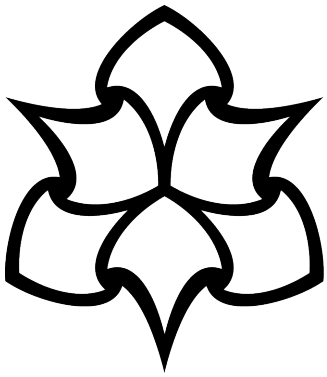


COMPREHENSION BASED ADAPTIVE LEARNING SYSTEMS

Michael Henry Holmes

A thesis submitted in partial fulfilment of the requirements of Manchester Metropolitan University for the degree of Doctor of Philosophy

School of Computing, Mathematics and Digital Technology

Manchester Metropolitan University

2017

"Computer science will transform our cultures [...] It will help us learn what knowledge is. It will teach us how we learn, think and feel. Then we'll be able to change ourselves. This will change all our sciences and humanities."

Minsky (1996)

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Michael Henry Holmes

2017

Acknowledgements

I would like to thank my supervisors, Annabel, Keeley, Jim and Cathy, for their guidance. I would also like to give thanks to my husband, Sami, and to my mum, Mary, for their support and encouragement over the years. I would also like to thank those students of Manchester Metropolitan University's Faculty of Science & Engineering who took part in the research; their contribution has been invaluable.

Abstract

Conversational Intelligent Tutoring Systems aim to mimic the adaptive behaviour of human tutors by delivering tutorial content as part of a dynamic exchange of information conducted using natural language.

Deciding when it is beneficial to intervene in a student's learning process is an important skill for tutoring. Human tutors use prior knowledge about the student, discourse content and learner non-verbal behaviour to choose when intervention will help learners overcome impasse. Experienced human tutors adapt discourse and pedagogy based on recognition of comprehension and non-comprehension indicative learner behaviour.

In this research non-verbal behaviour is explored as a method of computationally analysing reading comprehension so as to equip an intelligent conversational agent with the human-like ability to estimate comprehension from non-verbal behaviour as a decision making trigger for feedback, prompts or hints.

This thesis presents research that combines a conversational intelligent tutoring system (CITS) with near real-time comprehension classification based on modelling of e-learner non-verbal behaviour to estimate learner comprehension during on-screen conversational tutoring and to use comprehension classifications as a trigger for intervening with hints, prompts or feedback for the learner.

To improve the effectiveness of tuition in e-learning, this research aims to design, develop and demonstrate novel computational methods for modelling e-learner comprehension of on-screen information in near real-time and for

adapting CITS tutorial discourse and pedagogy in response to perception of comprehension indicative behaviour. The contribution of this research is to detail the motivation for, design of, and evaluation of a system which has the human-like ability to introduce micro-adaptive feedback into tutorial discourse in response to automatic perception of e-learner reading comprehension.

This research evaluates empirically whether e-learner non-verbal behaviour can be modelled to classify comprehension in near real-time and presents a near real-time comprehension classification system which achieves normalised comprehension classification accuracy of 75%. Understanding e-learner comprehension creates exciting opportunities for advanced personalisation of materials, discourse, challenge and the digital environment itself. The research suggests a benefit is gained from comprehension based adaptation in conversational intelligent tutoring systems, with a controlled trial of a comprehension based adaptive CITS called Hendrix 2.0 showing increases in tutorial assessment scores of up to 17% when comprehension based discourse adaptation is deployed to scaffold the learning experience.

Contents

List of Figures

List of Tables

Nomenclature

Roman Symbols

API Application Programming Interface

CA Conversational agent

CBFV Cumulative Behavioural Feature Vector

CI Construction-integration theory

CITS Conversational Intelligent Tutoring System

CLT Cognitive load theory

CNN Convolutional Neural Networks

CSV Comma Separated Value

DLL Dynamic Link Library

FILO First in, last out

GQM Goal Question Metric

GUI Graphical User Interface

MCQ Multiple Choice Quiz

MMU Manchester Metropolitan University

NVB Non-verbal behaviour

RNN Recurrent Neural Networks

TF-IDF Term frequency - inverse document frequency

Chapter 1

Introduction

This chapter introduces the research project, the research aims and objectives, and the research questions to be answered. The chapter gives an overview of the aims, outcomes and contributions of the research and the structure of this thesis.

1.1 About this research

The research presented in this thesis is motivated by the need to improve the effectiveness (Husbands and Pearce, 2012) of e-learning platforms. Literature review of educational practice (chapter 2), comprehension assessment (chapter 3) and the behaviour of intelligent tutoring systems (chapter 4) suggests that the timing of feedback micro-adaptations (sections 4.8.2 and 4.8.3) in relation to task context and comprehension state (section 3.2) plays an important role in the effectiveness of tuition.

This research presents a review of the best practice of human one-to-one tuition (section 2.6), explores comprehension and assessment to identify methods of computational analysis (3.6) and identifies the opportunity for improvement to timing of micro-adaptive behaviour by integration of near real-time comprehension assessment by analysis of learner non-verbal behaviour (3.4) as a trigger for immediate and contextualised feedback.

The research experimentally evaluates the effectiveness of CITS which embody adaptive behaviours drawn from literature on education and CITS design (chapter 8), experimentally evaluates whether reading comprehension states can be classified by non-intrusive analysis of learner non-verbal behaviour (chapters 10 and 12), experimentally evaluates the accuracy of comprehension classifications made during discourse with a conversational agent (chapter 14) and experimentally evaluates the effect of comprehension based micro-adaptation on learning outcomes for students using a conversational intelligent tutoring system (CITS) (chapter 15).

The aim of this research is to design, develop and evaluate novel computational methods for accurately classifying e-learner reading comprehension of on-screen information by near real-time observation of non-verbal behaviour using consumer grade computer hardware and peripherals, to demonstrate that comprehension classification can be used to time feedback micro-adaptation within a conversational intelligent tutoring system, and that timing feedback by comprehension assessment has an effect on learning outcomes.

1.2 Aims

1. Identify the effective behaviours of human tutors in one-to-one tuition

2. Identify the role of comprehension in learning and how comprehension assessment is used in appraisal of understanding

3. Identify the current state of the art in adaptive behaviours for CITS and assess whether they meet best practice for human tuition

4. Design, develop and evaluate a conversational intelligent tutoring system (CITS) that is capable of tutoring computer programming effectively

5. Design, develop and evaluate an automatic comprehension assessment and classification system for integration with e-learning environments, specifically CITS

6. Design, develop and evaluate a CITS system which improves learning outcomes by making timely pedagogic interventions based on assessment of learner comprehension

1.3 Research questions

To achieve the aims of the research (section 1.2), to improve e-learner educational outcomes by development of comprehension based adaptive CITS, six important research questions arise:

1. What are the effective behaviours of human tutors in one-to-one tuition?

2. What is comprehension and how is it assessed?

3. How do intelligent tutoring systems enact the effective behaviour of human tutors and could improvements be made?

4. Can a CITS tutor computer programming effectively at undergraduate level?

5. Can comprehension of on-screen information be classified accurately based on observation of e-learner non-verbal behaviour?

6. Does adaptation within a CITS, based on detection of non-comprehension states, improve learning outcomes for students?

In this research, questions are addressed using literature review, software design and development and empirical evaluation by experimentation with student participants at Manchester Metropolitan University.

1.4 Contribution

This thesis details a programme of research, software design and development, and empirical evaluation through studies conducted with learners, that presents five contributions in the advancement of intelligent adaptive agent-based and conversational e-learning systems.

Literature review ties together prior work on educational practice, cognitive psychology, adaptive system design and affect-cognitive detection using machine learning to identify an opportunity to improve the micro-adaptive behaviour of CITS.

The literature review that is presented highlights the effective behaviour of tutors in one-to-one tuition (section 2.6), finds definition and explanation of reading comprehension (sections 3.2) and identifies the opportunity for improving the effectiveness of micro-adaptive behaviour in CITS (sections 4.8) by integration of comprehension assessment through computational analysis of learner behaviour (sections 3.4, 4.8.4 and 6.2).

To evaluate whether comprehension based micro-adaptation can improve learning outcomes for students using a CITS. A CITS named Hendrix has been designed and developed to tutor computer programming (Holmes et al., 2015a) (chapters 7 and 8). Hendrix contribution is two-fold: in its design and architecture and in its ability to appraise and use natural language, mathematics and programming code as part of a conversational tutorial.

To meet the aim of having Hendrix adapt to e-learner comprehension of on-screen information in near real-time, a novel comprehension classification system, named COMPASS, has been designed, developed and evaluated (chapters 11, 12 and 14). COMPASS consists of three primary components - the comprehension model, the behaviour extraction algorithm and the comprehension classifier.

The model of e-learner comprehension has been designed to encapsulate non-verbal behaviour over time and learner attributes (Holmes et al., 2017b) (chapters 11 and 12). The contribution of the model is in joining rich patterns

of non-verbal behaviour with stratifying learner attributes including education level, gender and ethnicity. Doing so allows for distinct patterns of behaviour indicative of e-learner comprehension to be modelled within demographic strata.

To extract the comprehension model from web camera image data, a novel algorithm has been designed, developed and evaluated (Holmes et al., 2017a,b) (chapters 11 and 12). The contribution of the algorithm is in pairing near real-time face and facial feature detection using scale invariant detectors (Haar cascades) with a bank of artificial neural networks capable of producing descriptors of discrete non-verbal behaviours evident in sequential web camera images.

To classify the comprehension level expressed by the model, a novel artificial neural network based comprehension classifier has been designed, developed and evaluated (Holmes et al., 2017a,b) (chapters 11 and 12). The contribution of the classifier is in its learning of discriminant features so as to generalise about e-learner comprehension across multiple e-learning platforms.

COMPASS demonstrated a technique for modelling and classifying comprehension in near real-time without use of specialist hardware or body attached sensors, and showed that patterns of non-verbal behaviour can be used to estimate comprehension with accuracy greater than 75%.

Finally, COMPASS has been integrated into a comprehension based adaptive conversational intelligent tutoring system, Hendrix 2.0. Hendrix 2.0 has been designed, developed and evaluated to assess whether comprehension based adaptation improves learning outcomes for students (Holmes et al., 2017a) (chapters 13, 14 and 15). The contribution of Hendrix 2.0 is its ability to detect e-learner comprehension of on-screen information during conversational tutoring and to adapt discourse and interaction to support students displaying non-comprehension indicative behaviours.

Hendrix 2.0 demonstrated that patterns of non-verbal behaviour learned from web camera image data, during a simple on-screen multiple choice quiz

activity, are sufficiently generalised to successfully transfer across multiple on-screen learning environments without significant loss of accuracy.

1.5 Summary of research outcomes

This thesis details a programme of iterative research, software design, development and evaluation which has produced three novel systems - Hendrix 1.0, COMPASS and Hendrix 2.0. In this section a summary of the research outcomes is provided with these three distinct systems in mind.

Hendrix 1.0 is a conversational intelligent tutoring system designed to deliver short discursive tutorials on topics relating to computer programming at undergraduate level. Hendrix 1.0 was trialled in a pilot study (chapter 8) and shown to be a reliable conversational system which users found both useful and enjoyable to interact with. Hendrix 1.0 also showed tangible benefits (section 8.8.2) for users, with learners showing post-tuition learning gain of 22% ($p = 0.2$) for the cohort and 39.11% for the non-computing student group (see section 2.3.1 for learning gain calculation).

COMPASS is an automatic near real-time comprehension classification system which uses computer vision and machine learning to automatically model patterns of learner non-verbal behaviour during mental processing of on-screen information. COMPASS was trained and evaluated using data gathered in a study (chapter 12) of e-learner behaviour during on-screen learning interactions. COMPASS achieved 75.8% classification accuracy for 'comprehension' and 'non-comprehension' states using test set data from the study.

Hendrix 2.0 integrated Hendrix 1.0 and COMPASS to create a comprehension aware conversational intelligent tutoring system. Hendrix 2.0 is able to enact timely interventions in the learning process in response to near real-time estimates of learner comprehension.

Hendrix 2.0, using the COMPASS classifier, achieved 75.44% normalised classification accuracy (table 14.9) for 'comprehension' and 'non-comprehension'

states. The result demonstrates that analysis of e-learner comprehension of on-screen information from non-verbal behaviour during conversation with a virtual tutor performs comparably with reported accuracy of analysis of learner comprehension during human-to-human verbal information recall tasks (Buckingham et al., 2014).

Maintaining accuracy from training and test data sets to real-world data (Chapter 14) demonstrated that the COMPASS classifier had successfully learned generalised patterns of comprehension indicative behaviour.

In a study of the effect of adaptation on learning outcomes, comprehension based adaptation by Hendrix 2.0 (Chapter 15) resulted in a 17% increase ($p = 0.2$) in tutorial question scores for learners using the adaptive system compared to learners using the non-adaptive system (section 15.6.2).

1.6 Structure of thesis

This research combines literature review, intelligent system design and development, and empirical evaluation. The thesis begins in Chapter 2 with a review of educational theory relevant to the practice of tutoring and methods for producing classroom oriented short tutorials, highlighting the importance of scaffolding and fading for developing cognitive skills in learners. Chapter 2 identifies comprehension as a key cognitive state on which scaffold learning is contingent.

Chapter 3 explores literature on both human and computerised analysis of comprehension in learning and examination scenarios. The chapter finds that human tutors effectively identify comprehension states in learners and that some successes have been reported using automatic computerised classification techniques.

Chapter 4 discusses the current state of the art in conversational intelligent tutoring systems, incorporating review of literature on issues of structuring knowledge, enacting pedagogy in intelligent agents, natural language processing,

directing goal-oriented conversation and evaluating conversational systems. The chapter includes an in-depth review and discussion of adaptive behaviour in CITS and highlights the importance of *timing* feedback adaptations. The chapter concludes with an outline, from literature, of the structure and behaviour of the CITS to be developed in this work.

Chapter 5 gives an introduction to machine learning with artificial neural networks. Chapter 6 gives an overview of the computational methods used in extracting NVB from image data, modelling behaviours and classifying behavioural patterns.

Chapters 7 and 8 detail the design, development and evaluation of a novel conversational intelligent tutoring system called Hendrix, designed to teach computer programming. Chapter 9 details the tools developed for image processing and comprehension classifier training. Chapter 10 presents an initial study and evaluation of techniques for computerised analysis of e-learner non-comprehension and comprehension classification during recorded on-screen tutoring with Hendrix.

Chapters 11 and 12 detail the design, development and evaluation of COMPASS, an artificial neural networks based comprehension assessment and scoring system. Chapter 13 details the design and development of a comprehension based adaptive conversational intelligent tutoring system integrating Hendrix with COMPASS. Chapter 14 presents an empirical study of the accuracy of COMPASS comprehension classifications in the context of a CITS and Chapter 15 presents an empirical study of the effectiveness of tuition using the comprehension adaptive CITS.

Finally, Chapter 16 gives a summary of conclusions resulting from the research and an indication of developmental possibilities in this field.

Chapter 2

Theories for e-learning

2.1 Introduction

This chapter presents a review of literature on cognitive apprenticeship (section 2.2), evaluation of learning (section 2.3) and methods for producing instructional materials and systems (section 2.4).

Cognitive apprenticeship (section 2.2) is a style of teaching, a pedagogy that focuses on incremental development of cognitive skill in applied problem solving through exploration, challenge, discussion and collaboration between tutor and student.

Instructional design is a production method for creating educational tools and materials which is used widely in e-learning. Instructional design places a focus on identifying the purpose, objectives and function of design choices to ensure that the product, the educational tools and the materials can be evaluated effectively and improved.

2.2 Theories of learning and teaching

Ertmer and Newby (2013) emphasise the importance of theory in designing effective instruction, stating that 'theories are a source of verified instructional strategies, tactics and techniques'. Literature (Ertmer and Newby, 2013; Hay-

lock and Thangata, 2007; Jonassen et al., 1995) highlights three competing educational philosophies – Behaviourism, Cognitivism and Constructivism.

Behaviourism is primarily concerned with the performance of actions in response to environmental factors. The goal of behavioural learning theories is to elicit the desired response from a learner, given a certain stimulus. Ertmer and Newby (2013) define behaviourism by its focus on the consequence of performance. They suggest that behaviourism 'equates to learning with changes in either the form or frequency of observable performance' and assert that from a behaviourist view point, the learning process happens by repeated stimulus, performance and feedback cycles. However, in behavioural theory *feedback* is ill-defined, its purpose is only to create a reinforcement between the expressed behaviour and either a positive or negative outcome.

Cognitivism comes from the models of cognitive science and places emphasis on the importance of understanding how learning occurs, rather than focusing on the behaviour expressed after learning. Cognitivist learning theories stress the supervision of the learning process and the shaping of internal mental models built through learning. As with behavioural theories, feedback is important. However, unlike behavioural theories, feedback is viewed as a mechanism for *guiding* the learner to a correct mental model.

Constructivist principles describe a learning environment in which learners build 'meaning, understanding and relevant practice' (Jonassen et al., 1995) through authentic experiences and reflection. Haylock and Thangata (2007) describe constructivism at an 'active process' through which learners 'construct new ideas or concepts based upon their current and prior knowledge'. They comment that, from a constructivist point of view, knowledge is not out there to be acquired but is constructed by the individual.

In this research a cognitivist approach has been adopted. Constructivism, with its focus on guided learning and collaborative development of ideas from

building-blocks of knowledge, appears most applicable in a conversational scenario where interactions between learner and tutor are incremental in nature.

2.2.1 Cognitivism

Literature (Ertmer and Newby, 2013) describes cognitivist learning theories as focusing on the conceptualisation of learning processes. From a cognitivist viewpoint learning occurs through reception, organisation, storage and retrieval of information. Cognitivist learning theories are concerned not with what learners do, but with how they acquire the knowledge. A key strategy of cognitivist theories is simplification and standardisation (Ertmer and Newby, 2013). By representing information in simple chunks, or building blocks, the learners themselves are responsible for placing them in the correct order to create meaning in their own terms. Cognitivist learning theories use devices such as explanations, demonstrations and examples to guide learning. Cognitivist theories focus on the processes leading up to a learner response, such as mental planning, goal-setting and organisational strategies. Ertmer and Newby (2013) state that instructional situations should support learners to 'code, transform, rehearse, store and retrieve' information. Feedback is important for shaping mental models and should therefore be explicitly corrective.

