Author(s): Lee, Sin Wee; Palmer-Brown, Dominic; Draganova, Chrisina.
Article title: Diagnostic Feedback by Snap-drift Question Response Grouping
Year of publication: 2008
Citation: Lee, S. W., Palmer-Brown, D. & Draganova, C. (2008), 'Diagnostic Feedback by Snap-drift Question Response Grouping', In proceedings of the 9th WSEAS International Conference on Neural Networks (NN'08), pp 208-214.
ISSN: 1790-5109
ISBN: 9789606766565

Diagnostic Feedback by Snap-drift Question Response Grouping

SIN WEE LEE, DOMINIC PALMER-BROWN, CHRISINA DRAGANOVA
Innovative Informatics Research Group, School of Computing and Technology
University of East London
Dockland Campus, University Way, London E16 2RD
UNITED KINGDOM
{s.w.lee, d.palmer-brown, c.draganova}@uel.ac.uk

Abstract: - This work develops a method for incorporation into an on-line system to provide carefully targeted guidance and feedback to students. The student answers on-line multiple choice questions on a selected topic, and their responses are sent to a *Snap-Drift* neural network trained with responses from a past students. Snap-drift is able to categorise the learner's responses as having a significant level of similarity with a subset of the students it has previously categorised. Each category is associated with feedback composed by the lecturer on the basis of the level of understanding and prevalent misconceptions of that category-group of students. In this way the feedback addresses the level of knowledge of the individual and guides them towards a greater understanding of particular concepts. The feedback is concept-based rather than tied to any particular question, and so the learner is encouraged to retake the same test and receives different feedback depending on their evolving state of knowledge.

Key-Words: - Snap-Drift, Diagnostic Feedback, e-learning, personalized learning, diagnostic feedback, online multiple choice questions

1 Introduction

The *snap-drift* learning algorithm first emerged as an attempt to overcome the limitations of ART learning in non-stationary environments where self-organisation needs to take account of periodic or occasional performance feedback. Since then, the *snap-drift* algorithm has proved invaluable for continuous learning in several applications.

The r*einforcement* versions [1], [2] of *snap-drift* are used in the classification of user requests in an active computer network simulation environment whereby the system is able to discover alternative solutions in response to varying performance requirements. The *unsupervised snap-drift* algorithm, without any form of reinforcement, has been used in the analysis and interpretation of data representing interactions between trainee network managers and a simulated network management system [3]. New patterns of the user behaviour were discovered.

Snap-drift in the form of a classifier [4] has been used in attempting to discover and recognize phrases extracted from Lancaster Parsed Corpus (LPC) [5]. Comparisons carried out between *snap-drift* and MLP with back-propagation, show that the former is faster and just as effective. It is also been used in Feature Discovery in Speech. Results show that the snap-drift Neural Network (SDNN) groups the phonetics speech input patterns meaningfully and extracts properties which are common to both non-stammering and stammering speech, as well as distinct features that are common within each of the utterance groups, thus supporting classification.

In most recent development, a supervised version of snap-drift has been used in grouping spatio-temporal variations associated with road traffic conditions. Results show that the SDNN used is bale to group read features such that they correspond to the road class travelled even under changing road traffic conditions.

This paper describes a further exploration of SDNN, in unsupervised form, as an automatic diagnostic tool in a virtual learning environment, which the students will be given feedback based on their level of knowledge has been accomplished.

2 Snap-drift Neural Network (SDNN)

One of the strengths of the SDNN is the ability to adapt rapidly in a non-stationary environment where new patterns (new candidate road attributes in this case) are introduced over time. The learning process utilises a novel algorithm that performs a combination of fast, convergent, minimalist learning (snap) and more cautious learning (drift) to capture both precise sub-features in the data and more general holistic features. Snap and drift learning phases are combined within a learning system that toggles its learning style between the two modes. On presentation of input data patterns at the input layer F1, the distributed SDNN (dSDNN) will learn to group them according to their features using *snap-drift* [2]. The neurons whose weight prototypes result in them receiving the highest activations are adapted. Weights are normalised weights so that in effect only the angle of the weight vector is adapted, meaning that a recognised feature is based on a particular ratio of values, rather than absolute values. The output winning neurons from dSDNN act as input data to the selection SDNN (sSDNN) module for the purpose of feature grouping and this layer is also subject to snap-drift learning.

