A Critique of Using Machine Intelligence for e-Decision-Making

Gasser Auda

College of Arts and Sciences
American University of Sharjah
Sharjah, United Arab Emirates 26666
Tel: +9716-5152559, E-mail: gauda@ausharjah.edu

ABSTRACT

This paper is a critique of Artificial Intelligent agents used in the hearts of e-planning and e-decision-making systems in general. It refers to a case study of a web site that teaches JAVA programming language and is designed as a multi-level curriculum, each level with specific curricular objectives designed for users with certain background knowledge. The Instructor, to be replaced with the Intelligent Expert System, decides on the student's level based on his/her answer to a detailed questionnaire. The paper critically compares the Instructor's performance to three systematic approaches - procedural programs, logic expert systems, and neural networks.

Keywords

Logic Programming, Artificial Intelligence in e-learning, Expert Systems, Neural Networks.

1.0 INTRODUCTION

Could artificial intelligence replace human intelligence in making decisions in e-environments? This paper proposes a negative answer to this question based on a critical examination of three different typical designs versions of an educational expert system advisor (Hasset et al., 2003). Examples of similar critiques are (Sowa, 2000) and (Masao, 2001).

Figure 1 shows a general block diagram of the system at the operating mode, i.e., after designing/training the intelligent system. A JAVA applet displays a questionnaire to the student, marks the answers and, depending on the results of the answers, the intelligent system recommends one of the available curriculum plans for the student. At the end of every chosen plan, the student goes through an outcome test, which assesses his/her understanding of the studied material. Finally, the student re-takes the questionnaire in order to be guided to the next appropriate study plan, i.e., curricular level. At the end of every chosen and completed plan, the student takes an outcome test, which assesses his/her understanding of the studied material. Finally, the student re-takes the questionnaire in order to be guided to the next appropriate study plan.

In order to design the evaluating questionnaire, a standard JAVA tutorial was studied carefully and groups of questions on each section were designed. The questionnaire included twenty-five questions in the form of main questions followed by sub-questions.

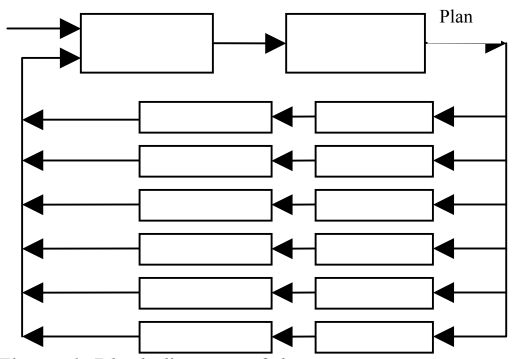

Figure 1. Block diagram of the system.

The main question, with a "yes" or "no" answer, asks the student about his/her knowledge on a specific topic. If the answer is "yes," the student will be asked a few further sub-questions on that topic to confirm his/her knowledge level of the specific topic. If the answer is "no," the program skips the sub-questions and goes to the next main question. Precedence was given to practical programming questions that cover more material in the chapters and many questions were merged into multiple-choice forms. The final questionnaire contained the questions shown in Figure 2. Note that the "main questions" are marked by an "*" and the sub-questions, which are to be skipped if the answer of the main question was "no," are marked by a "--". Also note that the actual screens were much more user friendly.

To design the curricula (levels), the course's topics were first divided into a number of modules, each contains a few interrelated topics. The following list shows the details.

Plan 1: contains all the categories (for beginners).
Plan 2: contains categories C, D, E, F and G. Suitable for students how have the basic concepts of JAVA.
Plan 3: contains categories C, E, F and G.
Plan 4: contains categories D, F and G.
Plan 5: contains categories D and G.
Plan 6: means that the student is good in the material of our tutorial and does not need this course.
Where the categories (A to F) are defined as follows.