Memory plays a critical role in cognitivist theories. Cognitivist instructional designers aim to help learners create effective mental models and associations. They use devices such as analogies and hierarchies to represent the relations between new and prior knowledge (Ertmer and Newby, 2013). It is through associations and mental models that cognitivists believe transference can occur. By understanding rules, concepts and boundaries a learner can identify similarities and differences between information and apply existing knowledge to novel information.

Cognitivist learning theories are associated with higher order skills such as problem-solving, reasoning and information processing. Cognitivist theories

work well with the ASSURE model of instructional design, placing an emphasis on learner participation. Ertmer and Newby (2013) highlights planning and revision as examples of learner participation. Instructional designers should develop experiences and materials structured around hierarchies of knowledge to demonstrate relationships in information, emphasising structure, organisation and sequencing of concepts so as to guide the learner in making connections.

Apprenticeship and scaffolded learning

Cognitive apprenticeship (Collins et al., 1988) is a pedagogy for collaborative learning, intended to help bridge the gap between expert and novice knowledge and practice. It is suggested (Collins et al., 1988) that collaborative learning is facilitated by reciprocal teaching, during which both tutor (expert) and learner (novice) engage in an active exchange of question formation and solution discovery.

To illustrate a viable cognitive apprenticeship scenario, Collins (Collins et al., 1988) discusses an activity in which both parties - expert and novice - appraise the same information. The expert leads with an opening question, requiring the novice to summarise, clarify or predict an aspect of the information. The novice can then explore the topic further by asking the expert a question. In the scenario, it is the expert's responsibility to set direction and to provide support. The tutor uses corrective feedback, hints and suggestions to encourage the novice to formulate appropriate answers and questions. Support is faded in and out of the exchange so that the novice receives only as much support as absolutely necessary to complete a task.

Cognitive apprenticeship has been shown to be effective both in tutoring mathematical subjects (Schoenfeld, 1992) and in computer supported e-learning environments (Saadati et al., 2015). The strong focus on interaction, discussion and information exploration lends itself to conversational e-learning, such as OSCAR (Latham et al., 2014).

Collins (Collins et al., 1988) describes a number of activities that support cognitive apprenticeship including reciprocal teaching in which learners are required to ask the tutor questions, summarising texts or other information and clarifying concepts through definition. Collins (Collins et al., 1988) also highlights six modes of teaching for delivering cognitive apprenticeship including expert demonstration (modelling), corrective feedback (coaching), prompting for knowledge (articulation), suggestions and hints (scaffolding) and fading support to allow the learner to take ownership of the learning process (exploration).

Scaffolding and fading are important concepts for cognitive apprenticeship. Wood and Wood (1996) describe 'scaffolding' as the 'support that an adult provides in helping children to learn'. They draw a parallel between the notion of scaffold learning and Zone of Proximal Development (ZPD) (lon, 2016). The ZPD describes the gap between what a learner can achieve alone and what they could achieve, given guidance. Scaffolding attempts to bridge the ZPD by guiding a learner, using both cognitivist and constructivist methods, through problem-solving. Wood and Wood (1996) highlight five effective features of scaffolding:

1. The tutor serves to bridge the gap between a learner's existing knowledge and skills, and the demands of the new task

2. The tutor provides structure to problem-solving by providing help and instruction in context

3. Guided participation allows the learner to participate actively in the learning process

4. Effective guidance transfers responsibility from the tutor to the learner

5. Not all guided participation is a deliberate attempt to teach

Wood and Wood (1996) describe a process of 'contingent instruction', whereby a tutor hands over responsibility for problem-solving to the learner. They suggest that in the first instance of difficulty, the tutor will immediately offer

more specific instruction to guide the learner. However, they suggest that as the learner achieves greater mastery of the subject the tutor should reduce the specificity of the instruction – for example, replace explanation with metaphor. Wood and Wood (1996) refer to this process as 'fading'.

Domain contingency is an important part of scaffolding learning. The tutor should offer suggestions of what to try next but should not prevent the learner from exploring – even if the motivation is seen as incorrect. The tutor should support the student in discovering the mistake for themself. This is a highly constructivist approach to learning and one which Wood and Wood (1996) recognise as difficult to achieve in a computer-based system. However, it is in this respect that there is clear advantage in using conversational agents to deliver instruction. While a conversation script cannot account for full freedom, sufficient conversational routes can be created to allow a degree of localised freedom to ask questions and explore ideas.

Wood and Wood (1996) highlight the following responsibilities for a contingency oriented computer-based tutoring system:

1. Provide instruction in the problem-solving context
2. Give immediate response to learner errors
3. Support successive approximations to competent performance
4. Provide reminders of the learning objective

Wood and Wood (1996) describe 'temporal contingency' as the timing of contingent intervention in relation to indicators of a learner's cognitive-affective state. They suggest that a human tutor responds to body language, facial expressions and other non-verbal behaviour to decide when to intervene. They suggest that a human tutor acting contingently, having observed non-verbal behaviour that indicated learner difficulty, would intervene to scaffold the learning experience. While they suggest that this would not be possible in a computer-based tutoring system, recent advances in computerised analysis of

non-verbal behaviour (see Chapter 3) suggests that artificial intelligence could be used to perceive learner comprehension in near real-time during learning activities.

2.3 Evaluating learning

This section presents literature and methods for evaluating learning outcomes. Evaluation of learning outcomes is an important consideration in this research as a measure of success for a comprehension based adaptive learning system. In this section two approaches to measuring learning outcomes are highlighted: learning gain metrics (section 2.3.1) and a taxonomy of educational objectives (section 2.3.2). Both play an important role in defining objective measures of learner performance and goal attainment.

2.3.1 Learning gain

Learning gain is a simple but widely adopted measurement of the effectiveness of tuition, for example in Colt et al. (2011); Graesser et al. (2003); Latham et al. (2014); Saadati et al. (2015); VanLehn (2011). Learning gain attempts to quantify the amount of *value-add* by measuring the distance between pre- and post-tuition test scores.

There are multiple mathematical definitions of learning gain to be found in literature. Latham (2011) uses equation 2.1. This interpretation of learning gain provides an absolute measure of difference between pre- and post-tuition scores. The approach does not account for prior knowledge and the learning gain is liable to be skewed upwards by under-performance in the pre-tutorial test.

Colt et al. (2011) uses a relative measure (equation 2.2) which gives a percentage gain of those marks available in addition to the pre-tutorial test. For example, if a learner scores 80% on the pre-tutorial test then there are 20%

of marks to be gained in the post-tutorial test. If the learner increases their score to 90% in the post-tutorial test then they have gained 10% and have a learning gain of 50%.

$$post\text{-}tutorial\ test\ score\ -\ pre\text{-}tutorial\ test\ score \tag{2.1}$$

$$\frac{post\text{-}tutorial\ test\ score\ -\ pre\text{-}tutorial\ test\ score}{100.0\ -\ pre\text{-}tutorial\ test\ score} \tag{2.2}$$

In this research equation 2.2 taken from Colt et al. (2011) will be used to represent learning gain. The result of equation 2.2 can be converted to a percentage by multiplying by 100.

Learning gain has proved to be a contentious topic (Cronbach and Furby, 1970; Hake, 2010). The arguments against such a simplistic appraisal technique, discussed in (Hake, 2010), and originally presented in (Cronbach and Furby, 1970), are aimed at the reliability of the statistics produced.

Boyer et al. (2008) investigated the role of cognitive and emotional scaffolding on learning gain. A surprising finding from their work is that *praise* during conversational tutoring has a negative effect on learning gain, a result of increasing confidence and misplaced self-efficacy. The result highlights the unanticipated complexity of variable interactions when motivation, emotion and social interaction are at play during learning.

With this in mind, learning gain alone is an insufficient measure of success. In this research learning gain will be used as part of an ensemble of measurements along with objective attainment, participant feedback and analysis of system function error rates.

2.3.2 Anderson and Krathwohl's taxonomy of educational objectives

Definition of learning objectives is an important aspect of learner outcome evaluation. Defining objectives is no small task and, if done incorrectly, can undermine the learning process and make accurate evaluation impossible. To aid designers in creating effective objectives Anderson et al. (2000) propose a taxonomy based on Bloom's original Taxonomy of Educational Objectives. Their model has two dimensions:

- **Knowledge dimension**
  - **Factual**: Basic elements of knowledge required to solve a problem
  - **Conceptual**: The relationships between basic elements in a broader context
  - **Procedural**: Methods, algorithms and techniques for applying knowledge
  - **Meta-cognitive**: Understanding of cognition
- **Cognitive dimension**
  - **Remember**: Recalling previously learned information from memory
  - **Understand**: Constructing meaning from various data sources
  - **Apply**: Using knowledge to implement a procedure
  - **Analyse**: Differentiating, attributing and organising to distinguish between components information
  - **Evaluate**: Make judgements based on evaluation of evidence
  - **Create**: Manufacturing, reorganising or synthesising information to produce a novel output

The model is principled on the view that an objective is a combinatorial statement of intent along these two dimensions. Anderson et al. (2000) suggest

that an objective should contain a verb, representing the Cognitive Process Dimension and a subject word representing the Knowledge Dimension.

For example, 'As a learner I should be able to *create* a for loop constructor to iterate ten times'. This learning objective exposes factual, conceptual and procedural knowledge of loop constructors and iteration and allows the learner to demonstrate creativity as cognitive process.

Learning objectives can provide assessment milestones which are clearly defined and evaluable. Unlike learning gain, goal attainment does not need to map complex socio-cultural, linguistic or emotional interactions. The measurement of success in any given learning outcome is the student's demonstrated competence within the bounds of the objective. Learning objective attainment can be measured within a tutorial by assessing competence on individual tasks, for example by marking answers to questions or appraising solutions to problems.

2.3.3 Question response complexity

In designing questions for a tutorial, it is important to consider the required complexity of response. For example, a simple question requiring only a single word answer, such as 'true' or 'false', may allow a learner to guess the answer easily. Complex questions, which cannot be guessed easily may require learners to formulate complex multi-part answers containing multiple keywords, phrases, mathematics or programming code, which can contain multiple conceptual elements.

2.4 Instructional design

Instructional design is the process through which an educator determines the best teaching methods to help specific learners achieve a specific goal IEEE (2015).

Gustafson and Branch (1997) define three orientations of models – classroom, product and system. The taxonomy describes the scope and requirements of a model in relation to instructional objectives, ranging from classroom models, which they describe as concerning 'one or a few hours' of instruction, to system models which are concerned with 'course or entire curricula'.

Product oriented models, such as CADMOS-D and SCORM, are also associated with the design of e-learning and are commonly focused on the production of materials for distribution using computer systems (Botturi, 2003; Lai and Liou, 2007). Many product oriented models focus on the definition of learning objects and make use of standardised markup languages such as XML for modelling learning materials and interaction as objects.

While product oriented models appear relevant to Conversational Intelligent Tutoring System tutorial design, due to use of online computer technologies, the nature and intent of CITS is to mimic natural language interactions with a human tutor. As this research is oriented towards development of adaptive technology for the delivery of short tutorials, classroom models will be used to develop and evaluate learning materials.

Classroom models are typically concerned with only a few hours of instruction often provided by a single instructor with limited resources. Simple design processes, such as ASSURE (Heinich et al., 1995), prescribe an iterative approach to materials development and evaluation.

The ASSURE model has six phases – Analyse Learners, State Objectives, Select Methods, Media and Materials, Utilise Media and Materials, Require Learner Participation, and Evaluate and Revise. ASSURE places a clear focus on identifying and defining learning objectives. Learning objectives are the evaluable outcomes by which iterative improvement is made. To design, develop and evaluate tuition for this research, ASSURE will be used as a process.

2.5 Discussion

The research suggests that the most appropriate model of instructional design for the development of tutorials for a conversational intelligent tutoring system will be a classroom oriented model. The ASSURE model is particularly well suited to an interactive online system as it explicitly requires the participation of students – it is therefore geared towards this type of online tool. The ASSURE model provides the greatest specificity in terms of guiding a novice designer in understanding learners, creating achievable outcomes and evaluating the success of the design:

1. Analyse learners (accounting for demographics such as age, gender, ethnicity and level of prior experience)

2. State objectives as behavioural outcomes using Bloom's taxonomy for objective measurement of attainment

3. Select media, methods and materials by analysis of the learning context and technological environment

4. Utilise media, methods and materials in development of a system

5. Require student participation in a pilot study

6. Evaluate the performance based on goal attainment, appropriateness of materials and learner satisfaction

This process will be used to structure the design, development and evaluation of resources and computer systems in this research project. Anderson and Krathwohl's taxonomy can provide a framework for the development of objectives, and as such are used to design evaluable learning outcomes for tutorial content.

Although behavioural learning theories would certainly be the simplest form of learning to implement in an online tutoring system, there are questions about the validity of the techniques used in promoting understanding. These

theories would be applicable in an online instructional system but do not provide the kind of learning experience required to achieve the higher order skills at undergraduate level.

Constructivist learning theories promote the higher order skills an undergraduate tutor would need to promote, but such unstructured and free-roaming learning would be extremely difficult to bake-in to a scripted conversational agent. I also have concerns that the lack of clear behavioural objective outcomes in constructivist theories would make a resultant tutorial difficult to evaluate using statistical measures. Constructivist learning theories appear to be more appropriate for research, collaborative or flipped learning environments.

Cognitivist learning theories span the greatest range of Bloom's cognitive dimension without giving total freedom to the student. The cognitivist viewpoint allows for higher order learning by guiding the learner through demonstration and example. Associations and hierarchies allow students to develop a cognitive map of the knowledge domain in a structured way. This type of learning is well suited to a pre-determined scripted conversation in which a student can ask limited questions and be shown limited support information, as it mirrors strongly the knowledge domain programmable into a conversational script. Cognitivism is the most appropriate learning theory for the development of conversational tutorials as it places emphasis on discursive devices including:

- Explanation, demonstration and example
- Analogy and metaphor
- Hierarchy and association representation
- Challenge and corrective feedback

Cognitive apprenticeship sits between constructivist and cognitive theories. The approach sees the tutor as a guide, and places responsibility for learning on the student. In this approach the learner drives the process but the tutor makes

helpful suggestions of what to try next and provides contextual information to help the learner decide which strategies to attempt.

These aspects of cognitivism, as part of a cognitive apprenticeship strategy, provide a definition of the requisite knowledge entities and behaviours for a conversational tutoring system. These will be used in design and implementation of the software solution in this research project.

Feedback is vital to apprenticeship but may be immediate or delayed – allowing the learner to try and fail, reflect and try again. This type of learning is often facilitated through scaffolding learning. Scaffolding provides an optimal combination of pedagogic devices for implementation in a dynamic e-learning system. The approach lends itself to conversational interaction, such as in a CITS, by promoting both aspects of guided learning and discovery. The limitations identified in literature, relating to timely and appropriate interventions in learning, are the motivation for this research. Contingent scaffolding for computer-based tutoring would be fully realised should comprehension classification based on learner non-verbal behaviour be integrated into a conversational intelligent tutoring system.

2.6 Conclusion

This chapter has presented an overview of literature relating to the methods, design strategies and evaluation of classroom based tuition.

The conclusions made here inform design choices in developing the comprehension based adaptive conversational *intelligent* tutoring system, which aims to mimic the behaviour of a human tutor. The best practice strategies identified in this chapter provide an initial blue print of system behaviour.

The literature reviewed in this chapter suggests a guided process of learning is appropriate, where the *tutor* leads the discussion, challenges the learner to demonstrate knowledge, provides feedback on both content and methods to support advancement and gives correction when impasse is reached. The

literature suggests that concepts should be decomposed, with content and tasks placed in a meaningful order so as to provide structure to the learning, and that the tutor should engage the learner in activities using explanation, discussion, summary and demonstration.

The tutoring behaviours identified are conversational, adaptive and highly contextualised. In Chapter 4, the behaviour of conversational intelligent tutoring systems are examined in detail to assess whether current technology is capable of performing human-like tutoring behaviour.

Literature has shown that adaptation of feedback is an important feature of human tutoring. Human tutors need to adapt to the needs of the learner, responding in context with summary, explanation or demonstration, as and when the learner is failing to comprehend information key to understanding of a concept.

Chapter 3 reviews and discusses literature on *comprehension* and describes how *comprehension* is defined and appraised in the context of computer mediated conversation.

This chapter has presented a review of literature on techniques for evaluating learning in a classroom setting. The conclusions made on evaluation of learning inform the design of experimental methods. When testing the new comprehension based adaptive conversational *intelligent* tutoring system, in addition to system functionality and user feedback, a measure of effect is sought to indicate whether the system itself, and its novel *intelligent* behaviours, have meaningfully improved learning.

The conclusion drawn is that both objective attainment and learning gain are useful in evaluation. In evaluation objective attainment can give a granular, per topic, view on learning. Learning gain can provide a statistical measure of the *effect* learning on demonstrable knowledge.

Chapter 3

Comprehension assessment

3.1 Introduction

Comprehension is a key cognitive state for learning and one which educationalists strive to assess accurately (Alibali et al., 1997; Cain and Oakhill, 2006; Machida, 1986; Webb et al., 1997). In this research, comprehension and non-comprehension states are sought to inform dynamic pedagogic adaptation in a conversational intelligent tutoring system.

This chapter presents an overview of literature on reading comprehension (section 3.2), reading comprehension assessment in education (section 3.3) and computational methods for automatic reading comprehension detection by analysis of non-verbal behaviour (section 3.4). Discussion and synthesis of literature is presented in section 3.5.

3.2 What is comprehension?

Snow (2002) comments on the difficulty of laying out a single definition of what reading comprehension actually entails. They state (Snow, 2002, p. 9), "A formal definition of reading comprehension may seem unnecessary because the term is used so widely and its meaning is assumed to be generally understood. Teachers think of reading comprehension as what students are taught to do in reading

instruction during the early school years and as the reading capacities they are expected to display throughout the middle and high school years. Taxpayers and employers think of reading comprehension as one of the capabilities that high school graduates should have acquired during their years in school. University faculty view high levels of reading comprehension as a prerequisite to a student's success. Yet, coming to a formal definition that is widely accepted turns out to be rather difficult. We believe that it is necessary, as a prerequisite to mapping the domains of knowledge relevant to formulating a research agenda in this area, to define comprehension in a way that clearly specifies its key elements."