The learning process is unlike error minimisation and maximum likelihood methods in MLPs and other kinds of networks which perform optimization for classification or equivalents by for example pushing features in the direction that minimizes error, without any requirement for the feature to be statistically significant within the input data. In contrast, SDNN toggles its learning mode to find a rich set of features in the data and uses them to group the data into categories. Each weight vector is bounded by snap and drift: snapping gives the angle of the minimum values (on all dimensions) and drifting gives the average angle of the patterns grouped under the neuron. Snapping essentially provides an anchor vector pointing at the 'bottom left hand corner' of the pattern group for which the neuron wins. This represents a feature common to all the patterns in the group and gives a high probability of rapid (in terms of epochs) convergence (both snap and drift are convergent, but snap is faster). Drifting, which uses Learning Vector Quantization (LVQ), tilts the vector towards the centroid angle of the group and ensures that an average, generalised feature is included in the final vector. The angular range of the pattern-group membership depends on the proximity of neighbouring groups (natural competition), but can also be controlled by adjusting a threshold on the weighted sum of inputs to the neurons. The output winning neurons from dSDNN act as input data to the selection SDNN (sSDNN) module for the purpose of feature grouping and this layer is also subject to snap-drift learning.

3 E- learning Snap-Drift Neural Network (ESDNN)

3.1 The ESDNN Architecture

This version of ESDNN is a simplified unsupervised version of the *snap-drift* algorithm [6], as shown in Figure 1.

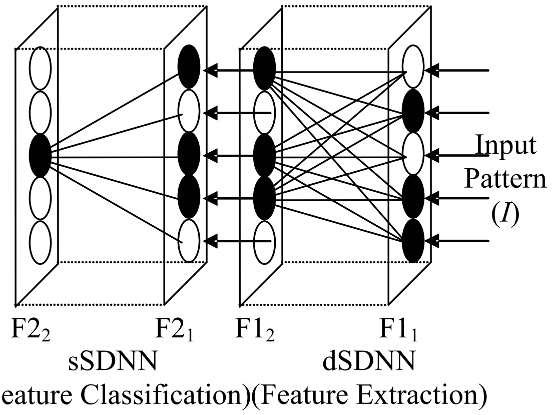

Fig. 1 E-learning SDNN architecture

During training, on presentation of an input pattern at the input layer, the dSDNN will learn to group the input patterns according to their general features. In this case, 5 $F2_1$ nodes, whose weight prototypes best match the current input pattern, with the highest net input are used as the input data to the sSDNN module for feature classification.

In the sSDNN module, a quality assurance threshold is introduced. If the net input of a sSDNN node is above the threshold, the output node is accepted as the winner, otherwise a new uncommitted output node will be selected as the new winner and initialised with the current input pattern.

The following is a summary of the steps that occur in ESDNN:

Step 1: Initialise parameters: ($\alpha = 1, \sigma = 0$)
Step 2: For each epoch (t)
For each input pattern
Step 2.1: Find the D (D = 5) winning nodes at $F2_1$ with the largest net input
Step 2.2: Inhibit the $F2_1$ node for weights re-initialization
Step 2.3: Weights of dSDNN adapted according to the alternative learning procedure: (α,σ) becomes Inverse(α,σ) after every successive epoch
Step 3: Process the output pattern of $F2_1$ as input pattern of $F1_2$
Step 3.1: Find the node at $F1_2$ with the largest net input
Step 3.2: Test the threshold condition:
IF (the net input of the node is greater than the threshold)
THEN
Weights of the sSDNN output node adapted according to the alternative learning procedure: (α,σ) becomes Inverse(α,σ) after every successive epoch
ELSE
An uncommitted sSDNN output node is selected and its weights are adapted according to the alternative learning procedure: (α,σ) becomes Inverse(α,σ) after every successive epoch

