Modeling and Simulation of an Adaptive Neuro-Fuzzy Inference System (ANFIS) for Mobile Learning

Ahmed Al-Hmouz, Jun Shen, *Senior Member*, *IEEE*, Rami Al-Hmouz, and Jun Yan

Abstract—With recent advances in mobile learning (m-learning), it is becoming possible for learning activities to occur everywhere. The learner model presented in our earlier work was partitioned into smaller elements in the form of learner profiles, which collectively represent the entire learning process. This paper presents an Adaptive Neuro-Fuzzy Inference System (ANFIS) for delivering adapted learning content to mobile learners. The ANFIS model was designed using trial and error based on various experiments. This study was conducted to illustrate that ANFIS is effective with hybrid learning, for the adaptation of learning content according to learners' needs. Study results show that ANFIS has been successfully implemented for learning content adaptation within different learning context scenarios. The performance of the ANFIS model was evaluated using standard error measurements which revealed the optimal setting necessary for better predictability. The MATLAB simulation results indicate that the performance of the ANFIS approach is valuable and easy to implement. The study results are based on analysis of different model settings; they confirm that the m-learning application is functional. However, it should be noted that an increase in the number of inputs being considered by the model will increase the system response time, and hence the delay for the mobile learner.

Index Terms—Mobile learning, learner modeling, ANFIS, adaptation

1 Introduction

Adaptive mobile learning systems aim to adapt learning content to each individual learner's context (characteristics, interests, needs) in order to improve the effectiveness of accomplishing a learning activity. Mobile learning has become widespread. Personal digital assistants (PDAs) and mobile phones are being used by a range of learners in a variety of different environments.

Awareness of context is important because it allows the environment to be used in a way that supports the learner. Context and context awareness are key aspects to consider in the design of adaptive learning systems in order for mobile learning systems to be truly effective [1].

Personalisation systems are based on storing and exploiting information about each learner. Learner preferences can predict learner behavior, which is considered as a vital concept in order to make appropriate decisions for each learner. To provide personalized services, systems need to be able to make inferences about their learners' preferences and expectations; this can be achieved by making assumptions about learners based on their interaction with the system. To achieve this we apply techniques such as machine learning algorithms.

- *A. Al-Hmouz, J. Shen, and J. Yan are with the School of Information Systems & Technology, Faculty of Informatics, University of Wollongong, Wollongong, NSW 2522, Australia. E-mail: aa998@uowmail.edu.au, {jshen, jyan}@uow.edu.au.*
- *R. Al-Hmouz is with the Department of Electrical and Computer Engineering, Faculty of Engineering, King Abdulaziz University, PO Box 80204, Jeddah 21589, Kingdom of Saudi Arabia. E-mail: ralhmouz@kau.edu.sa.*

Manuscript received 24 Apr. 2011; revised 25 Nov. 2011; accepted 5 Dec. 2011; published online 12 Dec. 2011.
For information on obtaining reprints of this article, please send e-mail to: lt@computer.org, and reference IEEECS Log Number TLT-2011-04-0052.
Digital Object Identifier no. 10.1109/TLT.2011.36.

Webb et al. [2] categorize the purposes for which learner models are developed as: classifying the cognitive processes underpinning a user's actions; depicting the differences between the skills of a specific user and an expert; and identifying the behavioral patterns and characteristics.

Learner adaptive systems should be able to infer learner behavior in order to provide personalized services. A learner model is generated to include all the information that the system knows about a learner. It is generally initialized either with default values or by querying the learner. The reasoning engine combines the learner profile with other models within the system to derive new "facts" about the learner.

An adaptive mobile learning system is a mechanism that performs adaptation based on a learner model, and updates that learner model with the newly derived facts. A mobile learning application consists of interactive systems that deal with imprecise information. Its interpretation is typically vague and uncertain. Several systems for user modeling apply uncertainty techniques, as described in [3].

Machine learning techniques are commonly used for learner modeling because of the complexity of relationships between learner contexts, which are difficult to represent. Webb et al. [2] listed four main issues related to the use of machine learning techniques for modeling purposes: the need for large data sets, the need for labeled data, concept drift, and computational complexity. Most personalisation approaches depend on machine learning techniques that require a large amount of labeled data in order to provide proper results. The continuous changes in learners' interests and profiles are referred to as concept drift.

This paper presents a mobile learning reasoning engine. The main objective of this reasoning engine is to provide a

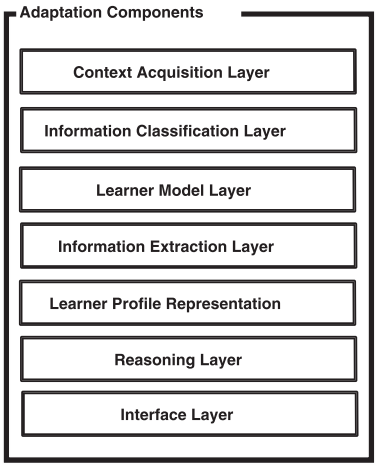

Fig. 1. Adaptation Components [4].

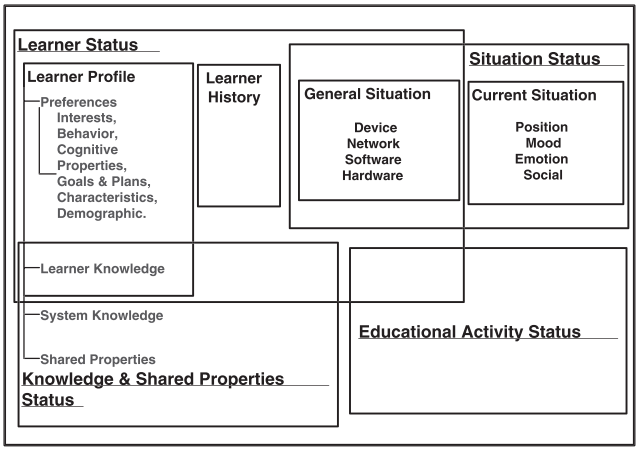

Fig. 2. Learner Model components [6].

suitable learning content format for a mobile learning application, using an Adaptive Neuro-Fuzzy Inference System (ANFIS) based on the acquired learner profile. The learner profile contains a learner's preferences, knowledge, goals, plans, place, and possibly other relevant aspects that are used to provide personalized learning content. The Adaptation Framework and the Enhanced Learner Model in Section 2 provide an overview of some of the key components in our system. Section 2 also provides a brief review of related work. Section 3 presents the structure of the proposed ANFIS. Learning context scenarios, input selection for the system, and the ANFIS system training process are explained in Section 4. Section 5 presents the results and discussion of the proposed system. Finally, Section 6 presents concluding comments about future developments related to this work.

2 Research Background

2.1 Machine Learning-Based User Profile Framework

In [4], we presented a new framework that depicted the process of adapting learning content to satisfy individual learner characteristics by taking into consideration a learner's learning style. In the case of a mobile learning adaptation, seven adaptation layers need to be considered (see Fig. 1).

Context Acquisition Layer. This layer is used to gather the information required for adaptation. It relies on both explicit and implicit information collection. Explicit information relies on information provided by the learner and implicit information is gathered by monitoring the learner's interactions with a system and making assumptions as to his/her needs.

Information Classification Layer. This layer deals with all data obtained from the previous layer by categorizing the data into several class types.

Learner Model Layer. A learner model is defined as a set of information structures designed to represent one or more of the following elements [5]: goals, plans and preferences; characteristics of learners; learner stereotypes and learner behavior.

Information Extraction Layer. This layer assesses, analyses, verifies, and filters the data based on the current learner's situation.

Learner Profile Representation. In order to achieve personalized services, we should be able to specify learner interests. Creating a learner profile allows for much more accurate results, given a sufficiently expressive keyword. Precise information allows the system to more accurately support learner decision making.

