how it (machine learning) can be incorporated in existing software systems to "train" the respective software to perform its task(s) effectively.

The main focus entails introducing components that "learn" to the already existing software frameworks.

The specific objectives include:

- Study a subset of already existing machine learning algorithms.
- Design a sample learning component that can be integrated in the existing ATM system.

1.4 Research questions

In order to achieve the objectives stated above the following questions need to be answered by this study.

i) What are the ATM users' needs and/or expectation(s)?

ii) How can machine learning can be incorporated to enable the ATM achieve most if not all of the user needs?

This research seeks to establish whether a learning algorithm could be introduced to the already existing framework of ATMs. The preceding chapter contains a review of existing learning algorithms detailing how they work and are applied. This review will act as a guide in choosing the learning algorithm to employ.

CHAPTER 2: LITERATURE REVIEW

For a computer system to adapt it has to incorporate some mode of learning. On the other hand the converse of adaptive systems is static systems. Static systems have no self repair ability and they typically remain in a single non-adaptive state until their termination. Static systems are not equipped for self-modification and are only able to function within a small range of environmental change. An intelligent system, on the other hand, will be equipped to self-modify into different system-states in order to navigate, function and succeed within different environments.

Majority of the existing software systems are static. This means that the software systems only perform simple duties though they have the capability of extending their functionality, efficiency and effectiveness.

By use of existing learning algorithms systems can be analyzed and redesigned using the algorithms, thereafter be run through training examples and finally after successful training the new versions of the systems can be implemented.

Generally machine learning includes a program that improves its performance through experience (Mitchell, 1997). In order to have a well defined learning problem, three features must be identified; the class of tasks, the measure of performance to be improved, and the source of experience. For example, a computer program that learns to play checkers might improve its performance as measured by its ability to win at the class of tasks involving playing checkers games, through experience obtained by playing games against itself.

Machine learning mainly involves running a machine through a set of training examples.

There are two ways a system can improve/learn:

1. By acquiring new knowledge For example: acquiring new facts or skills

2. By adapting its behavior For example: solving problems more accurately or efficiently

A program incorporating machine learning improves its performance with experience. The ATM system presently has only one way of recognizing customers which is the two-level authentication system of having the ATM card and entering the PIN code.

For a deeper understanding of machine learning, we now look at a subset of existing machine learning algorithms.

2.1 General-to-Specific Ordering in Concept Learning

Concept learning can be formulated as a problem of searching through a predefined space of potential hypotheses for the hypothesis that best fits the training examples. In many cases this search can be efficiently organized by taking advantage of a naturally occurring structure over the hypothesis space-a general-to-specific ordering of hypotheses (Mitchell, 1997).

Much of learning involves acquiring general concepts from specific training examples. People, for example, continually learn general concepts or categories such as "bird," "car," "situations in which I should study more in order to pass the exam," etc.

Each such concept can be viewed as describing some subset of objects or events defined over a larger set (e.g., the subset of animals that constitute birds). Alternatively, each concept can be thought of as a Boolean-valued function defined over this larger set (e.g., a function defined over all animals, whose value is true for birds and false for other animals).

Consider the example task of learning the target concept "days which Pablo enjoys his favorite water sport." The table below describes a set of example days, each represented by a set of ***attributes.*** The attribute ***EnjoySport*** indicates whether or not Pablo enjoys his favorite water sport on this day. The task is to learn to predict the value of ***EnjoySport*** for an arbitrary day, based on the values of its other attributes.

We can consider a simple representation in which each hypothesis consists of a conjunction of constraints on the instance attributes. In particular, let each hypothesis be a vector of six constraints, specifying the values of the six attributes ***Sky, AirTemp, Humidity, Wind, Water,*** and ***Forecast.*** For each attribute, the hypothesis will either

- Indicate by a "?' that any value is acceptable for this attribute,
- Specify a single required value (e.g., ***Warm)*** for the attribute, or
- Indicate by a *"0"* that no value is acceptable.

If some instance x satisfies all the constraints of hypothesis h, then h classifies x as a positive example (h(x) = 1).

To illustrate, the hypothesis that Pablo enjoys his favorite sport only on cold days with high humidity (independent of the values of the other attributes) is represented by the expression (?, Cold**,** High, ?, ?, ?)

Table 1: Positive and negative training examples for the target concept EnjoySport.

Example	**Sky**	**AirTemp**	**Humidity**	**Wind**	**Water**	**Forecast**	**EnjoySport**
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

The most general hypothesis-that every day is a positive example-is represented by

(?, ?, ?, ?, ?, ?)

and the most specific possible hypothesis-that no day is a positive example-is represented by

(Ø, Ø, Ø, Ø, Ø, Ø)

The set of items over which the concept is defined is called the set of instances, which we denote by X. In the current example, X is the set of all possible days, each represented by the attributes Sky, AirTemp, Humidity, Wind, Water, and Forecast. The concept or function to be learned is called the target concept, which we denote by c. In general, c can be any Boolean valued function defined over the instances X; that is, $c : X \quad \{0, 1\}$

In the current example, the target concept corresponds to the value of the attribute EnjoySport (i.e., c(X) = 1 if EnjoySport = Yes, and c(X) = 0 if EnjoySport = No).

The set of items over which the concept is defined is called the set of instances, which we denote by X. In the current example, X is the set of all possible days, each represented by the attributes Sky, AirTemp, Humidity, Wind, Water, and Forecast. The concept or function to be learned is called the target concept, which we denote by c. In general, c can be any Boolean valued function defined over the instances X; that is, $c : X \quad \{0, 1\}$

In the current example, the target concept corresponds to the value of the attribute EnjoySport (i.e., c(X) = 1 if EnjoySport = Yes, and c(X) = 0 if EnjoySport = No).

Table 2: The EnjoySport concept learning task.

Given

Instances ***X:*** Possible days, each described by the attributes

- ***Sky*** (with possible values ***Sunny, Cloudy,*** and ***Rainy),***
- ***AirTemp*** (with values ***Warm*** and ***Cold),***
- ***Humidity*** (with values ***Normal*** and ***High),***
- ***Wind*** (with values ***Strong*** and ***Weak),***
- ***Water*** (with values ***Warm*** and ***Cool),*** and
- ***Forecast*** (with values ***Same*** and ***Change).***

- Hypotheses ***H:*** Each hypothesis is described by a conjunction of constraints on the attributes ***Sky, AirTemp, Humidity, Wind, Water,*** and ***Forecast.*** The constraints may be "?" (Any value is acceptable), “ø” (no value is acceptable), or a specific value.
- Target concept ***c: EnjoySport*** : X $\rightarrow$ {0, 1}

Training examples D: Positive and negative examples of the target function (see Table 1).

Determine

A hypothesis ***h*** in ***H*** such that ***h(x)*** = ***c(x)*** for all *x* in X.

When learning the target concept, the learner is presented a set of ***training examples,*** each consisting of an instance ***x*** from X, along with its target concept value ***c(x)*** (e.g., the training examples in Table 1). Instances for which ***c(x)*** = 1 are called ***positive examples,*** or members of the target concept. Instances for which ***C(X)*** = 0 are called ***negative examples,*** or nonmembers of the target concept.

Given a set of training examples of the target concept ***c,*** the problem faced by the learner is to hypothesize, or estimate, ***c.*** We use the symbol H to denote the set of ***all possible hypotheses*** that the learner may consider regarding the identity of the target concept. In general, each hypothesis ***h*** in H represents a boolean-valued function defined over X; that is, ***h*** : X $\rightarrow$ {O, 1}.

The goal of the learner is to find a hypothesis ***h*** such that ***h(x)*** = ***c(x)*** for all ***x*** in X.

2.2 INDUCTIVE LEARNING

Inductive learning started around 1960's, when Banerji (1964) used predicate logic as a description language, (Sammut, 1993). The predicate logic would then be the foundation in inductive learning and inductive logic programming. Meanwhile in 1969, Cohen suggested that it would be possible to create effective descriptions by learning domain knowledge, which brought to the creation of

CONFUCIUS (Cohen and Sammut, 1982). CONFICIUS was the first program that learned description in first-order logic and be able to reuse the learned concept (Sammut, 1993).

Vere (1975) had developed formal induction algorithms for expressions in predicate calculus. He stated that if the concept being learned is conjunctive, it is sufficient to find the intersection of all products to produce the generalization

Given a pair *(x, f(x))*, where *x* is the input and *f(x)* is the output of the function applied to *x*. The task of pure inductive inference is this:

Given a collection of examples *f*, return a function *h* that approximates *f*.

The function *f* is called a hypothesis. The reason that learning is difficult, from a conceptual point of view, is that it is not easy to tell whether any particular *h* is a good approximation of *f*. A good hypothesis will *generalize* well-that is, will predict unseen examples correctly. This is the fundamental *problem of induction*. The possibility or impossibility of finding a simple, consistent hypothesis depends strongly on the hypothesis space chosen.

Inductive rule-learning methodology: One popular technique of machine learning involves the learning of paradigm classification rules from a set of training examples.

In this paradigm the learning system searches a space of rules to find out those which best classify the training examples where best is defined in terms of accuracy and comprehensibility

Algorithms ID3 and AQ have been used as a basis of several machine learning systems. The form of inputs and outputs which these algorithms receive and produce are shown in the example.

