Expert Systems with Applications 38 (2011) 9522–9529

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

EDUCA: A web 2.0 authoring tool for developing adaptive and intelligent tutoring systems using a Kohonen network

Ramón Zatarain Cabada [a,*], María Lucía Barrón Estrada [a], Carlos Alberto Reyes García [b]

[a] Instituto Tecnológico de Culiacán, Juan de Dios Bátiz s/n col. Guadalupe, Culiacán, Sinaloa, CP 80220, Mexico
[b] Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), Luis Enrique Erro No. 1, Sta. Ma. Tonanzintla, Puebla, CP 72840, Mexico

ARTICLE INFO

Keywords:
Intelligent tutoring systems
Cooperative/collaborative learning
Authoring tools and methods
Learning communities

ABSTRACT

This paper presents a Web 2.0 Learning Environment, for a systematic creation of adaptive and intelligent tutoring systems. Authoring contents is made by a community of users including teachers and students. The tutoring systems adapt the contents according to the best learning style using self-organizing maps (SOMs). The SOM was trained for classifying Felder–Silverman learning styles. The most important advantage of these unsupervised neural networks is that they do not require an external teacher for presenting a training set. The approach was implemented under an authoring tool that allows the production of personalized learning material to be used under collaborative and mobile learning environments. The tutoring systems together with the neural network can also be exported to mobile devices. We present different results to the approach working under the authoring tool.

1. Introduction

Since the beginning of Intelligent Tutoring Systems (ITS) research, more than twenty authoring systems for ITS have been developed (Murray, Blessing, & Ainsworth, 2003). Those authoring tools can be classified according to what type of tutoring system they produce. For example the author tool SIMQUEST (de Jong et al., 1999) produces "simulation-based learning" systems, IRIS (Arruarte, Fernández, Ferrero, & Greer, 1997) creates "multiple knowledge types" systems and InterBook (Brusilovsky & Schwarz, 1998) generates Intelligent/adaptive Hypermedia. One shared feature of all of those tools is the separate role of authors and learners. For instance, in SIMQUEST one author creates the simulation model, the learner interface, the instructional design, and the environment. In this list of tasks, the learners are never involved. In IRIS, the author produces an ITS by following two main phases: Specifying the general requirements of the tutoring system, and filling the learning contents of the ITS. On the other hand, building adaptive educational material is a similar procedure: the developer has to create content objects and specify the links between them. Authoring tools for developing adaptive educational/tutoring systems like InterBook, SIGUE (Carmona, Bueno, EduardoGuzman, & Conejo, 2002), NetCoach (Weber, Kuhl, & Weibelzahl, 2001), and Ale (Specht, Kravcik, Klemke, Pesin, & Hüttenhain, 2002) differentiate only by the interface tools they operate. Some of them use special markup languages and some GUI interfaces. The author creates the learning environments and the learner read the learning contents.

In the last years many research groups in the field of education are using Web 2.0 technologies, such as wikis, blogs, recommendation systems and social networking (O'Reilly, 2005). This kind of technology is now identified as *Social Software*. In the field of e-learning, a "tutor centered" concept is shifting to become more learner centered (coined as *e-learning* 2.0), where learners are also part of a community of authors and users of the learning resources (Hage & Aimeur, 2008). On the other hand, Learner or Student Models are the core of the personalization of Intelligent Tutoring Systems. They make available tutoring tailored or adapted to the needs of the individual students (Kerly & Bull, 2008). Many approaches and implementations have been developed in recent years in order to model students' learning styles (Carmona, Castillo, & Millán, 2008; Graf, Kinshuk, & Liu, 2008). Most of those implementations use Bayesian Networks (Carmona et al., 2008), Linear Temporal Logic (Limongelli, Sciarrone, & Vaste, 2008), or Neuro-fuzzy Networks (Zatarain-Cabada et al., 2008). In the case of using a learning model like Felder–Silverman (Felder & Silverman, 1988), we also use the Index of Learning Style Questionnaire (Felder & Solomon, 2004).

In this work we propose a different approach for selecting a student's learning style using self-organising feature maps (Kohonen neural networks) (Kohonen, 1989). Advantages of Kohonen networks include implementation simplicity, execution speed and a shorter training process; however maybe the most important advantage of these unsupervised neural networks is that they do

* Corresponding author.
E-mail address: rzc777@hotmail.com (R.Z. Cabada).

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.01.145

not require an external teacher for presenting a training set. During a training session, our Kohonen network receives a number of different input patterns (the student learning style obtained from the ILSQ, the course learning style, and the student's grade in the course), discovers significant features in these patterns (Felder–Silverman learning styles) and learns how to classify input.

In this context, we have designed and implemented a software tool (EDUCA) to create adaptive learning material in a Web 2.0 collaborative learning environment. The material is initially created by a tutor/instructor and later maintained and updated by the user/learner community to each individual course. The courses can dynamically recognize user learning characteristics and be displayed on mobile computers (cell phones, PDA, etc.). EDUCA makes use of Web 2.0 technologies as a recommendation system for filtering future Web learning resources, and Web mining for discovering such resources.

