Expert Systems With Applications 75 (2017) 94–108

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Learning style Identifier: Improving the precision of learning style identification through computational intelligence algorithms

Jason Bernard [a,*], Ting-Wen Chang [b], Elvira Popescu [c], Sabine Graf [a]

[a] School of Computing and Information Systems, Athabasca University, 1200 10011-109 Street, Edmonton, AB T5J3S8, Canada
[b] Smart Learning Institute, Beijing Normal University, China, 19 Xinjiekou Outer Street, Haidian, Beijing, 1008700875, China
[c] Faculty of Automation, Computers and Electronics, University of Craiova, Strada Alexandru Ioan Cuza 13, Craiova 200585, Romania

ARTICLE INFO

Article history:
Received 11 November 2016
Revised 3 January 2017
Accepted 24 January 2017
Available online 24 January 2017

Keywords:
Intelligent tutoring systems
Distance education and telelearning
Interactive learning environments
Computational intelligence

ABSTRACT

Identifying students' learning styles has several benefits such as making students aware of their strengths and weaknesses when it comes to learning and the possibility to personalize their learning environment to their learning styles. While there exist learning style questionnaires for identifying a student's learning style, such questionnaires have several disadvantages and therefore, research has been conducted on automatically identifying learning styles from students' behavior in a learning environment. Current approaches to automatically identify learning styles have an average precision between 66% and 77%, which shows the need for improvements in order to use such automatic approaches reliably in learning environments. In this paper, four computational intelligence algorithms (artificial neural network, genetic algorithm, ant colony system and particle swarm optimization) have been investigated with respect to their potential to improve the precision of automatic learning style identification. Each algorithm was evaluated with data from 75 students. The artificial neural network shows the most promising results with an average precision of 80.7%, followed by particle swarm optimization with an average precision of 79.1%. Improving the precision of automatic learning style identification allows more students to benefit from more accurate information about their learning styles as well as more accurate personalization towards accommodating their learning styles in a learning environment. Furthermore, teachers can have a better understanding of their students and be able to provide more appropriate interventions.

1. Introduction

Learning styles describe the preferences a student has for how material is presented, how to work with material and how to internalize information (Felder & Soloman, 2000). Knowing a student's learning styles can help in several ways to improve the learning process. For example, personalizing content to the learning styles of students has been found to be beneficial to learning in several ways such as improving satisfaction (Popescu, 2010), learning outcomes (Bajraktarevic, Hall, & Fullick, 2003), and reducing the time needed to learn (Graf, Chung, Liu, & Kinshuk, 2009). Students may also be enlightened by understanding their own learning styles (Felder & Spurlin, 2005). This enlightenment on their strengths and weaknesses allows them to make better choices when self-regulating their learning. Furthermore, teachers can benefit from knowing students' learning styles as they can then provide more appropriate interventions based on an individual student's learning styles.

In order to identify students' learning styles, dedicated questionnaires can be used. Although typically valid and reliable, such questionnaires have notable drawbacks. First, they take the student away from the actual learning task. Second, such questionnaires may sometimes misidentify students due to several factors. A student's perceived importance of the questionnaire can lead to a misidentification of their learning styles as they may answer the questions very quickly without much thought. Further, students' answers may be biased by personal misconceptions or from perceived expectations. To overcome these drawbacks, research has focused on automatic approaches that identify students' learning styles from their behavior in a learning system (e.g., Carmona, Castillo, & Millán, 2008; Cha et al., 2006; Dorça, Lima, Fernandes, & Lopes, 2013; García, Amandi, Schiaffino, & Campo, 2007; Graf, Kinshuk, & Liu, 2009; Latham, Crockett, McLean, & Edmonds, 2012; Özpolat & Akar, 2009; Villaverde, Godoy, & Amandi, 2006). Such an automatic approach reduces intrusiveness by working in the background as the student uses the learning system. Furthermore, automatic approaches are not influenced by students' perceived im-

* Corresponding author.
E-mail addresses: c.j.bernard@ieee.org (J. Bernard), tingwenchang@bnu.edu.cn (T.-W. Chang), popescu_elvira@software.ucv.ro (E. Popescu), sabineg@athabascau.ca (S. Graf).

http://dx.doi.org/10.1016/j.eswa.2017.01.021
0957-4174/

portance, preconceptions or expectations with respect to learning styles as only their actual behaviors are considered.

As the current automatic approaches yield results between 66% and 77% average precision at identifying learning styles, there is a need for improvement before such approaches can be effectively used. The goal of this research is to investigate the use of computational intelligence (CI) algorithms to improve the precision of automatic approaches to identify learning styles. In this paper, we introduce four approaches to identify learning styles (LSID-ANN, LSID-GA, LSID-ACS and LSID-PSO), each using a different CI algorithm (i.e., artificial neural network (ANN), genetic algorithm (GA), ant colony system (ACS), and particle swarm optimization (PSO)). These approaches are designed to work in any learning system and have been evaluated with real data from the learning management system Moodle.

The remainder of this paper is structured as follows. Section 2 provides background information on learning styles and describes related works on automatic approaches used to identify learning styles. Section 3 provides an overview of the CI algorithms used in this research. Section 4 describes the four proposed approaches and how the CI algorithms were adapted to identify learning styles. Section 5 presents the evaluation of the approaches. This includes the performance metrics used, the process for optimizing the control parameters and examining overfitting reduction strategies, and the results and observations on the results. Finally, Section 6 concludes this work and suggests future directions to be taken.

2. Related work

This section starts with providing some background information on learning styles. Furthermore, related works on automatic approaches for identifying learning styles are presented.

2.1. Learning styles

Individual students learn in different ways (e.g., Felder & Soloman, 2000; Kolb, 1971). For example, some students may learn best while working in groups, while others learn best working alone. Another example is that some students may prefer to learn by doing something while others prefer to read and reflect about it. Learning styles are not just about a preference for a particular type of activity, but rather describe the entirety of the preferences a student has for how learning material is presented, how they process information and how they internalize the information (Felder & Soloman, 2000). While there is not a single definition for learning styles, some of the more popular definitions indicate that learning styles are “a description of the attitudes and behaviors which determine an individual’s preferred way of learning” (Honey & Mumford, 1992), “characteristic strengths and preferences in the ways they (learners) take in and process information” (Felder, 1996) and “a complex manner in which, and conditions under which, learners most efficiently and most effectively perceive, process, store, and recall what they are attempting to learn” (James & Gardner, 1995). Based on different definitions of learning styles, different learning style models have been proposed, including, for example, those by Felder and Silverman (1988), Kolb (1981), Pask (1976) and Honey and Mumford (1992).

As already mentioned, the knowledge about and the identification of learning styles can benefit learners in many ways and can significantly help in improving the learning process. However, there also has been some critique on learning styles. For example, a criticism of learning styles is that using learning styles in a face-to-face environment requires an impractical effort for teachers (Coffield, Moseley, Hall, & Ecclestone, 2004a). In a classroom with many students and therefore, many different learning styles represented, there is no possibility that a teacher could adapt to and consider each student’s individual learning style. Solving that issue by splitting classrooms by learning styles is also logistically impractical in large scales (Coffield et al., 2004a). However, with the increase in online education and the use of learning systems (Dahlstrom, Walker, & Dziuban, 2013) this criticism becomes less meaningful as it has already been shown that learning systems can adapt to individual students’ learning styles and that such approach benefits learners (Bajraktarevic et al., 2003; Popescu, 2010; Graf, Chung et al., 2009). Furthermore, even those who raise these criticisms agree that identification of learning styles is useful as a means of self-awareness for students (Coffield et al., 2004a). When students know their own learning styles it allows them to use their strengths to their advantage to succeed at learning and understand their weaknesses when they struggle (Coffield et al., 2004a; Felder & Spurlin, 2005). Another point of criticism is the method for measuring learning styles. Most often, learning styles are identified through questionnaires. Such questionnaires have some general drawbacks, including that they make certain assumptions about learners that may not be true. For example, it is assumed that learners are motivated to fill out such questionnaires, that they will fill out the questionnaire truthfully (without the influence of perceived expectations) and that they actually know how they prefer to learn. In addition, questionnaires have the drawback that they are typically filled out only once and if learning styles change over time, then the results of these questionnaires would become outdated. Furthermore, some currently available questionnaires have issues with respect to validity and reliability. Coffield, Moseley, Hall, and Ecclestone (2004b) argued that from the 13 major learning style models they have identified and studied, only three of the models are close to being considered valid and reliable. Given this critic on the current method of measuring learning styles, it is only natural to investigate other approaches to identify learning styles. As such, automatic approaches, where data from students’ behaviors within a learning system are used, seem ideal as they overcome most of the drawbacks of questionnaires. Such approaches do not rely on any assumptions with respect to students’ motivation and attitudes, as they are purely based on student behavior. Furthermore, such approaches can use data over a period of time and therefore, update information on students’ learning styles frequently, ensuring that this information never gets outdated.

For this research, the Felder-Silverman learning styles model (FSLSM) (1988) has been selected for several reasons. The FSLSM brings together different elements from the models by Kolb (1981), Pask (1976) and Myers-Briggs (1962). The FSLSM uses four dimensions: active / reflective (A/R), sensing / intuitive (S/I), visual / verbal (V/V) and sequential / global (S/G), allowing the students’ learning styles to be described in great detail. The A/R dimension describes a preference for processing information, with active students preferring to learn by doing, experimentation and collaboration, while reflective students preferring to think and absorb the information alone or in small groups. The S/I dimension describes how students prefer to gather information. Students with a sensing preference tend to gather information by the use of their senses, i.e. interacting with the real world. Students with an intuitive learning style tend to prefer to gather information indirectly, through the use of speculation or imagination. The V/V dimension describes the input preference of the student. Visual students prefer materials such as graphs, charts or videos, while verbal students prefer words whether written or spoken. Lastly, the S/G dimension describes how students prefer to have information organized. Sequential students prefer information to be provided in a linear (serial) fashion and tend to make small steps through learning material. Global students tend to make larger leaps from non-understanding to understanding and tend to require seeing the “big picture” before understanding a topic. Another reason for se-

Table 1
Summary of existing automatic approaches to identify learning styles.

Study	Algorithm	Evaluated	Limitations
Latham et al. (2012)	Rule-based	Yes	Non-generic
Graf, Kinshuk et al. (2009)	Rule-based	Yes	–
García et al. (2007)	Bayesian network	Yes	Cannot identify V/V dimension
Özpolat and Akar (2009)	NB tree classification	Yes	–
Cha et al. (2006)	Decision tree Hidden Markov model	Yes	Identifies subset of students only
Villaverde et al. (2006)	Artificial neural network	No	Cannot identify V/V dimension Simulated data
Carmona et al. (2008)	Bayesian network	No	Students must rate LOs
Dorça et al. (2013)	Reinforcement learning	No	Simulated data

lecting this model is that unlike many other learning style models the FSLSM treats each dimension as a tendency instead of an absolute type. This is done by representing each dimension on a scale from +11 to −11, allowing the strength of each preference to be described. The FSLSM also has a valid and reliable questionnaire (Felder & Spurlin, 2005) for identifying learning styles, the Index of Learning Styles (ILS) (Felder & Solomon, 1998). Research has found that the FSLSM is well-suited to be used as a model for providing personalized courses in learning systems (Kuljis & Liu, 2005) and correspondingly much research has been done using the FSLSM to provide personalization (Carver, Howard, & Lane, 1999; Filippidis & Tsoukalas, 2009; Graf, 2007; Klašnja-Milićević, Vesin, Ivanović, & Budimac, 2011).