Attributes

Table 3: Sample data for inductive learning algorithms

Furry?	*Age?*	*Size?*	*Class*
Furry	Old	Large	Lion
Not furry	Young	Large	Not lion
Furry	Young	Medium	Lion
Furry	Old	Small	Not lion
Furry	Young	Small	Not lion
Furry	Young	Large	Lion
Not furry	Young	Small	Not lion
Not furry	Old	Large	Not lion

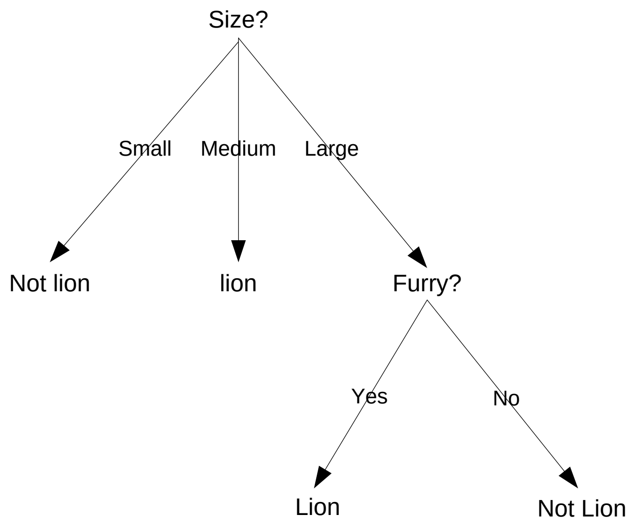


Figure 1: Decision tree output by ID3

Decision rules output by AQ

If furry = y and size = large

Then class = lion

If size = medium

Then class = lion

If furry = no

Then class = not lion

If size = small

Then class = not lion

2.2.1 ID3 Algorithm

Itemized Dichotomozer 3 algorithm, or better known as ID3 algorithm was first introduced in the late 1970's. The algorithm 'learned' from relatively small training set of data to organize and process very large data sets.

ID3 is based off the Concept Learning System (CLS) algorithm.

The basic CLS algorithm over a set of training instances C:

> **Step 1**: If all instances in C are positive, then create YES node and halt.
>
> If all instances in C are negative, create a NO node and halt.
>
> Otherwise select a feature, F with values v1, ..., vn and create a decision node.
>
> **Step 2**: Partition the training instances in C into subsets C1, C2, ..., Cn according to the values of V.
>
> **Step 3**: apply the algorithm recursively to each of the sets Ci.

Note, the trainer (the expert) decides which feature to select.

Ballard (2002), stated that ID3 algorithm is a greedy algorithm that selects the next attributes based on the information gain associated with the attributes. The information gain is measured by entropy, where Shannon (1948), first introduced the idea.

ID3 algorithm prefers that the generated tree is shorter and the attributes with lower entropies are put near the top of the tree (Ballard, 2002). These techniques satisfy the idea of Occam's razor. Occam's razor stated that, "one should not increase, beyond what is necessary, the number of entities required to explain anything", which means that one should not make more assumptions than minimum needed (Heylighen, 1997).

Hild (1989) described the basic technique on the implementation of ID3 algorithm and it is as shown below.

i) For each uncategorized attribute, its entropy would be calculated with respect to the categorized attribute, or conclusion.

ii) The attribute with lowest entropy would be selected.

iii) The data would be divided into sets according to the attribute's value. For example, if the attribute 'Size' was chosen, and the values for 'Size' were 'big', 'medium' and 'small, therefore three sets would be created, divided by these values.

iv) A tree with branches that represent the sets would be constructed. For the above example, three branches would be created where first branch would be ‘big’, second branch would be ‘medium’ and third branch would be ‘small’.

v) Step 1 would be repeated for each branch, but the already selected attribute would be removed and the data used was only the data that exists in the sets.

vi) The process stopped when there were no more attribute to be considered or the data in the set had the same conclusion, for example, all data had the ‘Result’ = yes.

ID3 improves on CLS by adding a feature selection heuristic. ID3 searches through the attributes of the training instances and extracts the attribute that best separates the given examples. If the attribute perfectly classifies the training sets then ID3 stops; otherwise it recursively operates on the n (where n = number of possible values of an attribute) partitioned subsets to get their "best" attribute. The algorithm uses a greedy search, that is, it picks the best attribute and never looks back to reconsider earlier choices.

ID3 is a non-incremental algorithm, meaning it derives its classes from a fixed set of training instances. An incremental algorithm revises the current concept definition, if necessary, with a new sample. The classes created by ID3 are inductive, that is, given a small set of training instances, the specific classes created by ID3 are expected to work for all future instances. The distribution of the unknowns must be the same as the test cases. Induction classes cannot be proven to work in every case since they may classify an infinite number of instances. Note that ID3 (or any inductive algorithm) may misclassify data.

Data Description

The sample data used by ID3 has certain requirements, which are:

- Attribute-value description - the same attributes must describe each example and have a fixed number of values.
- Predefined classes - an example's attributes must already be defined, that is, they are not learned by ID3.
- Discrete classes - classes must be sharply delineated. Continuous classes broken up into vague categories such as a metal being "hard, quite hard, flexible, soft, quite soft" are suspect.
- Sufficient examples - since inductive generalization is used (i.e. not provable) there must be enough test cases to distinguish valid patterns from chance occurrences.

Attribute Selection

How does ID3 decide which attribute is the best? A statistical property, called information gain, is used. Gain measures how well a given attribute separates training examples into targeted classes. The one with the highest information (information being the most useful for classification) is selected. In order to define gain, we first borrow an idea from information theory called entropy. Entropy measures the amount of information in an attribute.

Given a collection S of c outcomes

Entropy(S) = S -p(I) $\log_2$ p(I)

where p(I) is the proportion of S belonging to class I. S is over c. Log_2 is log base 2.

Note that S is not an attribute but the entire sample set.

Example 1

Based on Table 3.

Furry

3 lions 2 non lions

Size

Large 2 lion 2 non lion

Medium 1 lion

Small 3 non lion

Entropy furry:

-(0.6 $\log_2$ 0.6)-(0.4 $\log_2$ 0.4) = -(-0.44218)-(-0.52877) = 0.97095

Information gain = (1*3/8) + (1*2/8) = 0.625

Entropy size

Large = -(0.5 $\log_2$ 0.5)-(0.5 $\log_2$ 0.5) = 1

Medium = -(1 $\log_2$ 1) – (0 $\log_2$ 0) = 0

Small = -(1 $\log_2$ 1) – (0 $\log_2$ 0) = 0

Information gain = (1*4/8) + (0*1/8) + (0*3/8) = 0.5

Messages with a high probability of arrival are not as informative as messages with low probability. Hence the attribute size is chosen as the root for the ID3 tree. Small and Medium both yield an entropy of 0, which means that these attributes result in a perfect classification of not-lion and lion respectively.

The attribute chosen next based on the entropy is furry. This process continues until all attributes are classified.

Example 2

Suppose S is a set of 14 examples in which one of the attributes is wind speed. The values of Wind can be Weak or Strong. The classification of these 14 examples are 9 YES and 5 NO. For attribute Wind, suppose there are 8 occurrences of Wind = Weak and 6 occurrences of Wind = Strong. For Wind = Weak, 6 of the examples are YES and 2 are NO. For Wind = Strong, 3 are YES and 3 are NO. Therefore

Gain(S,Wind)=Entropy(S)-(8/14)*Entropy(S_{weak})-(6/14)*Entropy(S_{strong})

= 0.940 - (8/14)*0.811 - (6/14)*1.00

= 0.048

Entropy(S_{weak}) = - (6/8)*$\log_2$(6/8) - (2/8)*$\log_2$(2/8) = 0.811

Entropy(S_{strong}) = - (3/6)*$\log_2$(3/6) - (3/6)*$\log_2$(3/6) = 1.00

For each attribute, the gain is calculated and the highest gain is used in the decision node.

Example of ID3

Suppose we want ID3 to decide whether the weather is amenable to playing baseball. Over the course of 2 weeks, data is collected to help ID3 build a decision tree (see table 1).

The target classification is "should we play baseball?" which can be yes or no.

The weather attributes are outlook, temperature, humidity, and wind speed. They can have the following values:

outlook = { sunny, overcast, rain }

temperature = {hot, mild, cool }

humidity = { high, normal }

wind = {weak, strong }

Examples of set S are:

Table 4. Positive and negative training examples for the target concept "PlayBall".

Day	Outlook	Temperature	Humidity	Wind	Play ball
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes

D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

We need to find which attribute will be the root node in our decision tree. The gain is calculated for all four attributes:

Gain(S, Outlook) = 0.246

Gain(S, Temperature) = 0.029

Gain(S, Humidity) = 0.151

Gain(S, Wind) = 0.048 (calculated in example 2)

Outlook attribute has the highest gain, therefore it is used as the decision attribute in the root node.

Since Outlook has three possible values, the root node has three branches (sunny, overcast, rain). The next question is "what attribute should be tested at the Sunny branch node?" Since we've used Outlook at the root, we only decide on the remaining three attributes: Humidity, Temperature, or Wind.