The arrangement of the paper is as follows: Section 2 gives a general structure of the tool. Section 3 presents the neural network and predictive engine used in the tool. Results are shown in Section 4. Discussions and conclusions are given in Section 5.

2. The methodology

2.1. Building a tutoring system with Educa

The process of constructing a new tutoring system (an adaptive or intelligent system) consists of three main steps (Fig. 1). During Step 1 a tree structure of the adapted or intelligent tutoring system is designed by the main instructor(s) of the learning material. In the structure the designer specifies the course information (title, general description, author name, etc.), the unit names, and for each sub unit, the designer defines names and tags related to that sub unit, and all the prerequisites the student should get done. In the tree structure, the designer also inserts quizzes (multiple selection and choice). Quizzes are an important element to provide adaptation capabilities to the produced tutors.

In step 2 the tree structure is filled with domain contents (a knowledge base), and some other learning resources. At the beginning of the creation the instructor or teacher authors the tutoring system by inserting different learning objects like text fragments, images, audio/video clips, and by defining learning styles, prerequisites, tags and quizzes. At a later time, learning resources are added to the tutoring system by learners, who make recommendation of resources they find commonly in the web. Those resources can be found also in a special resource repository of Educa. After the author creates the **tutoring system**, she/he can save it and export it to a Mobile Learning format used to display tutoring systems on mobile devices (step 3). The saved/exported file will enclose three elements: a XML file corresponding to the **learning resources or contents,** a **predictive engine** for navigation purposes, and the **SOM Neural Network** for learning style classification. Another option to the output of Educa is to export the tutoring system to SCORM format. The benefit of this format is the availability of the material in any distance learning environment.

2.2. Educa architecture

Educa Architecture is composed by five main components or modules (Fig. 2): an authoring tool, two repositories (for resources and courses), an intelligent delivery Engine, and a recommendation engine.

The authors create initially a tutoring or adaptive system (a special type of knowledge base) using Educa authoring tool which runs in a desk computer. The tutoring system then is exported to a mobile computer (it can also be run in one wireless emulator like Sun Java Wireless Toolkit). Once the system is installed in a mobile device, a student can display the learning objects (LO) by using an Intelligent Delivery Engine (installed together with a neural network in the mobile). The engine displays the LOs in the mobile according to the learning style of the student and with the help of the neural network.

When the authors create the LOs, they have two different options: One choice, the traditional one, is importing LOs from different sources like SCORM, HTML, .doc or .PDF files or by editing directly the learning material by using a text editor of Educa. The second choice is more Web 2.0 oriented, and it consists of using a recommendation engine. This application employs two repositories: a resource repository, which is used for recommending new resources found in the Web by community members; a course repository which stores all the courses created with Educa. Both repositories are stored in a DBMS.

2.3. Creating the knowledge base

As we have mentioned before (Section 2.1), a course is created by first step designing a tree structure of the adapted or intelligent tutoring system, which represents the knowledge of the system. In order to designing and implementing a course knowledge representation, we apply some of the concepts related to Knowledge Space Theory (Doignon & Falmagne, 1999). This theory provides a sound foundation for structuring and representing the knowledge domain for personalized or adapted learning. It applies concepts from combinatory theory and we use it to model particular or personalized courses according to different learning styles. Educa enables the creation or construction and application of discipline-specific knowledge spaces (or structures). An adapted assessment based upon knowledge spaces and student prerequisites (and employing a Kohonen neural network for identifying learning styles), will derive a particular or personalized path of learning objects. Fig. 3 shows an example of the knowledge domain for a Compiler course.

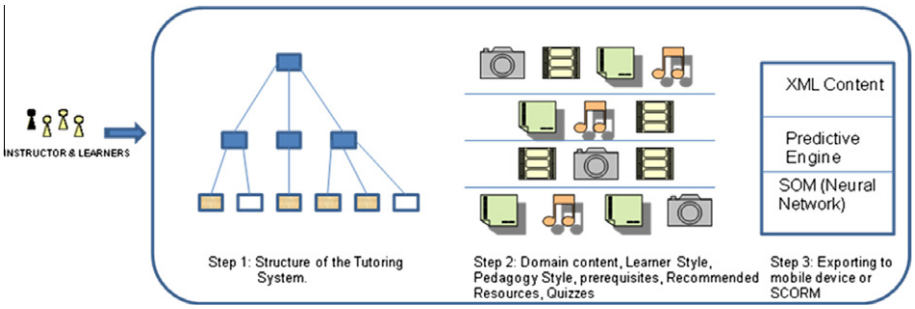

Fig. 1. Building a tutoring system with Educa.

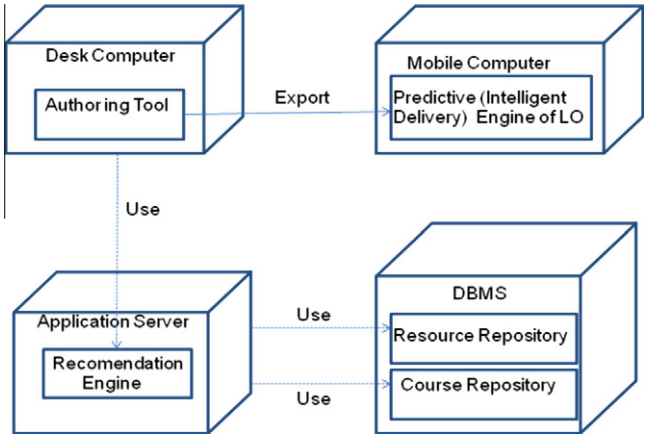

Fig. 2. Educa architecture.