Reasoning Layer. The output of this layer is a set of structural descriptions of what has been learned about learner behavior and interests.

Interface Layer. The layer is formed by the events that are processed by the adaptation system as well as the questions about the learner that it can answer.

This paper proposes a solution to adapt learning content through the use of mobile technology, based on modeling the learner and all possible contexts related to his/her current situation.

2.2 Enhanced Learner Model

In [6], we presented a new approach to combine learner information and contexts together to achieve maximum integration, which resulted in information rich learner profiles. This Enhanced Learner Model consists of four main components, namely the representation of the *Learner Status*, the *Situation Status*, the *Knowledge and Shared Properties Status*, and *Educational Activity Status* (see Fig. 2).

The first main component of the Enhanced Learner Model is the *Learner Status*. It encompasses the learner profile, learner history, general situation, learner knowledge, and the educational activity. It describes the learner using assumptions about his knowledge and preferences, the interaction history, and a description of his general situation. Preferences depict the learner's interest in certain topics; this category captures all the common information that can constitute a learner profile.

The second main component of the Enhanced Learner Model is the *Situation Status*. The learner's current situation and general situation are part of the situation status. This component describes the learner's current situation with its various characteristics in the real world. The general situation describes the overall status of the system, and

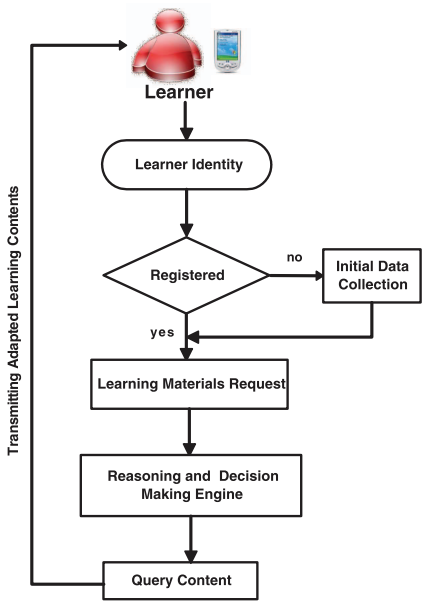

Fig. 3. Mobile learning application workflow.

consists of the following: devices, networks, hardware, and software resources.

The third main component of the Enhanced Learner Model is the *Knowledge and Shared Properties Status*. The learner knowledge is based on assumptions about the learner's knowledge of the system, relationships between inputs, and facts and rules regarding the mobile learning application. The shared properties represent the available tools that help learners to accomplish a task if that task involves teamwork.

The fourth main component of the Enhanced Learner Model is the *Educational Activity Status*, which consists of "any service or facility that supplies a learner with general electronic information and educational content that aids in acquisition of knowledge regardless of location and time" [2].

This paper is focused on modeling the learner and all possible contexts in an extensible way that can be used for personalisation in mobile learning.

2.3 Adaptive Mobile Learning Workflow

The reasoning engine consists of two stages: Fuzzy Logic and Neural Network. The inputs to the adaptation engine include the *Learner Status*, the *Situation Status*, the *Knowledge and Shared Properties Status*, and the *Educational Activity Status* in the form of Learner Profile Representation (see Fig. 1). In this section, we describe the workflow of the mobile learning application (see Fig. 3), which is summarized by the following steps:

The process begins when the learner carries out the learning activity by interacting with the application through the selection of some actions presented on the learner's device interface. If using the application for the first time, the learner will be asked to complete some forms to record personal information and other data through device sensors.

If the learner is registered (i.e., has used the application before), then the learner is willing to participate, and wants to begin the activity. The system issues a start up application to indicate that the learner wants to undertake a learning activity. Learner data are gathered to construct the learner model.

The reasoning engine transforms the content from one state to another in order to meet the constraints of the learner context. The system observes learner behavior and preferences, and recommends educational materials that are similar to those chosen in the past. The reasoning engine selects the proper media type for the learning content based on the learner constraints, and transmits the adapted content to the learner's interface.

2.4 Reasoning Techniques for Learner Modeling

Reasoning approaches try to make assumptions about individual learners based on each user's interaction with the system. Neural Networks, k-Nearest Neighbors (K-NN), Bayesian Network, Genetic and Fuzzy algorithms are some of the techniques that have been used to model the learner contexts. Hybrid systems, consisting of combinations of different machine learning techniques, have also been used. Numerous projects attempting to implement machine learning techniques for learner modeling have been undertaken in recent years. These projects have had a variety of aims and levels of adaptation for individual learners. A sample of such projects is discussed below.

The K-Nearest Neighbor (K-NN) model was proposed by Tsiriga and Virvou for the initialization of the student model in web-based educational applications (web-based Passive Tutor). The main goal of this system was initializing and updating the learner model with the combination of stereotypes. The proposed system was implemented with 117 learners belonging to different stereotypes [7].

A Bayesian Network model for detecting learning styles as defined by Felder and Silverman [8] was proposed by Garcia et al. [9]. The proposed system was implemented with 27 computer science engineering learners taking an Artificial Intelligence course. Garcia et al. compared the results of their approach with the results of a learning styles questionnaire completed by the same 27 learners. This proved that Bayesian networks were effective in predicting the learning styles of the learners with high accuracy.

In [10], another Bayesian learner model was proposed. Tests with simulated learners indicated that the Bayesian network model produced highly accurate estimations of learners' cognitive states.

In [11], a Feed Forward Neural Network model was used to identify the learners' learning styles (as defined by Felder and Silverman [8]). Artificial data were generated and used for experimentation by simulating the actions of learners.

A Bayesian network model to support educational tutors in making decisions under uncertain conditions was proposed by Xenos [12]. The proposed system was implemented with 800 learners studying in an informatics course. The main goal of this system was to model learner behavior in order to make predictions about the success of tutor decisions. Xenos found that the proposed system could be a valuable tool in decision-making under uncertain conditions.

An intelligent tutoring system was proposed by Saldías et al. [13]. The authors used Neural Networks to model learner preferences, which were then used in an adaptive

interface mechanism. The application's main focus was interface configuration.

A Neuro-Fuzzy learner model system was proposed by Stathacopoulou et al. to diagnose the errors of high school learners by collecting data with simulation tools related to a course, namely vectors in physics and mathematics [14]. The system was tested using simulated learner data with different knowledge level categories and behavior corresponding to fuzzy values. A Feed-Forward Neural Network was also trained for error classification.

Most of the projects referred to above have shown that machine learning techniques offer a set of powerful techniques either for learner modeling or for supporting decision-making tasks.

A Neural Network can perform pattern matching to a large number of highly interconnected processing elements. Decision making in a Fuzzy Logic system is based on inputs in the form of linguistic variables. These linguistic variables are derived from membership functions. Neural Networks have shortcomings with their representation of implicit knowledge, while Fuzzy Logic systems are subjective and heuristic. The determination of fuzzy rules, inputs, and outputs depends on trial and error; this makes the design of Fuzzy Logic systems a time-consuming task. These drawbacks of Neural Networks and Fuzzy Logic systems can be overcome by the integration of Neural Network technology with a Fuzzy Logic system. The Adaptive Neuro-Fuzzy Inference System is a Fuzzy Neural Network. The most significant advantage of using ANFIS is that all its parameters can be trained like a Neural Network within the structure of a Fuzzy Logic system [15].

3 Adaptive Neuro-Fuzzy Inference System

3.1 ANFIS Architecture

The Adaptive Neuro-Fuzzy Inference System technique was originally presented by Jang in 1993 [16]. ANFIS is a simple data learning technique that uses Fuzzy Logic to transform given inputs into a desired output through highly interconnected Neural Network processing elements and information connections, which are weighted to map the numerical inputs into an output.