S_{sunny} = {D1, D2, D8, D9, D11} = 5 examples from table 1 with outlook = sunny

Gain(S_{sunny}, Humidity) = 0.970

Gain(S_{sunny}, Temperature) = 0.570

Gain(S_{sunny}, Wind) = 0.019

Humidity has the highest gain; therefore, it is used as the decision node. This process goes on until all data is classified perfectly or we run out of attributes.

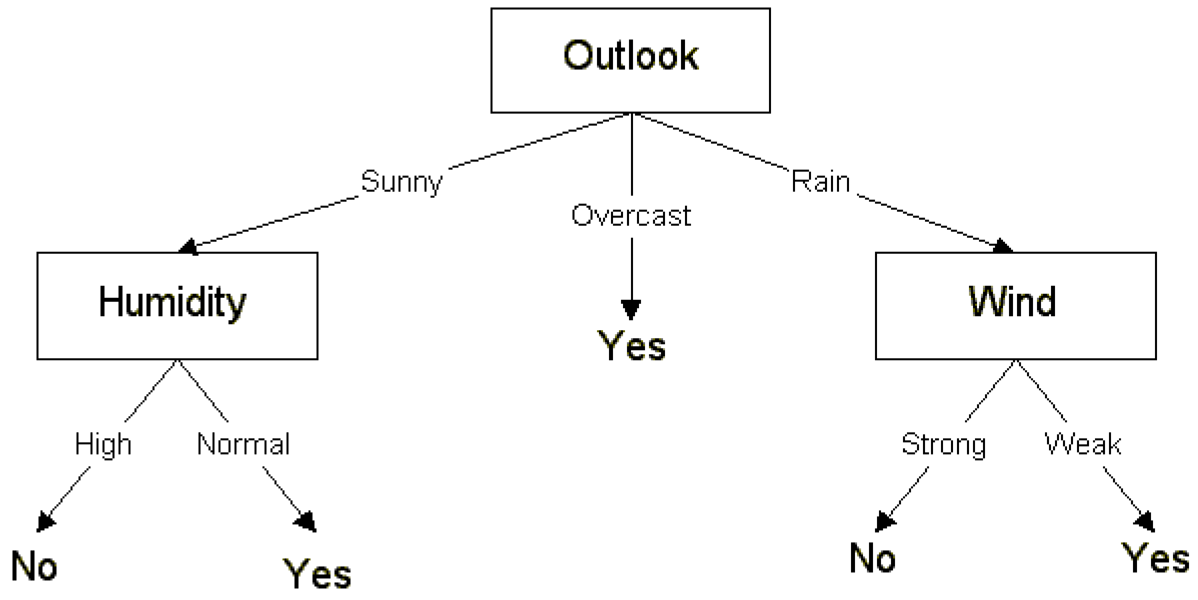


Figure 2: Decision Tree output by ID3 (from example of ID3)

The final decision = tree

The decision tree can also be expressed in rule format:

IF outlook = sunny AND humidity = high THEN playball = no

IF outlook = rain AND humidity = high THEN playball = no

IF outlook = rain AND wind = strong THEN playball = yes

IF outlook = overcast THEN playball = yes

IF outlook = rain AND wind = weak THEN playball = yes

ID3 has been incorporated in a number of commercial rule-induction packages. Some specific applications include medical diagnosis, credit risk assessment of loan applications, equipment malfunctions by their cause, classification of soybean diseases, and web search classification.

2.2.2 The AQ Algorithm

It was originally created in 1969. The AQ algorithm was designed to solve general covering problems of high complexity. In other words, the algorithm was designed to generate generalization or induction from very complex problems.
Clark (1990) stated that AQ algorithm output a set of 'if…then…' rules. This is different compared to ID3 algorithm, which output decision tree to represent the rules. AQ algorithm used 'separate and conquer' approach, where the data would be separated and general rules would be created from the separation.

According to Mihai (2001), the central concept of the algorithm is the 'STAR' defined as a set of general descriptions of a particular event (a 'seed') that satisfies given constraints. In finding the rules, the AQ algorithm used the 'beam search' method to explore the data. The 'beam search' method is actually a breadth first search technique with a heuristic function. Therefore, the progression would not go into ‘blind’ method, but instead, on every node, there would be a consideration whether or not to progress further.

Clark, (1990) and Sammut (1993) described the AQ algorithm as below:

i) The data set would be divided into two parts, according to the conclusion. These parts are known as positive data set and negative data set.

ii) One data would be selected randomly from the positive data set. This data would then be extended against the negative data set by using the STAR method as described above.

iii) All the positive data that satisfy the STAR would be removed and one of the remaining positive data would be selected. The STAR method would be applied again.

iv) The process stopped when there are no more data in the positive data set.

The AQ algorithm can be implemented using incidence calculus methods. Using this method a bit string for each attribute test is constructed and each bit representing a different example (“1” if test is passed and “0” if test is failed), (Bundy, 1985).

The AQ algorithm was chosen as the basis for the learning component since it produces a set of if…Else…Then rules which is the application of the divide and conquer approach. Since the attributes here are not deterministic at the time of customer application (customer opening customer account), the rules for the AQ algorithm will be specific for each and every customer who may want to employ the Learning approach. This in essence means that each customer will have their own specific knowledge base. The AQ rules will be induced from the particular knowledge base for a specific customer.

For instance, if customer XX has answered two questions in his knowledge base; e.g. What's your favorite football team and What was the name of your first pet; and the answers were Arsenal and Oyugi respectively, the AQ rules induced would be

```
If favorite football team = Arsenal then
 Customer=true
Else if first pet name = Oyugi then
  Customer = true
Else
  Customer = false
```

Every time a customer adds new information in his/her knowledge base the rules are updated to reflect this change.

CHAPTER 3: DESIGN

In chapter two a few learning algorithms were studied that would be the basis of developing the learning component. This chapter uses the discussion in the previous chapter to design a prototype learning component.

The main issue that most users have in regards to ATMs is that not physically having the ATM card translates to no service. Many times customers forget their ATM cards and there is no other way they can access their monies via the ATM. What this study produces is an ATM that could offer services to a customer without the customer necessarily having to swipe the actual card.

The ATM services offered are good, but to cater for instances where a customer forgets their ATM card and also better increase security, (e.g. in many carjacking events the victims are normally driven to their ATMs and forced to withdraw money from their accounts) the learning component will enable the customers to be able to perform their transactions normally without having to necessarily have their ATM cards with them.

The management of some local banks described the type of security that was currently offered by their systems as two tiers; one needed the physical ATM card and the PIN code for the ATM to access ATM services. They maintained that regardless of the solution this research would propose these two ‘tier’ security model had to be enforced.

The three alternatives that could be applied to facilitate learning in ATMs were suggested. Namely;

- Iris identification.
- Fingerprint recognition.
- Picture recognition.

Based on the three alternatives above, fingerprint identification would be the most popular. Iris and picture identification would be a major leap that would result in hesitation when it comes to application by customers.

This was however one tier. To fulfill the management requirement of two tiered authentication another component had to be introduced.

The basic idea being introduced by this research was to facilitate for the ATM to be able to recognize the customer without the ATM card.

In regards to the second tier, options had to be evaluated and the most common and popular methods that could be used were:

- User/password authentication
- Question answer authentication

Based on the fact that the component was to be used to identify the customers the question/answer authentication would suffice since the user/password authentication suffers weakness in that it’s static and is prone to shoulder surfing.

Listed below are the requirements the learning component had to fulfill:

- The system (learning component) must be fast.
- The system (learning component) must be user-friendly.
- The output generated by the component must be correct.

A biometric component would be integrated in the already existing framework which would be the level one authentication

After successful level one authentication, one would access level two which is where the learning component lies.

The level one authentication component will accept input fingerprint images; match in a database against a customer’s record. The customer places the finger on the optical scanner by the ATM; the fingerprint is classified and correctly matched to one in the database if it exists.

If the fingerprint match is successful the user enters level two authentication. Here the user is prompted to answer a question they had previously answered. If the answer is right the ATM may ask another question to the customer and store this against his/her record. These questions will constitute the knowledge base for the customer. The question-answer pairs are stored for the particular customer account. When the customer comes to access the service, one of the answered questions is picked randomly and posed to the customer.

The threshold of the number of questions may be between 5 and 10, depending on the customer preference.

She/he has three attempts to input the correct answer or else service will be denied.

Using these questions the ATM “learns” the customers.