The domain represents the knowledge of two main chapters (Front-end and Back-end). The first chapter contains four topics and one quiz. Each topic is represented by a special type of concept map providing learning material of the topic knowledge domain. For each topic a set of prerequisites and quizzes are set. For example, in order to study syntactic analysis it is necessary first to study context-free grammars. Based on the prerequisite relationship identified in this way, a knowledge structure (a subset of the knowledge domain) represents the personalized or adapted learning material.

Fig. 4 shows the knowledge domain of the topic parsing (syntactic analysis). In each line we see different learning objects teaching the same subject under different learning style. In the first path (pointed by a star) the last component is shown using the combination of dimensions (visual, sensitive, global). In this way, we can have many combinations or paths for learning the same topic. Dashed lines represent those paths. We can also see the prerequisites for studying the topic (Lecture 2 – Lexical Analysis).

2.4. Adding learning objects

In Educa we find two main authors: the instructor o tutor and the learners. The instructor creates an intelligent tutoring (adaptive) system by first building the tree structure of the knowledge domain (a learning object container) by using the Edita visual editor (see Fig. 3). The content for each topic is added by importing already prepared learning material in different standard formats like html, pdf, doc or SCORM learning objects from any type of source (Fig. 5). The author can also add learning material by using a simple editor included in the tool.

Another important learning object the authors insert into the knowledge repository is a quiz. This object can be added in every part of the domain (see Fig. 3). The quiz is essential for the dynamic course adaptation because it provides important information to the neural network.

After the author creates the knowledge base of the tutoring system, she/he can save it and export it to a Mobile Learning format used to display tutoring systems on mobile devices. The saved/exported file will enclose three elements: a XML file corresponding to the learning style model, a predictive engine for navigation purposes, and the Kohonen Neural Network for learning style classification. Another option to the output of the fuzzy editor is to export the learning material to SCORM format. The benefit of this format is the availability of the material in any distance learning environment.

The other authors in Educa are the Learners. When a student or learner reads a tutoring system for the first time, a user profile is created in the system. This profile contains student data like student's learning style, academic records and GPA, past uploaded resources (recommendation), and general academic information.

Once a tutoring system has been created, the module *Published Course* stores it in a course repository. We know that learners usually read tutoring systems stored in some kind of repository, but they also consult other learning resources in different web sites. For example a student who needs to implement a LR parser in a compiler course, could use the tutoring system stored in the course repository, but she/he could also consult extra resources in the Web. This material found by the student would be rated and recommended to be added to the regular compiler course. Educa uses a Hybrid Recommendation System (Burke, 2002) which stores new resources into a Resource Repository and Text Mining to search resources in the Web. Fig. 6 represents the different modules and connections of Educa.

3. The predictive engine

The main goal of the predictive engine is to identify dynamically the student's learning style whenever he is running the tutoring

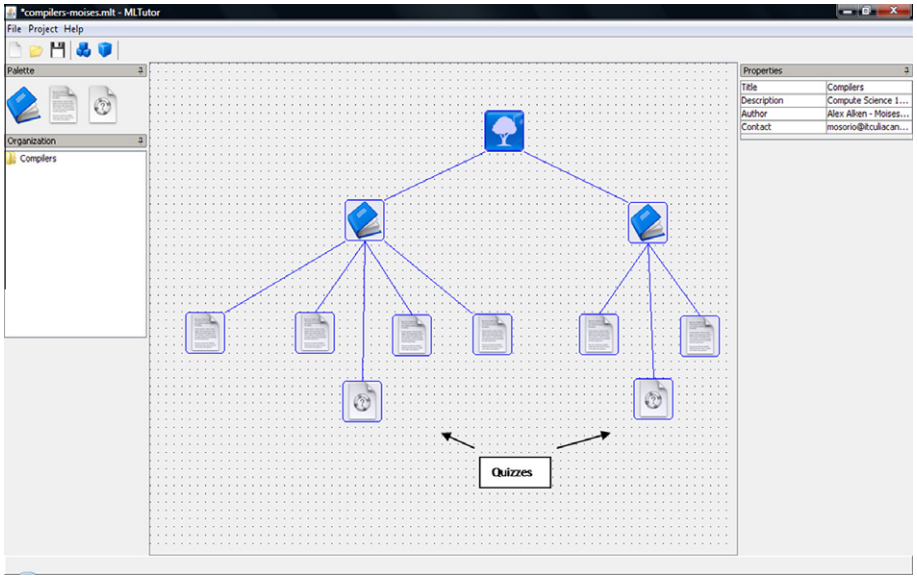

Fig. 3. Knowledge domain for a compiler course.