ANFIS combines the benefits of the two machine learning techniques (Fuzzy Logic and Neural Network) into a single technique [16]. An ANFIS works by applying Neural Network learning methods to tune the parameters of a Fuzzy Inference System (FIS). There are several features that enable ANFIS to achieve great success [17], [18]:

- It refines fuzzy IF-THEN rules to describe the behavior of a complex system;
- It does not require prior human expertise;
- It is easy to implement;
- It enables fast and accurate learning;
- It offers desired data set; greater choice of membership functions to use; strong generalization abilities; excellent explanation facilities through fuzzy rules; and
- It is easy to incorporate both linguistic and numeric knowledge for problem solving.

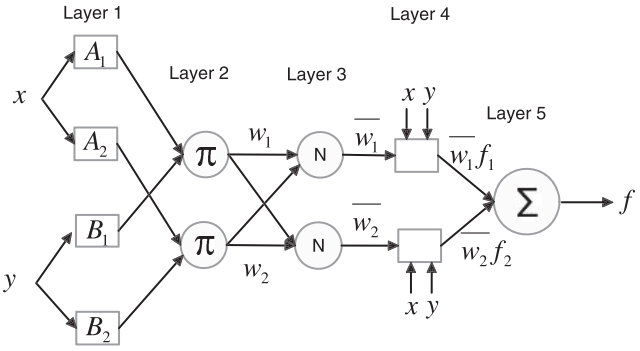

Fig. 4. ANFIS architecture [16].

Different rules cannot share the same output membership function. The number of membership functions must be equal to the number of rules. To present the ANFIS architecture, two fuzzy IF-THEN rules based on a first order Sugeno model are considered:

$$\text{Rule}_{(1)}\text{: } \mathbf{IF}\ x \text{ is } A_1\ \mathbf{AND}\ y \text{ is } B_1, \mathbf{THEN}$$
$$f_1 = p_1 x + q_1 y + r_1.$$
$$\text{Rule}_{(2)}\text{: } \mathbf{IF}\ x \text{ is } A_2\ \mathbf{AND}\ y \text{ is } B_2, \mathbf{THEN}$$
$$f_2 = p_2 x + q_2 y + r_2.$$

where:

- x and y are the inputs,
- A_i and B_i are the fuzzy sets,
- f_i are the outputs within the fuzzy region specified by the fuzzy rule, and
- p_i, q_i, and r_i are the design parameters that are determined during the training process.

Fig. 4 illustrates the reasoning mechanism for this Sugeno model, which is the basis of the ANFIS model.

The ANFIS architecture used to implement these two rules is shown in Fig. 4. In this figure, a circle indicates a fixed node, whereas a square indicates an adaptive node. ANFIS has a five-layer architecture. Each layer is explained in detail below.

In $\text{Layer}_{(1)}$, all the nodes are adaptive nodes. The outputs of $\text{Layer}_{(1)}$ are the fuzzy membership grade of the inputs, which are given by the following equations:

$$O_{1,i} = \mu_{Ai}(x),\ i = 1, 2, \tag{1}$$

$$O_{1,i} = \mu_{Bi-2}(y),\ i = 3, 4, \tag{2}$$

where x and y are the inputs to node i, and A_i and B_i are the linguistic labels (high, low, etc.) associated with this node function. $\mu_{Ai}(x)$ and $\mu_{Bi-2}(y)$ can adopt any fuzzy membership function. For example, if the bell-shaped membership function is employed, $\mu_{Ai}(x)$ is given by

$$\mu_{Ai}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right] b_i},\ i = 1, 2, \tag{3}$$

or the Gaussian membership function by

$$\mu_{Ai}(x) = exp\left[-\left(\frac{x - c_i}{a_i}\right)^2\right], \tag{4}$$

where a_i, b_i, and c_i are the parameters of the membership function.

In $\text{Layer}_{(2)}$, the nodes are fixed nodes. This layer involves fuzzy operators; it uses the **AND** operator to fuzzify the inputs. They are labeled with π, indicating that they perform as a simple multiplier. The output of this layer can be represented as

$$O_{2,i} = w_i = \mu_{Ai}(x) * \mu_{Bi}(y),\ i = 1, 2. \tag{5}$$

These are the so-called firing strengths of the rules.

In $\text{Layer}_{(3)}$, the nodes are also fixed nodes labeled by N, to indicate that they play a normalization role to the firing strengths from the previous layer. The output of this layer can be represented as

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2},\ i = 1, 2. \tag{6}$$

Outputs of this layer are called normalized firing strengths.

In $\text{Layer}_{(4)}$, the nodes are adaptive. The output of each node in this layer is simply the product of the normalized firing strength and a first order polynomial (for a first order Sugeno model). The output of this layer is given by

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i(p_i x + q_i y + r_i),\ i = 1, 2, \tag{7}$$

where $\bar{w}$ is the output of $\text{Layer}_{(3)}$, and p_i, q_i, and r_i are the consequent parameters.

In $\text{Layer}_{(5)}$, there is only one single fixed node labeled with $\sum$. This node performs the summation of all incoming signals. The overall output of the model is given by

$$O_{5,i} = \sum_i \bar{w} f_i = \frac{\sum_i w_i f_i}{\sum_i w_i}. \tag{8}$$

3.2 Hybrid Learning Algorithm

The learning algorithm for ANFIS is a hybrid algorithm that is a combination of gradient descent and least squares methods. In the forward pass of the hybrid learning algorithm, node outputs go forward until $\text{Layer}_{(4)}$ and the consequent parameters are determined by the least squares. In the backward pass, the error signals propagate backward and the premise parameters are updated using gradient descendent. The hybrid learning approach converges much faster by reducing search space dimensions of the original back propagation method [16]. The overall output can be given by

$$f = \frac{w_1}{w_1 + w_2} f_1 + \frac{w_2}{w_1 + w_2} f_2, \tag{9}$$

$$f = \bar{w}(p_1 x + q_1 y + r_1) + \bar{w}(p_2 x + q_2 y + r_2), \tag{10}$$

$$\begin{aligned} f = {} & (\bar{w_1} x) p_1 + (\bar{w_1} y) q_1 + (\bar{w_1}) r_1 + (\bar{w_2} x) p_2 \\ & + (\bar{w_2} y) q_2 + (\bar{w_2}) r_2, \end{aligned} \tag{11}$$

where p_1, q_1, r_1, p_2, q_2, and r_2 are the linear consequent parameters. The least squares method is used to identify the optimal values of these parameters. When the premise parameters are not fixed, the search space becomes larger and the convergence of the training becomes slower.

It should be noted that the ANFIS hybrid algorithm combines two methods, the least squares method and the gradient descent method, to solve the problem of search space. The hybrid algorithm is composed of a forward pass and a backward pass.

The least squares method (forward pass) is used to optimize the consequent parameters. The gradient descent method (backward pass) is used to optimize the premise parameters. The output of the ANFIS is calculated by employing the consequent parameters found in the forward pass. The output error is used to adapt the premise parameters by means of a standard backpropagation algorithm. It has been proven that this hybrid algorithm is highly efficient in training the ANFIS systems [18].

4 ANFIS Architecture for Adaptive Mobile Learning

ANFIS is an intelligent Neuro-Fuzzy technique used for the modeling and control of ill-defined and uncertain systems. ANFIS is based on the input/output data pairs of the system under consideration. ANFIS is selected to solve the problem of continuous changes in mobile learning environments, facilitating the delivery of adapted learning content. The proposed ANFIS model can be used for modeling the learner context. The steps required to apply ANFIS to learner modeling are: define input and output values; define fuzzy sets for input values; define fuzzy rules; and create and train the Neural Network.

To implement and test the proposed architecture, a development tool is required. MATLAB Fuzzy Logic Toolbox (FLT) from MathWorks was selected as the development tool. This tool provides an environment to build and evaluate fuzzy systems using a graphical user interface. It consists of a FIS editor, the rule editor, a membership function editor, the fuzzy inference viewer, and the output surface viewer.