What Snow (2002) highlight is that comprehension is context and task specific. Within reading comprehension, as highlighted in guidance to educationalists (Snowling et al., 2009) comprehension entails decoding of information, information reproduction and recall tasks. Anderson (1972) offers a tentative description, referring to comprehension as the 'processes entailed when ideas get off a printed page into a person's head'.

Snow (2002) define reading comprehension as "the process of simultaneously extracting and constructing meaning through interaction and involvement with written language.". They suggest that comprehension of written information entails three elements:

1. The reader who is doing the comprehending
2. The text that is to be comprehended
3. The activity in which comprehension is a part

(Snow, 2002) consider the learner and the task as important to comprehension, in addition to the text.

Woolley (2011) define reading comprehension by stating that "Reading comprehension is the process of making meaning from text.". They go on to comment that "the goal, therefore, is to gain an overall understanding of what is described in the text rather than to obtain meaning from isolated words or sentences.".

Woolley (2011) suggest that there are two types of reading comprehension, a "text-based model which is a mental representation of the propositions of the text" and "a situation model consisting of what the text is perceived to be about.". The text-based model is based on knowledge of language, the meaning of words and the structure of sentences; this is the **decoding** activity discussed in Snowling et al. (2009). The situational model requires the learner to place the information in a context by integrating the *decoded* information with prior knowledge. For the situational model (Woolley, 2011) refer to the work of linguist Walter Kitsch (Kintsch, 1998).

Constructions-integration (CI) theory (Kintsch, 1998) emerged from the field of linguistics as a conceptual model of the process for semantic understanding. CI describes comprehension as a mechanistic process of cognition through which symbols, such as words, are recognised and contextualised. According to CI, when an entity - a word, symbol or concept - is encountered then a network of related entities is automatically 'constructed' from memory. The network represents all the known attributes and relationships for the entity. 'Integration' of context then triggers the removal of irrelevant or incorrect relations. Successful integration results in a coherent model of an entity, a word, symbol or concept, based on comprehension of its relations to other entities. The process of 'construction' and 'integration' is described by Kintsch (1998) as being automatic or subconscious and requiring minimal cognitive effort. However, failure of the automatic processes necessitates active problem-solving through which new relationships and associations are reasoned and learned. Kintsch (1998) suggests that active problem-solving is a cognitively expensive manual, conscious process.

Construction-integration theory provides a particularly useful procedural definition of comprehension, moving beyond indicators of comprehension and providing an explanation of the mechanics of information decoding, contextual-

isation and understanding by reasoning of relations within encoded information - written or spoken language.

Importantly, CI (Kintsch, 1998) integrates well with cognitive load theory (CLT) (Sweller, 1999). CLT attempts to explain the effect of non-comprehension (*intrinsic cognitive load*), increased task complexity (*germane cognitive load*) and confusion (*extraneous cognitive load*) on the amount of mental effort required to understand information. CI would suggest that the cognitive load, mental *effort*, resulting from germane, extraneous and intrinsic sources are caused by a failure when integrating novel information into existing mental models.

3.3 Assessing learner comprehension

Formal assessments of comprehension most commonly depend on post-hoc testing of reading and listening competency (Cain and Oakhill, 2006; Oakhill and Yuill, 1986; Swan and Walter, 2017). The Neale Assessment of Reading Ability (NARA) is an example of a standardised reading comprehension assessment based on post-hoc testing. The NARA examines a learner's ability to comprehend written content by presenting a text and then questioning the learner about the text. The test provides a single outcome for a given question, indicating the level of comprehension for a given text. However, as observed in review of such assessment techniques (Cain and Oakhill, 2006; Swan and Walter, 2017), post-hoc testing cannot provide specific detail on the cause or context of non-comprehension. Review highlights that test question performance can be affected by compounding factors such as memory (Cain and Oakhill, 2006), speech reproduction ability (Cain and Oakhill, 2006), Verbal IQ (Oakhill and Yuill, 1986), or decoding problems such as unfamiliar presentation or complex grammar (Swan and Walter, 2017).

Literature shows (Oakhill and Yuill, 1986) that attempts have been made to differentiate between the confounding factors of post-hoc test performance

by using combinations of closed or *forced choice* questions which may mitigate for some confounding factors such as memory, vocabulary, unfamiliar grammar or confusing presentation. However, the results presented in Oakhill and Yuill (1986) are inconclusive as to the success of the techniques employed.

Cain and Oakhill (2006) suggested that real-time monitoring by computational methods may help to improve comprehension assessment, identifying indicators of comprehension or non-comprehension as learners absorb and process written information. They concluded however, that the technical implementations available, dependent on eye-tracking alone, cannot provide enough information about the cognitive state to classify comprehension. However, literature presented in section 3.4 suggests their view is no longer correct.

Real-time comprehension assessment is performed by experienced human tutors during classroom interactions. Literature (Alibali et al., 1997; Machida, 1986; Webb et al., 1997) shows that classroom tutors use informal assessment of non-verbal behaviour (NVB) to infer learner comprehension during tuition, using experiential knowledge of learner NVB to recognise patterns of non-verbal behaviour indicative of comprehension. Non-verbal behaviour can be described as any behaviour which does not use verbalisation. This broad category of behaviour includes movement, posture, gestures, facial expressions and audible, chemical and physiological expressions.

Literature (Barry et al., 2011) reviewing classroom non-verbal behaviour highlights facial expressions, posture, gestures and eye movements as NVB channels commonly incorporated into models. Literature Barry et al. (2011); Zoric et al. (2007) supposes that NVB is complex, occurring not in isolation but in simultaneous clusters. For example, a gaze shift, a mouth movement and a blush to the skin. Zoric et al. Zoric et al. (2007) also make note of the important difference between conscious and subconscious non-verbal behaviours - in other words deliberate and involuntary behaviours. They suggest that it is the involuntary behaviours, the momentary, fleeting, subconscious and

uncontrolled gaze shifts, micro-expressions and skin-tone changes, rather than planned and controlled wink, smile or nod, that give indication of the true mental state of the learner.

Despite the apparent complexity of the task, literature (Alibali et al., 1997; Machida, 1986; Webb et al., 1997) shows that experienced tutors are highly skilled in recognising learner non-comprehension by assessing NVB.

Research into affective state dynamics (D'Mello and Graesser, 2012) maps affective states to cognitive states during complex learning. D'Mello and Graesser (2012) show that boredom, frustration and confusion are the result of repeated or sustained *impasse*. *Impasse* is a point at which the learner cannot complete a given task due to non-comprehension. The map of learning presented in D'Mello and Graesser (2012) suggests that once *impasse* occurs, the learner attempts to problem-solve until either they succeed in resolving the problem or they lose motivation and succumb to negative affect such as boredom, frustration or anger. Recognising affect is one way human tutors can monitor for comprehension states. However, the dependency on manifest affect is limited by the subjective nature of affect interpretation and the temporal inaccuracy of the observable behaviours. As negative affect occurs only after sustained or repeated non-comprehension, it therefore cannot give insight into the specific concept, word or symbol which failed to *integrate*.

3.4 Automatic assessment of learner comprehension

In section 3.3 literature highlighted how human tutors use learners' displays of NVB to assess comprehension levels during reading, problem-solving and information recall tasks.

To transfer the approach from human-to-human classroom tuition to human-to-computer e-learning it is necessary to detect and interpret comprehension

indicative NVB automatically using computational methods. This section presents a survey of relevant literature on cognitive analysis by computational modelling and classification of non-verbal behaviour in a range of learning contexts.

3.4.1 Assessing cognition by an affect proxy

Much of the recent e-learning research on non-verbal behaviour (NVB) analysis (D'Mello and Graesser, 2010; Landowska, 2013; Lin et al., 2014; Rajendran et al., 2013; Whitehill et al., 2008, 2011) has focused on identifying a learner's emotional state from macro level NVB such as facial expressions or facial actions. The aim of affect-responsive learning systems is to respond to a learner's emotional state to support learning objectives. For example, Calvo and D'Mello (2010) suggest a learning system that responds to learner frustration would increase learning gain when compared to a non affect-responsive system. Mindspark (Rajendran et al., 2013) is one such intelligent tutoring system, capable of predicting learner frustration based on facial actions.

Commonly, affective systems use facial expressions (Ekman and Friesen, 2003) to match observed behaviour to a pre-defined cognitive-affective or emotional state (Calvo and D'Mello, 2010; D'Mello and Graesser, 2010; Rajendran et al., 2013; Sidney et al., 2005). While facial expressions and facial actions can be identified effectively using machine learning and image processing techniques (Bartlett et al., 1999; Lin et al., 2014; Whitehill et al., 2008), data sets of posed facial expressions and theoretically prescribed facial actions are assumed authentic, often relying on posed images staged using theoretical models provided by psychological and behavioural theory, rather than based on authentic observed behaviour in context. Where research such as Chen et al. (2014); D'Mello and Graesser (2010) has ventured away from prescribed facial expressions and actions towards statistical measurements of observed behaviour, there is often dependence on impractical technologies such as intru-

sive body-attached sensors or high specification high-speed or RGB-D cameras. The problem with the current technical methods used in this field are their lack of practical applicability outside of research, where in a real classroom, home-school or self-directed learning environment learners will most likely not have access to expensive specialist equipment.

A limitation of affect-responsiveness is evident when considering the affect dynamics shown by D'Mello and Graesser (2012). D'Mello and Graesser (2012) present a compelling argument that target affects such as frustration and boredom only occur after sustained impasse is experienced. D'Mello and Graesser (2012) suggest that learners experience repeated cycles of failure, leading to feelings of hopelessness, during sustained impasse. With this dynamic in mind, affect-response appears to necessitate that sustained impasse be endured by the learner before affect-response can occur. In this way affect-response is unsatisfactory. From a pedagogic perspective, it would appear desirable to respond directly to non-comprehension events as they occur, rather than wait for impasse to manifest as negative affect.

3.4.2 Atheoretic modelling of physical behaviour

Temporally accurate methods have used surveys of non-verbal behaviour to model the atheoretic patterns of behaviour indicative of underlying cognitive states. Unlike affect proxies, atheoretic models attempt to discriminate between patterns of observed behaviour in a given context. Research using both skeletal movement (Won et al., 2014) tracking and broad survey coarse-grained multi-channel NVB analysis (Buckingham et al., 2012, 2014) seek to map complex behavioural patterns to objective ground-truth comprehension states. In Buckingham et al. (2012, 2014); Won et al. (2014) comprehension is assessed under interview scenario, as learner (subject) and tutor (interviewer) interact verbally. The research suggests that learner NVB is a strong indicator of learning and subject comprehension when analysed in human-to-human verbal

information recall tasks, such as post-tuition interviews. However, from a methodological perspective, as with formal comprehension assessment examinations, literature (Cain and Oakhill, 2006; Oakhill and Yuill, 1986) indicates that an approach dependent on information recall and discussion may conflate comprehension states with compounding factors such as memory and speech reproduction difficulties. From a technical perspective, the use of standard video camera equipment in FATHOM (Buckingham et al., 2014) indicates a practical real-world technical solution for objective comprehension classification, but the literature lacks demonstration that the approach or findings are transferable from human-to-human verbal interaction to non-communicative on-screen information processing.

Analysis of NVB shows promising results when applied to predicting self-reported task difficulty (Hrubes and Feldman, 2001; van Amelsvoort et al., 2013). The research (van Amelsvoort et al., 2013) shows that even coarse NVB such as head movement can be a strong indicator of the perception of task difficulty. While the results are promising, both studies rely on subjective post-hoc self-reporting of difficulty, rather than establishing an objective measure of comprehension as a ground-truth.

Analysing objective learner comprehension of on-screen information has been approached using eye-tracking techniques and body attached sensors (Bednarik and Tukiainen, 2006; Chen et al., 2014; Copeland et al., 2014; D'Mello and Graesser, 2010; Gerjets et al., 2014; Yusuf et al., 2007). The literature shows positive results in providing real-time comprehension classifications for mental processing of on-screen materials in a variety of laboratory experiments. Objective measures of comprehension or learner performance are established by appraising demonstrable performance under examination (e.g. when answering questions or identifying a region of programming code). However, the technical approach is limited in real-world applications due to the high costs and impracticality of using specialist hardware, such as high speed eye-tracking

cameras, head mounts, chin rests, special chairs or body attached sensors, in a real classroom environment with many students. An eye-tracking and heat-map approach (Chen et al., 2014) presents a second problem. The approach depends on modelling discriminant fixation heat-maps for each state, but also for each information display. In this way, the approach lacks generality. A concern in adopting such an approach would be that information could not be easily changed without the classifier needing to learn new discriminative behavioural patterns.

3.4.3 A simple machine learning approach to atheoretic behavioural analysis and cognitive classification

FATHOM (Buckingham et al., 2014) provides a viable technical approach on which this research can build. The approach uses practical, low-cost and non-intrusive camera hardware to monitor multiple channels of coarse non-verbal behaviour, such as head movement, gaze direction, blink rate and skin tone change (blushing and blanching) during verbal communication of recalled information. FATHOM (Buckingham et al., 2014) uses a trained neural network to estimate the strength of association between the input behaviour pattern, a 40 variable numeric vector representing the average or cumulative behaviour of the subject over a given time period, and the polar classes *comprehension* and *non-comprehension*. The neural network outputs a single real number value on the scale of -1.0 (polar non-comprehension) to +1.0 (polar comprehension). A threshold function, for example ± 0.5, is applied to the output scale to allow for binary classification. Network outputs not meeting the threshold are disregarded. FATHOM (Buckingham et al., 2014) is reported to have achieved normalised classification accuracy of 76%.

FATHOM (Buckingham et al., 2012, 2014), and its methodological parent Silent Talker (Rothwell et al., 2006), monitor the subtle, subconscious and uncontrolled behavioural changes which occur in response to stress during

information recall, the same NVB discussed and advocated as authentic in the classroom (Barry et al., 2011; Zoric et al., 2007). The systems leverage the physical effects of active problem-solving during information recall to create distinct patterns of behaviour where information has caused low or high cognitive load. In information recall tasks, such as those examined in (Buckingham et al., 2014; Won et al., 2014), cognitive load increases when a study subject attempts to recall poorly comprehended information. The additional cognitive load causes a stress response which is expressed subconsciously by small momentary changes in the subject's non-verbal behaviour. CI and CLT (section 3.2) can provide an explanation for the mechanistic relationship between comprehension and stress responses.

3.5 Discussion

The combination of CI and CLT provides a workable explanation for the manifest learner behaviours identified in guidance to educationalists (Snowling et al., 2009) on assessing comprehension. According to CI, cognitive load is increased when non-comprehension occurs and according to CLT, increased *intrinsic cognitive load* will manifest in stress-response behaviours such as changes in subconscious non-verbal behaviour, information recall and reproduction errors, delayed responses and increased inter-utterance corrections.

Review of computational methods for automatic detection of comprehension by analysis of non-verbal behaviour has highlighted a specific gap in the literature. While affect detection by classification of facial expressions is a low-cost and widely available technology, the classifications cannot be considered *real-time*. Temporally accurate learning and comprehension systems, such as FATHOM (Buckingham et al., 2014), have demonstrated success in classifying broad survey coarse-grained multi-channel NVB during dyadic human-to-human verbal information recall tasks. However, the methods have not been evaluated for analysis during on-screen information consumption and processing tasks.

Arithmetic comprehension has been explored using practical technology but the study models subjective post-hoc reviews of task difficulty without an objective measure of true comprehension. Where objective measures of comprehension during information reading and processing have been the context, technical solutions such as eye-tracking, heat-maps and body attached sensors prove cumbersome and impractical. None of the approaches discussed in literature demonstrate the requisite criteria for a practical, low-cost, non-intrusive, real-time classification method capable of analysing NVB to estimate and classify learner comprehension during consumption and mental processing of on-screen information.

This thesis discusses the design, development and evaluation of a novel, practical, low-cost, non-intrusive classifier for estimation and classification of e-learner comprehension during reading and mental processing of on-screen information.

In designing a novel system to solve the problem, the literature has provided a suitable starting point in FATHOM (Buckingham et al., 2014). Although the system was designed and evaluated for recognising comprehension indicative patterns of subconscious behaviour in pre-recorded videos of human-to-human verbal information recall tasks (e.g. a recorded interview), CI and CLT (and the observable subconscious stress-response behaviours) should be applicable to an active learning context. This thesis presents and discusses a novel model of e-learner comprehension from non-verbal behaviour, and a computational method for estimation and classification of comprehension in near real-time, as learners read and process information from the screen. The CI and CLT theories support the hypothesis that patterns of learner NVB may be discriminable depending on the outcome of the CI process for on-screen information. On-screen information which cannot be *integrated* will cause increased cognitive load, stress response and some degree of change in physical or physiological non-verbal behaviour.

3.6 Conclusion

The review of literature in this chapter has provided definition of comprehension in the context of computer mediated, text-based, discursive tutoring. Literature has highlighted that for a text-based tutoring system, reading comprehension is the focus of concern.

In literature, reading comprehension is explained as the process of making meaning from textual information, entailing decoding, reproduction and recall tasks.

Construction-integration theory (CI) has explained *how* decoding of textual information occurs through cognitive processes. The CI process of comprehension suggests that non-comprehension states are accompanied by increased mental effort, as problem-solving is required to construct a mental model in which information can be successfully decoded.

Literature reviewed in this chapter has shown how reading comprehension is formally assessed using post-hoc examination and highlighted that there is an active research interest in developing real-time comprehension assessment technologies.

Classroom studies have shown that human tutors perform real-time comprehension assessment by appraising the non-verbal behaviour (NVB) of learners. As is suggested in reviewed literature, NVB may be an indicator of increased mental stress which can be determined visually by a tutor.

Literature shows that systems have been developed to model reading comprehension in computer assisted learning environments but have done so using impractical technologies. Literature on modelling comprehension during verbal information recall tasks has highlighted a practical technical approach which overcomes many of the barriers to technological application in a real-world classroom environment. FATHOM uses only a camera to track learner behaviour and is capable of predicting comprehension with reasonable accuracy when recorded videos are analysed.

Literature highlights that to achieve the type of adaptation functions defined in educational literature on tutoring (section 2.6), an *intelligent* virtual tutor must be able to infer reading comprehension states, in real-time, during problem solving with on-screen textual information.