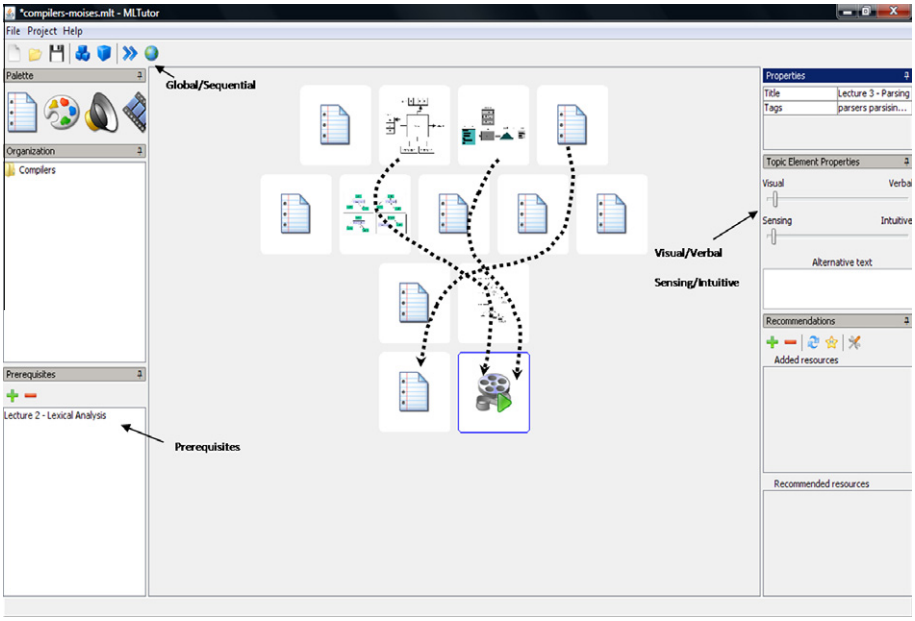

Fig. 4. Concept map for parsing topic.

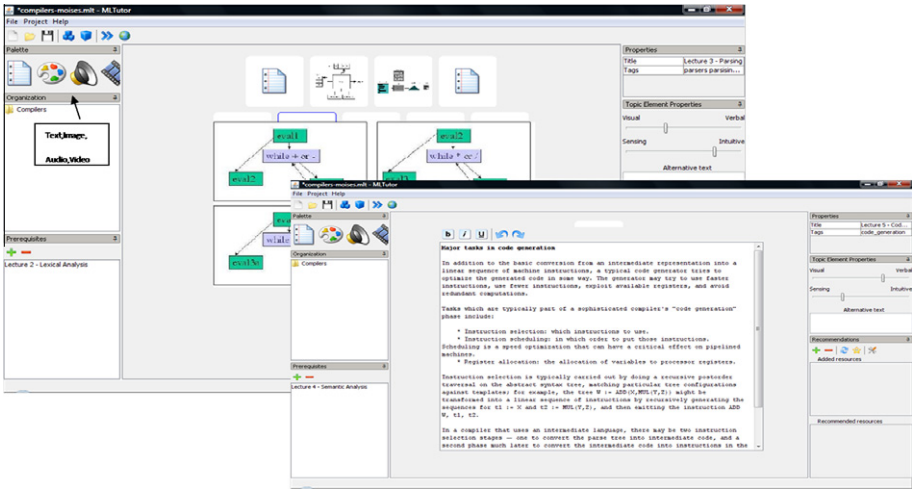

Fig. 5. Adding content to the tutoring system.

system. At the beginning an interpreter select content (learning objects) based upon the student's learning style, obtained from the student profile. The learning style of a student can be modified according to evaluations applied to the student (see Fig. 7).

3.1. The Kohonen neural network

The predictive or intelligent delivery engine is implemented by a self-organizing feature map or Kohonen neural network which is trained using unsupervised learning. In competitive learning, neurons compete among themselves to be activated. The Kohonen model provides topological mapping, placing a fixed number of input patterns from the input layer into a higher-dimension output or Kohonen layer (Fig. 8). The Kohonen layer consists of a single layer of computation neurons, with two types of different connections: forward connections, from input layer to the Kohonen layer, and lateral connections between neurons in the Kohonen layer. During training, the neural network receives new input patterns presented as vectors. Every neuron in the Kohonen layer receives a modified copy of the input pattern. The lateral connections produce excitatory or inhibitory effects, depending on the distance from the winning neuron.

3.2. Training the Kohonen network

An input vector in our neural network is described as:

$$\boldsymbol{D} = [\boldsymbol{d}_{fs}\boldsymbol{d}_a p],$$

where D is the input vector, and it is formed by three elements: $\boldsymbol{d}_{fs}$ which is the student learning style identified by applying the Felder–Silverman ILSQ questionnaire; $\boldsymbol{d}_a$ which is the learning style used in the learning material read by the student; and $\boldsymbol{p}$ is the student grade obtained in the test when the student has learning style $\boldsymbol{dfs}$ and the course was offered using learning style $\mathbf{d}_a$.