3.2 The ESDNN Learning Algorithm

The learning algorithm combines a modified form of Adaptive Resonance Theory (*snap)* (Carpenter, 1987) and Learning Vector Quantisation *(drift)* (Kohonen, 1990). In general terms, the snap-drift algorithm can be stated as:

$$\textit{Snap-drift} = \alpha(\textit{Fast\_Learning\_ART}) + \sigma(\textit{LVQ}) \quad (1)$$

The top-down learning of both of the modules in the neural system is as follows:

$$w_{Ji}^{(new)} = \alpha(I \cap w_{Ji}^{(old)}) + \sigma(w_{Ji}^{(old)} + \beta(I - w_{Ji}^{(old)}))(2)$$

where w_{Ji} = top-down weights vectors; I = binary input vectors, and β = the drift speed constant = 0.1. In successive learning epochs, the learning is toggled between the two modes of learning. When α = 1, fast, minimalist (snap) learning is invoked, causing the top-down weights to reach their new asymptote on each input presentation. (2) is simplified as:

$$w_{Ji}^{(new)} = I \cap w_{Ji}^{(old)} \quad (3)$$

This learns sub-features of patterns. In contrast, when σ = 1, (2) simplifies to:

$$w_{Ji}^{(new)} = w_{Ji}^{(old)} + \beta(I - w_{Ji}^{(old)}) \quad (4)$$

which causes a simple form of clustering at a speed determined by β.

The bottom-up learning of the neural system is a normalised version of the top-down learning:

$$w_{IJ}^{(new)} = w_{Ji}^{(new)}/| w_{Ji}^{(new)}| \quad (5)$$

where $w_{Ji}^{(new)}$ = top-down weights of the network after learning.

In ESDNN, *snap-drift* is toggled between snap and drift on each successive epoch. The effect of this is to capture the strongest clusters (holistic features), sub-features, and combinations of the two.

4 E-learning System

4.1 Motivation

Formative type of assessment provides students with feedback that highlights the areas for further study and indicates the degree of progress [7]. This type of feedback needs to be timely and frequent during the semester in order to really help the students in learning of a particular subject. One effective way to provide students with immediate and frequent feedback is by using Multiple Choice Questions (MCQs) set up as web-based formative assessments and given to students to complete after a lecture/tutorial session. MCQs can be designed with a purpose to provide diagnostic feedback, which identifies misconceptions or adequately understood areas of a given topic and explains the source of misconceptions by comparing with common mistakes. There are many studies (e.g. [8], [9], [10], [11]) investigating the role different types of feedback and MCQs used in web-based assessments, that report on positive results from the use of MCQs in online tests for formative assessments. However none of these studies have employed any intelligent analysis of the students' responses or providing diagnostic feedback in the online tests.

The ESDNN system presented in this paper enhances students learning through:

- Providing diagnostic feedback which is automatic, immediate and individual to large number of students based on intelligent analysis of real data
- Encouraging independent and deeper learning;
- Providing a tool for self-assessment accessible anywhere and anytime

The ESDNN is a simple tool that can be

incorporated in a VLE system or alternatively can be installed on a PC, configured to run as a web server. The student responses are recorded in a database and can be used for monitoring the progress of the students and for identifying misunderstood concepts that can be addressed in following face-to-face sessions. The collected data can be also used to analyse how the feedback influences the learning of individual students and for retraining the neural network. Subsequently the content of the feedback can be improved. Once designed MCQs and feedbacks can be reused for subsequent cohorts of students.

4.2 E-Learning System Architecture

The E-learning system has been designed and built using the JavaServer Faces Technology (JSF), which is a component-based web application framework that enables rapid development. The JSF follows the Model-View-Controller (MVC) design pattern and its architecture defines clear separation of the user interface from the application data and logic [12].

The ESDNN is integrated within the web application as part of the model layer. The ESDNN is trained for each set of questions offline with data available from previous years of students, and the respective weight text files are stored on the application server. The feedback for each set of questions and each possible set of answers is grouped according to the classification from the ESDNN and written in an XML file stored on the application server.