Chapter 4 presents literature review and discussion on the behaviour of *intelligent tutoring systems*, with a focus on implementation of behaviours highlighted in educational literature. In section 4.8.4 discussion returns to the relationship between adaptive tutoring behaviour, feedback and comprehension assessment.

Chapter 4

Conversational intelligent tutoring systems and expert systems

4.1 Introduction

Conversational Intelligent Tutoring Systems (CITS) are advanced agent-based e-learning systems which use natural language, conversation, to deliver tutorial content, challenge and feedback through naturalistic interactive discourse. Using conversational strategies, a CITS is able to contextualise and personalise tuition in response to a learner's discursive contributions.

CITS, such as OSCAR (Latham et al., 2014), advance traditional content management systems, such as Moodle (Moodle, 2017), by facilitating the use of advanced pedagogies such as cognitive apprenticeship and learning scaffolding (section 2.2). In this research near real-time comprehension classification is used as an adaptation trigger for interventions in conversational tutoring. The comprehension based adaptive conversational intelligent tutoring system will adapt the tutorial conversation, adding scaffolding, in response to indicators of non-comprehension.

This chapter presents background literature on the evolution of CITS (section 4.2) and the structure and architecture of CITS (section 4.4). A detailed review and discussion of existing techniques for natural language understanding is presented in section 4.5, with specific attention given to the benefits and limitations of semantics (section 4.5.1), pattern matching (section 4.5.2) and combinatorial (section 4.5.3) approaches to extracting, matching and understanding information contained within free text. Knowledge representation and encoding is surveyed and discussed in section 4.6 and goal-oriented conversation strategies in 4.7. Important to this research is the topic of personalisation and adaptation, discussed in section 4.8).

To inform the approach taken in evaluating the comprehension based adaptive conversational intelligent tutoring system developed through this research, section 4.9 surveys and discusses CITS specific evaluation frameworks and techniques.

Discussion and synthesis of the literature relating to conversational intelligent tutoring systems are presented in section 4.10.

4.2 Background

Intelligent tutoring systems (ITS) have been an active research area since the 1980s. Early systems focused on the delivery of instructional training (Swigger and Holman, 1988) by digitising instruction manuals and automating content delivery. As early as 1991 research (Burns et al., 1991) suggested that basic artificial intelligence, comprising decision trees and control logic, could improve learning outcomes by personalising the learning experience. In 1995 a major advancement was made when it was suggested that systems should provide students with real-time feedback indicating whether a learner had gone 'off track' (Anderson and Koedinger, 1995). Introducing real-time feedback alongside existing curriculum sequencing transformed the information delivery

system into a dialogue through which students were to be challenged and critiqued.

Both cognitive apprenticeship and scaffolding learning use corrective feedback to overcome sustained learner impasse. Graesser et al (Graesser et al., 2001) recognise the importance of immediate, meaningful and contextualised feedback in learning. Unlike ANDES' (Vanlehn, 2006) visual feedback, AutoTutor's (Graesser et al., 2001) feedback is delivered in natural language as part of an ongoing dialogue between the learner and the virtual tutor. To achieve this they integrated a conversational agent (CA) which was able to receive a learner's dialogue, analyse the content for correctness of concepts and respond with contextually relevant feedback. The combination of ITS and CA is known as a conversational intelligent tutoring system (CITS).

A CITS is a computer program which delivers tutorial instruction through natural language. CITS are able to deliver instruction in small chunks, challenge learners, analyse discourse and provide corrective feedback. More recent CITS (D'Mello et al., 2005; Latham et al., 2012a, 2014; Latham, 2011; Lin et al., 2014) adapt to the learner, personalising the media and methods selected for tuition depending on the attributes of the learner. The ambition of such systems is to mimic a human tutor in implementing pedagogic devices to support learning.

4.3 Can computers teach?

VanLehn (VanLehn, 2011) surveyed a number of learning environments including human tutoring, book based learning and intelligent tutoring systems. VanLehn (VanLehn, 2011) calculated the resultant learning gain (equation 2.2) for a period of study in each environment. The results show that step-by-step conversational tutoring using a computer system out-performs book-based learning and is comparable with human tutoring.

Research on pedagogy in the classroom (Cazden and John, 1985) suggests that learners' engagement can be better supported when sociolinguistic factors

are considered. Literature on the effective behaviours of human tutors (Cazden and John, 1985; Lee, 1995), using sociolinguistic and cultural factors to engage learners, suggests that a conversational e-learning system can better support learning outcomes by incorporating, as in OSCAR CITS (Latham et al., 2012a), specific cultural dialect, domain knowledge, slang and informal grammar and vocabulary, along with the conversational strategies for intervention, support, feedback and challenge which are at the heart of effective pedagogy (see section 2.2.1).

It is the CITS' ability to adapt and respond to the learner, in context, to employ effective conversational and pedagogic strategies and to recognise and respond to learner needs that gives the CITS an advantage over traditional static content delivery, such as books or digital content management systems.

4.4 The structure of a CITS

Sani and Aris (Sani and Aris, 2014) review a generic model of a CITS, as described in (Vanlehn, 2006), based on four primary components (figure 4.1).

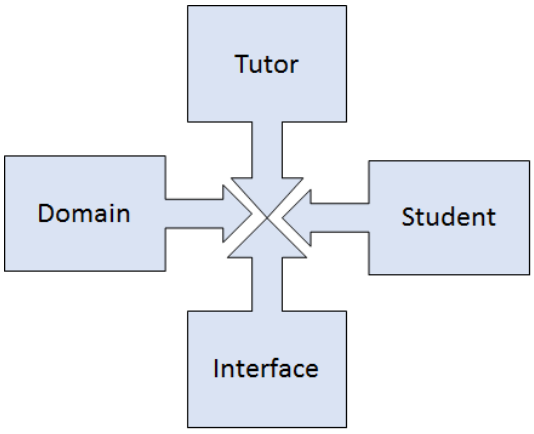


Figure 4.1 Generic model of a CITS

1. Domain model

   The domain model contains the expert knowledge for the CITS. This may include conversational scripts (Latham et al., 2012a), search indexes (Figa and Tarau, 2004) or graph data (Kim et al., 2007). The information

encoded within these repositories is used to conduct the conversation and the tutorial.

2. Tutor model

   The tutor model contains the business logic for the application, namely the conversational and pedagogic rules by which the tutorial conversation will operate. The tutor model decides what to do and when, based on the domain and the student. The tutor model is responsible for natural language understanding and maintaining the state of the conversation, student information and tutorial progress.

3. Student model

   The student model contains demographic and personal information about the learner, learning performance metrics and a log of activities undertaken. In addition to tutorial progress and scores, the student model may contain information about learning style (Latham et al., 2012a), affective state (D'Mello et al., 2005; Lin et al., 2014) or other data for personalisation.

4. Interface model

   The interface model is most commonly a Graphical User Interface (GUI) which resembles a chat or messenger interface. Commonly the GUI has additional windows for display of supporting content such as images, video or text. Embodied (Graesser et al., 2001; Lin et al., 2014; Rickel and Johnson, 2000) CITS include an animated avatar to represent the tutor. Full embodiment, for example within a virtual reality environment, gives the virtual tutor the ability to interact using non-verbal behaviour, gestures and facial expressions, and interact with objects in the environment (Rickel and Johnson, 2000). To a lesser extent a simple animated avatar image (Lin et al., 2014) can provide a focal point for interaction.

4.5 Natural language interpretation

Both AutoTutor (Graesser et al., 2003) and OSCAR (Latham et al., 2012b) are advanced conversational intelligent tutoring systems which use natural language to structure and deliver tuition. In this section technical approaches to natural language systems are discussed.

4.5.1 Semantic analysis

Semantic Analysis, such as used in AutoTutor (Graesser et al., 2000), attempts to understand the meaning of words used in utterances. AutoTutor uses a combination of latent (Landauer et al., 2014) and non-latent (Cai et al., 2004) semantic features to compare learner conversational dialogue with model dialogues. Using the semantic similarity, AutoTutor is able to navigate the conversation, appraise discursive contributions by the learner and prepare responses.

Latent semantic information can be derived by analysis of corpus linguistics, for example the frequency with which one word occurs in relation to another, by measuring the distance between words in a semantic graph or by analysing the function of words within a utterance. The structure of utterances can give insights into the function of words within an utterance - for example, identifying nouns, adjectives and verbs can highlight the subject of a sentence.

Table 4.1 Example of POS tagging

Utterance	**POS tags**
The quick brown fox	DT JJ JJ NN
The brown fox is quick	DT JJ NN VBZ JJ
The brown fox is slow	DT JJ NN VBZ JJ

POS tagging is a powerful tool for identifying the purpose, subject and orientation of an utterance. Row 1 of table 4.1 shows the utterance ‘The quick brown fox’ and its part of speech tags. ‘The’ is the determiner of the sentence, ‘quick’ and ‘brown’ are both tagged as adjectives and ‘fox’ is identified as a

singular noun. Row 2 of table 4.1 shows a very different sentence to row 1 (table 4.1), but the computer still correctly identifies the subject of the sentence (the noun) and the descriptive words (the adjectives). In both forms, the subject and descriptions can be extracted despite the varying grammatical structure. POS therefore provides a method of generalising about the content of natural language. Using POS on simple sentences, rudimentary assumptions can be made. For example, with a single noun present it can be assumed that the adjectives relate to the noun. However, POS cannot give a full understanding of semantics.

WordNet (Miller, 1995) is a large graph of words, connected by their relationships: hyponym and hypernym, meronym, synonym and antonym. Figure 4.2 (Bird et al., 2009a) shows a fragment of the concept hierarchy for hyponyms and hypernyms of 'motorcar'. Leveraging the relations within WordNet it is possible to understand how words and their semantic abstractions relate and thereby convey their meaning.

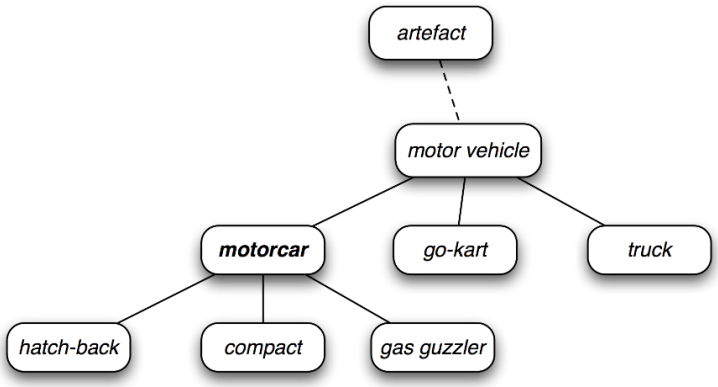


Figure 4.2 Example of WordNet semantic ontology (source: Bird et al. (2009b):online)

Row 3 of table 4.1 shows that 'the slow brown fox' has the same POS tags as row 1, while clearly the two utterances express opposite opinions. The relative orientation of the two utterances is determined by prior knowledge of the antonym relationship between the words 'fast' and 'slow'. The relations in WordNet show that 'fast' and 'slow' are linked by the antonym relation

- indicating that the *descriptor* (adjective) of the *subject* (noun) is inverse between the two utterances. Using WordNet therefore would allow a computer to determine that two utterances which look similar and function similarly, are semantically highly dissimilar.

However, while WordNet is a useful and widely adopted tool for analysis of semantics, the success of an algorithm dependent on WordNet is limited by the completeness of the graph for words which are domain specific to the conversation being analysed.

4.5.2 Pattern matching

Pattern matching is a simple approach to pairing input and response dialogues by matching strings. Pattern matching languages, such as AIML (Wallace) or PatternScript (ConvAgent), allow input string patterns to be defined, along with response strings. The agent finds the best match for the input string and retrieves the response string from the script. Both AIML and PatternScript allow for more complex implementations, incorporating variables, wildcards and conditions. However, pattern matching is a simpler approach to conversational direction because unlike a semantic approach, the system does not have to *understand* the content of the utterance.

Pattern matching languages have been widely adopted for conversational systems, primarily through the success of A.L.I.C.E and AIML (Wallace, 2009). CITS, as an off-shoot of conversational systems, have therefore followed suit (Fonte et al., 2009; Latham et al., 2014; Latham, 2011; Mikic et al., 2008, 2009). Pattern matching has distinct advantages over semantic analysis alone, as the script author can embed their specific domain knowledge in the model patterns and responses. The patterns can represent words, numbers or symbols - so can be applied in contexts where semantics fail. The patterns also allow for use of abbreviations, informal language, slang or metaphors - all aspects of authentic conversation in which semantic analysis would fall short. For example, a simple

pattern to match a sentence where the user gives their name is shown in the table below.

Table 4.2 Example of simple pattern matching

Utterance	**Pattern**	**Match**
Hello, my name is Philip.	*name is*	true
I think my name is great.	*name is*	true

The pattern defines that the presence of the string 'name is' surrounded by any other content is a match for a sentence where a name is given. For input row 1, this simple pattern correctly identifies that the utterance contains a statement of the speaker's name. However, row 2 highlights the drawback of this approach. In row 2 the *function* of the sentence is entirely different, yet the pattern still matches. The solution to this, in terms of pattern matching, is to create many patterns to match increasing specific sub-cases.

Table 4.3 Example of extended pattern matching rules

Utterance	**Pattern**	**Match**	**Is best match**
Hello, my name is Philip.	*my name is*	true	true
I think my name is great.	*name is*	true	false
I think my name is great.	*name is great*	true	true
Philip is a great name.	*name is great*	false	false

The examples in table 4.3 highlight both an advantage and a limitation of the approach. The patterns can be defined to match any arbitrary string, meaning that highly idiosyncratic utterances can be matched. The approach could allow for slang words, spelling mistakes or even non-verbal symbols such as mathematics or programming code. However, as the conversations expands the number of patterns needed grows exponentially.

The pattern matching approach has been successfully implemented in OSCAR (Latham et al., 2012b) using Convagent's (ConvAgent) pattern based scripting language PatternScript. PatternScript is a pattern matching language for natural language which uses a combination of regular expression like descriptive patterns and weightings to trigger responses to conversational input.

Other CITS (Mikic et al., 2008, 2009) use similar pattern matching languages to encode the structure and content of tutorial conversations, answers to questions and feedback.

OSCAR faced a novel problem in tutoring SQL by natural language, as the system had to allow for specific but comprehensive pattern matching. Latham et al. (Latham et al., 2012a, 2014; Latham, 2011) defined thousands of rules to achieve a conversational flow across 10 SQL programming questions. The approach allowed learners a degree of linguistic freedom in interacting with OSCAR, while still enforcing the strict syntactic rules of SQL and checking for conceptual correctness in learners' answers.

AIML script file repositories (Wallace, 2017) highlight the major drawback of the pattern matching approach. For example, the pattern matching file just for handling past participial phrases contains 8,980 distinct and individually encoded patterns. A similar issue exists for the OSCAR CITS (Latham et al., 2012a), where pattern files for interactions contain many thousands of encoded strings - each representing a contextual model interaction.

4.5.3 A combinatorial measure of text similarity

STASIS (Li et al., 2006), a short text similarity measure, uses a combination of POS tagging and WordNet queries to assess the semantic similarity of utterances and has been extended to provide a framework for conversational systems (O'Shea, 2014; O'Shea et al., 2009).

O'Shea et al. (O'Shea, 2012) used the short text similarity measure (Li et al., 2006) to compare the semantic and syntactic similarity of two sentences, allowing for greater generalisation of written form and reducing the number of phrase patterns required for each conversational interaction. The short text similar measure algorithm (Li et al., 2006) produces a single similarity score between 0.0 and 1.0, based on a weighted sum of two measures - semantic similarity based on the distance between words in WordNet and the similarity

of syntax based on POS tagging. The approach still depends on pre-defined patterns to match against but because there is some generalisation through semantic and syntactic analysis, the number of patterns needed for each case is greatly reduced.

A comparative study (Cai et al., 2011) of the effectiveness of LSA, pattern matching languages and regular expressions finds that for short texts, regular expressions are more accurate and reliable than complex attempts at semantic evaluation. The findings fit with the preliminary evaluation of STASIS (Li et al., 2006; O'Shea et al., 2009) for use in a tutorial for computer programming. When comparing the utterances 'the quick brown fox' and 'the brown fox is quick' the short text similarity algorithm produces a semantic similarity of 0.99, syntactic similarity of 0.73 and overall similarity score of 0.96. In this example generalisation works well, correctly identifying that the two utterances express the same meaning despite different syntactic form. However, when comparing two 'for loop' constructors, such as '*for(int i = 0; i <= 0; i++) and for(int i = 0; i <= 0; i–)*', the algorithm produces an overall match score of 1.0. In this programming example the generalisation is unwelcome as the two loop constructor forms describe dramatically different algorithmic behaviour but are considered identical.

The literature has highlighted that there are practical problems with the approaches adopted for both pattern matching and semantic analysis. In this research, a combinatorial approach will be devised to take advantage of the merits of each while minimising the effect of their limitations. For example, when the tutor needs to know if the learner is asking a question then semantic analysis would provide sufficient generalisation to limit the number of patterns needed. When the tutor needs to make a highly precise evaluation of a mathematical formula provided by the student, regular expressions can be used.

4.6 Encoding knowledge as a graph

CITS belong to a class of system called *expert systems*. Expert systems use domain knowledge to enact the effective behaviours of human experts in a given practice. The ability for a CITS to enact the discursive and pedagogical expertise of a human tutor depends on encoding that expert knowledge in a format usable by the virtual tutor. Information for conversational systems is commonly encoded in mark-up scripting languages such as AIML (Wallace) or PatternScript (ConvAgent). (ConvAgent; Latham et al., 2012a; Mikic et al., 2009; O'Shea et al., 2009).

As highlighted in the review of educational theory (Chapter 2), hierarchy and structure are key aspects of a cognitivist learning strategy. Ontologies are a method of representing information and knowledge (Tian et al., 2007; Wongthongtham and Zadjabbari, 2009) through structures, hierarchy and relationships. Used in areas such as intelligent reasoning (Besnard et al., 2007), information retrieval (Wongthongtham and Zadjabbari, 2009) and knowledge management (Wang et al., 2007), ontologies represent a simplified version of the world (Wang et al., 2007), giving representation to the content of, and relationships between, shared information in a given knowledge domain.