The FIS editor displays general information about a fuzzy inference system. The membership function editor is the tool that displays and edits the membership functions associated with all input and output variables. The rule editor allows the user to construct the rule statements automatically, by clicking on and selecting one item in each input variable box, one item in each output box, and one connection item. The rule viewer allows users to interpret the entire fuzzy inference process at once. The ANFIS editor GUI menu bar can be used to load a FIS training initialization, save the trained FIS, and open a new Sugeno system to interpret the trained FIS model.

4.1 Learning Context Scenarios

Our goal is to construct a system that will help learners to accomplish learning activities by delivering adapted learning content based on the learner's current context. The process of learner modeling has three steps: gathering data related to the learner, creating the learner model, and updating the learner model. To better understand the idea of the enhanced learner model structure, we will take the following typical learning scenarios, and analyze the situation and activities involved:

It is 8:00 am Monday morning and Alison is traveling by bus/car/train from her flat to attend her lecture at the university which

starts at 8:30 am. Alison checks her PDA (3G network, 24 kbps bandwidth) to review her lecture notes.

Sara is in a restaurant for 20 minutes and wants to review her assessments using her smart phone (GSM network, 4 kbps bandwidth).

It is the weekend and Mark is at home for an hour and wants to review his reading materials using his Nokia phone (3G network, 60 kbps bandwidth).

Many contextual elements are contained in the scenarios described above. We can identify different activities, places, times, events, constraints, locations, and other environmental variables. The overall activities in the above scenarios could be labeled as "Learning Activity." These activities can be separated into different elements:

- learning materials—lecture notes, assignments, etc.
- personal information—name, gender, language, occupation, learning style, etc.
- location—home, class, bus, train, road, etc.
- time—weekdays, weekend.
- mobile device—PDA, smart phones, etc.
- environment—sunny, windy, noisy, raining, etc.
- network—3G, GSM, WIFI; and
- bandwidth—low, high, middle.

4.2 Inputs Selection for the System

It is common to have tens of potential inputs that could be used for mobile learning modeling. Input selection is critical when creating a mobile learning system. An excessive number of inputs will increase the computation time necessary for building the model using ANFIS. As a result, it is essential to prioritize all system inputs and design the system accordingly. To build an accurate model for prediction, significant inputs must be selected. In the literature, many techniques have been proposed for input selection. These include ARD [19], CART [20], and the δ-test [21], which determines dependencies within data in order to select relevant inputs. In [22], the author listed some practical considerations for inputs selection:

- Remove noise/irrelevant inputs.
- Remove inputs that are dependant on other inputs.
- Make the underlying model more concise and transparent.
- Reduce the time for model construction.

The method presented in [22] takes advantage of the ANFIS structure [18]. It is based on the assumption that the ANFIS model with the smallest Root Mean Square Error (RMSE) after a small number of epochs has a greater potential of achieving a lower RMSE when given more epochs of training. Hence, if there are 10 candidate inputs and we need to determine the three most influential inputs, then we can construct $(C_3^{10} = 120)$ ANFIS models and the model with the smallest RMSE will be chosen.

In the case of mobile learning, hundreds of potential inputs can be selected. If there are seven inputs, the four most influential inputs could be determined. Each time, the RMSE is calculated for combinations of variables $(C_4^7 = 35)$ and the minimum RMSE is recorded. The algorithm is run for one input at a time, then two inputs at a time, and so on, until it reaches seven inputs at a time.

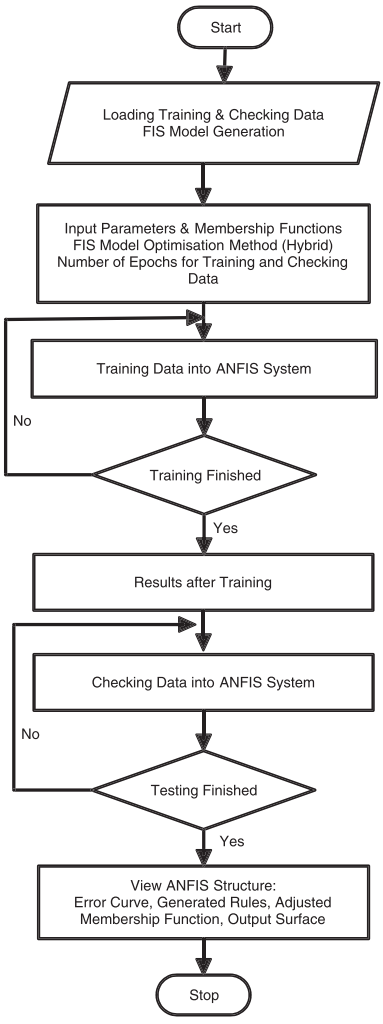

Fig. 5. ANFIS training system.

4.3 ANFIS System Training Process

The ANFIS system training methodology is summarized in Fig. 5. The process begins by obtaining a training data set (input/output data pairs) and checking data sets. The training data is a set of input and output vectors. Two vectors are used to train the ANFIS system: the input vector and the output vector. The training data set is used to find the premise parameters for the membership functions. A threshold value for the error between the actual and desired output is determined.

The consequent parameters are found using the least-squares method. If this error is larger than the threshold value, then the premise parameters are updated using the gradient decent method. The process is terminated when the error becomes less than the threshold value. The checking data set is then used to compare the model with the actual system [16].

ANFIS training learning rules use hybrid learning, combining the gradient descent and the least squares method. The aim of using ANFIS for adaptive mobile learning is to achieve the best performance possible. ANFIS training begins by creating a set of suitable training data in order to be able to train the Neuro-Fuzzy system. The obtained training data must include as many mobile

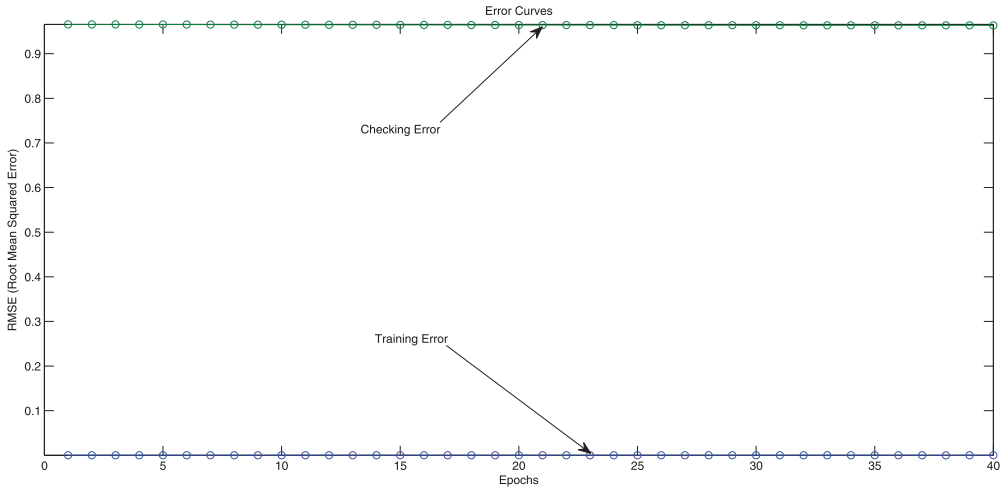

Fig. 6. Error curves.

learning situations (different locations, devices, network bandwidth, etc.) as possible. ANFIS training uses the *anfis* function. Evaluation of the system compared to the desired output is conducted using the *evalfis* function.

The first step is to prepare the training data to work with ANFIS in MATLAB. The data set used as the input to the *anfis* function must be in a matrix form, where the last column in the matrix is the output, and the matrix contains as many columns as needed to represent the inputs to the system. The rows represent all the existing data situations.

Creation of the membership functions is dependent on the system designer. The designer may create the parameters of the membership functions if they have knowledge of the expected shapes, or they can use the command *genfis1* from MATLAB to help in the creation of the initial set of membership functions. This work uses the *genfis1* command to create the membership functions.