Vectors $\boldsymbol{d}_{fs}$ and $\boldsymbol{d}_a$ are also composed as:

$$\boldsymbol{d}_a = \boldsymbol{d}_{fs} = [\boldsymbol{c}_1\boldsymbol{c}_2\boldsymbol{c}_3],$$

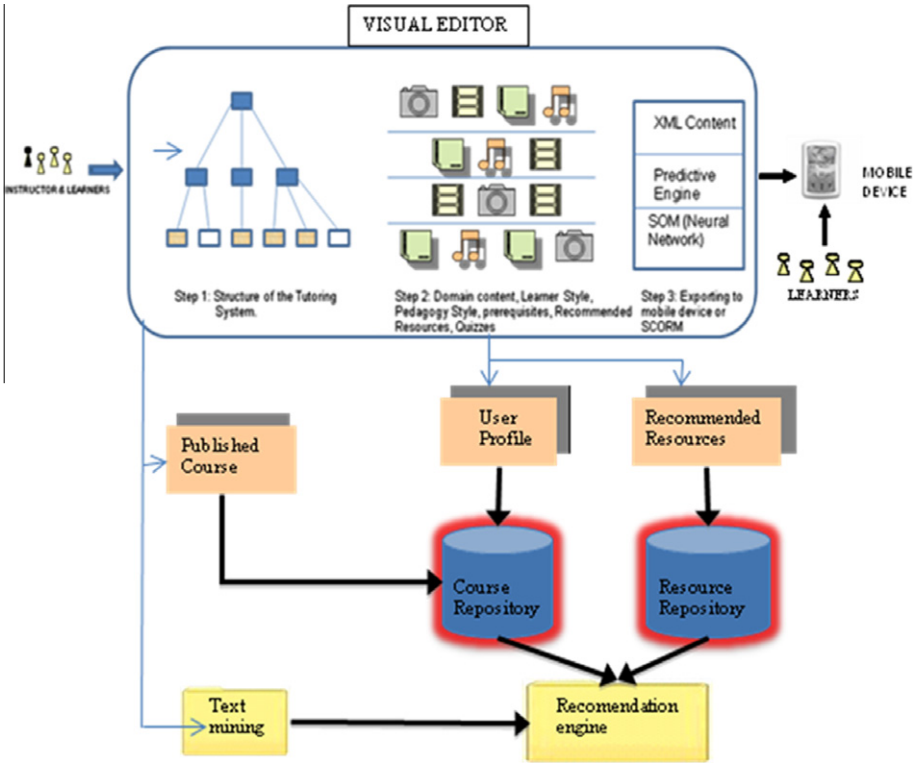

Fig. 6. General architecture of Educa.

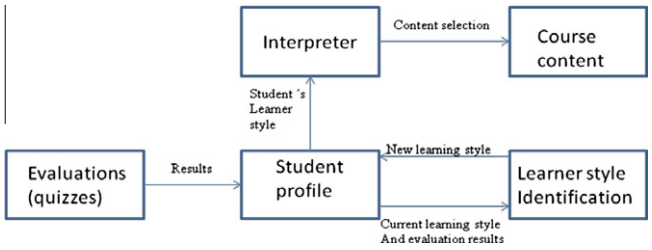

Fig. 7. The Predictive engine.

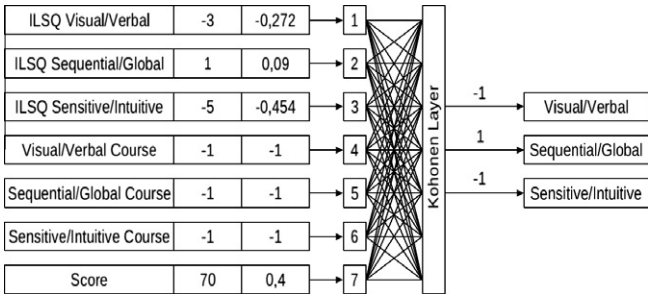

Fig. 8. Kohonen network for learning styles.

where $\boldsymbol{c}_1$ $\boldsymbol{c}_2$ $\boldsymbol{c}_3$ represents three scores for Perception, Input, and Understanding Dimension in the ILSQ questionnaire.

Fig. 8 shows part of the Kohonen network architecture. We can see that the input layer has seven neurons, corresponding to three-dimension vectors $\boldsymbol{d}_{fs}$ and $\boldsymbol{d}_a$, and the student grade $\boldsymbol{p}$. Vector's input data vary between −11 and +11, which is massaged to the range −1 to +1 with the equation:

$$\boldsymbol{Massaged\ value}_{11} = \boldsymbol{vi}/|\boldsymbol{vimax}|,$$

where *vi* is a ILSQ score, and *vimax* is the maximum ILSQ score value (+11). In Fig. 8, as we observe, −3, an ILSQ score for visual/verbal learning style, is mapped to −0.272. On the other hand, the student grade, which varies between 0 and 100, is also massaged to the range −1 to +1, with equation:

$$\boldsymbol{vnj} = \boldsymbol{vj}\ \times \mathbf{2}/\boldsymbol{vjmax} - \mathbf{1},$$

where $\boldsymbol{vj}$ is the student grade or score, and *vjmax* is the maximum student grade (+100). For example, 70 is mapped to 0.4 (Fig. 8).