As with WordNet (Miller, 1995), the *relationship* is key to the structural representation of knowledge within an ontology. Figure 4.3 shows that apple is a fruit, using a directional relationship. Unlike other information storage and representation systems, an ontology comes to define its entities, such as 'apple' or 'fruit', primarily by their relationships with other entities. This unique property of the ontology makes it an ideal method of representing hierarchical information, such as that used when teaching and learning using cognitivist methods (section 2.2.1).

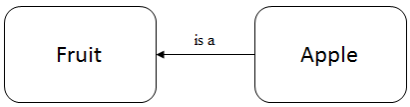


Figure 4.3 Example of a basic ontology

Ontology has been used in the past to map the semantic content of knowledge domains for use in e-learning applications (Serón and Bobed, 2016; Vesin et al., 2012). Protus 2 (Vesin et al., 2012) is a Java programming tutorial system which uses ontologies to personalise content and recommend onward learning, based on rules, derived from relationships between programming concepts in the ontology. Protus 2 (Vesin et al., 2012) contains multiple, discrete ontologies representing Knowledge, User, Task and Pedagogy.

Given the knowledge domain of a Java programming tutorial, an ontology (as shown in figure 4.4) representing the integrated domains of Knowledge, Pedagogy and Context can be created so as to map the knowledge domain of a human tutor. The ontology contains sufficient entities and relations so as to represent the recommended features of cognitive apprenticeship (section 2.2.1). The ontology shown in figure 4.4 defines structure, hierarchy, example, definition and assessment.

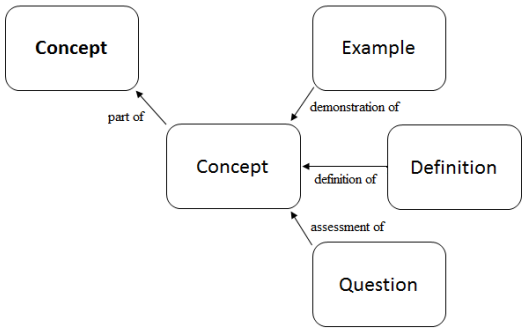


Figure 4.4 Example of a basic ontology

The *part of* relationship in the ontology shown in figure 4.4 provides directional structure to the *path* of concepts linking any two *concepts* in the network.

The ontology can be viewed as a directed graph. Entities within the ontology are nodes within the graph, connected together with directed relationships. Interestingly, so long as the graph is fully connected, any given node can be linked to any other node in a Hamiltonian path (Harris et al., 2008). The approach allows for novel journeys and relations to be explored within the knowledge domain. For example, the graph allows queries such as 'which Y are part of X?' or 'how does Y relate to X?'. The relationships between concepts within a graph can be used to structure the direction of a conversational tutorial.

4.7 Directing goal-oriented conversation

The tutor model encapsulates the business logic for the CITS, a combination of natural language understanding, data layer integration, event handling and state maintenance. The primary role of the tutor model is to decide what actions to take and when to take them, based on the tutorial, the student and the conversational context.

VanLehn (Vanlehn, 2006) discussed a two loop algorithm to achieve long term goals through short term conversational interactions. The outer loop is a macro structure of goals which deliver a learner from prior state to the desired new state. The inner-loop handles the specific conversational moves required to progress through each long-term goal. To implement the two loop algorithm a tutor must be able to construct a tutorial.

Defining tutorial content within a graph structure allows tutorial structure, the outer loop, to be discovered as a Hamiltonian path (Harris et al., 2008), a walk through a graph of concepts, from prior knowledge to a new learning objective.

Handling of short term goals on the inner loop requires a combination of domain knowledge reasoning and natural language understanding.

(Austin, 1962) observed that in *saying* something, one was usually *doing* something. Rather than viewing utterances purely as expressions containing evaluable information, utterances have an intent to produce consequential effects. Searle (Searle, 1969) linked this to goals, as speech acts. The complexities of analysing speech acts are well documented (Geis, 1995); nevertheless the use of such analysis for goal-oriented conversational systems is widely accepted (Grünberg, 2014; Kang et al., 2010; Ko, 2015; Litman and Allen, 1987). Goal-driven speech act identification has been used effectively to categorise natural language dialogue for education (Porayska-Pomsta et al., 2000), suggesting that a taxonomy of goals and associated speech acts can be used to drive conversational dynamics between student and tutor.

In the absence of standard user interaction devices, such as buttons or menus, speech act classification allows a system to process heterogeneous interactions appropriately. For example, an utterance may contain a request or a statement or a domain specific task. Speech act classification provides a mechanism by which to determine what type of action the learner is trying to take in any given conversational move. While cross-domain speech act classification (Grünberg, 2014; Kang et al., 2010; Ko, 2015) is complex as it requires more statistical analysis, corpus linguistics and use of complex classifiers such as neural networks. A simpler approach is evident in OSCAR, a CITS developed by Latham et al. (2012a, 2014); Latham (2011). OSCAR does not perform explicit speech act analysis but does pre-parse utterances against generic patterns before parsing against domain specific patterns. By doing so, OSCAR is able to handle a limited number of conversational moves out of context. A similar approach could be used to identify the acts contained within an utterance.

Review of literature suggests that there are three steps to a goal-oriented conversational system:

1. **Identify a process to achieve a stated goal.**

   This is the outer loop and can be either reasoned from an ontology or pre-defined in scripts.

2. **Identify the intended consequence of a learner's current act of speech.**

   This step opens the inner loop and can be achieved either by speech act classification or by pre-parsing against patterns.

3. **Execute conditional response based on the context of outer loop and intended consequence of the current speech act.**

   This step closes the inner loop and can be performed by calling on a combination of contextual logic for the speech act or by outputting pre-canned scripted responses.

4.8 Adaptation and personalisation

As highlighted in 2.2.1, a key concept for effective tuition is *scaffolding*.

Clark and Graves (2005) suggest that an effective technique for scaffolding text comprehension (see section 3.2) is to decompose complex tasks into small chunks with expressive interrelations. To achieve task decomposition in a conversational framework, Graesser et al. (1995) define a five-step dialogue pattern, developed by analysis of tutorial corpora.

1. Tutor asks a question
2. Learner answers the question
3. Tutor provides feedback
4. Tutor and Learner iterate answer and feedback cycles to improve solution
5. Tutor assesses Learner's understanding

This five step dialogue pattern aligns with both Clark and Graves (2005) effective technique for scaffolding and is demonstrated in the behaviour of CITS such as AutoTutor (Graesser et al., 2004), OSCAR (Latham et al., 2012a) and Hendrix (Holmes et al., 2015a).

Clark and Graves (2005) suggest that the tutor's role is to prompt the learner, ask questions and elaborate on the learner's responses. In addition, they state (Clark and Graves, 2005), that tutors must consider how a learner moves closer to an end goal over time and how feedback on responses can be used to make a learner aware of the mental processes required to comprehend the text.

Model progression, a gradual increase in complexity and specificity, is also highlighted in VanLehn et al. (2017), as a means of meta-tutoring- guiding the learner from simple to complex tasks as understanding is demonstrated.

For conversational interfaces, this progress can be managed using the two loop algorithm (Vanlehn, 2006). As discussed in literature (Clark and Graves, 2005; VanLehn et al., 2017), interrelation, progression, increasing complexity and meta-tutoring of process structure are taken as key functions of the tutoring behaviour.

The methods for adaptation discussed in literature (Clark and Graves, 2005; VanLehn et al., 2017) relate to adaptation of either content or process. In Siler and VanLehn (2015) definitions are given for two different types of adaptation, namely micro and macro adaptation

Macro is focused on selecting appropriate tasks to be undertaken, for example the number of challenges for a learner, the order of tasks, the complexity or the materials used (e.g. text or diagram). For example, OSCAR (Latham et al., 2012a) is a macro-adaptive CITS capable of detecting learning style preference and selecting appropriate materials to support learning.

Micro is focused on adaptation within the conversational turn, for example if a student is taking a long time to answer a question, the tutor may intervene to correct an error or provide a process hint to aid the learner.

The essential difference between micro and macro adaptation types is the scope of the adaptation. Macro adapts based on information which is constant across all interactions- such as learning style or prior test performance. Micro adapts based on more immediate information such as the content of an utterance or an affective expression (Lin et al., 2014).

The study presented in Siler and VanLehn (2015) is particularly important in understanding the types of behaviours and adaptations human tutors use and provides an illuminating and surprising conclusion which is highly relevant for the advancement of intelligent tutoring systems, particularly conversational and agent based systems such as Hendrix Holmes et al. (2015a).

In Siler and VanLehn (2015), the authors investigate whether a tutor's choice to adapt (macro-adaptation) related to the tutor's prior knowledge of learner competence. In the study, tutors were given the opportunity to enact a macro-adaptation, by asking a single harder question or deploying a series of simpler questions. The effect of prior knowledge is evaluated by comparing the adaptive choices of a tutor group where the student and tutor remain paired across activities and conversely, a tutor group where the student and tutor pairing is changed between activities.

The conclusion of Siler and VanLehn (2015) relevant to the research explored in this thesis is that for computer mediated tutors who could make accurate holistic appraisals of learner competence in advance of tutorial activities, their macro-adaptive choices did not appear to reflect their appraisals.

The results suggest that regardless of a-priori assessment of the learner, tutors didn't pose harder questions to more competent learners. However, the paper (Siler and VanLehn, 2015) goes on to synthesise the findings with other literature in the field and the authors suggest that while tutors may

not choose harder questions for more competent learners, the level of detail provided by tutors to support learning varies depending on the assessment of learner comprehension.

Importantly, discussion presented in (Siler and VanLehn, 2015) suggests that human tutor behaviour is deficit oriented in that tutors are inclined to present a standard syllabus of challenges and then intervene with additional support if required on a task by task, interaction by interaction basis without consideration of holistic or course level assessment playing a role, and that it is the timely and contextually relevant feedback and explanation provided in overcoming difficulty that human tutors, whether face to face or computer mediated, choose to focus on.

This observed behaviour suggests that macro-adaptation, such as alternative question selections, difficulty scales and learning style adaptation are not behaviours which a human-like intelligent agent software would need to adopt, provided the aim was to mimic but not exceed human tutor behaviour.

4.8.1 Immediate vs coaching micro-adaptation

VanLehn et al. (2017) explore two types of micro-adaptive behaviour, which they refer to as 'immediate' or 'coaching'. In VanLehn et al. (2017), a custom Intelligent Tutoring System, Dragoon, is developed to implement immediate or coaching adaptations. Dragoon helps students to learn process modelling, a skill closely linked to software computer programming and the understanding of algorithms.

Immediate feedback is described in VanLehn et al. (2017) as concrete (correct or incorrect) feedback on the product of the learner's work, the solutions manifest and provided. For immediate feedback, whenever a solution is attempted a red or green colour indicates the correctness of the solution. Coaching is described as the provision of hints on how to approach the problem. VanLehn

et al. (2017) refer to coaching as meta-tutoring, as it is intended to help learners adopt successful cognitive strategies for comprehension.

VanLehn et al. (2017) note that immediate correct or incorrect feedback is problematic, as it could encourage learners to make errors repeatedly and intentionally in order to prompt the system for a solution.

4.8.2 Micro-adaptive behaviours for coaching

VanLehn et al. (2017) detail behaviours for coaching micro-adaptation. These are shown below:

1. feedback and hints on the model
2. feedback and hints on the learner's process (meta-tutoring)
3. reflective debriefing
4. concrete articulation strategy
5. decomposition of tasks
6. answering student questions

For the Dragoon ITS (VanLehn et al., 2017), behaviours 1, 2 and 3 are enacted whereas behaviours 4, 5 and 6 are not attempted. In this research the Hendrix CITS developed to investigate the effect of comprehension based adaptation will implement all the coaching behaviours suggested in VanLehn et al. (2017).

4.8.3 Timing feedback

Holmes et al. (2015b) suggest that effective feedback in electronic learning addresses three questions - what, how and *when*. Conclusions highlight the importance of *how* and the *when*, particularly in relation to interruptive feedback. Their study proposes a number of triggers for introduction of feedback:

1. On successful completion of a step

2. On a procedural error

3. On completing a task

4. After 4 seconds of hesitation

Of the four feedback triggers used in Holmes et al. (2015b), *on success*, *on error* and *on completion* are simple context-and-content based adaptation heuristics. Literature contains examples of studies which explore the effect of context-and-content based adaptations on learning outcomes in intelligent tutoring systems.

Context-and-content are used to provide feedback in Graesser et al. (2004) and to adapt tutorial content to learning style in Latham et al. (2014). In VanLehn et al. (2017), the Dragoon ITS is evaluated against a non-adaptive software. The comparison yields important results in terms of *how* micro-adaptation should be performed. The study finds that there was a significant ($p=0.021$) improvement in post-tutorial exam scores for students studying with the adaptive tutor over the non-adaptive software. The difference between the two systems is not the content available to the learner but the context and timing of feedback. In both adaptive and non-adaptive cases, students had access to the same overall content. However, the adaptive tutor was able to introduce feedback and hints into the learning process at an appropriate and contextually relevant point in problem-solving. Elsewhere, literature (Nicol and Macfarlane-Dick, 2006; Siler and VanLehn, 2015) supports the conclusion that context and content relevance are important attributes of effective feedback.

The final trigger suggested in Holmes et al. (2015b), *on delay*, poses a more difficult challenge. The *on delay* heuristic supposes that a 4-second response delay is sufficient evidence of *non-comprehension* such that a supporting dialogue should be used. However, Holmes et al. (2015b) point out that such *interruptive* feedback must be well timed so as not to distract the learner or prevent independent problem-solving. A delay in response is not adaptation based on the content of a solution, but on observation of learner non-verbal *behaviour*.

4.8.4 Beyond conversational content

While Holmes et al. (2015b) used a simple delay time heuristic to capture an aspect of behaviour, modelling of learner non-verbal behaviour has been used in more complex ways to adapt feedback responses in intelligent tutoring systems. Literature shows that affective adaptation is an emerging aspect of the student model. Affect detection and adaptation is a form of micro-adaptation which goes beyond content analysis during tutor-learner interactions.

D'Mello and Graesser (2012) studied the affective-state dynamics of students during tutorial learning. They found that affective state changes are strongly associated with learning impasse or success as well as with motivation. Whitehill et al. (2008) discuss a framework for integration of affective-state detection into a CITS.

Using the facial action coding system (FACS) they suggest that prototypical affective states such as sadness, happiness, anger, etc. can be decomposed into a set of state-indicative facial feature values and classified automatically using support vector machines (SVM). Marchand and Gutierrez (2012), and Huk and Ludwigs (2009), find that affective state is a useful proxy for cognitive state and that an appropriate response to affective state can support learning by reducing cognitive load. Lin et al. (2014) implemented affective feedback within a CITS by automatically identifying the affective state of a CITS user. Lin et al. (2014) developed an affective algorithm able to monitor a learner's facial expressions during learning via a web camera facing the learner. The research concludes that feedback of sympathetic emotional valence from the virtual tutor supports learning.

However, as discussed in section 3.4, affect comes late in the learning process (D'Mello and Graesser, 2012). As was shown by VanLehn et al. (2017), timing of adaptive feedback is important to the effectiveness of intervention. While work by Lin et al. (2014) showed that affect mirroring increased overall engagement, it is the intention of this research to explore timing interventions based on

non-verbal indicators associated with the cognitive processes of comprehension as defined in section 3.2.

The literature suggests that non-verbal behaviour expressed during learning may allow for effective timing of interruptive feedback should the behaviour detected be sufficiently immediate to the cognitive process such that context is not lost. In this research a novel adaptation based on e-learner comprehension of on-screen information is designed, developed and evaluated. Adaptation of dialogue in response to context and learner plays an important role in supporting the effective pedagogy in conversational intelligent tutoring systems.

4.9 Evaluating conversational intelligent tutoring systems

Evaluation of conversational systems can be made complex by the subjectivity of discourse appraisal. In section 2.5 literature highlighted that objective measures of goal-attainment should be combined with subjective measures of the appropriateness of materials and learner satisfaction. In this section a discussion of relevant literature on the the evaluation of conversational systems is presented and linked to considerations for evaluation of instructional systems. Three frameworks are discussed: the goal question metrics (Basili and Rombach, 1994), the PARADISE (Walker et al., 1997) framework and learning gain measurement (Latham et al., 2014; VanLehn, 2011).

The PARADISE framework (Walker et al., 1997) (figure 4.5) is an evaluation framework designed for use with conversational systems. The PARADISE framework places user satisfaction as the generalised objective of conversational systems and seeks to evaluate the performance of a system by analysing the effect of both success measures and dialogue cost measures on user satisfaction. The PARADISE framework recognises the need for objective measurements of

performance in assessment of goal success. The approach has been adopted successfully in conversational systems (Walker et al., 1998).

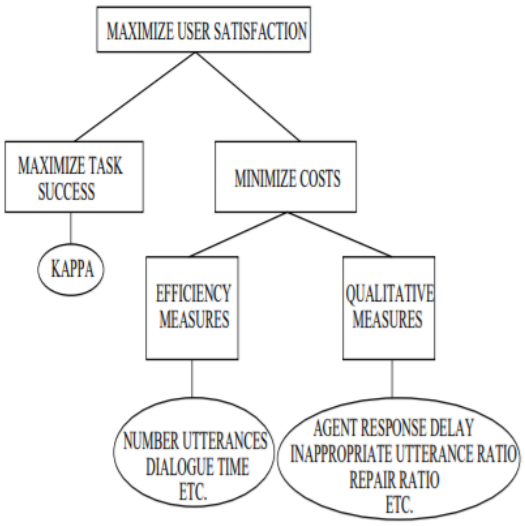


Figure 4.5 PARADISE evaluation framework diagram (Walker et al., 1997)

However, the supposition that user satisfaction is the ultimate goal of an educational conversational system is incorrect. The ultimate goal of an educational conversation system is to educate. While user satisfaction and educational attainment may correlate, it is not proven. As such, over-dependence on user satisfaction as a single measure of success appears flawed.

CITS (Latham et al., 2012a,b, 2014; Latham, 2011) have been evaluated using learning gain metrics adopted from educational theory (chapter 2, section 2.3.1). However, dependence on learning gain alone is questioned in literature (section 2.3.1).