2.2. Existing approaches to automatically identifying learning styles

Graf (2007) describes automatic approaches for learning style identification as either data-driven or literature-based. Data-driven approaches use algorithms from the fields of data mining, machine learning, artificial intelligence and/or computational intelligence to construct a model/network from existing student behavior data and their actual learning styles. Subsequently, this model/network can then be used to identify learning styles using as input student behavior data. Literature-based approaches do not build a model/network based on student data but use rules from literature to build a respective model/network. Similar to a questionnaire, the literature-based approach uses as input student behavior data to "answer questions" and applies predefined rules from literature to calculate learning styles from student behavior data. In the following paragraphs, an overview of existing automatic approaches that use the FSLSM is provided and these approaches are summarized in Table 1.

Latham et al. (2012) propose a literature-based approach used with a natural language conversational agent called Oscar. In their approach, logical rules are derived from literature, for example if a student answers a question correctly after being shown an image they may have a visual preference. The students' behaviors are extracted from the dialogs between the agent and student and the rule set is applied to the data to identify the learning styles. In an evaluation, learning styles have been identified with a precision of 86% for A/R dimension, 75% for S/I, 83% for V/V and 72% for S/G. While these are very promising results, a significant drawback of this approach is that it is tied to the Oscar system and cannot be generalized to other systems.

Another literature-based approach, "Detecting Learning Styles" (DeLeS) (Graf, Kinshuk et al., 2009) uses student behavior data related to different types of learning objects (e.g., content, quizzes, forums, etc.) to identify learning styles in learning systems in general. Rules have been derived from literature based on relations between student behavior and learning styles and such behaviors then give hints towards a particular learning style. For example, if a student has accessed many exercises, this gives a hint towards an active learning style. The proposed approach has been evaluated and learning styles have been identified with a precision of 79% for A/R, 77% for S/I, 77% for V/V and 73% for S/G.

García et al. (2007) used a Bayesian network (BN) in order to detect students' learning styles. They identified various behaviors that may be relevant to identifying learning styles in a given learning system. Then, a BN was trained with data from 50 students, using initial probabilities based on expert knowledge. The trained BN was then evaluated using 27 students. For each student, the BN provides a probability that the student has a particular learning style preference. As a result, the approach obtained an overall precision of 58% for A/R, 77% for S/I and 63% for S/G (the V/V dimension was not considered).

Özpolat and Akar (2009) developed and evaluated an approach that focuses on examining the learning objects (LOs) selected by students as being most useful in response to a keyword search. The keyword attributes of the LOs are mapped to learning style and NB tree classification is used to classify students with respect to their learning styles based on the selected LOs and their keywords. For example, if the student selects LOs with the keyword attributes such as graphs, charts or jpg then the student is more likely to be classified as having a visual preference. The approach has been evaluated with 40 students and as a result, a precision of 70.0% for A/R, 73.3% for S/I, 73.3% for S/G and 53.3% for V/V was obtained.

Cha et al. (2006) evaluate two approaches in their study, decision trees and hidden Markov models (HMM). Similar to other approaches, numerous behaviors are used as input such as the number of clicks on particular icons, time spent on some activities, quiz grades and reading or posting to forums. The decision tree as well as the HMM were evaluated with data from 70 students; however, data from students with a balanced learning style were removed from the dataset. A subset of the data is used to train the decision tree based on knowing the actual learning styles. The HMM-based approach is trained to recognize the sequence of buttons clicked from students with known learning styles. The HMM is used to identify a future student's learning styles from their click sequence. For the decision tree, they found a precision of 66.7% for A/R, 77.8% for S/I, 100% for V/V and 71.4% for S/G. The HMM approach is reported as having precision values of 66.7% for A/R, 77.8% for S/I, 85.7% for V/V and 85.7% for S/G. Although some of the results are quite high, as mentioned above they excluded all students with a balanced learning style preference. Thus, the approaches have a major limitation in that they can only identify students with a strong learning style preference one way or another, and they cannot identify students with a balanced preference.

Villaverde et al. (2006) proposed the use of a feed forward ANN (a 3-layer perceptron) with backpropagation under a supervised learning model to identify learning styles. Ten behavior patterns were used as inputs such as what kind of reading material did the student prefer, does the student revise their answers on exams prior to submission and does the student ignore, post or read forums? As output, the neural network produces three values, rep-

resenting the learning styles on three of the four learning style dimensions of the FSLSM. In an evaluation with simulated data, an average precision of 69.3% across the three dimensions was obtained. LSID-ANN is similar to this approach in that they both use a 3-layer perceptron with behavior patterns as inputs; however, LSID-ANN is expected to provide better results for two reasons. First, since a separate ANN is built for each FSLSM dimension each LSID-ANN is able to be more specialized and therefore more precise. Second, since Cha et al. (2006) used a single ANN, all of the behavior patterns are used as inputs for all three learning style dimensions they considered; however, it is likely that some of the behavior patterns were not relevant for some of the learning style dimensions. For each LSID-ANN only those behavior patterns expected to be relevant for each learning style dimension are used as inputs.

Carmona et al. (2008) also explored the use of a dynamic Bayesian network to identify learning styles. To LOs in the system they associated five learning style relevant attributes: format (image, text, etc.), resource type (exercise, example, etc.), interactivity level (very low to very high), interactivity type (active, expositive or mixed) and semantic density (very low to very high). Each of the attributes is mapped to one or more FSLSM dimension(s). Every time a student selected a LOs, they would be asked to rate the usefulness of the material from 1 to 4. After rating the LO, the rating is used as evidence to adjust the belief state of the network. The drawback to this approach, when compared to other automatic approaches, is that it requires input from the student instead of working solely on their behaviors. This is potentially intrusive for the student and there is no guarantee that the student will rate solely based on how it appealed to their learning styles. This approach was also not evaluated.

Dorça et al. (2013) presented an approach for identifying learning styles using reinforcement learning (Q-learning). In this approach, LOs in the system were associated to particular learning styles based on expert knowledge. The student is then presented LOs to achieve a learning goal followed by an assessment on how well they learned the material. The probability that the student has particular learning styles is then reinforced based on the performance assessment and how well the learning styles of the LO match the student's current predicted learning styles. Three different strategies are considered towards reinforcement, either reinforcing for high performance only, low performance only (inverse reinforcement) or both. So, for example, with a reinforcement with high performance strategy if the student performs well on an assessment it is more likely the student has the learning styles associated with the presented LO. The approach was only evaluated with simulated data.

3. Background on algorithms

Since many related works use classification algorithms for identifying learning styles, in this research we propose the use of an artificial neural network (ANN). An ANN is a universal approximator (Hornik, Stinchcombe, & White, 1990) and so it is reasoned that it may be a very suitable algorithm for identifying learning styles, given the typically small amount of data available in this line of research. In addition to ANNs, this research evaluates the use of three optimization algorithms: genetic algorithm, ant colony system and particle swarm optimization. These algorithms have been selected due to their wide-spread use to solve similar problems (Pothiya, Ngamroo, & Kongprawechnon, 2010; Robinson & Rahmat-Samii, 2004; Yao, 1999). In the following subsections, each algorithm is described in more detail.

3.1. Artificial neural network (ANN)

ANNs (Basheer & Hajmeer, 2000; Mitchell, 1997) are inspired by neurons in a brain. As such, they are described as a graph of artificial neurons. There are different ways these can be connected, with feed forward networks being the most commonly used ones, where circular connections are not permitted. ANNs are configured into layers, distinguishing between a single and multi-layer perceptron, where the latter is more common. One common configuration of a multi-layer perceptron consists of three layers: input, hidden and output. In this configuration, each neuron in each layer has a weighted connection (neural link) from each neuron in the preceding layer. Each input neuron has a single connection to an input variable. A sigmoid function (e.g., tanh) is used to calculate the strength (output) of the neuron. The input to the sigmoid function is the sum of each input multiplied by the weight of the neural link between the neurons. In this research, our approach, LSID-ANN, uses the commonly used configuration of a feed forward 3-layer perceptron.

To evaluate the ANNs outputs, one of two learning strategies are typically used: supervised or unsupervised learning. With supervised learning, the correct answer for the sample is known, so that the output can be directly compared with the answer. With unsupervised learning the correct answer for the sample is unknown. Unsupervised learning is generally used for finding clusters or other structures in data. LSID-ANN uses a supervised learning strategy since training data include the actual learning styles of students.

Lastly, there are numerous techniques for training an ANN. One of the most commonly used training algorithms is back propagation, which is also used for LSID-ANN. Using back propagation, the ANN is traversed in reverse along each neural link, adjusting the weights and thresholds by calculating a weight modification (ΔW) as a sigmoid function of the error between the output and actual values. The weight modification is further multiplied by a learning rate ($\eta > 0$) which helps control the size of the steps to take to keep the ANN from oscillating over the optimum. Training may be further optimized by adding momentum ($m \geq 0$) which aids in escaping local optima. Lastly, two training modes are possible: individual and ensemble. In individual training the weight modification is applied after the sample is run, whereas with ensemble training they are summed and applied at the end of the generation. The back propagation algorithm then runs until a termination condition is reached.

3.2. Genetic algorithm (GA)

A genetic algorithm (GA) (Grefenstette, 1986; Srinivas & Patnaik, 1994) is an optimization algorithm that utilizes concepts from evolutionary biology to solve optimization problems. In biology, when two individuals produce offspring, they intermix their genes. If one offspring is more fit than others, it will tend to survive and produce offspring of its own and so propagate its parents' genes. In this way, over time, increasingly fit offspring are evolved. With GA, a population of P genomes is produced. Each genome consists of N genes and each gene represents a component of the solution. In each generation, pairs of genomes are selected from the population favoring genomes with higher fitness (e.g., by using the roulette wheel technique where the chance of each genome being selected is equal to its fitness divided by the sum of all genomes' fitness values). A crossover process is conducted where each pair has some genes selected and swapped between them (based on the crossover weight); thereby, producing two new genomes (offspring). This crossover operation allows the genetic algorithm to make leaps far from its current position, making it useful for escaping local optima. Then each offspring can be mutated by se-

lecting some of its genes and changing them to a random value (based on the mutation weight). Mutation ensures that gene diversity does not stagnate by injecting new values into the genomes. Finally, the new population is built from a mix of the genomes and their offspring, depending on their fitness values. With the selection of a new population, a new generation starts.

The functioning of the GA is controlled by three parameters and a termination condition. The parameters are: population size (P), crossover weight (C) and mutation weight (M). The population size dictates the number of genomes in the population. The crossover weight determines the likelihood that any particular gene will be used for crossover; a high crossover weight encourages exploration but can disrupt good combinations of genes. The mutation weight determines the likelihood that a gene will be mutated, ensuring that particular gene values do not persist forever. The termination condition tells the GA when to stop processing.

3.3. Ant colony system (ACS)

ACS (Dorigo & Gambardella, 1997b) is one of several algorithms collectively referred to as ant colony optimization. These algorithms are inspired by the way ants share information while foraging for food. In such scenario, ants lay pheromone trails to food sources, thereby indirectly communicating to the other ants about the path to the food. With ACS, a graph is built where each node represents a part of a solution to the problem and the whole path through the graph represents a solution. Each link from one node to another has a problem specific inherent desirability (local quality) and may also have pheromone laid upon it. The overall quality (Q) of a link is a function of the local quality (l) and amount of pheromones (τ) on the link, with two parameters ($\alpha \geq 0$ and $\beta \geq 0$) weighting the local quality and amount of pheromones respectively (shown in Formula (1)).