3.1 High Level Design-Prototype Design

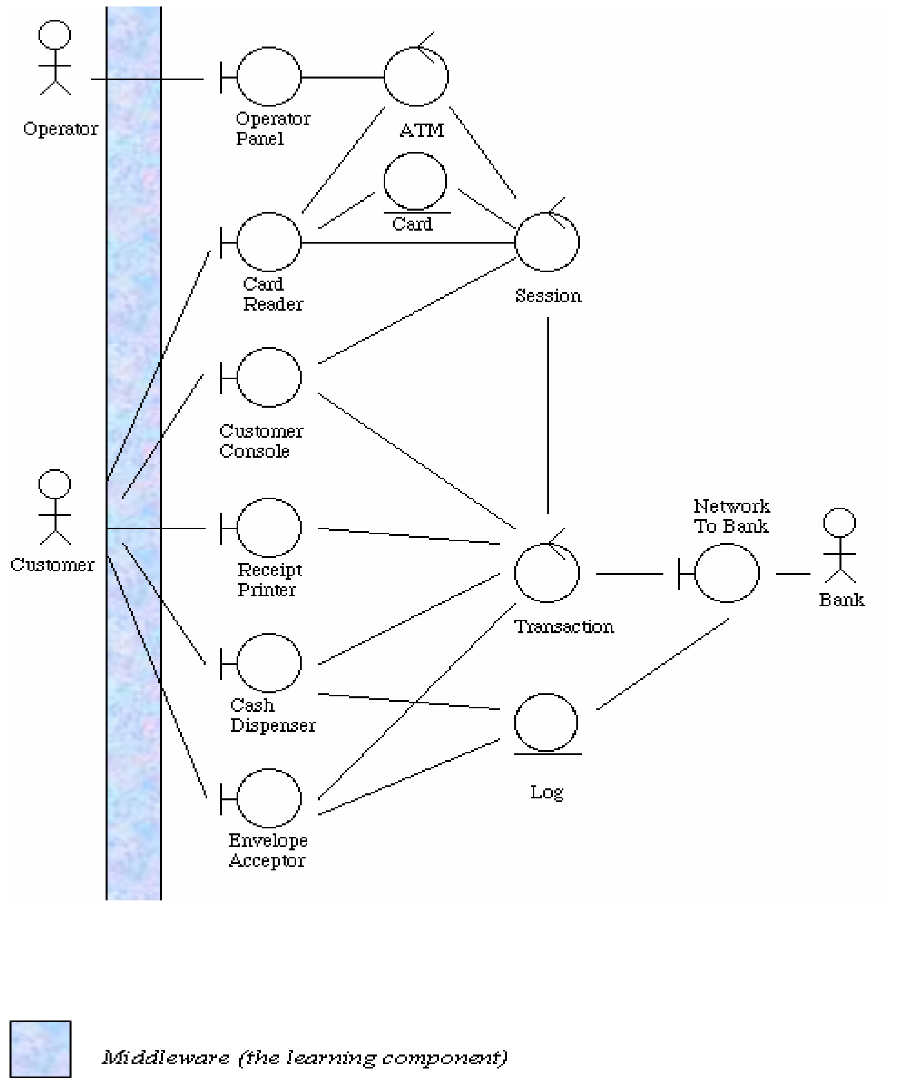


Figure 3: The learning component

The learning component includes the sub-system or middleware that will lie between the customer and the automated teller machine. It will involve acquiring fingerprints from a scanner integrated in the ATM and searching this against prints in the database.

The learning component then goes ahead and asks the user a question if his/her fingerprint is matched. If the user answers the question correctly, the component may ask another question this time not for authentication but for future use.

The main emphasis is for the ATM to “learn” the user. After the system stores a number of predefined questions about the user, it is said to have “learnt” the user. Every time the user comes to access their account and their print is matched. A question is chosen randomly and asked to the user in a way that he/she cannot predetermine what question will be picked.

Using this method a two-level method of security authentication is applied. This method of operation improves the security of the system in that for a customer to access services; first his/her fingerprint must match and secondly he/she must answer a question about them that may be very general or a bit personal. Even if someone was to shoulder surf and see the answer to one question, they cannot predetermine whether that is the same question that will be asked next time.

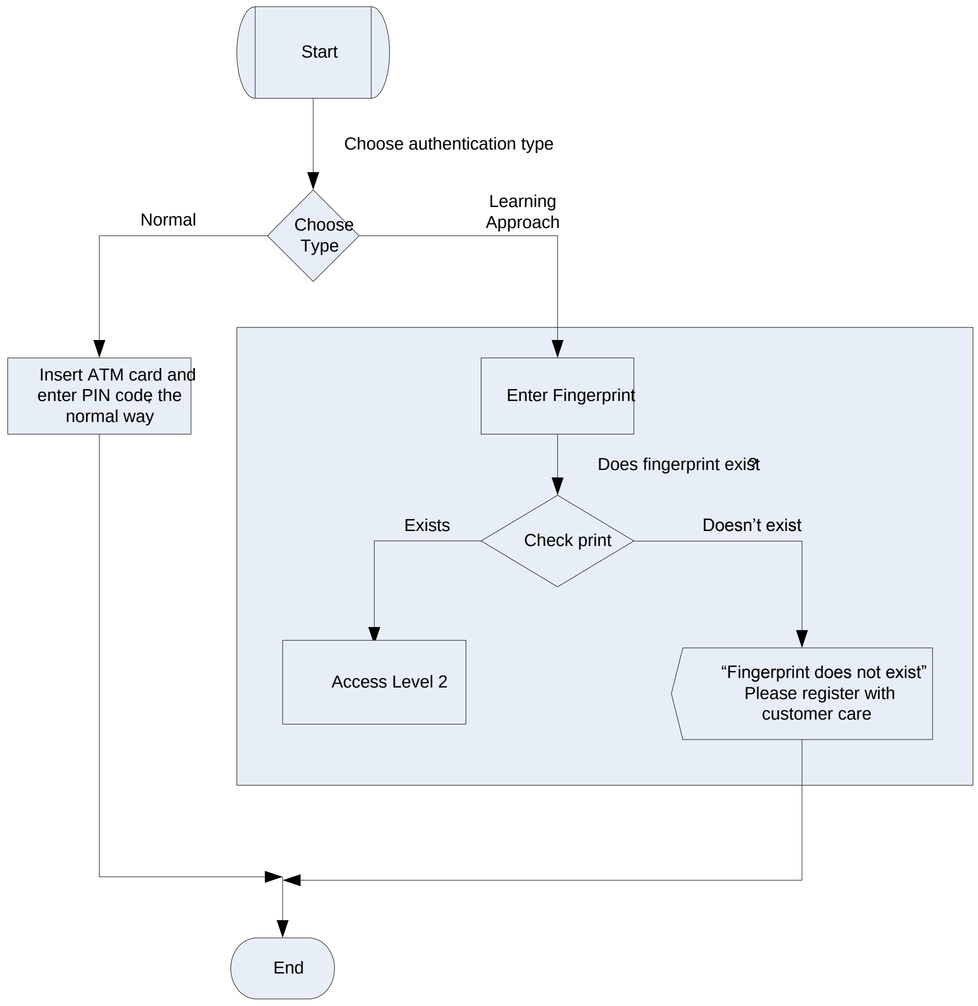

Start
Choose authentication type
Normal
Choose Type
Learning Approach
Insert ATM card and enter PIN code the normal way
Enter Fingerprint
Does fingerprint exist?
Exists
Check print
Doesn't exist
Access Level 2
"Fingerprint does not exist" Please register with customer care
End

Figure 4: Level 1 authentication

The proposed system will have a two-tier security structure. First the fingerprint is acquired and matched against existing prints in the database. If a match is found, the user is passed on to the second tier of authentication.

If the print search fails, the user is passed a message telling them to register their print probably at the customer care desk.

After successful passing of authentication level one, the customer enters authentication level two which is described below.