The Goal Question Metric (GQM) (Basili and Rombach, 1994) framework presents a more holistic approach to system evaluation. The GQM paradigm (Basili and Rombach, 1994) describes a three tier hierarchy of observable objectives. The GQM avoids over-dependence on a single metric and incorrect attribution of purpose by first defining the ultimate goal of the system based on purpose. The goal of the system is not prescribed by the evaluation framework. The goal is then broken down into key performance indicators, which are questions with observable answers. Finally, metrics are selected to answer each

question. Figure 4.6 shows an example GQM for a conversational intelligent tutoring system.

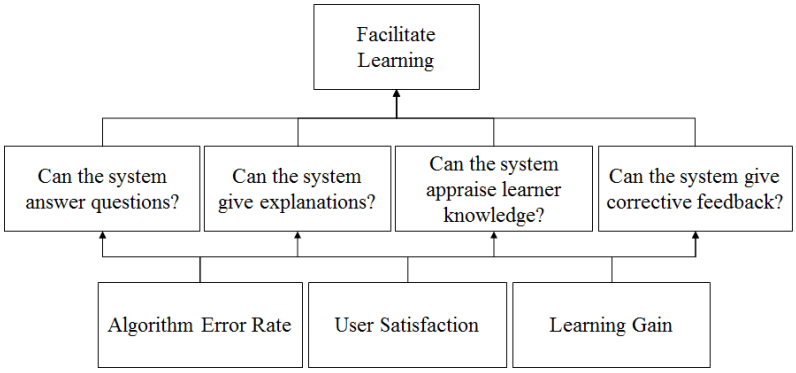


Figure 4.6 Example GQM for a conversational intelligent tutoring system

4.9.1 Evaluating the effectiveness of adaptation

An effective adaptation is one which increases an objective measure of learning. In this research, timing of micro-adaptive feedback dialogues is evaluated by controlled experiment. The objective learning outcomes of students encountering the adaptation is compared statistically to the performance of a control group in the absence of that adaptation.

4.10 Discussion

Sani and Aris (2014) provide a conceptual framework for the structure of a CITS, identifying the primary components and roles of system function. The framework will be used as a basis for structure of the CITS in this project.

The CITS developed should consider socio-linguistic adaptation to provide remedial support for low levels of confidence, interest and aptitude. Low confidence should be identified and corrected using supportive feedback and presentation of options for progression (e.g. suggesting the learner ask a question). Low interest, or wandering off-topic, can be corrected by reaffirming the learning objectives and guiding the conversation back onto track. Low aptitude should be identified by continuous appraisal of the correctness of

information provided by learners. When information is incorrect, follow-up dialogues should help to *fill* the gaps in knowledge and correct misconceptions.

Natural language understanding is an extremely complex topic. Although pattern matching provides the specificity needed to match highly idiosyncratic utterances and semantic analysis gives the ability to generalise about the meaning of utterances, neither approach alone provides a satisfactory solution for tutoring computer sciences, where programming and mathematics will play an important role. While structured mark-up languages such as PatternScript (such as in Latham et al. (2012a)) or AIML (such as in Fonte et al. (2009)) have been used to define CITS knowledge domain, they are verbose. Neither PatternScript nor AIML allow for the structure of knowledge to be adequately represented, which leads to case-specific repetition and definition of many patterns for different cases.

Given the two-loop algorithm described by Vanlehn (2006), the act of conversation can be viewed as a sequence of actions and sub-actions required to complete a process. Viewing the conversational structure as a macro-level process, the outer loop lends itself to using an ontology. Using an ontology on concepts and materials, such as that in Protus 2 (Vesin et al., 2012), allows for the relationships between knowledge entities to be used to structure conversation and contextualise patterns. Sub-processes, on the inner loop, the specific turn-by-turn conversational interactions, need to be brokered by some understanding of the intended *consequence* of an utterance. If the act of speech can be determined from the utterance, and the subject of the utterance found using syntactic analysis, then a robust understanding of direction is derived.

Given the limited number of action types required to support the domain activities, classification using generic syntactic patterns is possible. For example, the patterns can be defined to represent a request for definition, a request for demonstration, or a statement of confirmation. Once classification is made, the tutor is able to action the intended consequences, giving greater flexibility

and contextualisation to the response without the need to pre-define response patterns for each case in each conversational step.

Evaluation of the system should follow the goal-question-metric paradigm, as it provides the clearest definition and criteria for objective appraisal of qualitative objectives, such as '*converse effectively*'. User feedback scores and regression analysis can be incorporated from the PARADISE framework so as to explore the relationship between system function performance and subjective measures of learner satisfaction. Learning gain will be measured, but included as one of a number of evaluation metrics.

4.11 Conclusion

This chapter has presented an overview of literature on the concept, function and evaluation of conversational intelligent tutoring systems (CITS). Literature has provided a broadly accepted generic structure for CITS, containing domain, student, tutor and interface components.

In-depth review of literature on parsing conversational content has identified that for tutoring programming, both patterns and semantics will play important roles in appraising dialogue content.

Literature reviewed in this chapter has shown that goal directed conversations entail three actions - identify the process to achieve a goal, identify the consequence of learner intents in each interaction and execute a conditional response to support progression.

Review of literature has shown that domain knowledge can be effectively encoded as a graph, supporting the *process* and *decomposition* behaviours of tutors reviewed in Chapter 2.

In conversational systems, literature has shown *speect act classification* to be an effective method of extracting *intent* from free text. Speech acts can replace buttons in a dialogue driven system such that each *intent* can be mapped to a *consequence*.

To execute conditional responses, adaptation is required. Adaptation and conditional feedback have been highlighted as important in educational best practice, summarised in conclusions to Chapter 2. Again in CITS system design, adaptation is a focus of research to drive improvements in effective digital tutoring.

Literature has highlighted a dialogue pattern for interactions into which adaptation can be deployed, a process including questioning, response, feedback, iteration and assessment. This pattern defines the core behaviour of the *intelligent* tutor in each dialogue loop and fits precisely with the recommended practice of cognitive apprenticeship, as discussed in Chapter 2.

Literature has discussed two types of adaptation for *process* and *feedback*. Macro-adaptation focuses on selection of materials and questions, while micro-adaptation focuses on the content of discourse. Experiments drawn from literature indicate that human tutors are often deficit oriented, choosing not to enact macro-adaptation but to present the most challenging content first and then adapt to support the learner in overcoming the challenge.

Micro-adaptation, both immediate and coached, provide learners with context specific support during the learning process. Literature has highlighted six key behaviours for micro-adaptation in conversational tutoring, including feedback and hints on both solution and process, summary of content, immediate feedback on the correctness of solutions, decomposing tasks and iterating to build understanding and answering ad-hoc questions from the learner. The micro-adaptive behaviours highlighted in CITS literature correspond to the practices of human tutors, highlighted in Chapter 2.

Experiments drawn from literature and discussed in this chapter have shown that timing micro-adaptation, particularly feedback, is hugely important to the effectiveness of tuition. While existing systems have implemented dialogue-content based adaptation, there are few examples of sophisticated attempts to time micro-adaptation using other information channels available to human

tutors, such as learner non-verbal behaviour as discussed in review of automated comprehension assessment technologies in Chapter 3 sections 3.4 and 3.4.3. In Chapters 5 and 6, literature on methods for analysing learner non-verbal behavioural patterns is reviewed in greater depth.

Finally, review of literature has shown that evaluation of CITS is complex and requires integration of multiple performance metrics. The goal-question-matrix strategy provides a method of integrating observed and subjective performance indicators.

Chapter 5

Machine learning with Artificial Neural Networks

5.1 Introduction

In section 3.4.3, a demonstrated method of classifying comprehension by computational analysis of non-verbal behaviour during information recall tasks (Buckingham et al., 2014) was discussed. The approach depends on multiple layers of classifiers, producing binary classifications of complex visual input. These classifiers are part of an expansive field of computing, a branch of artificial intelligence, known as machine learning. As shown in a review of methods by Amancio et al. (2014), there are many different types of classifier used in machine learning. However, the classifiers discussed in section 3.4.3 and demonstrated in literature by Buckingham et al. (2012, 2014) for analysis of non-verbal behaviour are artificial neural networks (ANN).

To aid discussion of related works on face and facial detection, behaviour classification and comprehension classification (Chapter 6), this chapter presents an overview of literature on artificial neural networks, the development of the field and how artificial neural networks are constructed and trained.

ANN are statistical models based on a simplified representation of the human brain. Like the human brain, ANN have been discussed in literature as universal approximators capable of learning any pattern evident in data given enough examples to learn from (Hornik et al., 1989). ANN have been applied to a broad range of problems since their first inception in the late 1950s (Rosenblatt, 1958). In many pattern recognition problems ANN are now proving more accurate than humans. Figure 5.1 shows a comparison of the structure and function of biological and artificial neural networks.

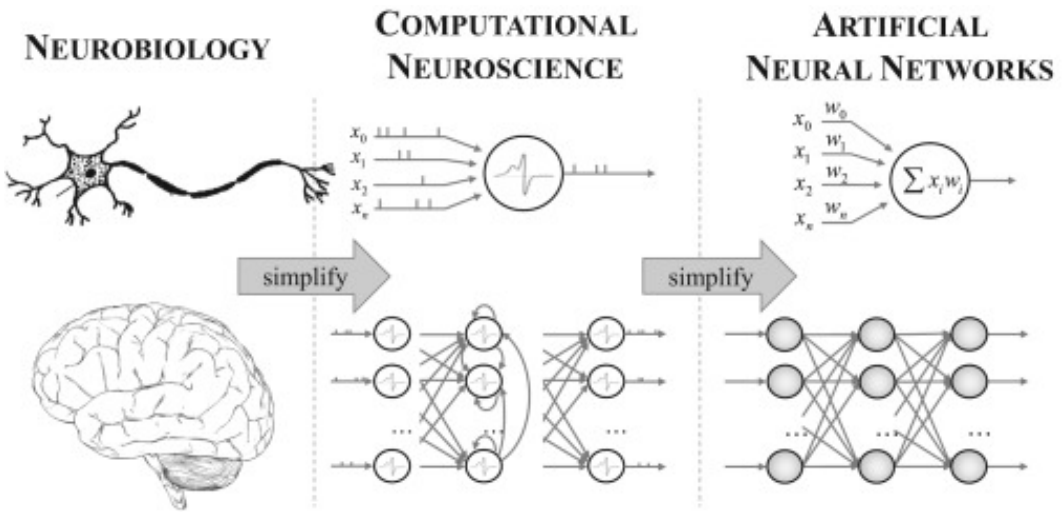


Figure 5.1 Comparison of biological and artificial neural networks (source: Almási et al. (2016):32

5.2 Simple artificial neural networks

Artificial neural networks (ANN) (Haykin, 1994) are models which can learn to map n inputs to k outputs. Based on a simplified model of biological neuron behaviour, the artificial neuron is able to accept multidimensional input, apply a transformation function and produce an output. In this chapter ANN will be discussed in the context of face detection, feature state classification and comprehension classification. This section presents an introduction to key concepts for machine learning with ANN.

The simplest form of artificial neural network is a Perceptron (Rosenblatt, 1958). As shown in figure 5.2 a Perceptron is a single layered feed-forward network with i inputs, i weights and n binary outputs. The output neuron sums the weighted inputs and applies a threshold, t, activation function to produce a binary output. Equation 5.1 shows the output, y, of the network. Analogous to biological neurons, the output neuron is said to *fire* if the activation function returns a positive value.

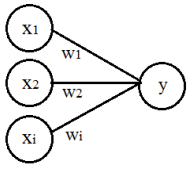


Figure 5.2 Perceptron Network

$$\text{if } \sum_i w_i x_i > t \text{ then } y = 1 \qquad (5.1)$$

$$\text{else } y = 0$$

To solve more complex multi-class problems the Perceptron network can be expanded (figure 5.3) to include multiple output neurons, each functioning as shown in equation 5.1.

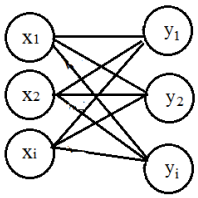


Figure 5.3 Multi-class Single Layer Perceptron Network

Perceptrons could learn using the Perceptron Learning Rule (equation 5.2) (Rojas, 1996b). The rule is applied for each example in a given set of pre-labelled data called a *training set*. By continuously updating both t and w_i when the network output is incorrect, the model can *learn* the parameters of the

problem. The precise amount to increase or decrease the threshold and weights is determined by a *learning rate* parameter. Convergence occurs when there is linear separation of classes and a decision surface can be drawn between each class (Rosenblatt, 1962). A neural network converges when no improvement can be made to the accuracy of the outputs.

$$\begin{aligned} &\text{if label} = 0 \text{ and } \sum_i w_i x_i > t \text{ then reduce } t \text{ and increase } w_i \\ &\text{else if label} = 1 \text{ and } \sum_i w_i x_i < t \text{ then increase } t \text{ and decrease } w_i \end{aligned} \tag{5.2}$$

However, by the late 1960s the evident limitations (Minsky and Papert, 1969) of single layer Perceptron networks had reduced their popularity. The problem was the networks' inability to model non-linear functions. To classify complex non-linear functions with disjoint decision boundaries the Perceptron had to evolve.

5.3 Multi-layer artificial neural networks

Minsky and Papert (1969) were aware that multi-layer networks could accommodate more complex computation but the theory was limited by the absence of a learning algorithm. Figure 5.4 illustrates the topology of a Multi-layer Perceptron Network (MLP).

In figure 5.4 inputs are shown as $\{x_1 \dots x_i\}$ and outputs as $\{y_1 \dots y_i\}$. As with the single layer Perceptron the output activation functions for $\{y_1 \dots y_i\}$ are calculated by comparing the the sum of weighted inputs to a threshold. However, the MLP introduced a *hidden* layer of neurons which act to provide intermediary computation between input and output. Equation 5.3 shows a linear activation for a hidden layer neuron in $\{h_{1,1} \dots h_{i,j}\}$, while equation 5.4 shows a non-linear activation function. Each hidden layer neuron passes forward its activation value to the next hidden layer, becoming the input vector

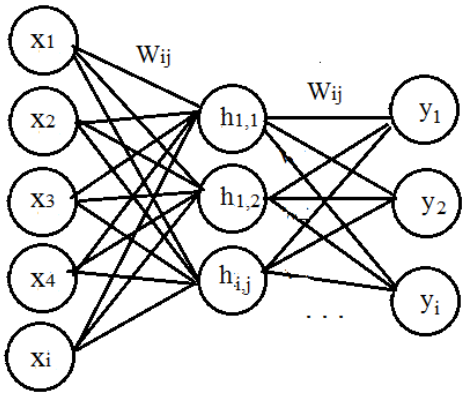


Figure 5.4 Multi-layer Perceptron Network

$\{x_1 \ldots x_i\}$ for the next layer. The final layer consists of the output nodes $\{y_1 \ldots y_i\}$. The output activation function for nodes in $\{y_1 \ldots y_i\}$ can similarly be represented by equation 5.4.

$$\text{if } \sum_{ij} w_{ij}x_{ij} > t \text{ then } activation = 1 \qquad (5.3)$$

$$\text{else } activation = 0$$

$$activation = tanh(\sum_{ij} w_{ij}x_{ij} > t) \qquad (5.4)$$

While the topology of the network was well understood by the late 1960s, a method of training the algorithm was still absent. The problem was how to apply the Perceptron Learning Rule (also known as the Delta Rule (Haykin, 1994)) when the hidden layers had no target output.

Werbos (Werbos, 1974) overcame the problem by transforming the discrete threshold function of the Perceptron's output (equation 5.1) to a differentiable output (equation 5.4). The shift allowed the Delta Rule to be generalised to

produce the back-propagation algorithm. Back-propagation is discussed in more depth in section 5.4.

MLP trained using back-propagation have been hugely successful in solving complex pattern recognition problems. MLP have been successfully used in a broad range of cognition and emotion recognition studies (Buckingham et al., 2012, 2014; Hai et al., 2015; Khandait et al., 2011; Owayjan et al., 2016; Rothwell et al., 2006, 2007).

MLP are just one of an ever increasing family of algorithms within the artificial neural networks. Recurrent neural networks (RNN) are a more recent addition to the family. RNN allow layers of hidden nodes to loop backwards within the network. RNN are particularly applicable to problems where sequence dependence is an important factor, such as time-series analysis (Hüsken and Stagge, 2003) and natural language understanding (Hu et al., 2017). Recent progress has been made in applying Convolutional Neural Networks (CNN) to image labelling, with Recurrent CNN showing promise in automatic video stream image segmentation and labelling (Graves et al., 2007; Kahou et al., 2016; LeCun et al., 1998; Lopes et al., 2017; Mayya et al., 2016; Pinheiro and Collobert, 2014; Teng and Yang, 2016)

In this research however, a simple MLP will be the starting position. The decision to adopt MLP over RNN or CNN as an assumed starting point is based on evidence that MLP have been successfully applied to similar problems (Buckingham et al., 2012, 2014; Hai et al., 2015; Khandait et al., 2011; Owayjan et al., 2016; Rothwell et al., 2006, 2007). In addition, the universality theorem of feed-forward networks (Hornik et al., 1989) suggests that a single MLP network, with at least one hidden layer, is capable of modelling any arbitrary problem given an error. Given these two findings, the least risk approach to developing a proof-of-concept system is to adopt MLP until evidence suggests a different model should be used.

5.4 Training artificial neural networks

Machine learning algorithms such as artificial neural networks require data to learn from. Sections 5.2 and 5.3 introduced training artificial neural networks in terms of *supervised learning*, where a training set of data is pre-labelled and the network is trained to meet those target labels. However, it is possible to train a network using *unsupervised learning* techniques.

Unsupervised learning, as used in (Graves et al., 2007; Kahou et al., 2016; Pinheiro and Collobert, 2014), does not depend on pre-defined labels but rather, organises the inputs into natural groups using algorithms such as clustering. Rather than learning the parameters of the class labels, as in supervised learning, the model learns the parameters of the natural clusters.

Supervised learning works by exposing the model to inputs for which there is a pre-determined output. The model self-corrects over the labelled inputs, the training set, until as many as possible of the inputs produce the pre-determined output. For the purpose of this research the discussion will focus on *supervised learning*, as this has been successfully applied in similar contexts (Buckingham et al., 2012, 2014; Hai et al., 2015; Kahou et al., 2016; Khandait et al., 2011; Owayjan et al., 2016; Rothwell et al., 2006; Teng and Yang, 2016; Tompson et al., 2014).