The Kohonen layer has 1600 neurons arranged in a structure of a 40 × 40 hexagonal grid. The output of the network consists of three signals or values ranging between −1 and +1. These data are then decoded to the ILSQ range (−11 to +11); they represent the learning style of the student.

The procedure to train the neural network consisted in three steps. The sample for this experiment was 140 high-school students. During the first step we applied the ILSQ questionnaire in order to identify the learning style of each student. In the second step we created three different introductory courses: Computer Science, Photography, and Eolic Energy. For each course we produced eight versions for each learning style (visual/verbal, sequential/global, sensing/intuitive, and active/reflective). Randomly, the students received each of the three courses, in only one learning style. We recorded the assigned learning styles of each student in the three courses. In the third step and after the students took the courses, we applied tests in order to evaluate the performance of the students in each course.

With these three elements (student learning styles, course learning styles, and student evaluation or results) we created 140 input vectors for training the network. We trained the network with 6000 iterations, an initial neighbourhood ratio of 50% of the lattice size and a learning rate of 0.1.

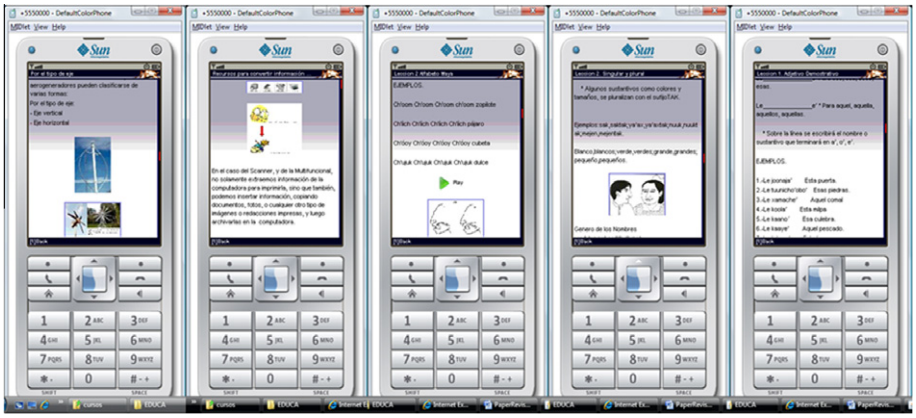

Fig. 9. Eolic energy, computer introduction, and Mayan language courses in a mobile phone.

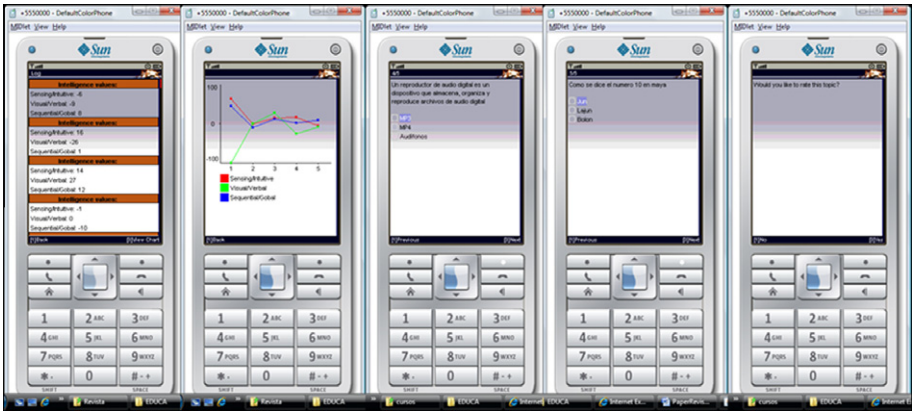

Fig. 10. Tracing student's learning style, presenting two quizzes, and asking to rate a course.

The network has 1600 neurons arranged in the form of a two-dimensional lattice with 40 rows and 40 columns, and every input vector has seven elements, as we established previously.

3.3. Using the Kohonen network

As we explained before the network is applied to identify learning styles of students. The network is then exported together with the ITS and the predictive engine to the mobile (see Fig. 6). When the student is displaying an ITS in a time $\boldsymbol{t}$, the learning material shown to the students is using a learning style $\boldsymbol{g}$. Whenever the student answers a quiz as part of the learning contents, he receives a grade $\boldsymbol{k}$.

Therefore, the network $\boldsymbol{r}$ takes as input the learning style of the displayed contents, and performs a search to find the winner-takes-all (best-matching) neuron $\boldsymbol{j_x}$ at iteration $\boldsymbol{p}$, using the minimum-distance Euclidean criterion

$$\boldsymbol{j_x(p)} = \boldsymbol{min}\|\boldsymbol{X} - \boldsymbol{W_j(p)}\| = [\Sigma(\boldsymbol{x_i} - \boldsymbol{w_{ij}})^2]^{1/2},$$

where $\boldsymbol{n}$ is the number of neurons in the input layer, $\boldsymbol{m}$ is the number of neurons in the Kohonen layer, $\boldsymbol{X}$ is the input pattern vector and $\boldsymbol{W}$ is the weight vector.