Access level 2

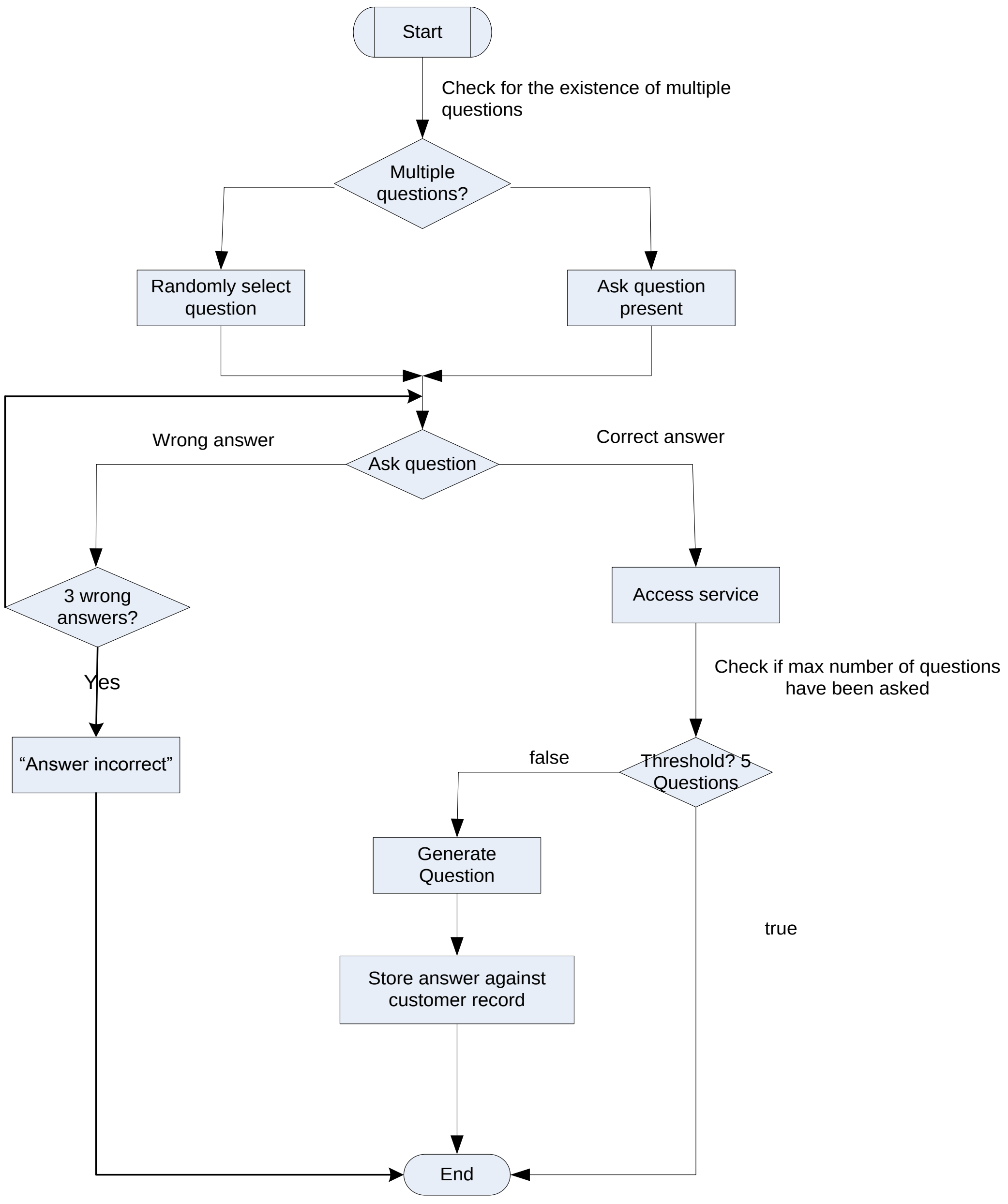


Figure 5: Level two authentication

This is the machine learning algorithm that the learning component introduces. The ATM system's main task will be "learning" its customers. The way it will know its customers is have a database of random questions that are mapped to existing customer records.

First the customer will have to pass their fingerprint against an embedded scanner and their fingerprint is checked for matches. If no match is found, the customer is informed. If on the other hand a match is found the customer accesses level two.

In access level 2, the ATM utilizes its learning component by pulling random questions from the user's profile. For example, the system can ask the user whether they are married or not, their birthday, how many children they have, their spouse's first names, their ID card number amongst others.

This randomly generated questions answers are then stored against the customer record in the database. Every new time a user approaches the ATM to use it and selects the learning approach a question they had answered previously is asked. If they answer correctly, which they should they are informed they are logged in. The user is asked another randomly generated question only that this time this question is just being stored under the user profile. The ATM will use it to further enhance its knowledge of the user up to a specified threshold value.

On the other hand if a user passes the fingerprint test (level one) and accesses level two they are asked a question they have only three attempts to provide the correct answer. Question with yes/no answers will be avoided.

The learning component will therefore entail the ATM learning the user through the questions and answers that the customers provide. Level two authentication is what includes machine learning.

Since both AQ and ID3 require Boolean functions to implement the classification a method of generating the questions asked has to be formulated so that it stays random for security to be enforced.

The randomness will be implemented using the following logic.

If a user has already registered him/herself with the bank, all the questions they have answered and their respective answers are pulled into a temporary table.

All this records are given random weights between 0 and 1. The random weights will be generated by the sql server functions RAND ( ) and NEWID( ). E.g. the statement select top 1 rand() as Weight, question from Question_Lookup order by weight desc. This statement picks the question that has the highest weight.

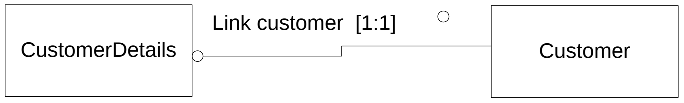
When the customer is being asked a question to answer, from their knowledge base, the sql server function NEWID () is used; e.g. select top 1 question from question_lookup order by newid() will pick a question randomly.

The one with the highest weight is picked and that is the question passed to the customer. Every time a customer comes their respective questions are loaded into the temporary table which lasts for the duration of the session the user uses the ATM.

If a user is new, all the questions in the question lookup file are assigned a random weight between 0 and 1. The question with the lowest weight is picked. All the questions picked from the question lookup files are the ones that have not been previously answered by the customer. For entirely first time customers the question asked may be related with their master record, e.g. Their ID card number or e-mail address.

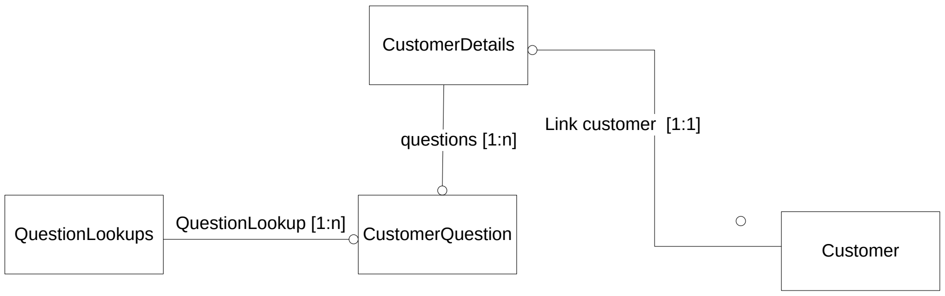
Thus by incorporating the logic above the learning component will be dynamic and thus no user can expect to be thrown the same question time and again. When the questions reach the threshold, between 5 and 10 based on customer preference, the user is exempted from being asked more questions for storage for a period of time. After a fixed duration of time the previously answered questions are answered and the user is taken through the process again, this time for the ATM to get the updated information.

3.2 Database Design



Since we are introducing a learning component, the database structure will not change much. A table customerDetails is added to the already existing framework. Its primary key will be customerId which creates a one-to-one relationship with the customer table.

Customer details will contain the to the tables that will contain questions that will be used to authenticate users. Lookups are the main tools that will be used to authenticate users using question answer authentication.



The E-R model above symbolizes the overall relationship between customer and the learning component tables.

The overall database schema is displayed below. Though the main concern is the learning component and the assumption is that we are not interested in other aspects of the system, the database model created is all inclusive so that the working data would be relevant.

The database schema below shows the entities developed and their respective cardinalities.

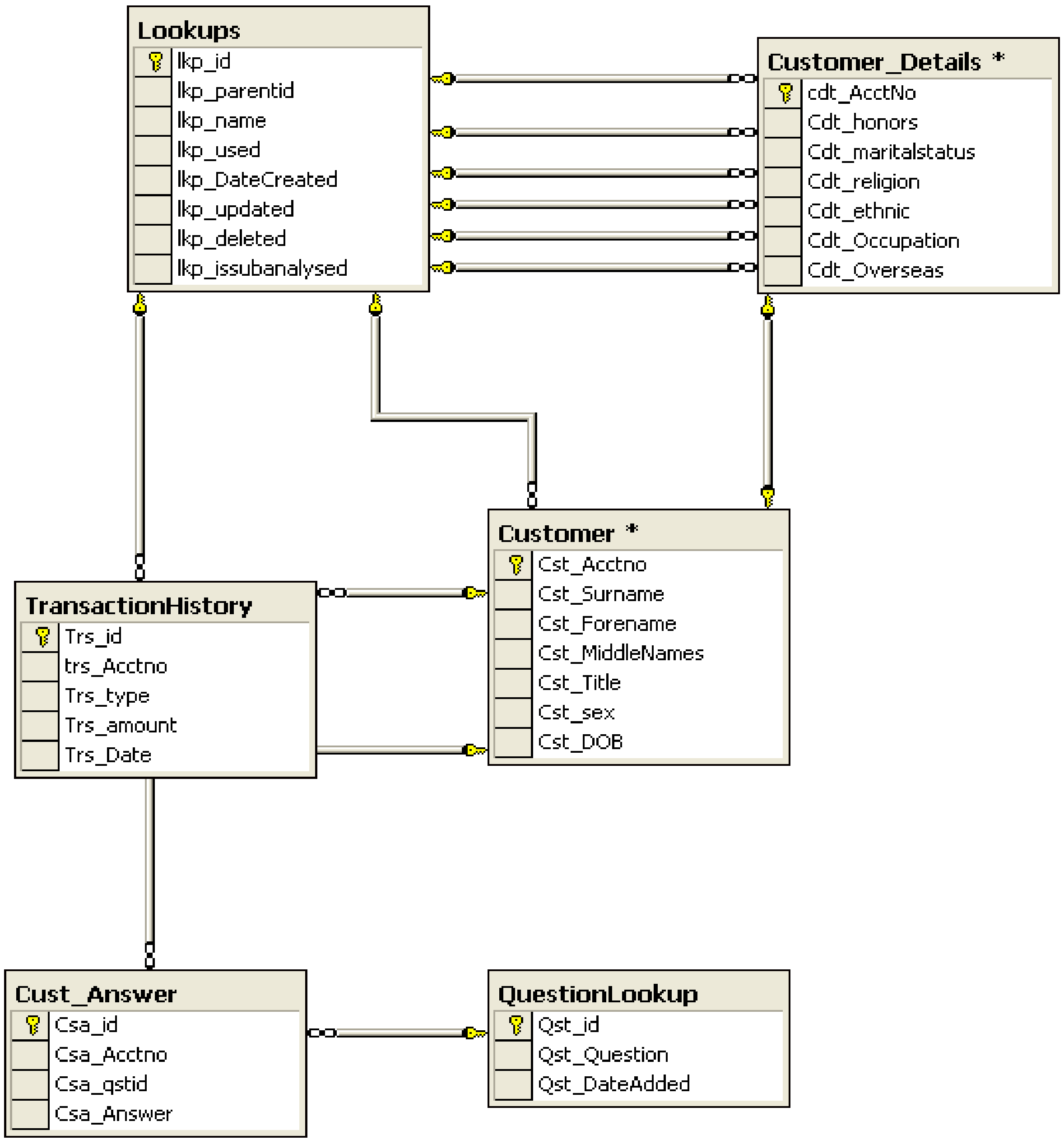


Figure 6: Learning component overall database schema

The data flows in the learning component are illustrated in the DFD below.