In order to analyse the progress of the students in using the system they have to login into the system with their student id numbers. The set of answers, time and student id are recorded in the database after each student's submission of answers. After login into the system the students are prompt to select a module and a topic and this leads to the screen with a set of multiple choice questions specific for the selected module and topic. On submission of the answers the system converts these into a binary vector which is fed into the ESDNN. The ESDNN produces a group number; the system retrieves the corresponding feedback for this group from the XML feedback file and sends it to the student's browser. The student is prompted to go back and try the same questions again or select a different topic. A high level architectural view of the system is illustrated in Figure 2.

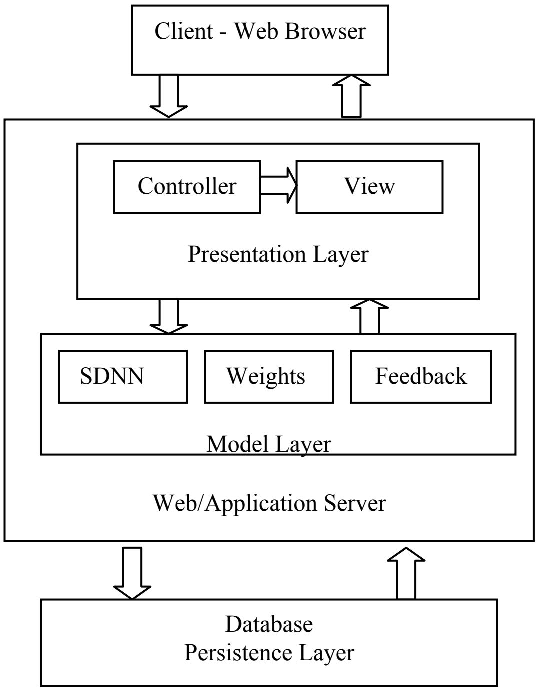

Fig. 2 E-learning system architecture

The features of the system can be surmised as follows:

1. Log in by student ID which allows the data to be collected and analysed.
2. Select a quiz related to a particular topic from a number of options.
3. Quiz page with questions in a multiple choice format
4. Classifications of the student response.
5. Displaying the corresponding feedback
6. Saving in a database the student ID, answers, topic ID and time of completion of the quiz.
7. Help which provides assistance to using the system

4 Trials and Results

4.1 Introduction

During training the ESDNN was first trained with the responses for 5 questions on a particular topic in a module/subject. In this case, the responses are obtained from previous cohort of students on the topic 1 of the module, *Introduction to Computer System*.

After training, appropriate feedback text is written by academics for each of the group of students' responses that address the conceptual errors implicit in combinations of incorrect answers.

During the trial, a current cohort of students is asked to provide responses on the same questions and they will be given the feedback on the combination of incorrect answers.

4.2 Trial Results and Analysis

A trial was conducted with 70 students. They were allowed to make as many attempts at the questions as they liked. On average they gave 7 sets of answers over 20 minutes.

Figure 2 illustrates the behaviour of students in terms of what might be called learning states. These states correspond to the output neurons that are triggered by patterns of question topic responses. In other words, the winning neuron represents a state of learning because it captures some commonality in a set of questions responses. For example, if there are several students who give the same answer (correct or incorrect) to two or more of the questions, snap-drift will form a group associated with one particular output neuron to include all such cases. That is an over simplification, because some of those cases may be pulled in to other 'stronger' groups, but that would also be characterized by a common feature amongst the group of responses.

Figure 2 shows the knowledge state transitions. Each time a student gives a new set of answers, having received some feedback associated with their previous state, which in turn is based on their last answers, they are reclassified into a new (or the same) state, and thereby receive new (or the same) feedback. The tendency is to undergo a state transition immediately or after a second attempt.

A justification for calling the states 'states of knowledge' is to be found in their self-organization into the layers of Figure 2. A student on state 14, for example has to go via one of the states in the next layer such as state 9 before reaching the 'state of perfect knowledge' (state 25) which represents correct answers to all questions. On average, and unsurprisingly, the state-layer projecting onto state 25 (states 20, 1, 9 and 4) are associated with more correct answers than the states in the previous layer. Students often circulate within layers before proceeding to the next layer. The may also return to previous layer, but that is less common. The commonest finishing scores are 3, 4 and 5 out of 5 correct answers; the commonest starting scores are 0,1,2,and 3. The average time spent on the questions was about 17 minutes, and the average increase in score was about 25%.