To start the algorithm, a colony (population) of P artificial ants is created and they are set to traverse the graph in the following manner. Each ant is placed at a starting node, and depending on the problem this may be a fixed node or selected in some fashion, for example, randomly. The ant then chooses a link to follow by a pseudorandom proportional rule described as follows. A random value from 0 to 1 is selected and if this value is less than the exploitation parameter ($0 \leq q_0 \leq 1$) then the link with the highest quality (Q) is selected. Otherwise the link is selected by roulette wheel technique (shown in Formula (2)), where the chance of selection is equal to the quality of the link divided by the total quality of all links from that node. Optionally, the list of nodes to be considered under the pseudorandom proportional rule can include only nodes that are from a candidate list, which is a pre-generated static list of preferred choices from each node. Typically, a tabu list is maintained for each ant so that it does not return to a previously visited node. If the ant cannot move to any node, then it is considered stuck and depending on the problem this may mean the solution is complete or invalid. In order to encourage exploration, as an ant traverses a link, it consumes a portion ($0 \leq \tau_0 \leq 1$) of the pheromone (shown in Formula (3)). This makes it less likely that ants which follow will take the same path. Once all of the ants have completed traversing the graph, a portion of pheromones ($0 \leq \rho \leq 1$) from each link in the graph is evaporated causing the colony to forget a little and so encourages exploration (shown in Formula (4)). Furthermore, the global best path or the iteration best path has pheromones laid on each link of the path as a function of the fitness of the solution. Subsequently, a new generation starts with ants starting again to move through the graph, until a termination condition is fulfilled.

$$Q = l^{\alpha} \times \tau^{\beta} \tag{1}$$

$$S_n = \frac{Q(l_n, \tau_n)}{\sum_{x=1}^{n} Q(l_x, \tau_x)} \tag{2}$$

$$\tau = (1 - \tau_0) \times \tau' \tag{3}$$

$$\tau = (1 - \rho) \times \tau' \tag{4}$$

3.4. Particle swarm optimization (PSO)

PSO (Eberhart & Kennedy, 1995) is an optimization algorithm inspired by flocking movements in nature, such as birds. PSO is used where the solution space to a problem can be represented as a hyperspace or hypershape. The position of a particle represents a candidate solution to the problem being solved. Each of the coordinates in the position represents a component of the solution although how the coordinates are decoded is problem specific. PSO uses search by social intelligence as the population of particles share information as they fly through the space and adjust their trajectories to focus on promising areas.

PSO has the following control parameters: population size, inertia, particle best acceleration, global best acceleration and maximum velocity which are described as follows (Clerc & Kennedy, 2002; Eberhart & Kennedy, 1995; Shi & Eberhart, 1998). The population size is the number of particles in the swarm. Inertia ($0 \leq w \leq 1$) keeps the particle moving in the same direction it is going and so higher values promote global exploration. The particle is encouraged to turn from its current position (X_{curr}) towards the particle best position so far (X_{pbest}) and the global best position (X_{gbest}) based on the particle best acceleration ($0 \leq c1 \leq 1$) and global best acceleration ($0 < c2 \leq 1$) (Eberhart & Kennedy, 1995). Higher acceleration rates promote exploitation of the most promising areas found so far. The maximum velocity ($Vmax$) helps to prevent the particles from exploring too far from promising areas, although if it is set too low the particles may not be able to explore enough space to find the best promising areas.

To use PSO, a population of particles is created and placed inside the hyperspace (or hypershape) with an initial position and velocity. In each generation, each particle's position is updated by a velocity vector calculated using Formula (5) where V_0 is the particle's current velocity and $rand_1$ and $rand_2$ are random values from 0 to 1. After each generation, the particle best position so far (X_{pbest}) and the global best position (X_{gbest}) may be updated based on found solutions and a new generation is started, until a termination condition is reached.

$$V = w \times V_0 + rand_1 \times c1 \times \left(X_{curr} - X_{pbest}\right) + rand_2 \times c2 \times \left(X_{curr} - X_{gbest}\right) \tag{5}$$

4. Learning styles identifiers

In this section, four Learning Styles Identifier (LSID) approaches are introduced, each using a different CI algorithm. These approaches aim at investigating the suitability of CI algorithms for improving the precision of identifying learning styles. Furthermore, it was a requirement that these approaches would be applicable and useful for any learning system rather than being specific to a particular system.

In order to ensure that the LSID approaches are generalizable and applicable for any learning system, it was key to base the approaches on generic behavior patterns that can be collected in any learning system. After a literature review, we found that the behavior patterns used in DeLeS (Graf, Kinshuk et al., 2009) are highly suitable for our approaches. These behavior patterns were designed to be generic and focus on elements that are available

Table 2
Relevant behavior patterns for each FSLSM dimension (Graf, Kinshuk et al., 2009).

Active/Reflective	Sensing/Intuitive	Visual/Verbal	Sequential/Global
content_stay (−)	content_stay (−)	content_visit (−)	outline_stay (−)
content_visit (−)	content_visit (−)	forum_post (−)	outline_visit (−)
example_stay (−)	example_stay (+)	forum_stay (−)	question_detail (+)
exercise_stay (+)	example_visit (+)	forum_visit (−)	question_develop (−)
exercise_visit (+)	exercise_visit (+)	question_graphics (+)	question_interpret (−)
forum_post (+)	question_concepts (−)	question_text (−)	question_overview (−)
forum_visit (−)	question_details (+)		navigation_overview_stay (−)
outline_stay (−)	question_develop (−)		navigation_overview_visit (−)
quiz_stay_results (−)	question_facts (+)		navigation_skip (−)
self_assess_stay (−)	quiz_revisions (+)		
self_assess_twice_wrong (+)	quiz_results_stay (+)		
self_assess_visit (+)	self_assess_stay (+)		
	self_assess_visit (+)		

in different learning systems. Furthermore, they are based on an extensive literature review and experimentations to link the behavior patterns with learning styles, ensuring that each behavior pattern is in fact indicating a certain learning style. The behavior patterns for each learning style dimension are shown in Table 2, using a "+" and "-" for indicating a high and low occurrence of the respective pattern from the viewpoint of an active, sensing, visual, and sequential learning style. The patterns are based on different types of learning objects including outlines, content, examples, self-assessment quizzes, exercises, and forums. Furthermore, general navigation of students through the course is considered. Patterns consider how long a student stayed on a certain type of learning object (e.g., content_stay) and how often a student visited a certain type of learning object (e.g., content_visit). Furthermore, questions of self-assessment quizzes were classified based on whether they are about facts or concepts, require knowledge about details or overview knowledge, include graphics or text only, and deal with developing or interpreting solutions. Patterns then consider how well students performed on such types of questions (e.g., question_concepts).

When designing the LSID approaches, the first step was to investigate how CI algorithms could be used to improve the precision of learning styles identification. Two possible directions were investigated in further detail. First, a data-driven approach could be used where the problem of identifying learning styles from student behavior data/patterns could be seen as a classification problem. Second, we looked into how CI algorithms can improve existing literature-based approaches. When investigating the currently leading approach, DeLeS (Graf, Kinshuk et al., 2009), in more detail, we found that DeLeS is based on the assumption that each behavior pattern contributes equally to the calculation of learning styles. While this is a reasonable assumption given the lack in studies on each behavior pattern and its weight in the calculation process of learning styles, CI algorithms can address this issue and can find optimal weights for behavior patterns based on data; hence approaching the identification of learning styles as an optimization problem. Accordingly, the proposed LSID approaches investigate the use of CI algorithms for identifying learning styles as a classification problem (described in more detail in Section 4.1) and as an optimization problem (described in more detail in Section 4.2).

From a technical viewpoint, all development and evaluation of the LSID approaches were done using the Windows 10 operating systems. The LSID approaches were developed using Microsoft's Visual C++ with the CLI extension and uses multithreading to improve runtime. Massive parallelism was investigated using C++ Accelerated Massive Parallelism; however, this did not increase speed by very much since the atomic operations, which can be parallelized in the LSID approaches, are all faster than the overhead of using massive parallelism. C++/CLI was used in lieu of C++ to facilitate targeting .NET platforms (Microsoft, 2016).

4.1. Improving learning style identification through classification

There are many different classification algorithms; however, for this research the use of ANNs is investigated with respect to its potential to improve the precision of identifying learning styles. ANNs work well with rather small datasets (Foody, McCulloch, & Yates, 1995; Swingler, 1996), which is important for this line of research as typical datasets are rather small. Furthermore, the problem can easily be translated to the network structure of an ANN. To the best of our knowledge, in previous works, only one study has investigated the use of ANNs for identifying learning styles so far. Villaverde et al. (2006) have designed one ANN to identify three out of four learning style dimensions of the FSLSM and achieved an accuracy of 69%. In our research, we propose LSID-ANN, which uses four ANNs, one per learning style dimension. Such design has the potential to improve the accuracy of learning style identification as it allows identifying all four learning style dimensions of the FSLSM and each ANN can focus on one separate task, namely the identification of learning styles for the respective dimension.

In LSID-ANN, each of its four ANNs uses the commonly used configuration of a feed forward 3-layer perceptron. Each ANN has as inputs the relevant behavior patterns for the respective learning style dimension, as listed in Table 1. Accordingly, the ANN for identifying the A/R dimension has 12 inputs, while for the S/I dimension the ANN has 13 inputs, for the V/V dimension the ANN has 6 inputs and for the S/G dimension the ANN has 9 inputs. As the inputs are of different scale, normalization was performed. Since for all patterns 0 was a valid value this was used as the lower bound. For the upper bound, for each behavior pattern the upper threshold value (T_{up}) introduced in the study by Graf, Kinshuk et al. (2009) was used. This threshold value considers the different scale of each pattern and represents a high occurrence of behavior for the respective pattern based on literature and adjusted based on the characteristics of the course. Each ANN has a single output which produces a value from 0 to 1 and represents the identified learning style value of the respective dimension. The number of hidden nodes is a control parameter and was optimized for each ANN (this process is described in Section 5.2). A sample of the topography used for the V/V dimension is shown in Fig. 1.

Each ANN uses a supervised learning strategy to train and backpropagation as training algorithm. To train the ANN, in each iteration, the ANN runs with the data of each student. Starting with the input data from the first student (i.e., data about behavior patterns), an output value (i.e., identified learning style) is produced. This output value is then compared to the actual learning style of that student (as identified by the ILS questionnaire (Felder & Solomon, 1998)). The difference between the output of the ANN

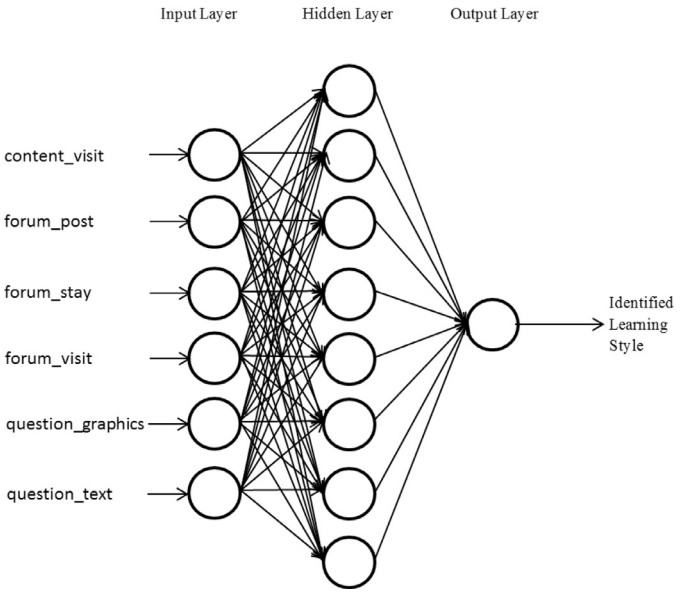

Fig. 1. LSID-ANN topography for the V/V dimension.

and the actual learning style is referred to as error (e) and the precision is calculated as 1-e. Based on the error, the weights of the ANN are corrected and the next student data are run through the ANN. At the end of an iteration (once all student data ran through the ANN), the fitness of the network is calculated to assess the overall quality of the ANN in the given iteration. This fitness is calculated as the average of the precision value over all students. Afterwards, the next iteration is started until the termination condition is met. For this research, the termination condition is selected to promote finding the optimal solution. When a new best fitness is found, the current iteration number (I_{best}) is recorded (e.g., 6000th iteration). When another I_{best} (e.g., 6000) iterations passed without finding a new best fitness, the algorithm stops. To prevent early termination, a minimum of 10,000 iterations must pass before the algorithm can terminate.