Once the initial membership functions are created, system training begins. The command provided by MATLAB to train an ANFIS system is *anfis*. We input the training data we defined, the membership functions we created using the *fismat* command, and some training options in order to produce the most acceptable output.

When the training process is finished (*trn-fismat*), the final membership functions and training error from the training data set are produced (see Fig. 6). The checking data set can be used in conjunction with the training data set for enhanced accuracy. It is possible for ANFIS to function with only one training data set, however input of checking data increases the possibilities to be understood by the system, and hence the system's effectiveness.

After the system training is complete, ANFIS provides a method to study and evaluate the system performance by using the *evalfis* function. The fuzzy system that was established based on the completed training (*trn-fismat*) is used. The process of studying and evaluating system performance starts by entering input data sets into the fuzzy system. These data sets do not include output values. The output of the *evalfis* function represents the response of the system or the final output of the ANFIS system. This response output can be measured by means of correlations between the desired learner contexts and the learning content format as the system output (i.e., input/output). Once the ANFIS is trained, we can test the system against different sets of data values to check the functionality of the proposed system.

To better understand the training steps explained in Fig. 5, the following mobile learning example will demonstrate the ANFIS modeling.

Four input parameters were controlled, namely: Learner Location (**LL**), Network Bandwidth (**NB**), Battery Life (**BL**), and Device Software Capabilities (**SC**); and one output parameter was controlled: Adapted Learner Content (**ALC**) (see Fig. 7).

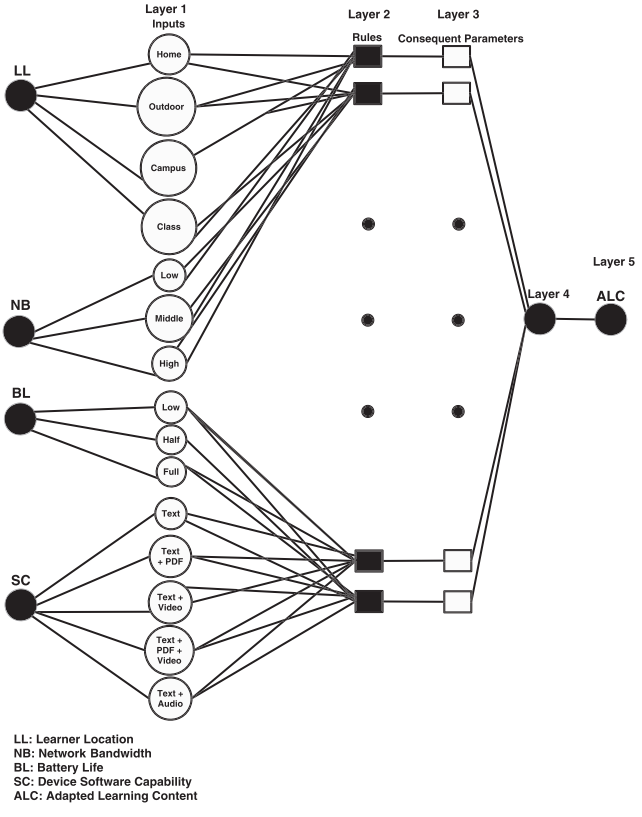

Fig. 7. ANFIS Structure with four inputs and one output.

The conditions that determine the learning content format depend solely on the criteria set by the human expert. Each of the four input conditions is represented by the following term sets. **LL** has four membership functions represented as *Home*, *Campus*, *Class*, and *Outdoor*. **NB** has three membership functions represented as *Low*, *Middle*, and *High*. **BL** has three membership functions represented as *Low*, *Half*, and *Full*. **SC** has five membership functions represented as *Text*, *Text+PDF*, *Text+Video*, *Text+PDF+Video*, and *Text+Audio*.

The Sugeno-type fuzzy-rule-based model has rules in the following forms:

IF (**LL** is *Outdoor*) **AND** (**NB** is *High*) **AND** (**BL** is *Full*) **AND** (**SC** is *Text+Video*) **THEN** (Learning Content Format is *Video*).

IF (**LL** is *Class*) **AND** (**NB** is *Low*) **AND** (**BL** is *Half*) **AND** (**SC** is *Text+PDF+Video*) **THEN** (Learning Content Format is *PDF*).

genfis1 is the function used to generate an initial single-output FIS matrix from training data. First we need to initiate our example model with default values for membership function numbers "4*3*3*5" and types "Gaussian curve (*gaussmf*), Triangular-shaped (*trimf*), and Generalized bell-shaped (*gbellmf*)." These defaults provide membership functions on each of the four inputs, 15 altogether. The generated fuzzy inference system structure contains 180 fuzzy rules. Each rule generated by *genfis1* has one output membership function, which is of type "linear" by default.

Each of the fuzzy rules of ANFIS is a multi input single output structure. The adjustment of membership function parameters is computed by gradient vector, which shows how well the ANFIS is modeled by a given different set of training data consisting of certain conditions.

Several experiments have been conducted to assess whether the proposed modeled ANFIS has produced acceptable results. ANFIS network training involves mapping inputs through input membership functions and mapping output through output membership functions. The parameters associated with each membership function will keep changing throughout the learning process.

The ANFIS training process starts by determining fuzzy sets and the number of sets of each input variable and shape of their membership function. All the training data passes through the Neural Network , to adjust the input parameters to find the relationships between input/output, and to minimize the errors. RMES is the function used to monitor the training errors. RMSE is used and defined as

$$RMSE = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(y_j - \hat{y}_j\right)^2}. \tag{12}$$

Mean Average Error (MAE) is used and defined as

$$MAE = \frac{1}{N}\sum_{j=1}^{N}|y_i - \hat{y}_i|, \tag{13}$$

where N is the total number of prediction, $\hat{y}_j$ is the predicted time series, and y_j is the original series.

Functions from the Fuzzy Logic Toolbox (FLT) were included in MATLAB code. This code was used to solve the input/output problem with different membership function numbers and types for each input. The difference of the RMSE between observed and modeled values was computed for each trial with different epoch numbers, and the best structure was determined by the lowest value of the RMSE.

Overfitting is a common problem in ANFIS model building, which occurs when the data are overtrained by ANFIS. Every data set that is trained using ANFIS has its maximum number of epochs before overfitting occurs; this causes the predicted output to be over its accuracy. The optimal number of epochs can only be found by conducting numerous experiments.

Overfitting is analyzed by plotting the training and checking error values from the ANFIS simulation (see Fig. 6). To avoid the overfitting problem, we will test our model by setting training epoch numbers to different values. This will allow us to determine the optimal epoch number with the lowest RMSE. The Surface Viewer is used to display the dependency of one of the outputs on any one or two of the inputs. It generates and plots an output surface map for the system. The output is presented by a 3D surface model.

In this study, an ANFIS model based on both Neural Network and Fuzzy Logic has been developed to adapt learning content formats for mobile learning application. The experiments were divided into four ANFIS structures to demonstrate learning activities in different contexts. For this purpose, computer simulations using MATLAB were carried out and statistical validation indexes were used for determining the performance of the best proposed model within each context. Once the inputs and output have been identified, it is necessary to validate the quality of the model results after training. Model validation will validate the accuracy and verify whether the model can be easily interpreted.

The performances of the ANFIS models of both training and checking data were evaluated and the best training/checking data set was selected according to RMSE. Prediction accuracy was calculated by comparing the difference of predicted and measured values. In order to find the ideal ANFIS system, we trained the system with a variety of settings for items such as data set sample, epoch number, membership function type and number, and number of inputs to achieve the best performance. The validation tests between the predicted results and the actual results for both training and testing phases are presented in Section 5.

5 Results and Discussion

Various experiments were conducted and the sizes of the training and checking data sets were determined by taking the classification accuracies into consideration.