Neuron $\boldsymbol{j_x}$, the output of the network, is the student learning style which is defined as

$$\boldsymbol{g(t+1)} = \boldsymbol{r(g(t), k)}.$$

This process of identifying the student learning style is performed every time the student answers a quiz on the ITS.

4. Results and discussion

The evaluation of an authoring tool can be given from two different perspectives: the authoring tool and the intelligent tutoring or adapted system. In this paper, we present the evaluation of the authoring tool. The authoring tool EDUCA was evaluated with arguments like software usability (human–machine interfaces), program content and functionality. Before the evaluation we offered a small course using the authoring tool. There were 30 learners who tested the tool. The volunteers consisted of graduate (master degree) and undergraduate students and professors in the computer systems department of the Instituto Tecnológico de Culiacán. The course was given in four hours. With regard of the results or products of the course, they produced different intelligent tutoring systems for topics like Compiler Construction, Introduction to Computer Science, teaching the Maya Language, introduction to Eolic Energy, etc. Fig. 9 shows different snapshots of such courses, displayed on mobile phones (Sun Java Wireless Toolkit 2.5.2).

One important application of Educa is tracing the student's learning style during the execution of a tutoring system or course (first and second mobile phones of Fig. 10). This information can be precious for research in the field of learning. The third and fourth phones show two quizzes of the Mayan language course. The last phone is just a question to the student about rating the course. This information is for recommendation purposes.

To assess the tool we applied a survey of 20 questions to each participant at the end of the mini course, asking the learners to

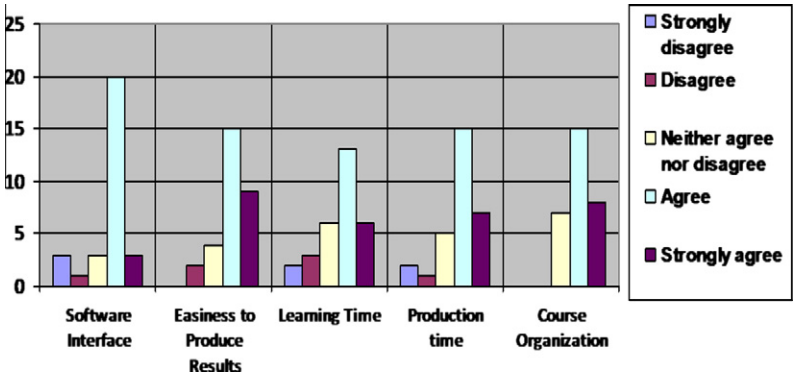

Fig. 11. Evaluation of the tool.

evaluate the software tool on the Likert scale according to their level of agreement or disagreement. Fig. 11 depicts the frequencies in the evaluation for five questions. As shown in the graph, the results tell us that most of the group in the workshop "agree" or "strongly agree" with respect to the interfaces, easiness to generate an intelligent tutor, learning time for using the tool, time to produce an intelligent tutor (1 h) and the course organization.

5. Discussions and conclusions

Authoring Systems for Intelligent Tutoring Systems and programs alike were introduced in the 1970s. Since then there have been a diversity of authoring tools dealing with different aspects and methods to be used. Murray et al. (2003) provides a full picture of the state of the art in ITS authoring systems. This work presents authoring tools in the categories of simulations, human expertise, instruction strategies, and special purpose. All of those tools are more PC-oriented. With respect to adaptation using learning styles, related work aims at identifying a learning style using approaches like Bayesian networks, decision trees, or Hidden Markov Models. In Carmona et al. (2008), Graf et al. (2008) and Limongelli et al. (2008) work, learning styles are calculated based only on the ILSQ questionnaire and none of those works are authoring tools. There are several authoring tools used to create mobile applications like *MyLearning* (Attewell et al., 2009), *Zirada* (http://www.itclearning.com.au/), Test Editor (Romero, Ventura, Hervás, & De Bra, 2006), or mediaBoard (Attewell et al., 2005). Some of them are more PocketPC's oriented and some are focused to quiz editing or game-based learning. None of those author tools however, have the ability of adapting to the user's learning style, nor have portability across different computer and operating system platforms.

The software tool Educa was implemented with Java version 1.6 and XML. The learning material produced with the tool is platform independent and it has been tested with different software emulators (NHAL Win32 Emulator, J2ME Wireless Toolkit and Sony Ericsson SDK), cell and smart phones (Sony Ericsson and Nokia). We presented a test with one of such mobile devices. Currently empirical studies are taking place to examine the students' reaction to the learning material produced using **Educa**. Future work should focus on continuing testing with official and complete material for course programs in public and private schools in Mexico. Right now, we are starting working with social networking. In this case, the main idea is to give support to create and maintain social learning networks with adaptation possibilities.

Acknowledgments

The work described in this paper is fully supported by a grant from the DGEST (Dirección General de Educación Superior Tecnológica) in México under the program "support and development of academic bodies" [Academic Body: Research in Software Engineering].

References

Arruarte, A., Fernández, I., Ferrero, B., & Greer, J. (1997). The IRIS shell: How to build ITSs from Pedagogical and design requisites. *International Journal of Artificial Intelligence in Education*, 341–381.