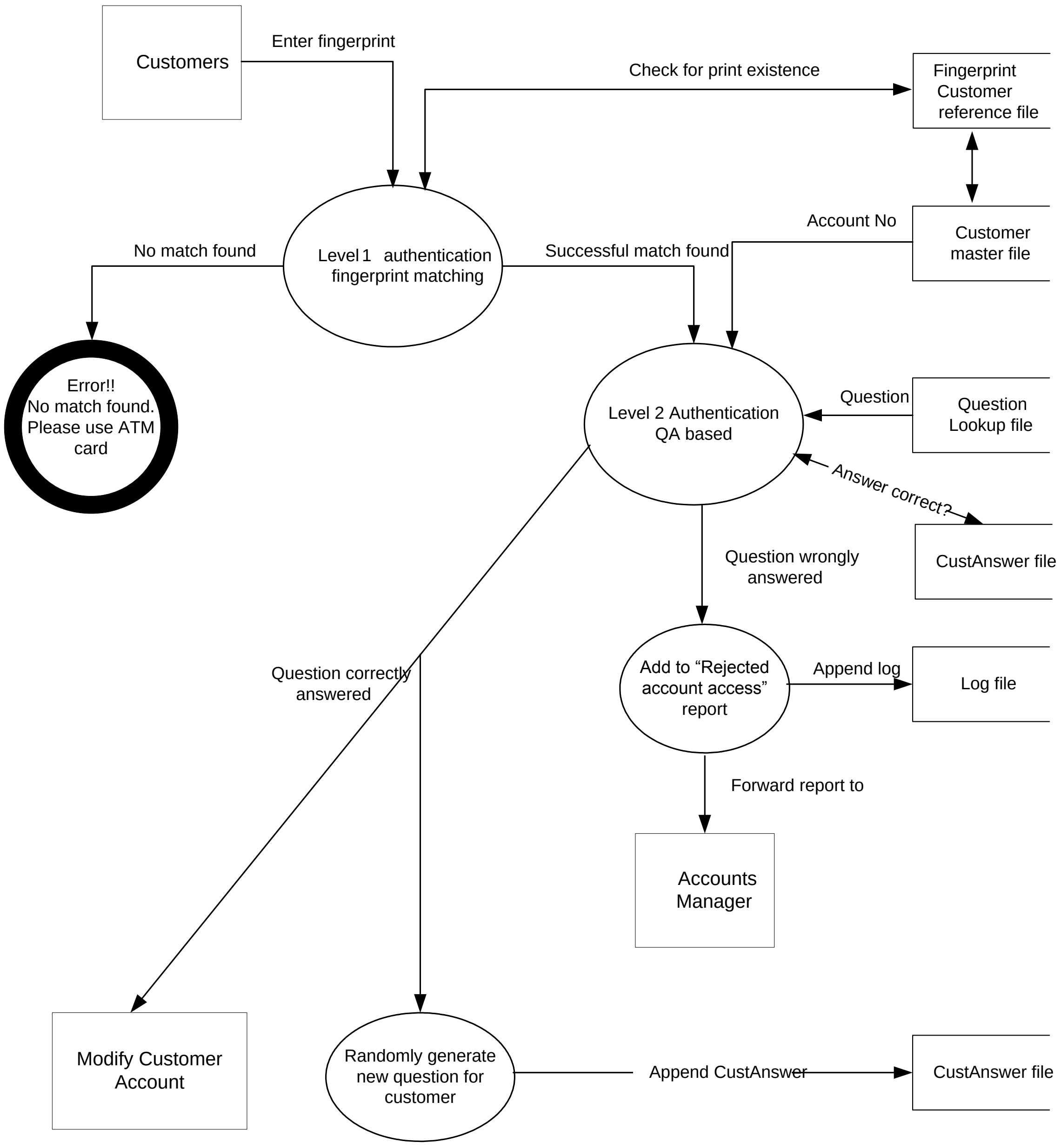

Customers
Enter fingerprint
Check for print existence
Fingerprint Customer reference file
Customer master file
Account No
No match found
Level 1 authentication fingerprint matching
Successful match found
Error!! No match found. Please use ATM card
Level 2 Authentication QA based
Question
Question Lookup file
Answer correct?
CustAnswer file
Question wrongly answered
Question correctly answered
Add to "Rejected account access" report
Append log
Log file
Forward report to
Accounts Manager
Modify Customer Account
Randomly generate new question for customer
Append CustAnswer
CustAnswer file

Figure 7: Overall system-wide DFD for the learning component

The conceptual design above is based on the general-to-specific ordering in concept learning approach as the base learning algorithm. The basis is on the type of classification used. To correctly classify a customer the learning component has to search from a list of possible classifications.

In level 1, the learning component first searches through the set of fingerprints to check whether it will be able to pull a match. If a match is found successfully it pulls the customer account number from the customer master record, searches the question lookup file to randomly pull out a question that directly involves the account number that has been pulled.

The customer is given one opportunity to answer correctly. If he does answer correctly he/she may be asked another question but this will be stored for future reference.

The tasks above can be summarized in the following pseudocode:

```
begin
choose method of service
if method = 'learning' then
begin
      prompt user to enter left thumbprint
      capture print
      compare print with already existing prints in database
      if no match then
      begin
```

```
          message "Print was not found. Please register with
customer care"
          exit (1)
     end
     else
     begin
          randomly generate question from database
          (using the sql server RAND() and NEWID() function)
          capture answer from customer
          match answer
          if match successful
          begin
               message "You are now successfully logged on to
               this session"
               Message "Please answer the following question
               for future reference"
               generate question
               capture answer
               map answer against customer record
          end
          else
          begin
               if wrong answers < = 3
                    message “Incorrect answer”
```

```
                else
                     message "Question incorrectly answered"
                     message "Please use your ATM card normally
                     to  enjoy our services"
                     store information in log
                exit(1)
           end
      end

end
```

end

This chapter detailed the overview of the learning component design. The preceding chapter gives an overview of the outcome, (the learning component developed).

CHAPTER 4: RESULTS

Chapters one to three introduce the concept of integrating a learning component in the already existing ATM framework. This chapter contains screen dumps of the learning component prototype developed to illustrate this concept.