The data were divided into two separate sets: the training data set and the checking data set. The training data set was used to train the ANFIS, whereas the checking data set was used to verify the accuracy and the effectiveness of the trained ANFIS model for the adaptation of learning content. In order to find out the optimal model to address the problem of mobile learning, we need to

TABLE 1
The $\mathrm{ANFIS}_{(1)}$ Structure Information (Membership Functions Number "3*3*3")

ANFIS parameter type	$\mathrm{ANFIS}_{(1)}$ (1)			$\mathrm{ANFIS}_{(1)}$ (2)			$\mathrm{ANFIS}_{(1)}$ (3)		
Number of Inputs	3								
Membership Functions type	Gaussian curve			Triangular-shaped			Generalised bell-shaped		
Number of Membership Functions	3*3*3								
Training Data Set	10	15	27	10	15	27	10	15	27
Checking Data Set	40	60	80	40	60	80	40	60	80
Epoch Number	10	25	40	10	25	40	10	25	40
Number of Nodes	78			78			78		
Number of Linear Parameters	108			108			108		
Number of Nonlinear Parameters	18			27			27		
Total Number of Parameters	126			135			135		
Number of Fuzzy Rules	27								
Input Combinations	LL, NB, SC								
RMSE	1.0177	0.9652	0.8265	0.9274	0.863	0.7426	1.0094	0.965	0.8217
MAE	0.7085	0.6611	0.5348	0.58	0.5377	0.4795	0.7037	0.6654	0.5314

investigate a number of factors that play important roles in determining this model.

The optimal ANFIS model setting will be selected based a comparison of RMSE values for different epoch numbers and the number of membership functions assigned to each ANFIS structure. Four main aspects will be taken into consideration in relation to ANFIS system training: overfitting, number of membership functions, type of membership functions, and training options (training data sample and epoch number).

$\mathrm{ANFIS}_{(1)}$. Three inputs and membership function number (3*3*3). The first system was structured by selecting three inputs and feeding them into the network. The selection of the inputs was based on the impact of these inputs on the output format for mobile learning. Three input parameters were controlled, namely: Learner Location, Network Bandwidth, and Software Capability. **LL** has three membership functions represented as "*Home*, *Class* and *Outdoor*." **NB** has three membership functions represented as "*Low*, *Middle*, and *High*." **SC** has three membership functions represented as "*Text*, *Text+PDF*, and *Text+PDF+Video*."

First we will test our model with default values for membership function number (3*3*3); membership function types "Gaussian, Triangular-shaped and Generalized bell-shaped" are used. These defaults provide membership functions on each of the three inputs, nine altogether. The generated fuzzy inference system structure contains 27 fuzzy rules.

Table 1 shows that the Triangular-shaped (*trimf*) membership function performs most effectively with minimum RMSE during validation with the epoch numbers 10, 25, and 40, and the Gaussian (*gaussmf*) membership function produced the worst results with epoch numbers 10, 25, and 40.

$\mathrm{ANFIS}_{(2)}$. Four inputs and membership function number (3*3*3*3). Similar to the $\mathrm{ANFIS}_{(1)}$, the only difference is the inclusion of another input which is Battery Life. Four inputs were considered in the second experiment. **LL** has three membership functions represented as "*Home*, *Class*, and *Outdoor*." **NB** has three membership functions represented as "*Low*, *Middle*, and *High*." **BL** has three membership functions represented as "*Low*, *Half*, and *Full*." **SC** has three membership functions represented as "*Text*, *Text+PDF*, and *Text+PDF+Video*".

LL, **NB**, **BL**, and **SC** are the model inputs, with default values for membership function number (3*3*3*3); membership function types "Gaussian, Triangular-shaped and Generalized bell-shaped" are used. These defaults provide membership functions on each of the four inputs, twelve altogether. The generated fuzzy inference system structure contains 81 fuzzy rules. The summary of the tested membership function numbers and types with their respective epoch numbers is tabulated in Table 2.

TABLE 2
The $\mathrm{ANFIS}_{(2)}$ Structure Information (Membership Functions Number "3*3*3*3")

ANFIS Parameter Type	$\mathrm{ANFIS}_{(2)}$ (1)			$\mathrm{ANFIS}_{(2)}$ (2)			$\mathrm{ANFIS}_{(2)}$ (3)		
Number of Inputs	4								
Membership Functions Type	Gaussian curve			Triangular-shaped			Generalised bell-shaped		
Number of Membership Functions	3*3*3*3								
Training Data Set	40	60	81	40	60	81	40	60	81
Checking Data Set	80	100	180	80	100	180	80	100	180
Epoch Number	10	25	40	10	25	40	10	25	40
Number of Nodes	193			193			193		
Number of Linear Parameters	405			405			405		
Number of Nonlinear Parameters	24			36			36		
Total Number of Parameters	429			441			441		
Number of Fuzzy Rules	81								
Input Combinations	LL, NB, BL, SC								
RMSE	0.9524	0.8202	0.7708	0.919	0.825	0.775	0.9442	0.8129	0.7665
MAE	0.6501	0.5041	0.494	0.626	0.5069	0.4965	0.6439	0.4995	0.4898

TABLE 3
The $\mathrm{ANFIS}_{(3)}$ Structure Information (Membership Functions Number "4*3*3*5")

ANFIS Parameter Type	$\mathrm{ANFIS}_{(3)}$ (1)			$\mathrm{ANFIS}_{(3)}$ (2)			$\mathrm{ANFIS}_{(3)}$ (3)		
Number of Inputs	4								
Membership Functions Type	Gaussian curve			Triangular-shaped			Generalised bell-shaped		
Number of Membership Functions	4*3*3*5								
Training Data Set	100	150	180	100	150	180	100	150	180
Checking Data Set	200	300	500	200	300	500	200	300	500
Epoch Number	10	25	40	10	25	40	10	25	40
Number of Nodes	397			397			397		
Number of Linear Parameters	900			900			900		
Number of Nonlinear Parameters	30			45			45		
Total Number of Parameters	930			945			945		
Number of Fuzzy Rules	180								
Input Combinations	LL, NB, BL, SC								
RMSE	0.8153	0.8047	0.8263	0.8059	0.7921	0.808	0.8185	0.8012	0.8248
MAE	0.5093	0.4971	0.5336	0.5182	0.5032	0.5354	0.5132	0.4958	0.5336

As shown in Table 2, for the model with four inputs, that the Triangular-shaped (*trimf*) membership function with the epoch number 10 and the Generalized bell-shaped (*gbellmf*) membership function with the epoch numbers 25 and 40 are selected as the best fit model for describing the mobile learning scenarios. The Triangular-shaped (*trimf*) membership function with epoch numbers 25 and 40 and the Gaussian (*gaussmf*) membership function perform the worst with maximum RMSE during validation.

$\mathrm{ANFIS}_{(3)}$. Four inputs and membership function number (4*3*3*5). Similar to the $\mathrm{ANFIS}_{(2)}$, four inputs were controlled: **LL**, **NB**, **BL**, and **SC**. This model was previously explained and shown in Fig. 7. Table 3 shows, for this model, that the Triangular-shaped (*trimf*) membership function with the epoch numbers 10, 25, and 40 is the best fit model, and that the Trapezidal-shaped (*trapmf*) and Gaussian (*gaussmf*) membership functions perform the worst.

$\mathrm{ANFIS}_{(4)}$. Five inputs and membership function number (2*2*2*2*2). Similar to the previous ANFIS structures, the only difference is that five inputs are included, with the addition of the input Environment Status (**ES**). Five inputs were considered in the fourth experiment. **LL** has two membership functions represented as "*Home* and *Outdoor*." **NB** has two membership functions represented as "*Low* and *High*." **BL** has two membership functions represented as "*Low* and *Full*." **SC** has two membership functions represented as "*PDF* and *PDF+Video*." **ES** has two membership functions represented as "*Sunny* and *Raining*." Table 4 shows, for this model, that the Generalized bell-shaped (*gbellmf*) membership function with the epoch numbers 25, 30, and 40 is the best fit model, and that the Triangular-shaped (*trimf*) membership function performs the worst.