4.2. Improving learning style identification through optimization

While the previous section showed how learning style identification can be achieved through classification, this section introduces our approaches to identify learning styles by the use of optimization algorithms, in particular an ant colony system, genetic algorithm and particle swarm optimization. As described before, the introduced learning style identification approaches (i.e., LSID-GA, LSID-ACS and LSID-PSO) are all based on the existing leading learning style identification approach DeLeS (Graf, Kinshuk et al., 2009). In DeLeS, the assumption is made that each behavior pattern contributes equally to the calculation process of learning styles which is an assumption that is reasonable given the lack of available studies on this issue but probably not optimal. The proposed learning style identification approaches address this issue by extending DeLeS in a way that optimization algorithms are used to find the optimal weight for each behavior pattern. The given optimization problem has a rather large solution space with about 10^{12} to 10^{26} combinations, depending on each learning style dimension. Although brute force searching is effective for small solution spaces and guarantees finding a globally optimal solution they become intractable as the solution space size increases (Russell & Norvig, 2010). Thus, to find the optimal pattern weights requires a more efficient searching mechanism. Accordingly, GA, ACS and PSO were selected given that they are commonly used optimization algorithms and studies have shown that they are effective in solving similar problems, particularly searching for optimal weight sets (Abido, 2002; Ericsson et al., 2002; Pothiya et al., 2010). In the next paragraphs, DeLeS is introduced in more detail in order to provide background information on how the proposed LSID approaches work. Subsequently, a general overview is provided on how the proposed LSID approaches work and after that, each algorithm is described separately in a designated subsection.

To identify learning styles, DeLeS extracts behavior pattern data from the learning system's database and translates the behavior pattern data into a learning style hint h. A hint value of 3 indicates that the behavior pattern data provide a strong indication for an active, sensing, visual or sequential learning style (according to Table 1). A hint value of 2 indicates that the student's behavior is average and therefore does not provide a specific hint towards a learning style. A hint value of 1 indicates that the behavior pattern data provide a strong indication for a reflective, intuitive, verbal or global learning style (according to Table 1). A hint value of 0 indicates that no information about the student's behavior is available with respect to the respective pattern. In order to classify the behavior pattern data into learning style hints, an upper and lower threshold has been derived from literature and is used for each pattern. To calculate the learning style of a student in a given learning style dimension, an average hint value is calculated by summing up all hint values of all patterns relevant for that dimension and dividing it by the number of patterns that include available information (for the respective dimension). The resulting value is then normalized, leading to a value between 0 and 1, where 1 indicates a strong preference for an active, sensing, visual or sequential learning style and 0 indicates a strong preference for a reflective, intuitive, verbal or global learning style.

Each LSID approach follows the exactly same overall process to calculate learning styles as DeLeS, with only one difference. When DeLeS is calculating learning styles from hint values, it calculates an average hint value where each hint from a behavior pattern contributes equally to the overall learning style value. The LSID approaches use different optimization algorithms to identify optimal weights for each behavior pattern and therefore, when calculating learning styles from hint values, a weighted average hint value is computed (instead of just an average hint value), where each hint value is multiplied by the optimal weight of the respective pattern.

In the following subsections, each algorithm is explained in further detail, focusing on how it has been designed to find optimal weights of behavior patterns.

4.2.1. Using GA for weight optimization

In LSID-GA, four genetic algorithms are used to identify the weights of behavior patterns, each for one learning style dimension. The genome structure for each GA uses N gene values, where N is the number of relevant behavior patterns for the respective learning style dimension and therefore, each gene represents a relevant behavior pattern in the respective learning style dimension. Each gene is permitted to have an integer value from 1 to 100, representing the weight of the respective pattern. Thus, each genome as a whole provides a candidate solution, representing a set of weights. The genome structure for the V/V dimension is shown in Fig. 2.

To calculate the fitness of a genome (and therefore a respective solution), the error between the actual learning style (as identified by the ILS questionnaire (Felder & Solomon, 1998)) and the calculated learning style for each student in a given dataset is computed and the average error over all students is used as fitness value. The calculated learning style for each student is computed from the relevant behavior patterns of the respective dimension, as described in Section 4.2, using the genome's gene values as pattern weights.

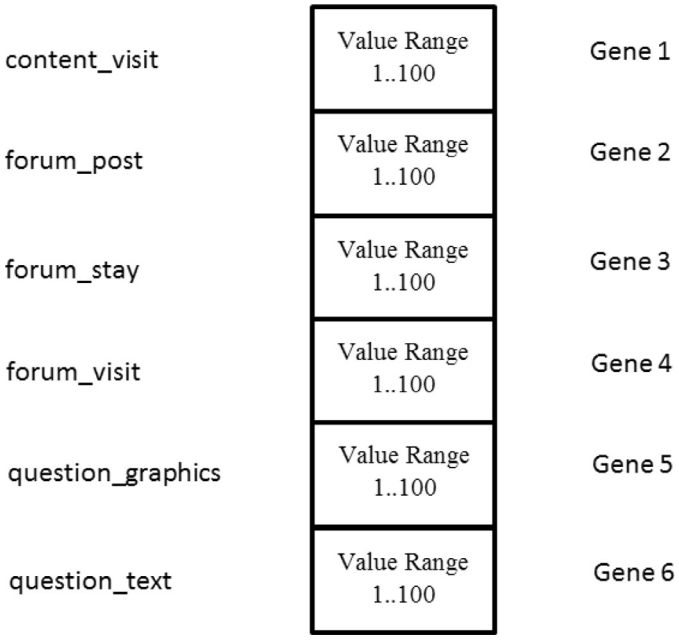

Fig. 2. Genome structure for V/V dimension.

LSID-GA uses well-known GA operators and works as follows. During initialization, the population is fully populated with P genomes using random gene values as no initial information is available on the potential quality of any weight value. For the selection operator, the roulette wheel technique is used where the probability of a genome being selected are equal to its fitness divided by the total fitness of all genomes in the population. The selection operator selects $P/2$ genome pairs. Each genome may be selected more than once; however, the same pairing is not permitted within the same generation. The crossover operator uses uniform crossover where each gene has a chance of being swapped equal to the crossover weight. The mutation operator is applied to each of the new offspring and uses uniform mutation where each gene has a chance of being mutated equal to the mutation weight. There is a small chance that no crossover will occur, depending on the crossover weight. In such case, at least one gene will be forced to mutate in each genome to create new offspring. After crossover and mutation, the new genomes are merged into the population and the survival process culls the genomes with the lowest fitness until the population is back to its original size (P). Then, a new generation starts until the termination condition is reached.

The termination condition for LSID-GA is the same as for LSID-ANN and again was selected to promote finding an optimal solution. When a new best solution is found, the current generation number (G_{best}) is recorded (e.g., 6000th generation). LSID-GA terminates when another G_{best} (e.g., 6000) generations passed without finding a new best solution. To prevent early termination, a minimum of 10,000 generations must pass before LSID-GA can terminate.

4.2.2. Using ACS for weight optimization

In LSID-ACS, four ACSs are designed, each for identifying the weights of patterns in one learning style dimension. ACS requires the problem to be converted into a graph for the ants to traverse. Accordingly, to find optimal pattern weights, a layered graph was designed as follows. The graph has a single start node and end node, where ants start and end their paths. In between the start and end node, there are N layers of nodes, representing the N relevant patterns for the respective learning style dimension. Each layer consists of 100 nodes with values from 0.01 to 1 in increments of 0.01, representing the possible weight values for a respective pattern. Each node in a layer is connected to every node in the next layer (or end node) but is not connected to any node of the same layer. When an ant has finished traversing the graph, it therefore has selected a single node from each layer, and this forms the candidate set of weights. A representation of the graph for the V/V dimension is shown in Fig. 3.

To determine the quality of a path (from start node to end node), the same fitness function is used as in LSID-ANN and LSID-GA. Therefore, for each student in a given dataset the learning style of the respective dimension is calculated from the relevant behavior patterns as described in Section 4.2, using the node values of the respective path as pattern weights. The error between the actual learning style and the calculated learning style for each student is then computed and the average error over all students is used as fitness value for the given path.

LSID-ACS works as follows: At the initialization stage, the local quality of the graph's links are populated; however, since there is no information on what weights might be good choices, each of the values are set to 1. Furthermore, with no initial information on the potential quality of weight values no candidate lists are constructed and the ants can transition to any node in the next layer. Since the graph is unidirectional, the ants cannot return to a previously selected node during a single pass. After the graph has been constructed and initialized, the population of ants is built. Each ant is placed on the "Start" node and permitted to traverse the graph using the pseudorandom proportional rule until it reaches the "End" node. When an ant traverses a link, it consumes a portion of the pheromone in proportion to the consumption ratio. The path from each ant is decoded into a set of optimal weights and assessed by calculating its fitness value. Once all ants have traversed the graph and fitness values have been calculated, if the best ant has a fitness value greater than the current global best path then its path is saved as the current global best path. The links along the global best path have their pheromone values updated in proportion to its fitness value. Finally, all the links in the graph lose a proportion of pheromone in proportion to the evaporation factor. Then, a new generation starts where ants are again traversing through the graph until the termination condition is reached, which is identical to that described for LSID-ANN and LSID-GA.

4.2.3. Using PSO for weight optimization

In LSID-PSO, four PSOs are designed, each for identifying the weights of patterns in one learning style dimension. PSO requires that the solution space for the problem be defined as a hyperspace or hypershape. For this research, an N-dimensional hypershape is defined, where N is the number of behavior patterns for the learning style dimension under consideration. Each hypershape dimension represents the range of possible weights for the behavior patterns and so each dimension is bounded with a minimum value of 0.01 and a maximum value of 1. As such, the location of a particle in the hypershape represents a set of weights. Fig. 4 shows the relationship between the particle's coordinates in the hypershape and the behavior patterns for the V/V learning style dimension.

To determine the quality of a candidate solution, in this case a specific position in the hypershape, the same fitness function is used as in LSID-ANN, LSID-GA and LSID-ACS. Therefore, for each student in a given dataset the learning style of the respective dimension is calculated from the relevant behavior patterns as described in Section 4.2, using the coordinate values of the respective position in the hypershape as pattern weights. The error between the actual learning style and the calculated learning style for each student is then computed and the average error over all students is used as fitness value for the given position in the hypershape.