4.1 Sample screens

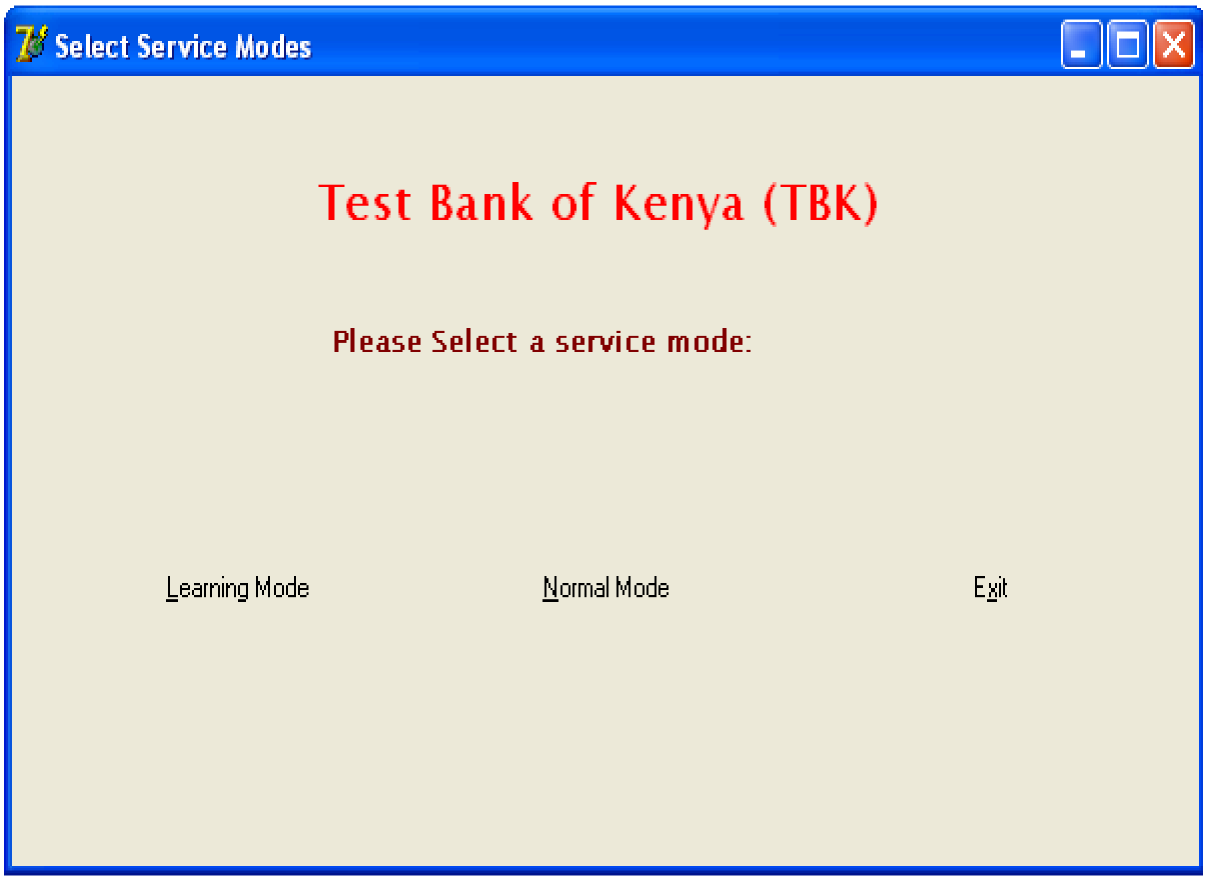
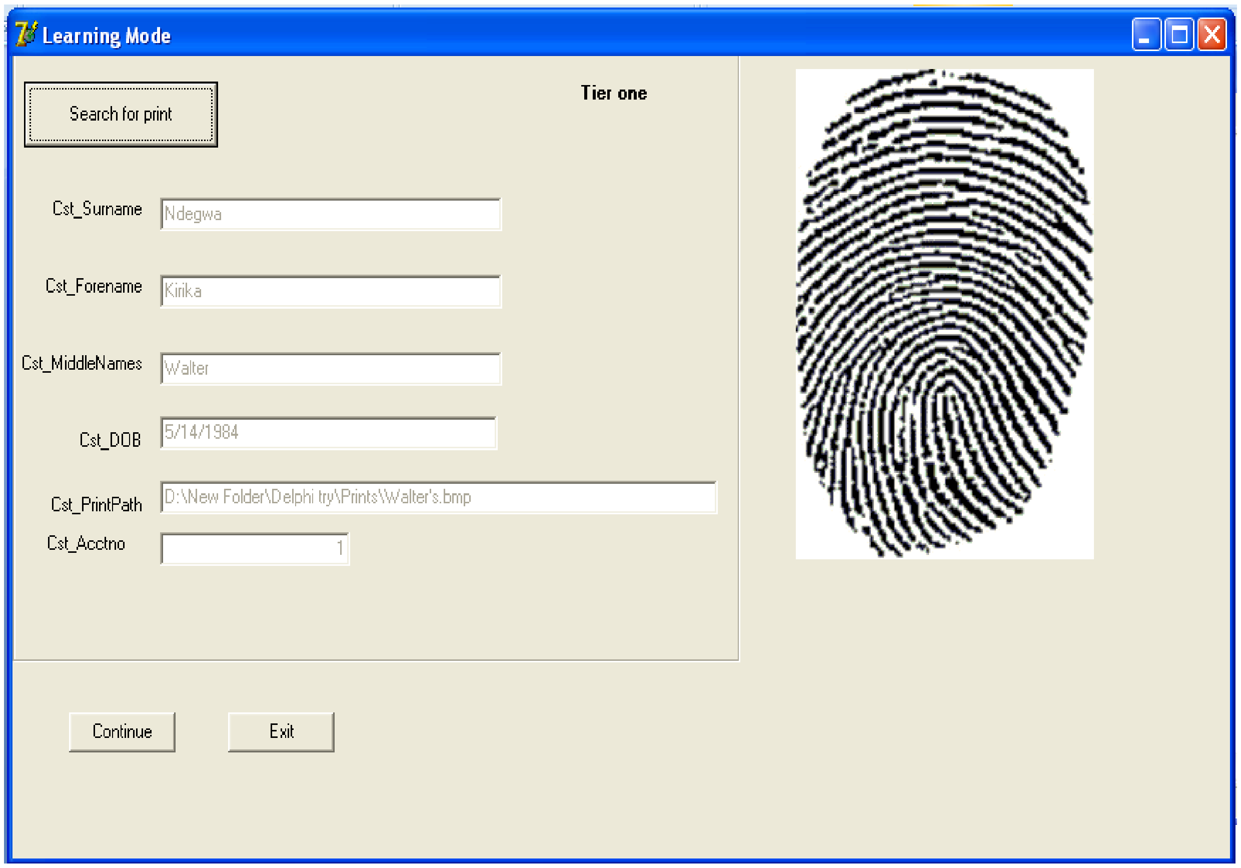


Figure 8: Screen dump of how the learning component (from prototype) is integrated

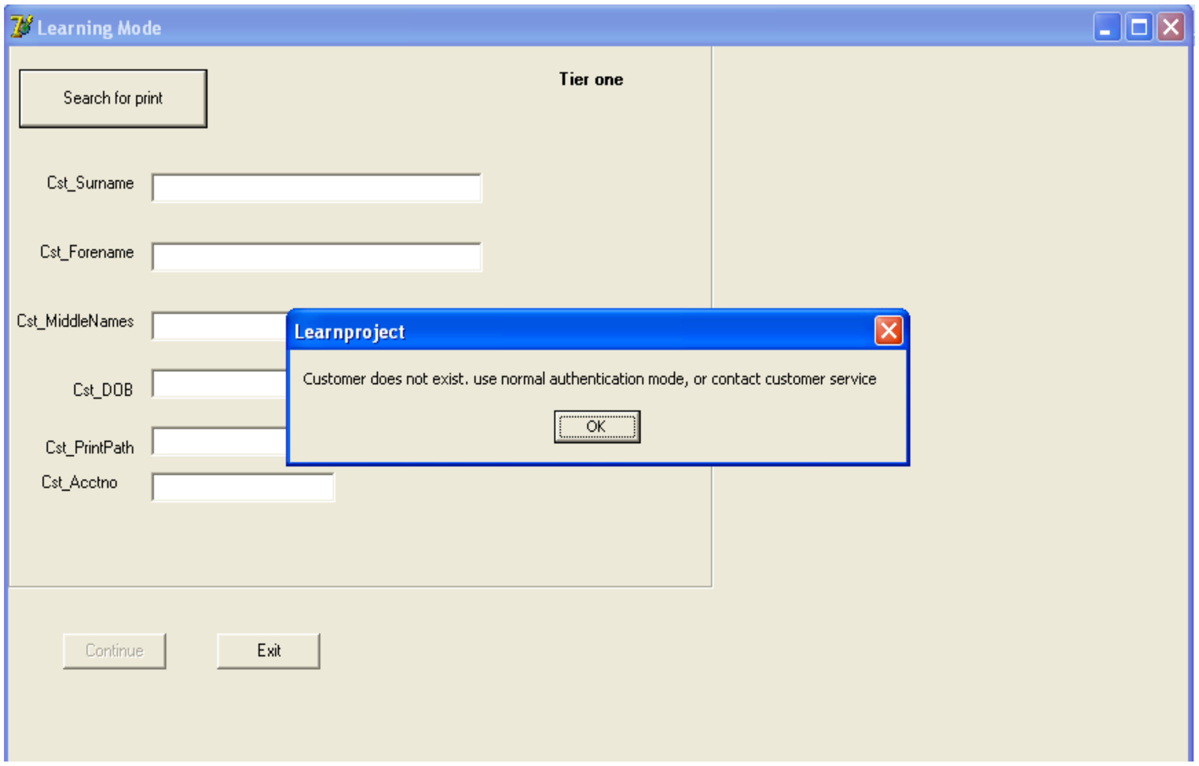
If service mode chosen is learning,



Cst_Surname: Ndegwa
Cst_Forename: Kirika
Cst_MiddleNames: Walter
Cst_DOB: 5/14/1984
Cst_PrintPath: D:\New Folder\Delphi try\Prints\Walter's.bmp
Cst_Acctno: 1