The experimental results shown in the tables above demonstrate that the fuzzy rule base selected by the human expert produces consistent results for the test data used [10, 15, and 27 ($\mathrm{ANFIS}_{(1)}$), 40, 60, and 81 ($\mathrm{ANFIS}_{(2)}$), 100, 150, and 180 ($\mathrm{ANFIS}_{(3)}$)) and 15, 25, and 31 ($\mathrm{ANFIS}_{(4)}$)]. However, not all situations are covered by the human expert's fuzzy rules and some missing rules are detected by ANFIS. Tables 1, 2, 3, and 4 show that all situations for all input attributes are covered by the sets of 27, 81, 180, and 32 rules; however some of the rules have been found to produce illogical decisions because the training sample size was not large enough to cover all possible cases.

The ANFIS models are evaluated based on their performance in training and checking sets as shown in Tables 1, 2, 3, and 4. The ANFIS models have shown significant performance variations against the evaluation criteria in terms of data sample, number of membership functions, and type of membership functions. It appears that the ANFIS models are accurate and consistent in different subsets, where all the values of RMSE and MAE

TABLE 4
The $\mathrm{ANFIS}_{(4)}$ Structure Information (Membership Functions Number "2*2*2*2*2")

ANFIS Parameter Type	$\mathrm{ANFIS}_{(4)}$ (1)			$\mathrm{ANFIS}_{(4)}$ (2)			$\mathrm{ANFIS}_{(4)}$ (3)		
Number of Inputs	5								
Membership Functions type	Gaussian curve			Triangular-shaped			Generalised bell-shaped		
Number of Membership Functions	2*2*2*2*2								
Training Data Set	15	25	31	15	25	31	15	25	31
Checking Data Set	50	70	90	50	70	90	50	70	90
Epoch Number	25	30	40	25	30	40	25	30	40
Number of Nodes	92								
Number of Linear Parameters	192								
Number of Nonlinear Parameters	20			30					
Total Number of Parameters	212			222					
Number of Fuzzy Rules	32								
Input Combinations	LL, NB, BL, SC, ES								
RMSE	0.5885	0.6198	0.6165	0.5948	0.6241	0.6166	0.5858	0.6171	0.6162
MAE	0.3544	0.3882	0.3802	0.3538	0.3895	0.3802	0.3535	0.3867	0.3804

are smaller. It also shows that the lowest value of RMSE is 0.7426 and the highest value of the RMSE is 1.0177 in $ANFIS_{(1)}$. The values of the RMSE and MAE in $ANFIS_{(3)}$ are much lower than in $ANFIS_{(1)}$ and $ANFIS_{(2)}$. $ANFIS_{(3)}$, which consists of four inputs and membership function (4*3*3*5), has shown the highest efficiency and correlation, and the minimum RMSE and MAE. It also shows that the lowest value of RMSE is 0.5858 and the highest value of the RMSE is 0.6241 in $ANFIS_{(4)}$. As a result, $ANFIS_{(3)}$ and $ANFIS_{(4)}$ were selected as the two best-fit models to deliver learning content for mobile learning.

Both the type and number of membership functions are important in building the ANFIS architecture. To investigate the effects of the number and types of membership functions, a variety of different training data sets and checking data were used to examine the impact of the type and number of membership functions. Based on the experiment results in Tables 1, 2, 3, and 4, a number of membership functions are needed to achieve desired performance of the model.

$ANFIS_{(3)}$, which is equivalent to $ANFIS_{(2)}$ in terms of the number of inputs, varies in the number of membership functions and the training sample. $ANFIS_{(3)}$ is more successful even though the same number of inputs are used in both structures. This shows that the number of membership functions and the training data sample have positive training effects on the production of an acceptable system output. Although the process used to create the network for $ANFIS_{(3)}$ was slightly more time consuming, it provided better results.

The required number of membership functions is determined through trials, and based on error values. It indicates that each ANFIS model is very sensitive to the type and number of membership functions. Increasing the number of membership functions per input does not necessarily increase model performance, but usually leads to model overfitting. The RMSE values were used to determine the best membership function and epoch number in order to select the best fit model. From these results, the Triangular-shaped membership function with optimized epoch numbers of 10, 25, and 40 was found to be the best model with the lowest RMSE value for two model structures ("3*3*3" and "4*3*3*5"). The Generalized bell-shaped membership function with epoch numbers 25 and 40 (in $ANFIS_{(2)}$) and 25, 30, and 40 (in $ANFIS_{(4)}$) was found to be the best fit model for describing the mobile learning scenarios in the model structures ("3*3*3*3") and ("2*2*2*2*2").

Tables 3 and 4 show different approximations regarding different epoch numbers and membership functions ("4*3*3*5" and "2*2*2*2*2"). It is shown that the number of training samples has influenced ANFIS behavior by adding new possibilities to produce more acceptable results. It is very important for Fuzzy Logic engine to train the system with a set of training data that includes many different data sample possibilities. Training options are important in building an ANFIS architecture; they include epoch numbers, the error goal, the initial step size, and decrease or increase rate of the step size.

For $ANFIS_{(1)}$, $ANFIS_{(2)}$, $ANFIS_{(3)}$, and $ANFIS_{(4)}$, different epoch numbers are used. It can be concluded that the increase of epoch number for a training data set does not necessarily improve the system performance significantly but it helps to overcome the problem of overfitting. The training data should cover all different possibilities to enhance system performance, as previously discussed. As the different tests have shown, each ANFIS structure has it own best and worst fit model according to the current ANFIS settings.

A number of tests were carried out with regard to the number of epochs. The results of these tests indicate that the most important factors to achieve good performance are usually the training data sample and the number/type of membership functions. The number of epochs and the training options do not appear to have a significant influence on performance.

Results in the tables demonstrate that ANFIS model performance is, in general, accurate and acceptable, where some rules are covered by human expert training data set and the missing rules are detected by ANFIS. The results of the ANFIS models demonstrate that ANFIS can be successfully applied to adapt learning content according to the learner's current situation or needs.

In this work, the applicability and capability of the ANFIS method are investigated through the use of a number of data sets, membership functions, and types of membership functions. Twelve ANFIS models were constructed using combinations of inputs, sample training data, membership function numbers, and membership function types for the purpose of mobile learning. These ANFIS models were trained and tested. The results of the ANFIS models and observation were compared and evaluated. ANFIS models were compared based on their performance in training and checking data sets.

One of the motivations behind our design of an adaptive mobile learning system was to support the learner throughout the learning activity by realising the adaptation principle (Section 2.1), which consists of: 1) capturing the learners' context (implicit and explicit) and then 2) utilizing the captured learner context to construct a Learner Model (Section 2.2). If the captured learner context, which forms the learner model, truly represents the entire learning activity with its changes over time, the use of ANFIS for a reasoning engine will keep up with the change in learner context and make accurate adaptation decisions.

A key contribution of this paper is the adaptation components, with a specific focus on the detailed learner model components. Adaptive results presented in this section confirm that the m-learning application is feasible. The feasibility of the Reasoning Layer confirms the assumptions made in Sections 2.1 and 2.2 of this paper. These results were based on analysis of a number of inputs; the research confirms that the m-learning application is functional, however an increase in the number of inputs will increase the response time.

One of the key challenges in m-learning is to ensure that an appropriate balance is struck between the conveniences provided by adaptation and the time taken to provide the user with such a personalized experience.