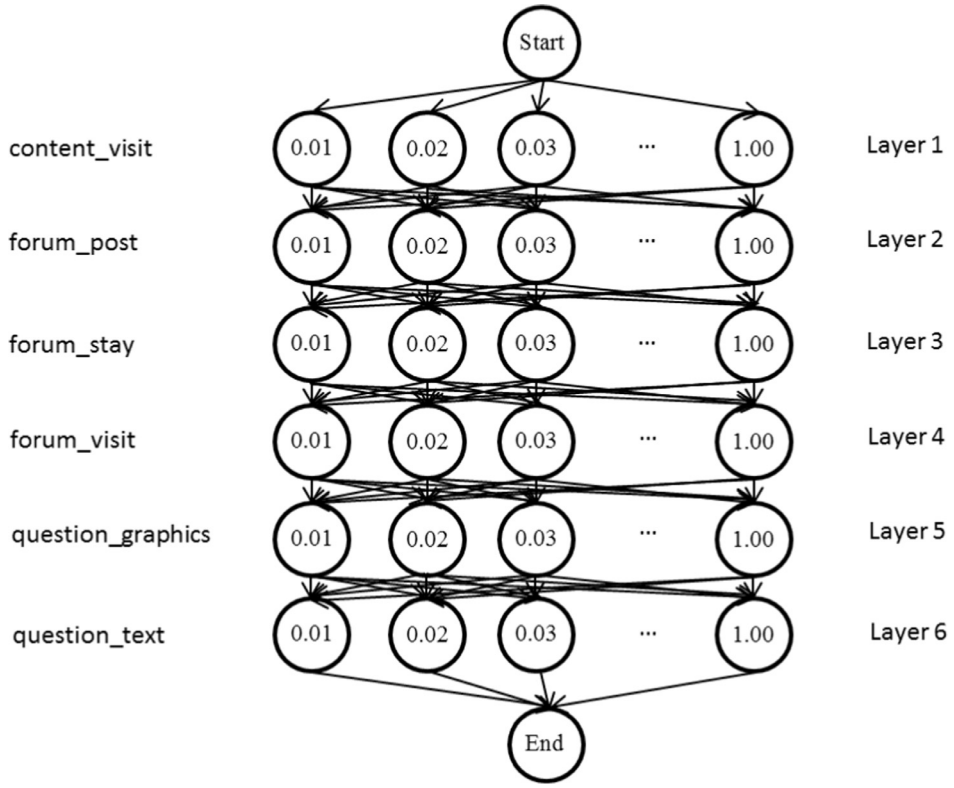

Fig. 3. Graph for finding set of weights for V/V dimension.

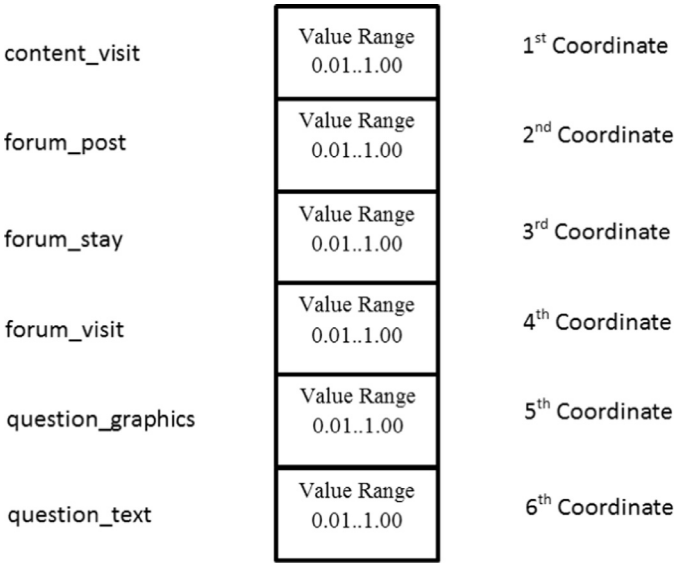

Fig. 4. LSID-PSO's coordinate to pattern relationship for V/V learning style dimension.

LSID-PSO works as follows: the population of particles is initialized with each particle receiving a random position and velocity. For each position coordinate, a random real value is picked between the hypershape bounds (0.01 to 1.00). Each component of the initial velocity vector is a random value from 0 to Vmax. After initialization, each particle is assessed using the fitness function and the global best position is updated to the position of the particle with the best fitness. The algorithm's main processing loop is then started. Each generation the particles are moved and have their velocity adjusted as previously described (Section 3.4). After each generation, each particle is re-assessed using the fitness function. If there has been any improvement, the global best position and the particles' individual best position are updated. This process continues until the termination condition is reached, which is identical to that described for LSID-ANN, LSID-GA and LSID-ACS.

5. Evaluation

To evaluate the proposed LSID approaches, the dataset used for each approach is the same dataset that has been used for evaluating DeLeS by Graf, Kinshuk et al. (2009). The dataset consists of both the behavior data from the learning management system Moodle and learning styles. The process used to acquire the behavior data from Moodle is described as follows. The data was gathered from 127 students in an undergraduate object-oriented modelling course held at a university in Austria. Moodle (2016) was used to host the course, and the built-in logging functionality of Moodle was used to acquire the data. One extension was added to Moodle to track how often students revise their self-assessment and exercise answers. Additionally, the Moodle methods used to distinguish between particular types of learning objects and questions were extended to make it easier to gather the data.

For gathering learning styles data, the ILS questionnaire was used (Felder & Solomon, 1998). In order to ensure that the identified learning styles are reliable, any student who spent less than 5 min filling out the questionnaire was eliminated from the dataset. In addition, only data from students who submitted more than half of the assignments and took the final exam were used, in order to ensure that behavior data were only used from students who tried completing the course rather than from students who dropped out early. After these removals, the final dataset consisted of 75 students. It should be noted that this dataset is of similar size or larger than those used in related works: Latham et al. (2012) used 75–95 students, García et al. (2007) used 77 stu-

dents, Özpolat and Akar (2009) used 40 students and Cha et al. (2006) used 23–49 students.

The evaluation of the LSID approaches consists of three parts. First, to find the optimal values for the parameters of each CI algorithm, an iterative experimental process was used. Second, an experimental process was also used to test overfitting reduction strategies and find optimal parameters for those strategies. Third, using the optimal parameters for each algorithm and the optimal overfitting reduction strategies, final results were obtained for each LSID approach.

In order to ensure generalizability to any dataset, a 10 fold cross validation process was used for finding the optimal parameters, finding optimal overfitting reduction strategies, and getting the final results. In a 10 fold cross validation, the algorithm is run 10 times, each time with a different training and assessment set, and the results are averaged over the 10 runs/folds. To build the different training and assessment sets, the dataset is split into 10 equal parts. For the first fold, the assessment set is comprised of the first 1/10th of data and the training set uses the remaining data; for the second fold, the assessment set is comprised of the second 1/10th of data and the training set uses the remaining data; and so on.

5.1. Performance metrics

The performance of the proposed LSID approaches is measured using four metrics. The four metrics are SIM (similarity), ACC (accuracy), LACC (lowest accuracy) and %Match (percentage of students identified with reasonable accuracy).

The first metric (SIM) is commonly used in literature to measure performance for learning style identification (García et al., 2007; Graf, Kinshuk et al., 2009; Özpolat & Akar, 2009). SIM works by dividing learning styles on each dimension into three regions. For example, for the A/R dimension, these three regions would be active, balanced and reflective. SIM returns 1 when the actual and identified learning styles are in the same region, 0.5 when they are in adjacent regions (i.e., one value is neural and the other value is pointing towards a particular learning style), and 0 when they are in opposite regions. SIM values are calculated for each student and then an average SIM value is built to measure the accuracy of the learning style identification approach. Formula (6) shows the calculation for SIM where the function R returns the region for a learning style value, LS_{id} is the identified learning style, LS_{actual} is a student's actual learning style and LS_B is the value returned by R for the balanced region. As suggested by Graf, Kinshuk et al. (2009), a threshold of 0.25 and 0.75 is used in this research to divide learning style values into the three regions, for the ILS questionnaire results and the results of the LSID approaches alike.

$$SIM = \begin{cases} 1.0 & if\ R(LS_{id}) = R(LS_{actual}) \\ 0.5 & if\ R(LS_{id}) \neq R(LS_{actual})\ and \\ & \quad (R(LS_{id}) = LS_B\ or\ R(LS_{actual}) = LS_B) \\ 0.0 & otherwise \end{cases} \tag{6}$$

While SIM is commonly used in literature, it has a drawback of reduced accuracy due to classifying results into regions. While some approaches for identifying learning styles return learning style regions as results (e.g., Bayesian networks), LSID approaches are capable of returning concrete learning style values. Accordingly, we are able to measure the exact difference between the result from LSID approaches and the actual learning style, leading to a more accurate performance metric, which we call ACC. As with SIM, ACC is calculated for each student and an average ACC is built as defined in Formula (7) where n is the number of students in the respective data set and $LS_{actual,x}$ and $LS_{id,x}$ are the actual and identified learning style values for the x^{th} student in the set.

$$\text{ACC} = \frac{\sum_{x=1}^{n} 1 - |LS_{actual,x} - LS_{id,x}|}{n} \tag{7}$$

While the above-mentioned performance metrics provide details on how accurately a proposed approach is able to identify learning styles for students on average, the next two performance metrics, LACC and %Match, investigate the accurate identification of learning styles for each single student. Looking into the accuracy of an approach for a single student is important since misidentifying a student's learning style leads to severe consequences for the respective student, in terms of providing him/her with mismatched material in an adaptive learning system or with misleading advice from a teacher who is misinformed by the system about the student's learning style. LACC is the lowest ACC value within a set of students (as shown in Formula (8)) and therefore measures the worst case scenario for any student. %Match measures the percentage of students who were identified with reasonable accuracy (as shown in Formula (9)). A threshold for reasonable accuracy was calculated by considering the range of learning style values in the dataset and assuming that ACC has to be at least higher than half of this range. Accordingly, the ACC threshold is assumed to be 0.5.

$$LACC = \min_{0<x<n} ACC\left(LS_{actual,x}, LS_{id,x}\right) \tag{8}$$

$$\%Match = \frac{\sum_{x=1}^{n} \begin{cases} 0.0 & if\ ACC\left(LS_{actual,x}, LS_{id,x}\right) < 0.5 \\ 1.0 & otherwise \end{cases}}{n} \tag{9}$$

5.2. Parameter optimization

The process to optimize the parameters for all CI algorithms is described as follows, differing only by the parameters of each algorithm. As a first step, a literature review was performed to find either a suitable range or principles for each parameter, resulting in a set of values for each parameter. A mid-range value was selected from the set of values to act as a default value (shown in bold), with the exception of the Vmax parameter for PSO as the highest value is clearly recommended. Then, for the first parameter, the respective CI algorithm is executed iteratively for each value in the set while using the default value for the remaining parameters. The parameter value which produces the best result is considered the optimal choice and used for all subsequent executions. This process is repeated for each parameter.

The control parameters for the ANN are optimized in the following order: number of hidden node (H), learning rate (η), momentum (m) and training mode. Literature suggests that the number of hidden nodes should be between $log\ T$ (where T is the size of the training set) (Wanas, Auda, Kamel, & Karray, 1998) and $2\times$ the number of inputs (Swingler, 1996). In our case, the lower bound is log 67 or 1.82. To maximize the chance to find an optimal value, the lower bound is reduced to 1 instead of being rounded up to 2. The upper bound varies by learning styles dimension since the number of behavior patterns (inputs) varies by learning styles dimension. Accordingly, values from 1 to 24 for A/R, 1 to 26 for S/I, 1 to 12 for V/V and, 1 to 18 for S/G were assessed, in steps of 1. For learning rate, a low value is suggested (Swingler, 1996) so that the ANN can converge to an optimal solution. The learning rates assessed were between 0.01 and 0.1, in steps of 0.01, leading to the following values: (0.01, 0.02, 0.03, 0.04, **0.05**, 0.06, 0.07, 0.08, 0.09, 0.1). Similarly, momentum should be low so that the ANN does not skip good areas, but needs to be large enough to aid in escaping local optima (Swingler, 1996). The values assessed for momentum were between 0 and 0.1, in steps of 0.01, leading to the following values: (0.00 0.01, 0.02, 0.03, 0.04, **0.05**, 0.06, 0.07,

Table 3
LSID-ANN optimal parameter settings.

	Number of hidden nodes	Learning rate	Momentum	Training mode
A / R	1	0.08	0.10	Individual
S / I	5	0.06	0.09	Individual
V / V	8	0.08	0.06	Individual
S / G	2	0.07	0.01	Individual

Table 4
LSID-GA optimal parameter settings.