5.4.1 Training by back-propagation

The most common supervised learning algorithm is back-propagation (Werbos, 1974) and well documented in literature (Rojas, 1996a). Back-propagation uses a derivative of the error to update the previous layer of connected neurons - the errors *propagate backwards* through the network. The algorithm is a generalisation of the Delta Rule (Haykin, 1994).

The network is created with n fully connected inputs, m non-linear hidden layer neurons and p outputs. Each connection is weighted, as w_{ij}. Weights are set to random initial values in the space $0 \pm 1/\text{fan-in}$, where fan-in is the

number of incoming connections. A learning rate η is set and error function defined $E(w_{ij})$. For each iteration of learning the weights on connected neurons are updated using the rule shown in equation 5.5. The process is repeated until the error reaches an acceptance level.

$$\Delta w_{ij} = \frac{-\eta \partial E(w_{ij})}{\partial w_{ij}} \tag{5.5}$$

5.5 Challenges in supervised learning

Supervised learning with Multilayer Perceptron Networks (Rosenblatt, 1958) has been successfully applied to the problem of NVB analysis found in literature (Buckingham et al., 2012, 2014; Khandait et al., 2011; Rothwell et al., 2006, 2007). However, as observed by Webb et al. (2001) there are practical challenges to supervised approaches. A significant challenge is gathering labelled datasets large enough to train effective models. Publicly available labelled datasets of text works, for example, can contain many millions of lines of text. Creating a labelled training set for a novel problem can be laborious and costly. Once a dataset has been gathered, a second problem arises: do the labels still reflect reality? The problem, called *concept drift* (Webb et al., 2001), occurs when target patterns change over time causing the training data to *drift* from observations.

5.6 Conclusion

Review of literature presented in this chapter has given an introduction to artificial neural networks, as a sub-field of machine learning. Literature has highlighted that multi-layer perceptrons, coupled with back-propagation training, have been hugely successful in applied pattern recognition problems.

Literature has shown that MLP have been successfully applied to analysis of behaviour and shown accuracy in classifying cognitive processes such as emotion (Khandait et al., 2011), deception (Rothwell et al., 2007) and comprehension (Buckingham et al., 2014).

In Chapter 6, literature on comprehension analysis by machine learning is reviewed in greater detail with the intention of identifying which methods, behaviours and measurements are optimal for machine learning.

Chapter 6

Modelling and classifying patterns of non-verbal behaviour

6.1 Introduction

This chapter gives an overview of the relevant literature on technologies and applications for pattern recognition problems relevant to this research, particularly focusing on face and facial feature detection and classifying cognitive states using patterns of of non-verbal behaviour.

These technologies will be used in this research to locate, extract and classify non-verbal behaviours visible in web camera image stream data.

6.2 Non-verbal behaviour

The non-verbal behaviour of learners has already been discussed in the context of educational practice (Chapter 2), comprehension assessment (Chapter 3) and with regard to timing micro-adaptation in CITS (Chapter 4).

In chapter 3, literature on comprehension assessment highlighted how human tutors use learners' displays of non-verbal behaviour as a means of perceiving

learner comprehension during reading, problem-solving and information recall tasks. In chapter 4, review briefly touched on how NVB could be used as a timing trigger for micro-adaptation in conversational systems.

Hrubes and Feldman (2001) asked whether displays of NVB, a social communicative behaviour, would persist when problem-solving was undertaken in a solitary environment. They find that displays of anxiety remain observable in both high and low self-monitoring students when there are no other humans to communicate with. The findings mirror cognitive load theory, discussed in section 3.2, suggesting that as cognitive load increases due to non-comprehension of information, subconscious non-verbal expressions of stress will emerge.

Anxiety responses to non-comprehension of information are found elsewhere in literature (Vrij et al., 2008). Vrij et al. (2008) suggests that increasing cognitive load will exacerbate a subject's observable stress responses. Similarly, Rothwell et al. (2006, 2007) show that the additional cognitive load involved in articulating false memories, rather than recalling the truth, is manifest and observable under computational analysis. Stress response to cognitive load has also been observed in learning contexts (Haapalainen et al., 2010).

In the context of both lie detection and comprehension assessment, the literature suggests that increased cognitive load will manifest in visible NVB such as blushing due to increased heart rate and breathing, blanching due to release of cortisol, and fluctuation in galvanic skin response (GSR), electroencephalogram (EEG) and electrocardiogram (ECG) measurements.

From literature it is possible to identify a model of target non-verbal behaviours. Table 6.1 shows non-verbal behavioural channels identified from literature.

Table 6.1 reflects the importance of facial features in non-verbal communication. Literature (Knapp and Hall, 2014; Mehrabian, 1968) suggests that the most informative region of the body for NVB cues in human to human communication is the face.

Table 6.1 Non-verbal behaviour channels

Channel	**References**
Head movement	Buckingham et al. (2014); Rothwell et al. (2006); van Amelsvoort et al. (2013); Won et al. (2014)
Posture / Body	Patterson et al. (1980); Won et al. (2014)
Eye gaze	Buckingham et al. (2012, 2014); Doherty-Sneddon and Phelps (2007); Emmorey et al. (2008); Ishii et al. (2013); Khandait et al. (2011); Rothwell et al. (2006)
Eye contact	Patterson et al. (1980); Vrij et al. (2000)
Eye brow position	Khandait et al. (2011)
Eye openness	Buckingham et al. (2012, 2014); Khandait et al. (2011); Rothwell et al. (2006)
Physiological	Buckingham et al. (2012, 2014); Haapalainen et al. (2010); Rothwell et al. (2006, 2007)

EEG, ECG and GSR channels are not used in this research, as they require body-attached sensors. Equally, high-speed or infra-red cameras will not be used. This choice has been made so as to ensure that the technology developed and evaluated could be deployed cheaply and conviently in a real-world classroom environment.

6.3 Face and facial-feature detection

Face and facial feature detection is an important component of any computational method for the analysis of NVB, as the majority of NVB channels highlighted in Table 6.1 are expressed by the face or head. Surveys of methods for detecting faces in images (Bakshi and Singhal, 2014; Gupt and Sharma, 2014; Hatem et al., 2015; Lu et al., 2012; Yang et al., 2002) show the diverse solutions available, each with benefits and limitations. Two approaches which regularly feature in literature are artificial neural networks (Haykin, 1994), as used in Buckingham et al. (2012, 2014); Gupt and Sharma (2014); Rothwell et al. (2006, 2007), and Haar cascades (Viola and Jones, 2004), as used in Castrillón et al. (2010); Castrillón-Santana et al. (2008).

Both artificial neural networks and Haar cascades have a role to play in this research. Due to the specific context in which image data is recorded, literature

suggests that a two-step combinatorial process will play to the strengths of each approach while mitigating the weaknesses. This section presents a description and evaluation of both approaches.

6.3.1 Artificial neural networks

Literature (Buckingham et al., 2012, 2014; Gupt and Sharma, 2014; Lu et al., 2012; Rothwell et al., 2006, 2007; Yang et al., 2002) shows that Artificial Neural Networks (ANN) (Chapter 5), specifically Multi-layer Perceptron Networks (MLP), have commonly been used to classify whether regions of an image contain a face. An advantage of ANN for face detection is the ability to train the classifier to recognise faces in almost any position (Yang et al., 2002). Unlike other methods, such as Haar cascades (Viola and Jones, 2004), where facial landmarks are important, an ANN can learn arbitrary discriminant patterns to identify a face. However, a neural network has a fixed input length.

A limitation of MLP is that they can only accept an input of a pre-defined length. When handling image regions, this input is a vector of pixel values. The fixed input length for an ANN meaning the region for classification is expected to be of a fixed height and width (Gupt and Sharma, 2014; Lu et al., 2012; Yang et al., 2002). Whether this aspect of the MLP is problematic depends on the specific application, context of use and conditions under which image data was recorded.

For facial detection, the fixed input of an MLP becomes more problematic due to scale variance of facial features. In situations akin to CCTV monitoring, where there is distance between camera and subject, the a region can be estimated inside which all faces will fit. In this scenario the height and width of the region of interest (ROI) can be well defined.

However, as the face moves closer to the camera the scale variance becomes more extreme. In the context of a web camera several inches away from the subject, an action such as leaning backwards can cause the face to halve in size.

In literature (Buckingham et al., 2012, 2014; Rothwell et al., 2006, 2007), practical applications have assumed a fixed size of face by controlling the distance and relative positions of camera and subject. Doing so ensures the face fits within known height and width boundaries, meaning the image can be searched efficiently for regions containing the target feature. However, if the size of the feature were not known the same approach would be inefficient. The image would need to be searched many times, each time with height and width for the region set to a different size.

One solution to achieve scale invariance is to use principal component analysis (PCA) (Pearson, 1901). PCA is a form of factor analysis which reduces the total variance of the data by removing highly variant weak interactions in order to produce a smaller set of linear factors. The effect of PCA is to reduce the overall dimensionality of the input data while maintaining the important factors, the principal components. PCA is commonly used (Bajwa et al., 2009; Cooray and O'Connor, 2004; Kamencay et al., 2013; Xiao, 2010) in image classification problems where the raw pixel data provides a large input vector with many redundant features and the classifier needs to learn meaningful pixel combinations.

While PCA provides a solution to the fixed input width problem with MLP, such extreme scale variation encountered when camera and subject are only inches apart still causes a high degree of error in recognising face patterns.

For this reason it is necessary to split the process of feature detection and feature-behaviour classification between two different technologies: a highly scale tolerant algorithm to locate the face, eyes, nose and mouth within the image and a pattern classifier to learn comprehension indicative patterns of NVB.

6.3.2 Haar cascades

Haar cascades are collections of weak classifiers designed to recognise haar-like features, wavelets, within images (Viola and Jones, 2004). Collectively, the ensemble of classifiers form a meta-algorithm capable of highly robust classification (Hatem et al., 2015). Haar cascades have gained popularity over recent years as they perform exceptionally well at detecting known-angle faces with variant scale.

First described by Viola and Jones (Viola and Jones, 2004), these simple but effective object recognition classifiers have become ubiquitous in modern technology. Haar cascades search images for simple patterns (figure 6.1) of intensity which are learned from training over large sets of specifically designed training images. To train a face detection Haar classifier a large set of images containing faces on random backgrounds, often with small random distortions applied to the face, is created.

Figure 6.1 Haar-like features (Hatem et al., 2015)

A major advantage to Haar cascades over neural networks, when used to locate a face, is speed of detection for variant scale faces. The ensemble of weak classifiers in the cascade is able to disregard the background of the image quickly, to focus search time effectively on the most face-like regions of the image (Hatem et al., 2015). The patterns of intensity can be applied at any scale in the image, meaning that no strict control of relative positions for camera and subject must be enforced. However, as observed in Gupt and Sharma (2014), Haar cascades perform best on forward facing faces and, unlike neural networks, they do not perform well on faces viewed at different angles.

Figure 6.2 shows an example of bounding boxes for face, eyes, nose and mouth located within a live web camera image stream using public domain, pre-trained, AdaBoost Haar cascades.

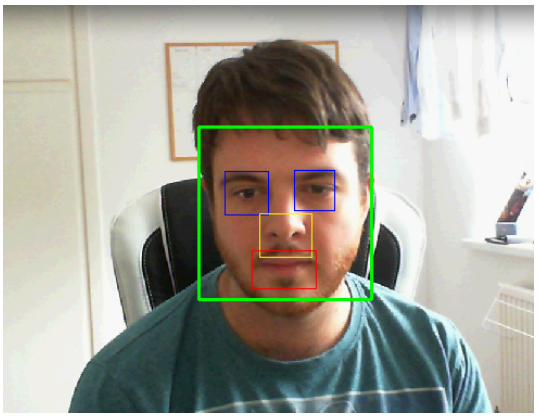


Figure 6.2 Facial feature detection with public domain haar cascades (Castrillón-Santana et al., 2008)

As commented in Hatem et al. (2015), the different face and facial feature recognition algorithms have advantages and disadvantages. The selection of an appropriate algorithm will depend on the context of its use. For example, neural networks will perform well in a scenario where the face is a consistent size, such as when using CCTV where there is significant distance between camera and subject. However, if the face is consistently front-facing then Haar cascades offer more flexibility when faces are of an unpredictable size, such as when the subject is very close to the camera and looking directly towards it.

6.4 Modelling non-verbal behaviour from image stream data

NVB can be modelled by interrogating the state and descriptive attributes of facial features identifiable within the image (section 6.3). Figure 6.3 shows a pipeline for modelling behaviour, similar to that found in Buckingham et al. (2014); Khandait et al. (2011); Rothwell et al. (2006).

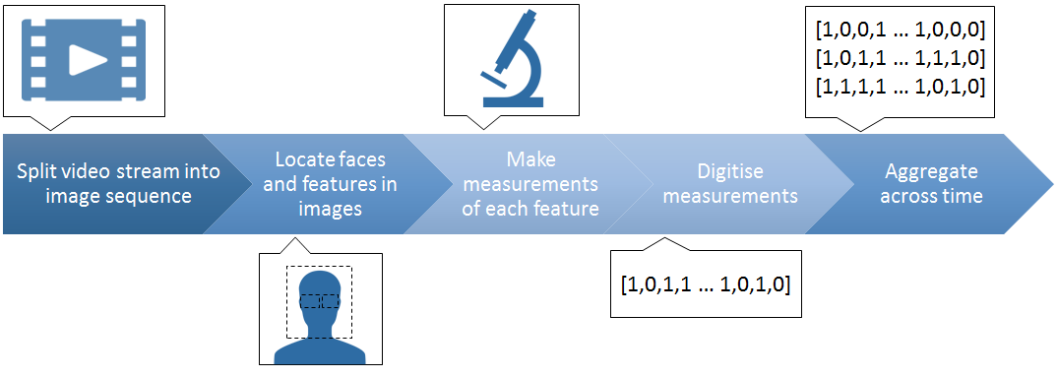


Figure 6.3 Process for modelling behaviour

Facial feature states and attributes for each image are collected into a discrete vector of observations representing the NVB for that moment in time. As shown in figures 6.3 and 6.4, a given period of time can be represented by a matrix of observations. The final step in modelling behaviour is to summarise the behaviour over a given period of time. In figure 6.4 the matrix is summarised by calculating the average value in each column to produce a summary vector of average behaviour over the given observation window.

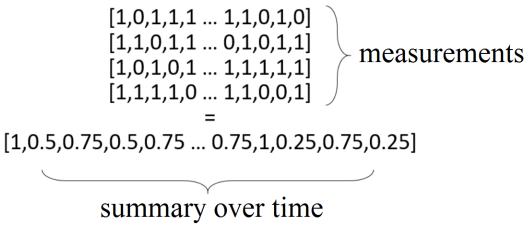


Figure 6.4 Summary of behaviour over time

6.5 Classifying cognitive states from non-verbal behaviour

Complex classification, such as modelling non-verbal behaviour, has been demonstrated using machine learning (Buckingham et al., 2012, 2014; Graves et al., 2007; Hai et al., 2015; Kahou et al., 2016; Khandait et al., 2011; Lopes et al., 2017; Mayya et al., 2016; Owayjan et al., 2016; Pinheiro and Collobert, 2014; Rothwell et al., 2006, 2007; Teng and Yang, 2016; Tompson et al., 2014).

Such algorithms *learn* the problem domain using statistical analysis of the data, producing a model with hyper-parameters optimised to describe the classes, or patterns, within the source data. The trained model can then use these learned parameters to recognise familiar patterns in new data.

Breakthrough research (Rothwell et al., 2006, 2007) from the field of deception detection has demonstrated the effectiveness of ANN (Chapter 5) for classifying deception from non-verbal behaviour during interrogative interviews. In recent research (Buckingham et al., 2012, 2014) the approach has been successfully modified to classify comprehension in both face-to-face medical interview and student oral examination contexts. Importantly, the approach depends entirely on the subject's expressed non-verbal behaviour during cognition. The generalised model of learner comprehension-indicative NVB demonstrated in Buckingham et al. (2014) requires no pre-planning of visual context or question content.

In work by both Rothwell et al. (2006, 2007) and Buckingham et al. (2012, 2014) participants were recorded while answering questions orally, under interview conditions. The video recordings were analysed computationally to extract a numerical model of learner non-verbal behaviour for a given time frame. The model was then classified using an MLP (Chapter 5) to detect the presence of a target cognitive process. The research indicates the MLP classifiers performed well, with Rothwell et al. (2006) reporting classification accuracy of 74% on deception and Buckingham et al. (Buckingham et al., 2014) reporting 76% classification accuracy on comprehension.

The success of the approach pioneered by Rothwell et al. (2006, 2007) and adopted by Buckingham et al. (2012, 2014) was partly owed to the high fidelity of the behavioural data model. In Buckingham et al. (2014) a classification was made on every 1 second of video. Each video frame in the 1 second tranche was analysed to produce a vector of 40 binomial ± 1.0 variables. For example,

+1.0 if the left eye was fully open, -1.0 if it was not. The matrix of values was then reduced to a single 40 variable vector using summary statistics.

6.6 Conclusion

The review of literature presented in this chapter has highlighted how machine learning, particularly artificial neural networks, can be used to model patterns of behaviour expressed during learning activities.

Literature has shown that non-verbal behaviour, such as stress responses, is not dependent on social interaction with a human. This finding highlights an opportunity for techniques, found in literature on analysis of human-to-human communicative behaviour (Buckingham et al., 2014; Khandait et al., 2011; Rothwell et al., 2007), to be applied in the classroom to analyse learner behaviour during e-learning.

Literature has provided a feature set of behavioural features, shown to be linked to cognitive processes, including head movement, posture, gaze and skin tone change. Literature has also shown that streams of image data can be analysed using Haar cascades and artificial neural networks to produce a numeric vector representative of learner behaviour over discrete windows of time.

The literature presented in this chapter demonstrates that learner non-verbal behaviour, such as that used by human tutors to assess comprehension in a classroom setting, can be extracted from digital information and automatically classified to inform on the cognitive state of a learner during a learning activity. The literature suggests an opportunity to develop a similar technology capable of classifying learner comprehension in near real-time, during learning, as a method of timing micro-adaptations in conversational tutoring.