* Do you know anything about Object-Orientation?
* Do you know how JAVA objects, classes & interfaces are used?
-- Which sentence is correct from the following:
- Class declaration must contain a superclass name.
- To declare that your class implement interfaces use the keyword extends.
- Using the final modifier you can declare that your class can not be subclassed.

* Do you know the Java's traditional language features?
-- opl | op2 means:
- op1 and op2
- op1 xor op2
- op1 or op2

* Do you know how strings are used in Java?
-- Which sentence of the following is incorrect:
- StringBuffer str = new StringBuffer;
- System.out.println(newstringBuffer().append(“con”).append (“cat”).append(“enation”);
- The value of sb after these statements is Drink Hot Java!
StringBuffer sb = new StringBuffer(“Drink Java!”);
sb.insert(6,”Hot”);

* Do you know how Java deals with **System** class and input, output streams?
-- To use a **System** class variables or methods:
- Instantiate a System class and declare a variable of type System and use its methods and variables as any other class.
- Append the variable or method name directly to the class name using (.) notation.

-- The **piped** input and output streams are used:
- To access files sequentially.
- For methods that produce output to be used as input by someone else.

-- True or False: The **ByteArayInputStream** read 8_bit data from a byte array in memory but **StringBufferInputStream** reads 16_bit Unicode data from a string buffer in memory.

* Do you know what Exception is in Java?
-- Which sentence is incorrect:
- The object that can be thrown must be a subclass of exception class.
- The object that can be thrown must be a subclass of a throwable class.
- Any object can be thrown.

* Do you know anything about threads?
-- Which sentence of the following is incorrect:
- Java program can have many threads, and those threads can run concurrently.
- Synchronized keyword is used in a program that accesses the same data from within separate, concurrent threads.
- We use wait() in conjunction with notify() to coordinate the activities of multiple threads using multiple resources.

-- In Threads execution, the Java runtime supports:
- Fixed priority scheduling.
- Time_slicing.

* Do you know what applets are and how they are written in Java?
--True or False: When you write an applet you should add it to an HTML page using the **<applet>** tag.
-- Which sentence of the following is correct:
- Applets can load libraries and define native methods.
- Applets can start any program from the host that executing it.
- Applets can invoke public methods of other applets on the same page.

-- The **init()** method should contain:
- The code that you would normally put into a constructor.
- The execution code.

-- Which sentence of the following is incorrect:
The applet creates its own thread.
- When it performs a task repeatedly.
- When it performs a time consuming task.
- When it performs several tasks.

* Do you know how to create a Java user interface?
-- CheckboxGroup, **Choice** and **List** share same properties but:
- CheckboxGroup have no index where Choice and List have.
- In CheckboxGroup and Choice only on item can be chosen where in List we can chose several items.
- Both 1 and 2 are correct.

-- To create submenu in a menu we write the following code:

```
MenuBar mb;
Menu m1, m2;
Menultem mi;
-  m1 = new Menu(“Menu 1”);
   mb.add(m1);
   m2 = new Menu(“subMenu”);
   ml.add(m2);
   mi = new Menultem(“Menu Item”);
   m2.add(mi);
- ml = new Menu(“Menu 1”);
   mb.add(m1);
   m2 = new Menu(“subMenu”);
   mb.add(m2);
   mi = new Menultem(“Menu Item”);
   m2.add(mi);
```

-- The **GridLayout**:
- Uses the component’s natural size.
- Display a few equal components in rows and/or columns use.
- Uses two_arguments add method.

-- In **GridBagLayout,** if we want Button1 to be in the first row and Button2 in the second row. Which code is correct

```
      GridBagConstraints c = new GridBagConstraints();
      GridBagLayout gridbag = new GridBagLayout();
- c.gridwidth = GridBagConstraints.Remainder();
makebutton = (“Button1”,gridbag,c);
makebutton = (“Button2”,gridbag,c);
- c.gridwidth = GridBagConstraints.Relative();
makebutton = (“Button1”,gridbag,c);
c.gridwidth = GridBagConstraints.Remainder();
makebutton = (“Button2”,gridbag,c);
```

-- Which sentence of the following is incorrect:
- Using the drawImage we can change the size of the image.
- To create an animation loop, call the repaint method to draw the current frame then it should sleep for a while.
- One of the benefits of using MediaTracker class, in loading sequence of images for the animation, is increasing the speed.