	Population	Crossover weight	Mutation weight
A / R	200	0.80	0.03
S / I	100	0.90	0.03
V / V	100	0.80	0.03
S / G	400	0.70	0.04

0.08, 0.09, 0.1). Both individual and ensemble training modes are assessed for LSID-ANN with individual used as the default. The optimal parameters for LSID-ANN are shown in Table 3.

The control parameters optimized for the GA are: population size (P), crossover weight (C) and mutation weight (M). In general, for larger populations it is suggested to use lower crossover and mutation weights and vice versa (Grefenstette, 1986; Srinivas & Patnaik, 1994); however, since no firm relationship is known between these parameters a variety of configurations has been assessed. Grefenstette (1986) recommends either very low or very high populations. His work examined up to 160 genomes (but does not recommend this as a strict upper limit). Therefore, we expanded on this upper limit to 400 in order to maximize the chances of finding an optimal setting. This results in a set of population values of (25, 50, 100, 150, 200, 400). The crossover weight is recommended to be above 0.6 (Grefenstette, 1986; Srinivas & Patnaik, 1994), therefore the set of values assessed was (0.6, 0.7, **0.8**, 0.9). Mutation weight is recommended to be between 0.05 and 0.01 (Grefenstette, 1986) or even between 0.05 and 0.0001 (Srinivas & Patnaik, 1994). Accordingly, the set of values assessed was (0.0001, 0.001, 0.01, 0.02, 0.03, 0.04, 0.05). The optimal parameters for LSID-GA are shown in Table 4.

The following parameters are optimized for the ACS: population size (P), local quality weight (α), pheromone weight (β), exploitation parameter (q_0), evaporation ratio (ρ), and consumption ratio (τ_0) (Dorigo & Gambardella, 1997b). For the population size, Stutzle et al. (2011) examined values up to 400 and found that values above 100 generally performed poorly; however, Dorigo and Birattari (2010) state that the optimal population size is problem dependent. Therefore, the maximum value for population was slightly expanded to 200 to maximize the chance of optimization giving the set (25, 50, 100, 200). As discussed previously, there is no local information available to determine the inherent quality for any particular pattern weight. Therefore, the local information for all links is set to 1. This means that an ant's decision is only based on the amount of pheromone on a link and therefore, there is no need to optimize α and β parameters, so both are set to 1. In addition, the lack of local information significantly changes the way exploitation works in the pseudorandom proportional rule. Normally, when exploiting an ant takes the link (l) with the highest overall quality (see Formula (1)) (Dorigo & Gambardella, 1997a) which is a function of local information and pheromone. For this problem, as the local information is always 1, the ant will take the link with the highest pheromone. Since this is such a significant change to the pseudorandom proportional rule, the exploitation parameter is also evaluated as off ($q_0 = 0$). In general, the exploitation parameter is preferred to be high so that the ants will use previously found good solutions (Dorigo & Gambardella, 1997b) and accordingly, the exploitation parameter is assessed with the set of values (0.0, 0.5, 0.6, **0.7**, 0.8, 0.9). When evaluated by Stutzle et al. (2011), it was found that for the evaporation ratio values higher than 0.6 performed best with slightly lower values performing not much worse. Since Dorigo and Gambardella (1997a) suggested that the evaporation ratio may be problem specific, the set of values was expanded to include 0.5 giving the set (0.5, 0.6, **0.7**, 0.8, 0.9). The consumption ratio parameter is recommended to be $0 < \tau_0 \leq 1$ (Dorigo & Birattari, 2010). Initially, the intent was to evaluate the values from 0.1 to 1.0 in increments of 0.1; however, for this problem, values above 0.3 were found to reduce pheromone trails very quickly to near zero thereby suggesting that low values may be best, and therefore, two values lower than 0.1 were added. So, the consumption ratio values assessed were (0.01, 0.05, 0.10, 0.20, 0.30, 0.40, **0.50**, 0.60, 0.70, 0.80, 0.90). The optimal parameter values computed for LSID-ACS are shown in Table 5.

Table 5
LSID – ACS optimal parameter settings.

	Population	Exploitation	Evaporation ratio	Consumption ratio
A / R	100	0.00	0.80	0.20
S / I	200	0.00	0.80	0.20
V / V	50	0.00	0.90	0.05
S / G	200	0.00	0.50	0.20

The following parameters are optimized for PSO: population size (P), acceleration rates ($c1$ and $c2$), inertia (w) and maximum velocity ($VMax$). In much of the research on PSO, a population size of 100 or less is used (Clerc & Kennedy, 2002; Eberhart & Kennedy, 1995; Shi & Eberhart, 1998); however, this is not recommended as an absolute maximum and so was expanded to 400 resulting in a set of values to be assessed of (25, 50, 75, 100, 200, 400). The recommendation is that acceleration rates should be found experimentally as they are problem specific (Clerc & Kennedy, 2002). The range of values for the individual acceleration rate ($c1$) is 0 to 1 (Clerc & Kennedy, 2002) and so the values assessed were (0.0, 0.25, **0.5**, 0.75, 1.0). The global acceleration rate ($c2$) should satisfy $0 < c2 \leq 1$ (Clerc & Kennedy, 2002), 0.0 is excluded as the global best position must always be considered, and so the set of values assessed were (0.25, **0.5**, 0.75, 1.0). The recommended range for inertia is 0.9 to 1.2 (Shi & Eberhart, 1998); however, this was expanded slightly to allow for a greater chance of optimization giving the set of values of (0.75, 0.9, **1.0**, 1.1, 1.2). For bounded problems, a good initial value for the maximum velocity ($Vmax$) is recommended to be set equal to the extent of the bounds ($Xmax$), however, a trial-and-error process is recommended to refine the value (Shi & Eberhart, 1998). The bounds for the problem in this research are the minimum (0.01) and maximum (1.0) weights, giving an $Xmax = 0.99$. Therefore, the default value for $Vmax$ is set to 0.99 instead of a mid-range value. There is no reason to assess a value of $Vmax > Xmax$ as this has the same effect as $Vmax = Xmax$. If a particle has a velocity (v) such that $v \geq Xmax$, it simply hits the hypershape boundary. However, there is a reason to assess $Vmax \leq Xmax$ as this promotes exploitation of promising areas by preventing particles from flying away from them too quickly. The set of $Vmax$ values was found by iteratively multiplying $Xmax$ by the values (0.05, 0.1, 0.25, 0.5 and 1.0) giving a set of value for $Vmax$ of (0.0495, 0.099, 0.2475, 0.495, **0.990**). The optimal parameters for LSID-PSO are shown in Table 6.

5.3. Overfitting reduction strategies

With CI algorithms, overfitting is a common problem where the solution is fit to noise in the training data and is not generalizable to other data. Such overfitting reduces the quality of the found solution (i.e., reduces the precision of learning styles iden-

Table 6
LSID-PSO optimal parameter settings.

	Population	Acceleration			
		Global	Individual	Inertia	Vmax
A / R	400	1.00	1.00	0.75	0.990
S / I	100	1.00	0.25	1.20	0.990
V / V	400	1.00	0.50	1.00	0.099
S / G	50	1.00	1.00	0.90	0.495

Table 7
Distribution of learning style data.

	Active	Sensing	Visual	Sequential
Data used in this research	52.7%	63.0%	81.9%	53.4%
Felder and Spurlin (2005)	47–70%	46–81%	66–91%	45–85%

tification). However, techniques exist which can reduce overfitting and three such techniques were investigated for this study: stratification (Kohavi, 1995), future error prediction (FEP) (Mitchell, 1997) and weight decay (Krogh & Hertz, 1992).

Stratification works by ensuring the training sets and assessment sets are picked such that they have a similar distribution to expected future datasets. Thus, even if the algorithm's solution is overfit it will be overfit to likely future data and so will be effective anyway. The distribution of learning style data used in this study is consistent with the range of values found by Felder and Spurlin (2005) in their summary of several studies, as shown in Table 7. Therefore, it can be assumed that future data will be distributed similarly. To ensure that each training set and assessment set has a proper distribution, students are first grouped in accordance to their preference (e.g., all students with an active preference are grouped together). Then for each fold's assessment set, students are picked so that the percentage of students for each learning style preference (e.g., active, reflective, sensing, etc.) is as close as possible to the percentage of students of that learning style preference in the overall dataset. For example, for an assessment set consisting of 7 students for the A/R dimension, 4 active students are selected as this gives the closest possible percentage to 52.7%. Stratification is assessed as either on or off.

FEP functions by attempting to detect when overfitting starts to occur and then stop the algorithm's processing. This is done by extracting a validation set from the training set and using it to represent future data samples. Whenever a new best solution is found, a result is produced using the validation set. If this validation set result is lower than the previous validation set's result, it indicates that overfitting might occur and the algorithm is stopped. FEP is assessed as either on or off. In early generations, results are likely to be unstable. To prevent this instability from causing early termination, a parameter G_{min} was used, indicating the minimum number of generations the algorithm will run. The minimum number of generations used by FEP overrides the minimum number of generation requirement used by the termination condition. As no literature could be found suggesting values for G_{min}, the values were found through experimentation, resulting in a set of values of (25, 50, 75, 100, 200, 300, 500) for G_{min}.

Weight decay is a technique specific to ANNs. The inspiration for weight decay comes from the observation that lower weight values for the neural links have been shown to be associated with better generalization in ANNs (Krogh & Hertz, 1992). With weight decay a percentage ($0 \leq \lambda < 1$) of the weight of each neural link is lost each generation. Therefore, a weight value on a neural link stays only high if there is a consistent pressure from many of the samples to keep it elevated. Although the hypothetical range of values for weight decay can rise to just less than 1.0, in practice weight decay is recommended to be low (Krogh & Hertz) and so the possible parameter values were from 0.0 to 0.1 in increments of 0.01, with 0.0 be used for considering it to be off.

Table 8
LSID –ANN overfitting settings.

	Stratification	FEP	G_{min}	λ
A / R	On	Off	–	0.05
S / I	On	Off	–	0.05
V / V	On	Off	–	0.01
S / G	Off	Off	–	0.10

Table 9
LSID –GA overfitting settings.

	Stratification	FEP	G_{min}
A / R	On	On	100
S / I	On	On	25
V / V	On	On	75
S / G	On	On	25

Table 10
LSID –ACS overfitting settings.

	Stratification	FEP	G_{min}
A / R	On	Off	–
S / I	On	Off	–
V / V	On	Off	–
S / G	On	Off	–

Table 11
LSID –PSO overfitting settings.

	Stratification	FEP	G_{min}
A / R	On	Off	–
S / I	On	Off	–
V / V	On	Off	–
S / G	On	Off	–

Stratification and FEP were used for each algorithm; whereas, weight decay was only used with the ANN as it is ANN specific. For LSID-GA, LSID-ACS and LSID-PSO, each combination of stratification and FEP was assessed (i.e., no overfitting reduction technique, only stratification, only FEP, and stratification and FEP). For LSID-ANN, weight decay was investigated first and the optimal weight decay setting was used when investigating stratification and FEP, where again all combinations were investigated. The optimal overfitting reduction settings are shown in Tables 8–11 for LSID-ANN, LSID-GA, LSID-ACS and LSID-PSO, respectively.

5.4. Results and discussion

This subsection presents and discusses the results of each LSID approach, using the optimal parameters and overfitting reduction settings. First, a comparison is made between LSID and related works using the SIM metric. Afterwards, a comparison is made between DeLeS, the top approach according to SIM, and the LSID approaches with the remaining metrics. Then, the practical implications of the results are discussed. This is followed by observations made on overfitting reduction strategies and lastly performance related issues for each algorithm are discussed.