Attewell, J. (2005). From research and development to mobile learning: tools for education and training providers and their learners. <www.mlearn.org.za/CD/papers/Attewell.pdf>, retrieved October 9, 2009.

Attewell, J. (2009). Mobile technologies and learning: A technology update and mlearning project summary. London: Learning and skills development agency. <www.m-learning.org/reports.shtml>, retrieved October 9, 2009.

Brusilovsky, P., & Schwarz, E. (1998). Web-based education for all: A tool for developing adaptive courseware. *Computer Networks and ISDN Systems, 30*(1–7), 291–300.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User Adapted Interactions, 12*(4), 331–370.

Carmona, C., Bueno, D., EduardoGuzman, & Conejo, R. (2002). SIGUE: Making web courses adaptive. In: *Proceedings of second international conference on adaptive hypermedia and adaptive web-based systems (AH'2002)*, Málaga, Spain, pp. 376–379.

Carmona, C., Castillo, G., & Millán, E. (2008). Designing a Bayesian network for modeling student's learning styles. In: *Proceedings of the eighth international conference on advanced learning technologies (ICALT 2008)*, pp. 346–350.

de Jong, T., Limbach, R., Gellevij, M., Kuyper, M., Pieters, J., & van Joolingen, W. R. (1999). Cognitive tools to support the instructional design of simulation-based discovery learning environment: The SIMQUEST authoring system. In *Design approaches and tools in education and training* (pp. 215–225). Dordrecht: Kluwer Academic Publishers..

Doignon, J. –P., & Falmagne, J. C. (1999). *Knowledge spaces*. Springer-Verlag.

Felder, R.M., & Solomon, B.A. (2004). Index of Learning Styles Questionnaire. <http://www.engr.ncsu.edu/learningstyles/ilsweb.html>, retrieved October 9, 2009.

Felder, R. M., & Silverman, L. K. (1988). Learning and teaching styles in engineering education. *Engineering Education, 78*, 674–681.

Graf, S., Kinshuk & Liu, T. (2008). Identifying learning styles in learning management systems by using indications from students' behavior. In *Proceedings of the Eighth International Conference on Advanced Learning Technologies (ICALT 2008)* (pp. 482–486). IEEE Computer Society.

Hage, H., & Aimeur, E. (2008). Harnessing learner's collective intelligence: A web 2.0 approach to e-learning. In *Proceeding of the Nineth International Conference on Intelligent Tutoring Systems (ITS 2008). LNCS* (Vol. 5091, pp. 438–447). Heidelberg: Springer.

Kerly, A., & Bull, S. (2008). Children's Interactions with Inspectable and Negotiated Learner Models. In *Proceedings of the Nineth International Conference on Intelligent Tutoring Systems (ITS 2008). LNCS* (Vol. 5091, pp. 132–141). Heidelberg: Springer.

Kohonen, T. (1989). *Self-organization and associative memory* (Third ed.). Springer-Verlag.

Limongelli, C., Sciarrone, F., & Vaste, J. (2008). LS-PLAN: An effective combination of dynamic courseware generation and learning styles in web-based education. In *Proceedings of the Fifth International Conference on Adaptive Hypermedia and Adaptive Web-based Systems (AH 2008). LNCS* (Vol. 5149, pp. 133–142). Heidelberg: Springer.

Murray, T., Blessing, S., & Ainsworth, S. (2003). *Authoring tools for advanced technology learning environments*. Kluwer Academic Publishers.

O'Reilly, T. (2005). What is Web 2.0. <http://oreilly.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html>, retrieved October 9, 2009.

Romero, C., Ventura, S., Hervás, C., & De Bra, P. (2006). An authoring tool for building both mobile adaptable tests and web-based adaptive or classic tests. In

Proceedings of the Third International Conference on Adaptive Hypermedia and Adaptive Web-based Systems (AH 2006) (pp. 203–212). Springer-Verlag.

Specht, M., Kravcik, M., Klemke, R., Pesin, L., & Hüttenhain, R. (2002). Adaptive learning environment (ALE) for teaching and learning in WINDS. In *Proceedings of the Second International Conference on Adaptive Hypermedia and Adaptive Web-based Systems (AH'2002)* (pp. 572–581). Springer-Verlag.

Weber, G., Kuhl, H.-C. & Weibelzahl, S. (2001). Developing adaptive internet based courses with the authoring system NetCoach. <http://wwwis.win.tue.nl/ah2001/papers/GWeber-UM01.pdf>, retrieved October 9, 2009.

Zatarain-Cabada, R., Barrón-Estrada, M. L., Sandoval, G., Osorio, M., Urías, E., & Reyes-García, Carlos A. (2008). Authoring neuro-fuzzy tutoring systems for M and e-learning. In *Proceedings of the Seventh Mexican International Conference on Artificial Intelligence (MICAI 2008)*. *LNCS* (Vol. 5317, pp. 789–796). Springer-Verlag.

Zirada Mobile Publisher, the Premier Mobile Content Creation Tool. <http://www.itclearning.com.au/>, retrieved October 9, 2009.