-- Eliminating flashing by overriding the **update()** method do the following except one:
- Move the drawing code from the **paint()** to **update()** method.
- The **update()** method clears the entire background only when necessary.
- The **update()** method creates **offscreen**, draw to it, and then display the resulting image onscreen.

* Do you know about networking and security in Java?
-- True or False: TCP is a connection based protocol that provides a reliable flow of data between two computers, while UDP is a protocol that sends independent packets of data with no guarantee for arrival.
-- Fill in the blank by choosing the correct answer:
The TCP and UDP protocols use___________ to map incoming data to a particular process running on a computer.
- Datagrams. - Ports. - HTTP.

-- True or False: Results of the following two lines are the same
- URL batelco = new URL(“http://www.batelco.com.bh/Inet.html”);
- URL batelco = new URL(“http”,”www.batelco.com.bh”,”/Inet.html”);

-- True or False: To use a **socket,** we must create input and output streams for reading from or writing to the **socket.** To close the **socket** after we finish using it, we must close the streams first.

```
-- socket = new DatagramSocket();
   packet = new DatagramPacket(buf,222);
   socket.receive(packet);
```

True or False: The **receive()** blocks until a packet is received.

Figure 2. The questionnaire that the system uses to evaluate students.

Module A:

- Object-Oriented Programming Concepts.
- Object, Classes and Interfaces in JAVA.

Module B:

- The Nuts and Bolts of JAVA Language.
- The String and String Buffers Classes.
- Setting Program Attributes.

Module C:

- System.
- Input and Output Streams.

Module D:

- Handling Errors Using Exceptions.
- Threads of Control.

Module E:

- Overview of Applets.
- Creating an Applet User Interface.
- Communication with Other Programs.
- Understanding Applet Limitations.
- Finishing an Applet.

Module F:

- Overview of JAVA UI.
- Using Components within a Container.
- Working with Graphics.

Module G:

- Overview of Networking.
- Working With URLs.
- All about Sockets.
- All about Datagrams.
- Providing Your Own Security Manager.

The levels were not arranged in a specific order in terms of difficulty. They are simply different scenarios tailored to the students' needs. For example, level 5 does not contain categories E and F, which come after categories D in order, because of the complexity of the material in Module D even for high-level students.

2.0 VERSION 1: PROCEDURAL PROGRAM

In this version, an attempt was made to implement the system using a conventional procedural approach. A procedural language was used, which was JAVA. The system was not feasible to be built, to start with, because of the nature of the problem in hand. Decision making at the level of the desired system cannot be implemented using a procedural language. The number of possible combinations of the different answers to the questions was too large for any practical implementation using this approach.

Thus, in order to make decisions at a practically complex level, one has to think in terms of artificial intelligence.

3.0 VERSION 2: LOGIC-LANGUAGE EXPERT SYSTEM

In this version, the human instructor was asked about what first-order logic rules he would use to "systemize" his thinking and the decision making process in order to copy these rules into a logic system. The large number of rules that the instructor came up with (in the order of a couple hundred) were feasible to implement using a logic programming language, PROLOG in this case.

However, the actual testing of the system showed numerous discrepancies in the logic of these rules. It seems that instructors who make their "human" decisions about these kinds of issues do not have a closed-end set of rules that lead to the same result, given that one might follow different lines of reasoning. The instructor was never able to refine his many rules in order to avoid logical conflict between their generated lines of reasoning that sometimes caused logical contradictions. Moreover, the logic-language environment assumes that the set of true facts or rules are the only sets of facts or rules that exist in the "true" space. In other words, the logic program assumed, by default, that any fact or rule that is not declared is false, which was an obvious limitation of the performance of the system.