As most related works use the SIM metric, the results of the LSID approaches were first compared to respective related works based on the SIM metric. Accordingly, the LSID approaches were compared to DeLeS (Graf, Kinshuk et al., 2009), a Bayesian network approach (García et al., 2007) and an NBTree approach (Özpolat & Akar, 2009) (shown in Table 12). The other related works are not included in the comparison, because they either have no evaluation (Carmona et al., 2008), use simulated data (Dorça et al., 2013)

Table 12
Comparison of SIM results (ranks in parentheses, top result bolded).

Algorithm	SIM				
	A/R	S/I	V/V	S/G	Average
LSID-ANN	0.802 (2)	0.741(6)	0.727 (5)	**0.825 (1)**	0.774 (4)
LSID-GA	0.781 (5)	**0.781 (1)**	0.755 (4)	0.818 (2)	**0.784 (1)**
LSID-ACS	**0.804 (1)**	0.762 (4)	**0.771 (1)**	0.785 (4)	0.781 (2)
LSID-PSO	0.801 (3)	0.755 (5)	0.756 (3)	0.810 (3)	0.781 (2)
DeLeS (Graf, Kinshuk et al., 2009)	0.793 (4)	0.773 (2)	0.767 (2)	0.733 (5)	0.767 (5)
Bayesian network (García et al., 2007)	0.580 (7)	0.770 (3)	–	0.630 (7)	0.660 (7)
NB tree (Özpolat & Akar, 2009)	0.700 (6)	0.733 (7)	0.533 (6)	0.733 (5)	0.675 (6)

Table 13
Comparison of ACC results (ranks in parentheses, top result bolded).

Algorithm	ACC				
	A/R	S/I	V/V	S/G	Average
LSID-ANN	0.802 (3)	0.790 (4)	**0.840 (1)**	**0.797 (1)**	**0.807 (1)**
LSID-GA	0.795 (5)	0.796 (2)	0.794 (3)	0.774 (2)	0.790 (3)
LSID-ACS	**0.819 (1)**	**0.797 (1)**	0.799 (2)	0.737 (4)	0.788 (4)
LSID-PSO	0.805 (2)	0.794 (3)	0.796 (4)	0.768 (3)	0.791 (2)
DeLeS	0.799 (4)	0.790 (4)	0.788 (5)	0.702 (5)	0.770 (5)

Table 14
Comparison of LACC results (ranks in parentheses, top result bolded).

Algorithm	LACC				
	A/R	S/I	V/V	S/G	Average
LSID-ANN	**0.610 (1)**	0.575 (2)	**0.656 (1)**	**0.613 (1)**	**0.614 (1)**
LSID-GA	0.584 (4)	0.557 (3)	0.541 (2)	0.522 (3)	0.551 (2)
LSID-ACS	0.599 (2)	**0.583 (1)**	0.534 (3)	0.426 (4)	0.536 (4)
LSID-PSO	0.596 (3)	0.551 (4)	0.482 (4)	0.524 (2)	0.538 (3)
DeLeS	0.435 (5)	0.389 (5)	0.226 (5)	0.134 (5)	0.296 (5)

Table 15
Comparison of %Match results (ranks in parentheses, top result bolded).

Algorithm	%Match				
	A/R	S/I	V/V	S/G	Average
LSID-ANN	0.986 (4)	0.961 (3)	0.986 (2)	**0.986 (1)**	**0.980 (1)**
LSID-GA	0.986 (4)	0.946 (5)	0.936 (3)	0.916 (3)	0.946 (4)
LSID-ACS	**1.000 (1)**	**0.971 (1)**	0.909 (4)	0.879 (5)	0.940 (5)
LSID-PSO	0.988 (3)	**0.971 (1)**	0.909 (5)	0.943 (2)	0.953 (3)
DeLeS	0.987 (2)	0.960 (4)	**0.987 (1)**	0.880 (4)	0.954 (2)

or can only classify a subset of students (Cha et al., 2006). Also, the learning style identification approach in Oscar (Latham et al., 2012) was not included in the comparison since that approach uses behaviors from dialogs between students and the natural language conversational agent Oscar, which makes it very specific to Oscar (or other systems that use dialogs and natural language conversational agents). The comparison shows that in each learning style dimension, an LSID approach achieved the highest results, and all of the LSID approaches are better than the related works when averaging over all dimensions.

Since the SIM metrics is not as accurate as ACC and we also aim at investigating the accurate identification of learning styles for each single student, raw results from DeLeS were obtained to calculate ACC, LACC and %Match for DeLeS, as DeLeS has clearly the best results in comparison to the other related works. Accordingly, the LSID approaches were compared to DeLeS with respect to the ACC, LACC and %Match metrics and results are shown in Tables 13–15, respectively. In the following paragraphs, the results from these three metrics are discussed in more detail.

In the A/R dimension, LSID-ACS is the top approach as it achieved the highest values in the ACC and %Match metrics, and for the LACC metric, it is on rank two slightly behind LSID-ANN. For the S/I dimension, LSID-ACS is the top approach again in all metrics (being tied with LSID-PSO in the %Match metric). With respect to the V/V dimension, LSID-ANN is the leading approach, with the top result in the ACC and LACC metrics and ranked second slightly behind DeLeS in the %Match metric. Lastly, for the S/G dimension, LSID-ANN is the best approach, achieving the highest results in every metric. When considering the average results across all learning style dimensions, LSID-ANN is the top approach in every metric.

When looking at the results from a practical point of view, even small improvements in the precision and accuracy of learning style identification can make a significant difference for students. When learning styles are used, for example, to provide students with advice and recommendations during the learning process, either by a teacher or an adaptive learning system, or simply to make students aware of their learning styles and related strengths and weaknesses, a poor identification of a student's learning style can have severe consequences for the student such as providing the student with mismatched materials in an adaptive learning system or with misleading advice from a teacher. When comparing the results of the best LSID approach, LSID-ANN, with DeLeS, it can be seen from Table 13 that LSID-ANN identifies learning styles on average with an accuracy of 80.7%, which is a 3.7 percent point increase compared to DeLeS. While this increase seems fairly small, every increase means a more accurate identification of students' learning styles and accordingly, more accurate information, interventions and advice for students. Furthermore, this increase is an average value across all students and therefore, can lead to a significant difference for individual students. The results for the LACC and %Match metrics illustrate this significant difference in more detail as these metrics focus on the performance of the approaches considering individual students. As mentioned before, LACC measures the worst case scenario for a student in terms of the lowest ACC value for any student. As can be seen in Table 14, the average LACC value for DeLeS is 29.6% while it is 61.4% for LSID-ANN. An accuracy of 29.6% means, for example, that if a student's learning style is very active (i.e., a value of 1 on a scale from 0 to 1, where 1 represents a strong active learning style and 0 represents a strong reflective learning style), the student's learning style would be identified with a value of 0.296, which represents a moderate reflective learning style. Such misidentification would have severe consequences for a student. In contrast, LSID-ANN would identify the learning style of the same student as a value of 0.614, which represents a mild active learning style. While this worst case scenario is still not optimal, the misidentification is far less severe and demonstrates a significant improvement for the student. For the %Match metric, which shows the percentage of students who were identified with reasonable accuracy, DeLeS already achieved rather high results with an average %Match value of 95.4%, as shown in Table 15. However, LSID-ANN achieved an average value of 98.0%, which is a percent point increase of 2.6. In practical terms, this means that when the learning styles of 100 students are identified,

Table 16
Average number of generations before termination.

Algorithm	A/R	S/I	V/V	S/G
LSID-ANN	10,000	10,000	10,000	10,000
LSID-GA	32,698	27,509	10,000	27,295
LSID-ACS	27,745	25,311	11,307	26,907
LSID-PSO	10,000	10,000	10,000	10,000

DeLeS identifies the learning styles of 95 students with reasonable accuracy and LSID-ANN identifies the learning styles of 98 students with reasonable accuracy. Again, even when the difference in results seems rather small, it can have a significant effect for single students.

The next two paragraphs provide further discussion on the use of overfitting reduction techniques and algorithm performance. With respect to overfitting reduction, weight decay was beneficial by improving the LSID-ANN results in every learning style dimension. Stratification also improved the results for every algorithm and learning style dimension except for LSID-ANN in the S/G dimension. As this was the only algorithm and dimension where stratification seems to have not improved the results, an additional experiment was performed for LSID-ANN with weight decay turned off (i.e., λ set to 0.0) and stratification turned on. The result for ACC was 0.795 which compares slightly favourably to the result with no overfitting reduction technique used at all (ACC = 0.792). Thus, stratification is useful in all instances; however, for the S/G dimension weight decay provides sufficient overfitting reduction so as to make stratification unnecessary. FEP only provided an improvement for LSID-GA. The drawback to FEP is that it reduces the training set size by 10%, so it is possible with a larger data set that FEP might be more helpful.

Several observations were made on algorithm performance starting with examining the average number of generations completed by the algorithm before terminating (shown in Table 16). Since the solution space for the V/V dimension is the smallest, the number of generations required is lowest. The minimum number of generations (10,000) is only exceeded in 3 of 10 folds when using LSID-ACS. LSID-ANN and LSID-PSO never exceeded the minimum number of generations for any learning style dimension. For LSID-ANN, no algorithmic issues could be found suggesting that it simply does not require that many generations to train successfully. For LSID-PSO, it seems that particles adopted partially inefficient trajectories based on the relative closeness of the individual and global best positions. When the individual and global best positions were close, the particle would tend to orbit them elliptically. When the individual and global best positions are distant, the particles tend towards a flat trajectory between both points. Thus, LSID-PSO quickly stopped finding good solutions and so terminated at the minimum (Dorigo & Gambardella, 1997b). For LSID-GA, the number of unique values was examined on a gene by gene basis to ensure that diversity was being maintained similar to the technique used by Leung, Gao, and Xu (1997) for binary-encoded GA. Their technique looks at how many genes (g) have the same value throughout the entire population and then define diversity (d) as shown in Formula (10) where G is the total number of genes. Diversity was found to be consistently equal to G, i.e. 100% diverse, until close to the termination point.

$$d = G - g \tag{10}$$

6. Conclusions

This paper presented four automatic approaches (i.e., LSID-ANN, LSID-GA, LSID-ACS and LSID-PSO) for identifying learning styles from the behavior of students in learning management systems, using four computational intelligence (CI) algorithms, namely an artificial neural network, genetic algorithm, ant colony system and particle swarm optimization. An evaluation with data from 75 students was conducted, demonstrating the overall precision of the approaches for each of the four learning style dimensions of the Felder-Silverman Learning Style Model (FSLSM) (1988) as well as the accurate identification of learning styles for each single student.

The results for the proposed approaches were compared to existing approaches using the similarity (SIM) metric, which is commonly used in research on identifying learning styles (García et al., 2007; Graf, Kinshuk et al., 2009; Özpolat & Akar, 2009). Based on SIM, the best solution for each of the four FSLSM dimensions always came from one of the proposed LSID approaches and on average, all LSID approaches outperformed the existing approaches. The results for the LSID approaches were then compared to the existing approach with the best SIM values, DeLeS (Graf, Kinshuk et al. 2009) using the three metrics: accuracy (ACC), lowest accuracy (LACC) and percentage matched with reasonable accuracy (%Match). With respect to ACC, LSID-ACS achieved the best results in the A/R and S/I dimensions while LSID-ANN obtained the best results in the V/V and S/G dimensions. For LACC, LSID-ACS outperformed the other approaches in the S/I dimension and LSID-ANN outperformed the other approaches in the remaining three dimensions. For %Match, LSID-ACS was best in the A/R and S/I dimensions, LSID-ANN was best in the S/G dimension and DeLeS was best (by a value of 0.01 over LSID-ANN) in the V/V dimension. Looking at the results as an average across all four FSLSM dimensions, LSID-ANN achieved the best results in all three metrics.