6 Conclusion

This paper introduced the Adaptive Neuro-Fuzzy Inference System as a reasoning engine to deliver learning content for

mobile learning applications. This study was conducted to illustrate the potential effectiveness of ANFIS with hybrid learning, for the adaptation of learning content format for mobile learning users. The performance of ANFIS was evaluated using standard error measurements which revealed the optimal setting necessary for better predictability. The numbers of fuzzy rules obtained from the human experts were insufficient, therefore, this paper adopted a hybrid approach that combined the Fuzzy Inference System with the Neural Network in determining a complete fuzzy rule system. The ANFIS approach has successfully solved the problem of incompleteness in the fuzzy rule base made by the human expert. By training the Neural Network to apply the human expert's fuzzy rule base to different training data, the Neural Network is expected to recognize other decisions that were previously not detected. Future research will revise the rules, inputs, number and type of membership functions, the epoch numbers used, and training sample to further refine the ANFIS model. This proposed future work will examine and increase the feasibility of developing a more effective and adaptive m-learning content delivery system.

REFERENCES

[1] J. Malek, M. Laroussi, and A. Derycke, "A Multi-Layer Ubiquitous Middleware for Bijective Adaptation between Context and Activity in a Mobile and Collaborative Learning," *Proc. IEEE CS Int'l Conf. Systems and Network Comm. (ICSNC),* p. 39, 2006.

[2] G.I. Webb, M.J. Pazzani, and D. Billsus, "Machine Learning for User Modeling," *User Modeling and User-Adapted Interaction,* vol. 11, nos. 1/2, pp. 19-29, 2001.

[3] A. Jameson, "Numerical Uncertainty Management in User and Student Modeling: An Overview of Systems and Issues," *User Modeling and User-Adapted Interaction,* vol. 5, nos. 3/4, pp. 193-251, 1996.

[4] A. Al-Hmouz, J. Shen, and J. Yan, "A Machine Learning Based Framework for Adaptive Mobile Learning," *Proc. Eighth Int'l Conf. Advances in Web Based Learning (ICWL),* M. Spaniol, Q. Li, R. Klamma, and R.W.H. Lau, eds., pp. 34-43, 2009.

[5] A. Kobsa, "Generic User Modeling Systems," *User Modelling and User-Adapted Interaction,* vol. 11, pp. 49-63, 2001.

[6] A. Al-Hmouz, J. Shen, J. Yan, and R. Al-Hmouz, "Enhanced Learner Model for Adaptive Mobile Learning," *Proc. 12th Int'l Conf. Information Integration and Web-Based Applications and Services (IIWAS),* G. Kotsis, D. Taniar, E. Pardede, I. Saleh, and I. Khalil, eds., pp. 781-784, 2010.

[7] V. Tsiriga and M. Virvou, "A Framework for the Initialization of Student Models in Web-Based Intelligent Tutoring Systems," *User Modelling and User-Adapted Interaction,* vol. 14, no. 4, pp. 289-316, 2004.

[8] R.M. Felder and L.K. Silverman, "Learning and Teaching Styles in Engineering Education," *Eng. Education,* vol. 78, no. 7, pp. 674-681, 1988.

[9] P. García, A. Amandi, S.N. Schiaffino, and M.R. Campo, "Evaluating Bayesian Networks' Precision for Detecting Students' Learning Styles," *Computers and Education,* vol. 49, no. 3, pp. 794-808, 2007.

[10] E. Millán and J.-L. Pérez-de-la-Cruz, "A Bayesian Diagnostic Algorithm for Student Modeling and its Evaluation," *User Modelling and User-Adapted Interaction,* vol. 12, nos. 2/3, pp. 281-330, 2002.

[11] D.A.A. Villaverde and J.E. Godoy, "Learning Styles Recognition in E-Learning Environments with Feed-Forward Neural Networks," *J. Computer Assisted Learning,* vol. 22, pp. 197-206, 2006.

[12] M. Xenos, "Prediction and Assessment of Student Behaviour in Open and Distance Education in Computers Using Bayesian Networks," *Computers and Education,* vol. 43, no. 4, pp. 345-359, 2004.

[13] G.C. Saldías, A.R. Barbosa, and F.M. de Azevedo, "Intelligent Tutoring Systems: Formalization as Automata and Interface Design using Neural Networks," *Computers and Education,* vol. 49, no. 3, pp. 545-561, 2007.

[14] R. Stathacopoulou, M. Grigoriadou, G.D. Magoulas, and D. Mitropoulos, "A Neuro-Fuzzy Approach in Student Modeling," *Proc. Ninth Int'l Conf. User Modeling,* P. Brusilovsky, A.T. Corbett, and F. de Rosis, eds., pp. 337-341, 2003.

[15] V.S. Kodogiannis and A. Lolis, "Forecasting Financial Time Series Using Neural Network and Fuzzy Systembased Techniques," *Neural Computing and Applications,* vol. 11, no. 2, pp. 90-102, http://dx.doi.org/10.1007/s005210200021, 2002.

[16] J.-S.R. Jang, "ANFIS: Adaptive-Network-Based Fuzzy Inference System," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 23, no. 3, pp. 665-684, May/June 1993.

[17] J.S.R. Jang and C.T. Sun, "Neuro-Fuzzy Modeling and Control," *Proc. IEEE,* vol. 83, no. 3, pp. 378-406, Mar. 1995.

[18] J.S.R. Jang, C.T. Sun, and E. Mizutani, *Neuro-Fuzzy and Soft Computing.* Prentice Hall, 1997.

[19] D.J. MacKay and C. Laboratory, "Bayesian Non-Linear Modelling for the Prediction Competition," *ASHRAE Trans.,* vol. 100, no. 2, pp. 1053-1062, 1994.

[20] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees.* Wadsworth and Brooks, 1984.

[21] M. Ohlsson, C. Peterson, H. Pi, T. Rögnvaldsson, and B. Söderberg, "Predicting System Loads with Artificial Neural Networks - Methods and Results From 'The Great Energy Predictor Shootout'," *Proc. Ann. Am. Soc. of Heating, Refrigerating and Air-Conditioning Eng., Inc.,* vol. 100, pp. 1063-1074, 1994.

[22] J. shing and R. Jang, "Input Selection for ANFIS Learning," *Proc. IEEE Fifth Int'l Conf. Fuzzy Systems,* Sept. 1996.

Ahmed Al-Hmouz received the BSc degree in computer science from Mutah University, Jordan, the MSc degree in internet technology, and the MSc degree in information systems from the University of Wollongong, Australia. He is currently working toward the doctorate degree in the School of Information Systems & Technology, Faculty of Informatics at the University of Wollongong, Australia. His primary research interests include mobile learning, learner modeling, ANFIS, and context adaptation.

Jun Shen was awarded the PhD degree in 2001 at Southeast University, China. He held positions at the Swinburne University of Technology in Melbourne and the University of South Australia in Adelaide before 2006. He is a senior lecturer at the University of Wollongong in NSW Australia. He has published more than 60 papers in journals and conferences in the areas of computer science and information systems. His expertise is on web services and the Semantic Web. He has been an editor, PC chair, guest editor, and PC member for numerous journals and conferences published by the IEEE, ACM, Elsevier, and Springer. He has been a chair of the Education Chapter of the IEEE NSW section since 2007. He is a senior member of the IEEE.

Rami Al-Hmouz received the BSc degree in engineering (electrical engineering/telecommunication) from Mutah University in 1998, the MSc degree in computer engineering from the University of Western Sydney in 2004, and the PhD degree in computer engineering from the University of Technology, Sydney, in 2008. Currently, he is an assistant professor at King Abdulaziz University in Saudi Arabia. His research interests are in computer vision, computer networks and sensor networks.

Jun Yan received the BEng and MEng degrees in computer application technologies from Southeast University, Nanjing, China, in 1998 and 2001, respectively, and the PhD degree in information technology from the Swinburne University of Technology, Melbourne, Australia, in 2004. He is currently a senior lecturer in the School of Information Systems and Technology, University of Wollongong, Australia. His research interests include software technologies, workflow management, service-oriented computing, and agent technologies.