Thus, the design of a first-order logic-based expert system proved its incapability to handle such a planning decision in a systematic and coherent form.

4.0 VERSION 3: NEURAL NETWORKS

In this version, and since the number of all possible combinations of answers to the questionnaire is very high, the neural network system is trained using 300 samples/prototypes that were created based on random combinations of answers to the questionnaire and their corresponding decisions taken by the human JAVA instructor as to which curriculum is most appropriate.

A Feedforward Backpropagation network was used (Hagan, Demuth, & Beale, 1996; Demuth &

Beale, 2000). The number of inputs was 25 (equal to the number of questions) and the number of outputs was 6 (corresponding to the 6 different decisions explained above). It was found that one hidden layer was enough to achieve satisfactory results. The transfer function of the hidden layer is chosen to be the sigmoid function. The size of the network, i.e., the number of hidden nodes, appeared to affect the results significantly. That is why 10 different network sizes were used and compared. An even distribution of input samples over the neural network outputs, i.e., classified for the different plans, was necessary for good learning results (Auda, 1996). And since the sample answers were generated randomly, some samples were manually changed in order to fall into the under-represented plans/levels.

In order to test the performance of the system, another group of 300 samples were presented to the instructor and the decisions of the neural network system were compared to the recommendations of the instructor.

The neural network made 90% of its decisions on the testing samples (that were not part of the training process) similar to the human instructor's decisions. Although we find this result interesting, we do not think that the neural network should replace the human planner for the following reasons.

1. The performance of the system depends on the data base used for training. This reminds us of the traditional criticism of theories based on induction (Sowa, 2000); that is: If you claim that you reached your theory based on exploring all possibilities, then your claim is false in case there are an infinite number of possibilities. And if you already explored all possibilities, then there is no need for your inductive theory because all possibilities are explored anyway!
2. The relative importance of the questions was filtered out in the numerical representation of the data at the inputs of the neural networks. To my knowledge, there is no neural design that considers this factor (except for the biological neural network, the human brain, for example).
3. The neural network solution does not consider the combinations of inputs that significantly affect the resulting decision. For a human decision maker, as well as for the logic expert system, combinations of certain inputs had their direct impact on the decision making process (for example: the failure to answer a certain question while answering another question perfectly, etc).
4. The hard boundaries of the neural network did not allow any fuzziness in the decision, something that a human decision maker might experience. Fuzzy logic, on the other hand, does not solve the fuzziness problem but rather adds more hard categories (Sowa, 2000).
5. The hidden nodes of the neural network supposedly represent higher-level concepts about the data. However, they are strongly coupled (Auda, 1996), which means their formation of the higher-level concepts was based on all inputs. In other words, the hidden nodes were not free, computationally, to form specific higher level correlations between groups of inputs.

5.0 CONCLUSION

Although modern machine-intelligent systems succeeded in modeling numerical abstractions of human decision-making, they cannot grasp the intricate complexities of human decision-making methods, especially in the educational field. The human mind remains the ultimate decision-maker, despite the machines' speed and accurate memory retrieval mechanisms.

6.0 REFERENCES

Auda, G. (1996). *Cooperative modular neural network classifiers.* Ph.D. Thesis. Systems Design Eng. Department, Univ. of Waterloo, Canada.

Demuth, H., & Beale, M. (2000). *Neural network Toolbox for MATLAB*.http://www.mathworks.com

Hassett, J., Ingram, A., Hassett, M., & Marino, E. (2003). What Do Learners Like? Ratings of Off-the-Shelf Web-Based Training Courses. *Intl. Journal on E-Learning 2*(1), 50-60.

Hagan, T., Demuth, H., & Beale, M. (1996). *Neural Network Design*. USA: PWS Publishing.

Masao, M. (2001). *Fuzzy Logic For Beginners.* Singapore: World Scientific.

Sowa, J. (2000). *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. USA: Brooks/Cole.