The feedback texts are composed around the pattern groupings and are aimed at misconceptions that may have caused the incorrect answers common within the pattern group. An example of a typical response to the questions is:

1. A common characteristic of all computer systems is that they

○ lower the operating costs of companies that use them

○ destroy jobs

○ increase the efficiency of the companies that use them

● process inputs in order to produce outputs

○ are used to violate our personal freedoms

2. A digital computer system generates, stores, and processes data in

○ a hexadecimal form

○ a decimal form

○ an octal form

● a binary form

○ none of the above forms

3. All modern, general purpose computer systems, require

○ at least one CPU and memory to hold programs and data

● at least one CPU, memory to hold programs and data and I/O devices

○ at least one CPU, memory to hold programs and data and long-term storage

○ at least one CPU, I/O devices and long term storage

○ at least one CPU, memory

4. Babbage’s 19th Century Analytical Engine is significant in the context of computing because it

○ was the first digital computer

○ was the first device which could be used to perform calculations

● contained all the essential elements of today’s computers

○ could process data in binary form

○ was the first electronic computer

5. According to Von Neumann s stored program concept

○ program instructions can be fetched from a storage device directly into the CPU

○ data can be fetched from a storage device directly into the CPU

● memory locations are addressed by reference to their contents

○ memory can hold programs but not data

○ both program instructions and data are stored in memory while being processed

This is classified into Group (state) 14, which generates the following feedback:

State 14 Feedback

1.John Von Neumann, whose architecture forms the basis of modern computing, identified a number of major shortcomings in the ENIAC design. Chief amongst these was the difficulty of rewiring ENIAC's control panels every time a program or its data needed changing. To overcome this problem, Von Neumann proposed his stored program concept. This concept allows programs and their associated data to be changed easily.

2.Memory acts as a temporary storage location for both program instructions and data. Data, including program instructions, are copied from storage devices to memory and viceversa. This architecture was first proposed by John von Neumann.

3.Much of the flexibility of modern computer systems derives from the fact that memory is addressed by its location number without any regard for the data contained within. This is a crucial element of the Von Neumann architecture.

Prompted by the group 14 feedback the student is able, either immediately or after some reflection, to improve their answer to the question 5 to “both program instructions and data are stored in memory while being processed”. This gives rise to the state 9 feedback below, and after perhaps another couple of attempted answers they correct their answer to question 3, to achieve the correct answers to all questions.

State 9 Feedback

The work of a modern computer system can be described in terms of an input-process-output model (IPO). To implement this model, a computer needs at least one means of both input and output and a means of processing the input. The design of Charles Babbage's Analytical Engine, which preceded the first digital computers by more than 100 years, also included a means of input (punched cards), a means of output (a printer) and a means of processing the input (a device which Babbage called the 'mill'). Babbage was a genuine visionary.