Figure 9: Screen dump when fingerprint is found in the database

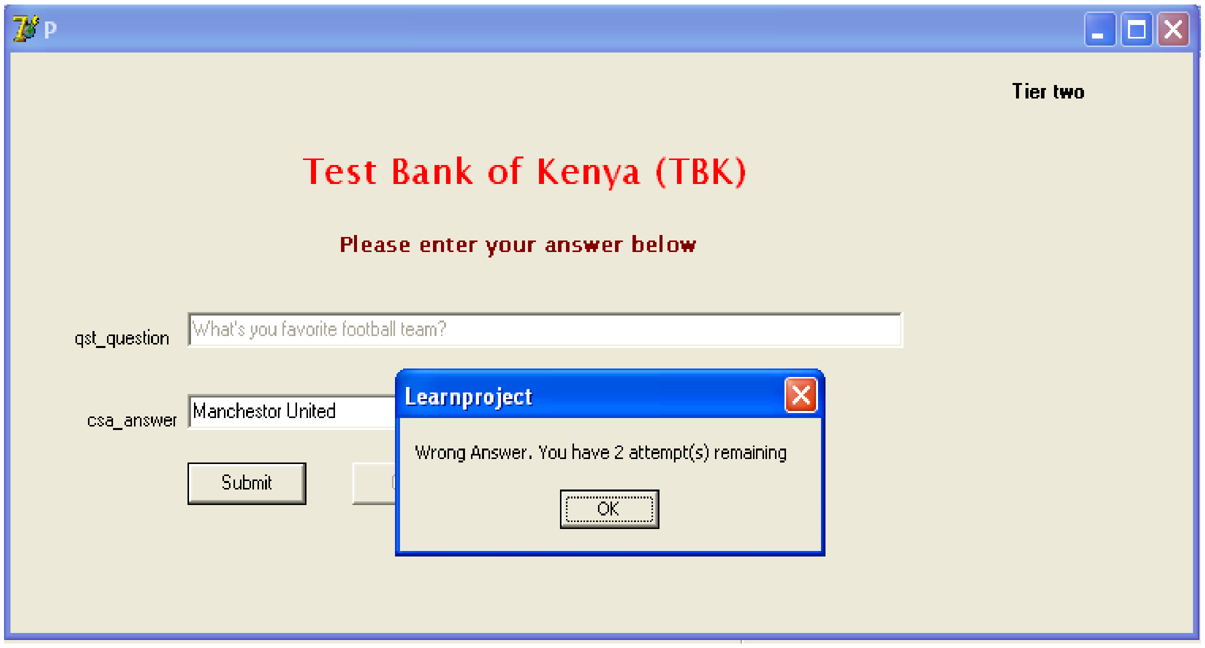
The user is prompted to enter their account number after which they swipe their fingerprint on the scanner. The fingerprint is searched against the users' database records. If a match is found the screen above is displayed. However if a match is not found the exception below is thrown.



Customer does not exist. use normal authentication mode, or contact customer service

Figure 10: Screen dump when fingerprint is not found in the database

If a wrong answer is submitted, the warning message below is displayed:



What's you favorite football team?

Manchestor United

Wrong Answer. You have 2 attempt(s) remaining

Figure 11: Screen dump when the user submits a wrong answer

The screens above depict how the final component looks like. The user makes a selection on which service method they require. If they choose normal, that is beyond the scope of the learning component.

4.2 White Box testing

Using the learning component pseudocode, we can determine the CC. The CC = the number of conditionals / predicates plus one.

The number of independent execution paths includes

- Normal use of ATM.

- Learning option, fingerprints match, wrong answer provided
- Learning option, fingerprint non-existence (no match)
- Learning option, fingerprints match, right answer provided

Test samples will be derived from each of the execution paths above. All the execution paths were tested on the learning component with the tests turning out successful.

4.3 Black Box Testing

The black box mode of testing used is applying V & V. Confirm whether the product is being built right and whether we are building the right product. Validation always involves comparison against requirements. Here functional testing will be done to ensure that the system performs its functions correctly and it produces expected results from the inputs provided.

The preceding chapter contains the conclusions drawn from this study.

CHAPTER 5: CONCLUSION AND FUTURE WORK

Chapters one through to four of this study included a study to identify how a learning component could be integrated to the already existing ATM framework. The study established that indeed a learning component could be added and the machine learning algorithm chosen was AQ. Chapter four included screen dumps of the prototype developed to illustrate the functioning of the learning component. The objectives of the study included:

- Study a subset of already existing machine learning algorithms. This was achieved by taking a comprehensive look at inductive learning algorithms and the algorithms mainly focused on were AQ and ID3.
- Design a sample learning component that can be integrated in the existing ATM system. This was achieved by developing a sample prototype of an ATM system that uses Biometrics and Question Answer authentication as its mode of learning.

The result of this study is a simple learning algorithm that can be used to train ATMs. The prototype solution can be used as a skeleton by other researchers to develop other fool proof learning algorithms that can be used to successfully train ATMs and other applications in a broad range.

The AQ algorithm was chosen as the basis for the learning component since it produces a set of if…Else…Then rules which is the application of the divide and conquer approach. Since the customer attributes here are not deterministic at the time of customer application (customer opening an account), the rules for the AQ algorithm will be specific for each and every customer who may want to use the Learning component. This in essence means that each customer will have their own specific knowledge base. The AQ rules will be induced from the particular knowledge base for a specific customer since a customer will first have to enter his/her account number before using the learning component.

For instance, if customer XX has answered two questions in his knowledge base; e.g. What's your favorite football team and What was the name of your first pet; and the answers were Arsenal and Oyugi respectively, the AQ rules induced would be

```
If favorite football team = Arsenal then
 Customer=true
Else if first pet name = Oyugi then
  Customer = true
Else
  Customer = false
```

The learning component can be developed but in employing it some benefits of the conventional ATM cards would be lost. These include:

i) Ability to conduct offline transactions. Since data is stored in the magnetic stripe in the ATM, it can still be used in an instance where a bank is offline for a while.

ii) The fingerprints would have to be stored in a central server, this increases computational overhead and also translates to, if the bank goes offline ATM services for the learning component will not be available.

On the flip side, the learning component provides more accessibility and more user mobility, since users don't have to carry their ATM card everywhere they go.

The other problematic area in the learning component is formulation questions which when combined with their answer pair will constitute the knowledge base.
The security of a challenge question system is related directly to the confidentiality of the challenge question answers. Other properties such as integrity and availability are also important to the security of the overall system.
The following security criteria apply primarily to the content of individual questions and

answers:

Guessing difficulty

Answers should be difficult to guess and have an answer space with a fairly uniform distribution.

Observation difficulty

The answers to challenge questions should be difficult for an attacker to retrieve or observe easily. In particular, the answers should not be available from public sources.

Questions that individuals are often asked to answer, such as “What is your mother’s maiden name?” do not pose much observation difficulty. Unlike guessing difficulty, a determination of a question’s observation difficulty is more subjective, as the difficulty of determining the answer is dependent upon a number of factors (e.g., the availability of the answer). In addition, observation difficulty will differ for individuals that have different relationships with the user—for example, family, friends, acquaintances, colleagues, or strangers.

New algorithms can be based on already existing ones or researchers could even develop a totally new machine learning algorithm that mainly focuses on learning the customer and later create a generic algorithm that could be used in a broad spectrum of applications.

REFERENCES

Alpaydin, E., 2004. Introduction to Machine Learning (Adaptive Computation and Machine Learning), MIT Press, ISBN 0262012111

Anthes, G. H., 2006. Machine Learning. Computerworld, February 6.

Ballard, L., 2002. Topics in Machine Learning: Decision Tree Learning. Available at: http://www.cs.brandeis.edu/~cs113/classprojects/~lballard/cs113/proj1.html [Accessed 17 November 2007].

Banerji, R. B. 1964. A Language for the description of Concepts. General Systems, 9, p.135-141.

Bundy, A., 1985. Incidence calculus: A mechanism for probabilistic reasoning, in: *Proceedings Workshop on Uncertainty in Artificial Intelligence*, UCLA, Los Angeles, CA, p.177-184.

Clark, P., 1990. Machine Learning: Techniques and Recent Development. Available at: http://citeseer.nj.nec.com/clark90machine.html   [Accessed 17 August 2007].

Cohen, B. L., & Sammut, C. A. 1982. Object Recognition and Concept Learning with CONFUCIUS. *Pattern Recognition Journal*, 15(4), p.309-316.

Heylighen, F., 1997. Occam's Razor. Available at: http://pespmc1.vub.ac.be/occamraz.html [Accessed 02 January 2008].

Hild, H., 1989. Variations on ID3 for Text to Speech Conversation. Available at: ftp.cs.orst.edu/pub/tgd/papers/hild-ms-thesis.ps.gz [Accessed 17 August 2007].

Michalski R.S., Carbonell J.G., & Mitchell T.M. (Eds), Machine Learning: An Artificial Intelligence Approach, Morgan Kaufmann, 1983 (Vol. 1), 1986 (Vol. 2).

Mihai, I. F. 2001. Exploring Places Rated and Body Fat Data Using AQ and Crystal Vision. Available at:

www.galaxy.gmu.edu/stats/syllabi/inft979/MihaiPaper.pdf [Accessed 17 August 2007].

Mitchell, T. 1997. Machine Learning, McGraw-Hill, ISBN: 0070428077.

Pomerleau, D. A. 1989. ALVINN: An autonomous land vehicle in a neural network. (Technical Report CMU-CS-89-107). Pittsburgh, PA: Carnegie Mellon University.

Sammut, C. 1993. The Origins of Inductive Logic Programming: A Prehistoric Tale. Available at: http://citeseer.nj.nec.com/sammut93origins.html [Accessed 17 August 2007].

Selfridge, O. G. 2000. AI's Greatest Trends and Controversies. Available at: http://www.aaai.org/AITopics/html/overview.html#trends [Accessed 10 January 2008].

Shannon, C.E. 1948. A Mathematical Theory of Communication, *Bell System Technical Journal*, 27, p. 379-423 & 623-656.

Vere, S. 1975. Induction of Concepts in the Predicate Calculus. In *Proceedings of the Fourth International Joint Conference on Artificial Intelligence*. p.351-356.