By identifying students' learning styles with higher accuracy and with less mismatches for single students, adaptive learning systems can use this learning style information to provide more accurate personalization, leading to even higher learning satisfaction (Popescu, 2010), improved performance (Ford & Chen, 2001) and a reduction in the time to learn (Graf, Chung, et al., 2009). Furthermore, students can directly benefit from a more precise identification of learning styles by being able to capitalize on their strengths with respect to learning styles, and by understanding their weaknesses. Additionally, teachers can use this learning style information to provide more accurate advice to their students, which is again more helpful for students the more accurate the learning style identification is.

Future work deals with the practical implementation of the proposed LSID approaches as well as with future research work. In order to make the LSID approaches useful for educators and learners, we already started with the development of a plugin for Moodle to support the identification of learning styles in online courses hosted within Moodle. Furthermore, we plan to expand the development of plugins also to other LMSs. Future research work deals with investigating the applicability of the proposed approaches to identify other student characteristics such as cognitive traits. Furthermore, hybrid algorithms may be investigated which could overcome to weaknesses of mono-CI algorithms.

Acknowledgements

The authors wish to acknowledge the support of Athabasca University, Alberta Innovates – Technology Futures, Alberta Innovation & Advanced Education, NSERC and the Smart Learning Institute of Beijing Normal University. This work was also supported by a grant of the Romanian National Authority for Scientific Research and Innovation, CNCS – UEFISCDI, project number PN-II-RU-TE-2014-4-2604.

References

Abido, M. (2002). Optimal power flow using particle swarm optimization. *International Journal of Electrical Power & Energy Systems, 24*(7), 563–571.

Bajraktarevic, N., Hall, W., & Fullick, P. (2003). Incorporating learning styles in hypermedia environment: Empirical evaluation. In *Proceedings of the workshop on adaptive hypermedia and adaptive web-based systems* (pp. 41–52). June 22 2003.

Basheer, I. A., & Hajmeer, M. (2000). Artificial neural networks: fundamentals, computing, design, and application. *Journal of Microbiological Methods, 43*(1), 3–31.

Carmona, C., Castillo, G., & Millán, E. (2008). Designing a dynamic Bayesian network for modeling students' learning styles. In *Proceedings of 8th international conference on advanced learning technologies* (pp. 346–350). IEEE. July 2008.

Carver, C. A., Jr, Howard, R. A., & Lane, W. D. (1999). Enhancing student learning through hypermedia courseware and incorporation of student learning styles. *IEEE Transactions on Education, 42*(1), 33–38.

Cha, H. J., Kim, Y. S., Park, S. H., Yoon, T. B., Jung, Y. M., & Lee, J.-H. (2006). Learning styles diagnosis based on user interface behaviors for the customization of learning interfaces in an intelligent tutoring system. In *Proceedings of the 8th international conference on intelligent tutoring systems, Berlin, June 26-30 2006: Vol. 4053* (pp. 513–524). Berlin Heidelberg: Springer.

Clerc, M., & Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a Mmultidimensional complex space. *IEEE Transactions on Evolutionary Computation, 6*(1), 58–73.

Coffield, F., Moseley, D., Hall, E., & Ecclestone, K. (2004a). *Learning styles and pedagogy in post-16 learning: A systematic and critical review*. London: Learning & Skills Research Centre.

Coffield, F. J., Moseley, D. V., Hall, E., & Ecclestone, K. (2004b). *Learning styles: What research has to say to practice*. London: Learning & Skills Research Centre.

Dahlstrom, E., Walker, J. D., & Dziuban, C. (2013). *Ecar study of undergraduate students and information technology*. Boulder, CO: Educause Center for Applied Research.

Dorça, F. A., Lima, L. V., Fernandes, M. A., & Lopes, C. R. (2013). Comparing strategies for Mmodeling students learning styles through reinforcement learning in adaptive and intelligent educational systems: An experimental analysis. *Expert Systems with Applications, 40*(6), 2092–2101.

Dorigo, M., & Birattari, M. (2010). Ant colony optimization. In *In* Encyclopedia of machine learning (pp. 36–39). New York, NY: Springer.

Dorigo, M., & Gambardella, L. M. (1997a). Ant colonies for the travelling salesman problem. *BioSystems, 43*(2), 73–81.

Dorigo, M., & Gambardella, L. M. (1997b). Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation, 1*(1), 53–66.

Eberhart, R. C., & Kennedy, J. (1995). A new optimizer using particle swarm theory. In *Proceedings of the sixth international symposium on micro machine and human science* (pp. 39–43). October 4-6 1995.

Ericsson, M., Resende, M. G. C., & Pardalos, P. M. (2002). A genetic algorithm for the weight setting problem in Ospf routing. *Journal of Combinatorial Optimization, 6*(3), 299–333.

Felder, R. M. (1996). Matters of style. *ASEE Prism, 6*(4), 18–23.

Felder, R. M., & Silverman, L. K. (1988). Learning and teaching styles in engineering education. *Engineering Education, 78*(7), 674–681.

Felder, R. M., & Soloman, B. A. (2000). *Learning styles and strategies* Retrieved October 7th 2016 from *https://www.dal.ca/content/dam/dalhousie/pdf/management/Faculty%20%26%20Staff/LEARNING%20STYLES%20AND%20STRATEGIES.pdf*.

Felder, R. M., & Solomon, B. A. (1998). *Index of learning styles* Retrieved October 7th 2016 from *http://www.engr.ncsu.edu/learningstyles/ilsweb.html*.

Felder, R. M., & Spurlin, J. (2005). Applications, reliability and validity of the index of learning styles. *International Journal of Engineering Education, 21*(1), 103–112.

Filippidis, S. K., & Tsoukalas, I. A. (2009). On the use of adaptive instructional images based on the sequential–global dimension of the Felder–Silverman learning style theory. *Interactive Learning Environments, 17*(2), 135–150.

Foody, G. M., McCulloch, M. B., & Yates, W. B. (1995). The effect of training set size and composition on artificial neural network classification. *International Journal of Remote Sensing, 16*(9), 1707–1723.

Ford, N., & Chen, S. (2001). Matching/Mismatching revisited: An empirical study of learning and teaching styles. *British Journal of Educational Technology, 32*(1), 5–22.

García, P., Amandi, A., Schiaffino, S., & Campo, M. (2007). Evaluating Bayesian networks' precision for detecting students' learning styles. *Computers & Education, 49*(3), 794–808.

Graf, S. (2007). *Adaptivity in learning management systems focussing on learning styles*. Vienna University of Technology.

Graf, S., Chung, H. L., Liu, T.-C., & Kinshuk (2009). Investigations about the effects and effectiveness of adaptivity for students with different learning styles. In *Proceedings of 9th international conference on advanced learning technologies* (pp. 415–419). Latvia: IEEE. July 200.

Graf, S., Kinshuk, & Liu, T.-C. (2009). Supporting teachers in identifying students' learning styles in learning management systems: An automatic student modelling approach. *Educational Technology & Society, 12*(4), 3–14.

Grefenstette, J. J. (1986). Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics, 16*(1), 122–128.

Honey, P., & Mumford, A. (1992). *The manual of learning styles* (3rd ed.). Maidenhead.

Hornik, K., Stinchcombe, M., & White, H. (1990). Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Networks, 3*(5), 551–560.

James, W. B., & Gardner, D. L. (1995). Learning styles: Implications for distance learning. *New Directions for Adult and Continuing Education*, (67), 19–31 1995.

Klašnja-Milićević, A., Vesin, B., Ivanović, M., & Budimac, Z. (2011). E-Learning personalization based on hybrid recommendation strategy and learning style identification. *Computers & Education, 56*(3), 885–899.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th international joint conference on artificial intelligence* (pp. 1137–1145). Morgan Kaufmann Publishers Inc. August 1995.

Kolb, D. A. (1971). *Individual learning styles and the learning process*. Cambridge, MA: Sloan School of Management, Massachusetts Institute of Technology.

Kolb, D. A. (1981). Learning styles and disciplinary differences. In Chickering and Associates (Ed.), *The modern american college* (pp. 232–255). San Franciso, CA: Jossey-Bass.

Krogh, A., & Hertz, J. A. (1992). A simple weight decay can improve generalization. *Advances in Neural Information Processing Systems, 4*, 950–957.

Kuljis, J., & Liu, F. (2005). A comparison of learning style theories on the suitability for elearning. In *Proceedings of the IASTED conference on web technologies, applications and services* (pp. 191–197). ACTA Press. July 17-19 2005.

Latham, A., Crockett, K., McLean, D., & Edmonds, B. (2012). A conversational intelligent tutoring system to automatically predict learning styles. *Computers & Education, 59*(1), 95–109.

Leung, Y., Gao, Y., & Xu, Z.-B. (1997). Degree of population diversity-a perspective on premature convergence in genetic algorithms and its Markov chain analysis. *IEEE Transactions on Neural Networks, 8*(5), 1165–1176.

Microsoft. (2016). *Net programming with C++/CLI* Retrieved October 7th, 2016, from *https://msdn.microsoft.com/en-us/library/68td296t.aspx*.

Mitchell, T. (1997). *Machine learning*: Vol. 45. Burr Ridge, IL: McGraw Hill.

Moodle. (2016). *Moodle* Retrieved December 1st, 2016, from *http://www.moodle.org*.

Myers-Briggs, I. (1962). *The Myers-Briggs type indicator: Manual*. Palo Alto, CA: Consulting Psychologists Press.

Özpolat, E., & Akar, G. B. (2009). Automatic detection of learning styles for an e-learning system. *Computers & Education, 53*(2), 355–367.

Pask, G. (1976). Styles and strategies of learning. *British Journal of Educational Psychology, 46*(2), 128–148.

Popescu, E. (2010). Adaptation provisioning with respect to learning styles in a web-based educational system: An experimental study. *Journal of Computer Assisted Learning, 26*(4), 243–257.

Pothiya, S., Ngamroo, I., & Kongprawechnon, W. (2010). Ant colony optimisation for economic dispatch problem with non-smooth cost functions. *International Journal of Electrical Power & Energy Systems, 32*(5), 478–487.

Robinson, J., & Rahmat-Samii, Y. (2004). Particle swarm optimization in electromagnetics. *IEEE Transactions on Antennas and Propagation, 52*(2), 397–407.

Russell, S. J., & Norvig, P. (2010). *Artificial intelligence: A modern approach*: Vol. 3. Upper Saddle River: Prentice Hall.

Shi, Y., & Eberhart, R. C. (1998). Parameter selection in particle swarm optimization. In *Proceedings of the 7th annual conferences on evolutionary programming* (pp. 591–600). Springer. March 25-27.

Srinivas, M., & Patnaik, L. M. (1994). Genetic algorithms: A survey. *Computer, 27*(6), 17–26.

Stutzle, T., Lopez-Ibanez, M., Pellegrini, P., Maur, M., Montes de Oca, M., Birattari, M., et al. (2011). Parameter adaptation in ant colony optimization. In *Autonomous search* (pp. 191–215). Berlin: Springer.

Swingler, K. (1996). *Applying neural networks: A practical guide*. San Franciso, CA: Morgan Kaufmann.

Villaverde, J. E., Godoy, D., & Amandi, A. (2006). Learning styles' recognition in e-learning environments with feed-forward neural networks. *Journal of Computer Assisted Learning, 22*(3), 197–206.

Wanas, N., Auda, G., Kamel, M, S., & Karray, F. (1998). On the optimal number of hidden nodes in a neural network. In *Proceedings of IEEE canadian conference on electrical and computer engineering* (pp. 918–921). IEEE. May 1998.

Yao, X. (1999). Evolving artificial neural networks. *Proceedings of the IEEE, 87*(9), 1423–1447.