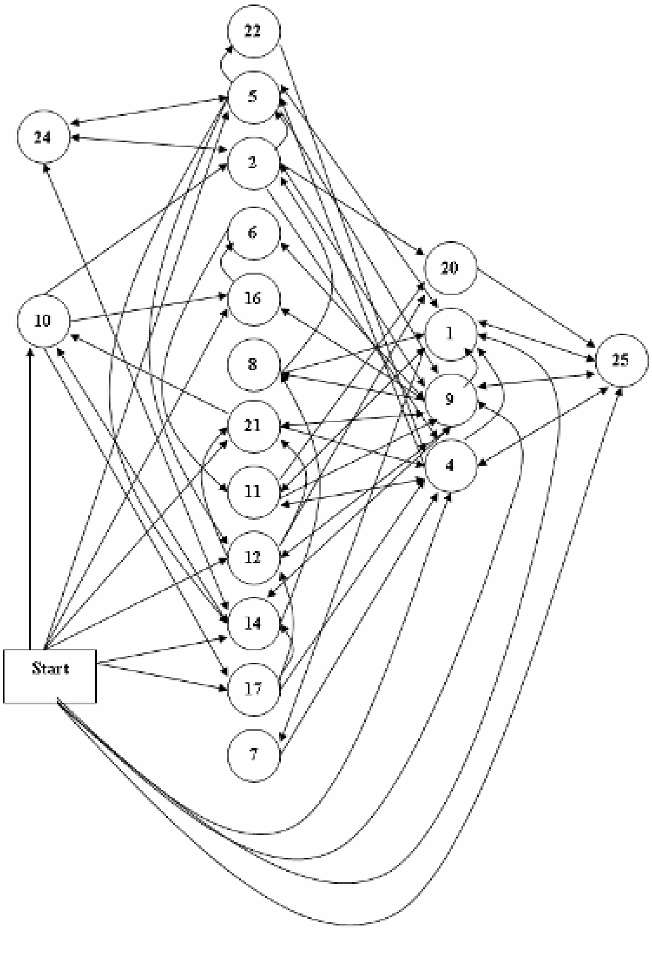

Fig.2 Knowledge state transitions

5 Conclusion

This paper presents a plausible method of using *snap-drift* in a diagnostic feedback tool to provide feedback addressing the level of knowledge of individuals, guiding them towards a greater understanding of particular concepts. Although this is still at the preliminary stages of research into this application, the results from a small cohort of 70 students have provided very beneficial insights into the knowledge state transition of the students. The next stage will be applying ESDNN for larger cohorts of students and using tests from other subject modules.

References:

[1] Lee, S. W., Palmer-Brown, D., Tepper, J., and Roadknight, C. M., *Snap-Drift: Real-time, Performance-guided Learning*, Proceedings of the International Joint Conference on Neural Networks (IJCNN'2003), 2003, Volume 2, pp. 1412 – 1416.

[2] Lee, S. W., Palmer-Brown, D., and Roadknight, C. M., Performance-guided Neural Network for Rapidly Self-Organising Active Network Management, *Neurocomputing*, Volume 61, pp. 5 – 20.

[3] Donelan, H., Pattinson, Palmer-Brown, D., and Lee, S. W., *The Analysis of Network Manager's Behaviour using a Self-Organising Neural Networks*, Proceedings of The 18th European Simulations Multiconference, 2004, pp. 111 – 116.

[4] Lee, S. W., and Palmer-Brown, D., *Phrase Recognition using* Snap-Drift *Learning Algorithm*, *Proceedings of* The International Joint Conference on Neural Networks, 2005, Volume 1, pp. 588-592.

[5] R. Garside, G. Leech and T. Varadi, *Manual of Information to Accompany the Lancaster Parsed Corpus*: Department of English, University of Oslo, 1987.

[6] Lee, S. W., and Palmer-Brown, D., *Phonetic Feature Discovery in Speech using Snap-Drift*, Proceedings of *International Conference on Artificial Neural Networks (ICANN'2006), S.* Kollias et al. (Eds.): ICANN 2006, Part II, LNCS 4132, pp. 952 – 962.

[7] Brown G., J. Bull and M. Pendlebury, *Assessing Students Learning in Higher Education*, Routledge, London, 1997

[8] Higgins E, Tatham L, *Exploring the potential of Multiple Choice Questions in Assessment*, Learning & Teaching in Action, Vol 2, Issue 1, 2003

[9] Payne, A., Brinkman, W.-P. and Wilson, F., *Towards Effective Feedback in e-Learning Packages: The Design of a Package to Support Literature Searching*, *Referencing and Avoiding Plagiarism*, Proceedings of HCI2007 workshop: Design, use and experience of e-learning systems, 2007, pp. 71-75.

[10] Dafoulas, G.A., *The role of feedback in online learning communities,* Fifth IEEE International Conference on Advanced Learning Technologies, 5-8 July 2005, pp. 827 – 831.

[11] Iahad, N.; Dafoulas, G.A.; Kalaitzakis, E.; Macaulay, L.A., *Evaluation of online assessment: the role of feedback in learner-centered e-learning*, Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 1 - Volume 1, 2004, pp. 10006.1

[12] Race, P., *The Open Learning Feedback*, 2^{nd} Ed, London:Kogan Page Ltd, 1994