Chapter 7

Hendrix 1.0: A conversational intelligent tutoring system for programming

7.1 Introduction

Hendrix 1.0 (Holmes et al., 2015a) is a conversational intelligent tutoring system (CITS) designed to tutor computer science and programming. At the heart of the CITS is a conversational algorithm which has to ability to interpret and respond to discourse using natural language. Using natural language interpretation, Hendrix 1.0 mimics a human tutor by guiding the learner through tutorial content using discourse, challenge and feedback. It converses with a learner to identify gaps in knowledge through questioning and expanding the curriculum when gaps in knowledge are identified. Hendrix 1.0 supports learners by detecting questions and providing definitions and examples. Hendrix 1.0 uses both syntactic and semantic language analysis to extract and match information from learner utterances. Its two loop algorithm is dependent on identifying the short term goal of a learner in each conversational turn.

Hendrix 1.0 makes both technical and educational contribution to the field of intelligent adaptive CITS. The key contributions of the system are discussed in section 7.3.1.

This chapter introduces Hendrix 1.0 (section 7.3), details the requirements and challenges in developing the system (sections 7.3.2 and 7.3.3), provides wire frames and explanation of the user interface (section 7.4.1), presents the architecture of the system (section 7.4) and discusses the core functions of the system (sections 7.4.2, 7.4.4 and 7.4.3).

7.2 Motivation

The motivation for designing and developing the Hendrix 1.0 CITS (Holmes et al., 2015a) is to provide a base e-learning environment in which comprehension based adaptation can later be evaluated. Hendrix 1.0 uses cognitive apprenticeship and scaffold learning (Chapter 2.2.1) to support and guide learners, giving them space to express their knowledge, discuss their ideas, create their own questions and utilise examples in applied contexts. Hendrix 1.0 adapts the specificity of guidance to provide additional prompting for learners who are not expressing a high degree of knowledge and the tutoring system, by recognising discourse indicators of low confidence, can provide additional supportive dialogue to encourage learners. Hendrix 1.0 has been designed, developed and evaluated as a candidate platform for incorporation of comprehension based adaptation.

7.3 Overview

Hendrix 1.0 Holmes et al. (2015a) is a Conversational Intelligent Tutoring System (CITS) for teaching Java programming at an undergraduate level. Hendrix 1.0 is an ontology based, goal oriented conversational system which is capable of delivering tutorial content through natural language. Adopting a

process of cognitive apprenticeship (see section 2.2), Hendrix 1.0 encourages students by challenging them to construct solutions to applied problems in an ongoing discussion of increasing complexity. As highlighted in literature (sections 2.2.1 and 4.2), timely and relevant feedback has an important role in guiding and correcting learners to develop appropriate conceptual models of the problem domain.

Hendrix 1.0 uses cognitivist devices (section 2.2.1) to guide a learner through a knowledge domain, giving introduction, providing examples and demonstrations and challenging the learner to solve problems and demonstrate understanding of concepts. Hendrix 1.0 provides additional support for learners by giving immediate feedback, guidance and explanation.

Hendrix 1.0 has been developed as a candidate platform for the integration of near real-time comprehension classification. While the pilot platform design and development focuses on the functions of a CITS, Hendrix 1.0 also records and stores web camera image data during conversational tutoring. This image data, along with logs of tutorial questions and answers, will be used to conduct an initial evaluative study of comprehension indicative non-verbal behaviour in e-learning.

7.3.1 Contributions

The educational contribution of Hendrix 1.0 is to implement micro-adaptive behaviours, discussed in section 4.8.2, including hints and feedback on solutions, hints and feedback on approach to problem-solving, decomposition of tasks and answering students' questions.

The two technical contributions of the Hendrix 1.0 system are the ability to structure tutorial plans dynamically using shortest path queries over a directional graph of concepts, reducing the overhead of pre-planning and encoding specific tutorial progressions in static scripting files, and its ability to

parse discursive, mathematical and programming language (Java) content, as part of a conversation.

7.3.2 Key functionality

Hendrix 1.0, the tutor, is a CITS designed for teaching computer programming. The functionality of Hendrix 1.0 is based on the requirements and specifications of cognitive apprenticeship (section 2.2). The high-level system function can be defined as:

- **Use information about the learner to personalise conversation**

  Capturing personal information allows Hendrix 1.0 to begin personalisation. For example, addressing the learner by name. This aims to increase engagement and motivation by developing the personal relationship between the learner and Hendrix 1.0.

- **Elicit a learning objective from the user**

  Identifying learning objectives in discourse allows the user to control their learning experience by expressing their own learning objectives. This aims to increase engagement and motivation, by placing the learner in control and re-positioning Hendrix 1.0 as an assistant.

- **Use pre-existing domain knowledge to dynamically structure a tutorial**

  Hendrix 1.0 dynamically structures tutorial plans, based on a graph of knowledge. Hendrix 1.0 organises concepts into a sequence of discussions and challenges for a learning objective, giving the learner opportunities to ask questions, work with applied examples and demonstrate competence over time. This aims to ensure that content, discussion and challenge is always relevant to the learning objective.

- **Discuss a topic with the learner**

  Hendrix 1.0 must be able to discuss a topic with the learner. Hendrix 1.0

will introduce new concepts and be able to give definition and example. Hendrix 1.0 will be both pro-active and responsive in providing definition and example when a learner meets a new concept. This aims to allow the learner to develop some contextualised understanding of the topic of discussion before applying the concept in problem-solving challenges.

- **Set contextually relevant challenges for the learner**
  Hendrix 1.0 must set challenges for the learner, within the context of the on-going dialogue. The challenges should push the learner to explore the information available on the topic under discussion and demonstrate comprehension of the information.

- **Appraise learner's dialogue in terms of accuracy**
  Hendrix 1.0 must be able to appraise and mark the statements made by a learner. When Hendrix 1.0 sets a challenge, the answers returned by the learner are marked to assess learner knowledge. Hendrix 1.0 is then able to provide guidance and feedback, follow-up questions and further examples and demonstrations to support understanding.

- **Tailor the specificity of guidance dialogue to the learner's demonstrated competence**
  This function allows Hendrix 1.0 to increase the degree of intervention in a learner's self-directed problem-solving, as demonstrable competence decreases. This aims to allow competent learners to develop self-directed problem-solving, while allowing less competent learners greater dependence on explicit guidance.

- **Provide feedback when an attempt at competence is made**
  Immediate, context specific, feedback allows Hendrix 1.0 to confirm or correct the learner's asserted understanding. This aims to increase engagement and motivation through positive feedback and prompt reflection through corrective feedback.

- **Record the non-verbal behaviour of learners during question-answer interactions**

  The system must record learners during their conversational question-answer interactions. The recordings, along with marks for the answers provided, will be used as a pilot data set for an experiment investigating comprehension classification from learner non-verbal behaviour.

7.3.3 Challenges

There are a number of technical challenges - requirements and limitations - which necessarily inform the design of the system. In this section each challenge is discussed and an approach defined so as to meet the functional requirements (section 7.3.2) of the system.

1. **Network**

   The software will need to record large volumes of image stream data from the web camera. It is not viable, due to upstream network bandwidth within the university, to stream the image data over the network to a server. This requirement rules out use of a client-server model architecture, such as a web application or thin-client. Image data will need to be securely stored locally for retrieval at a later point.

   The university network environment does allow machines on the same network to communicate with low latency downstream rates. As such, where possible, basic read-only data should be stored and maintained on a server. Domain knowledge for Hendrix 1.0 should reside in a centralised network-available database, rather than being distributed alongside the application within a local container.

2. **Software environment**

   To facilitate experimentation it is desirable to assume the software must function on PC terminals commonly available within the MMU computer network. Given the limitations of the network, the software must be a

locally executable program rather than a web application. As Microsoft Windows is the most common platform for PC terminals available to students, the software should be developed targeting the current network-wide installed version of the .NET framework.

As university PC terminals run restricted user accounts, the software developed must not require elevated permission to run or require installation of additional software or drivers. The software must not require any drivers which cannot be deployed as stand-alone DLLs. The software solution must be a portable application which can run from local user space without dependency on installed software beyond Microsoft Windows, .NET APIs or local DLLs.

3. **Data handling**

   The software will handle sensitive data which is personally identifiable of learners. In line with ethical practice, data from any use of the software should not be written to any public location. The software should write to an encrypted location on a portable storage device, such that information can be collected securely without writing data to the local drives of public MMU PC terminals.

4. **Peripherals**

   The software must work with low-cost and widely available plug-and-play camera hardware.

   The software should not depend on any body-attached sensors, eye-tracking glasses or other specialist peripheral hardware.

While the initial assumption had been made that the optimal design for this system would be a client-server web application, optimal for its inherent cross-platform and location independent availability, the technical requirements of the software and network environment dictate that the application must be a stand-alone portable executable. Due to limitations of network upstream bandwidth within the university it is not possible to store data in a centralised

repository. As such, data will need to be written locally. To ensure that personally identifiable and sensitive data is not written, unencrypted, to public machines, the application should run from and write to an encrypted portable storage device (i.e. an encrypted USB pen drive).

One of the novel challenges in developing this system is to perform analysis of non-verbal behaviour and comprehension classification from data gathered using consumer-grade camera peripherals. The software will capture live data from a Microsoft LifeCam USB web camera, which is a low cost web camera widely available through retail outlets. This technical requirement is driven in part by the usability benefit, compared to cumbersome and intrusive eye-tracking headsets or body-attached sensors, and partly by the cost benefit, compared to prohibitively high definition video cameras or specialist cameras (e.g. high speed eye-tracking or skeletal-tracking cameras).

While the technical limitations and requirements discussed have altered the assumed approach, the limitations highlighted represent real-world considerations for an e-learning platform deployed not in the lab but in a real life educational environment.

7.4 Hendrix 1.0 architecture and core components

Hendrix 1.0 is built around the four component models of a CITS described in Sani and Aris (2014): interface, tutor, learner and domain. In this section an overview of the Hendrix 1.0 architecture is provided along with discussion of the roles, duties and interactions of objects and services within each of the component models of a CITS. Each model in the architecture diagram (figure 7.1) is numbered, 1 to 4, enclosed by a dashed line and explained in the corresponding numbered list below.

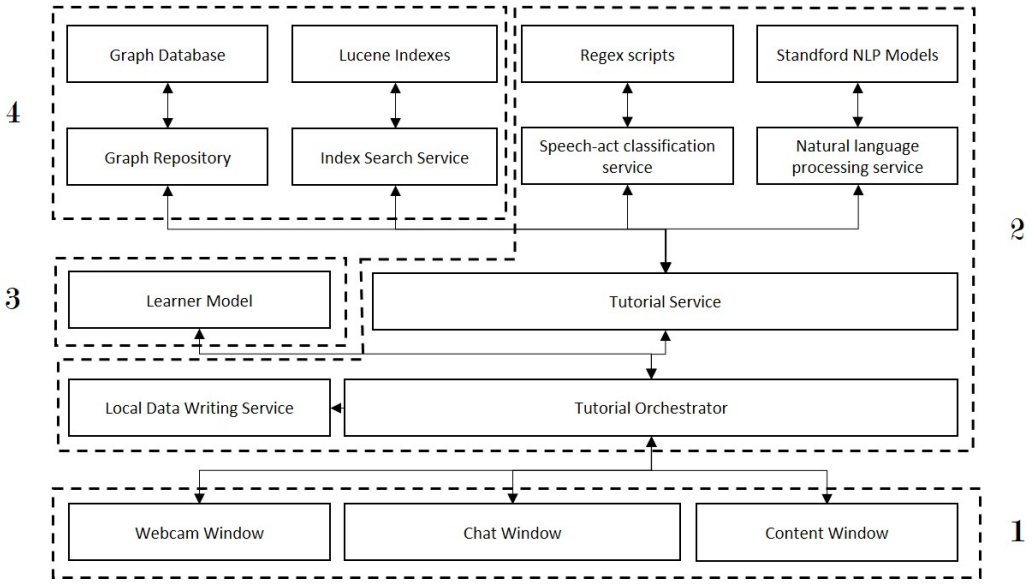


Figure 7.1 Hendrix 1.0 system architecture

1. **Interface**

   The interface model (figure 7.1:1) includes two graphical interface windows as shown in wire frame diagrams (section 7.4.1, figures 7.2a and 7.2b). As indicated in the architecture diagram, the *chat window* and *content window* objects interact with the *tutorial orchestrator*, displaying conversational and supporting content, and accepting free-text input from the user.

2. **Domain**

   The domain model (figure 7.1:4) contains the data layer, search and retrieval services utilised by the *tutorial service*. The domain model encapsulates access to an ontology of concepts, questions, answers, materials, definitions and examples, provided by a graph database, as well as free-text search and retrieval, provided by Lucene full-text indexes.

3. **Tutor**

   The tutor model (figure 7.1:2) encapsulates the natural language, reasoning and pedagogic algorithms which form the Hendrix 1.0 *agent*. The tutor controls the conversation, accepting new utterances from the learner and providing a set of methods for natural language processing functions,

speech-act classification and information search and graphing services. Hendrix 1.0 analyses discourse and pushes relevant conversational content, handouts, multimedia elements and code samples to the *content window*.

4. **Student**

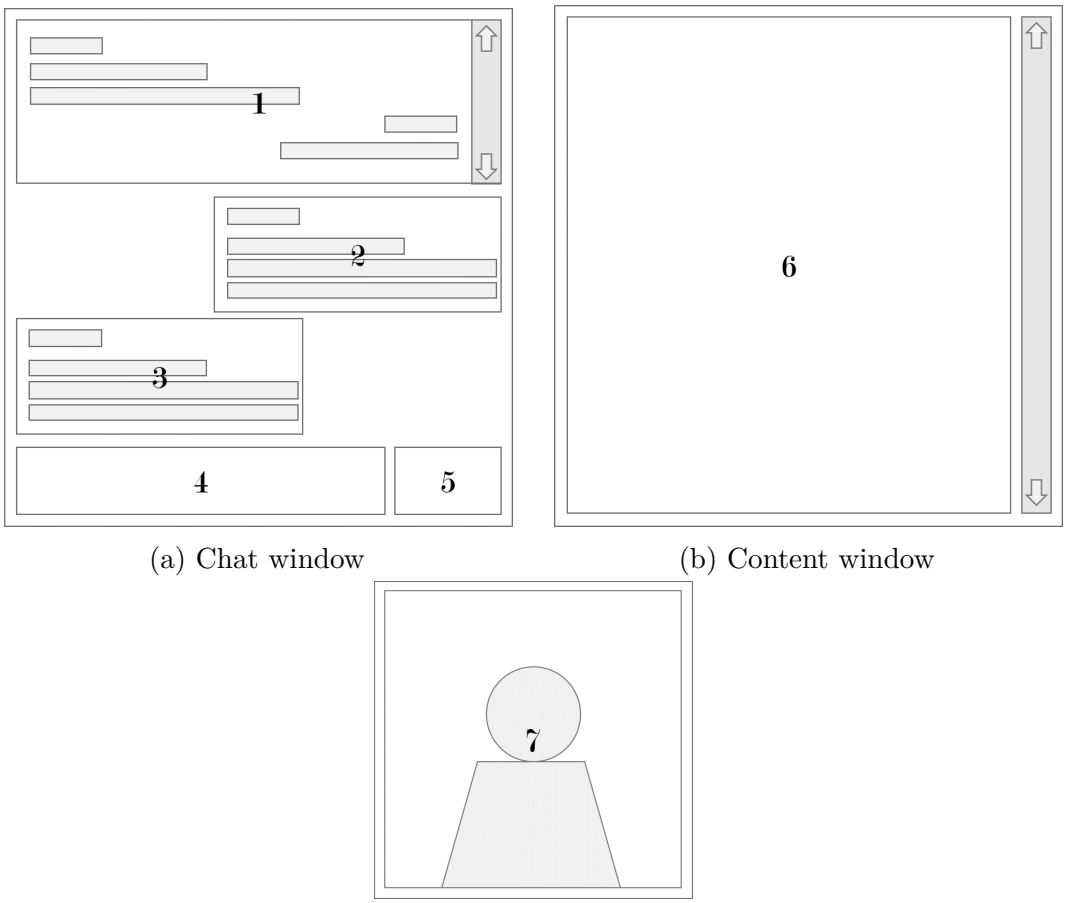
   The student model (figure 7.1:3) contains personal, demographic and performance information, such as the user's tutorial progress and answer scores. The student model is populated with demographic meta-data when a learner registers and logs into the system. Registration captures information including name, age, gender, ethnicity, academic level, whether the learner has programmed before, whether the learner is enrolled on a computing related degree course and whether the learner is wearing glasses. These learner attributes are captured as they may alter the patterns of non-verbal behaviour which are indicative of comprehension and non-comprehension states. The student model is then updated following each interaction with Hendrix 1.0 to track the learner's progress through the tutorial, answer scores and current conversational context.

7.4.1 Interface

Figures 7.2a and 7.2b show wireframe diagrams for the Hendrix 1.0 chat window (figure 7.2a) and supporting content window (figure 7.2b). Each region within the wire frame is numbered 1 to 6 and explained in the corresponding numbered list below.

1. **Chat history**

   Full history of the current tutorial conversation is available to the learner in the scrollable chat history panel (figure 7.2a:1). Full chat history is included to allow a user to review previous conversational interactions with Hendrix 1.0, review previous questions, answers, explanations and definitions, in support of their current learning objective.



(a) Chat window

(b) Content window

(c) Webcam window

Figure 7.2 Hendrix 1.0 wireframe diagrams

2. **Last message from learner**

   The last message from the learner is pulled out from the chat history to highlight the context of the current interaction. As such it is desirable to maintain a thread of conversational context by which each conversational interaction develops the content of the previous statement or question from the learner.

3. **Last response from Hendrix 1.0**

   The last response from Hendrix 1.0 is the current conversational entity to which the learner should respond. The last response from Hendrix 1.0 is given a prominent position directly above the text input field, to indicate that the learner should be replying in relation to this dialogue.

4. **Text input field**

   The text input field allows the learner to enter new dialogue to submit to Hendrix 1.0. The text input field is cleared when the dialogue is submitted.

5. **Submit button**

   The submit button sends the text input to Hendrix 1.0.

6. **Content window**

   The content window allows HTML pages to be loaded into the window. The content window is used to display 'hand-out' information sheets, multimedia such as videos and images and programming code excerpts.

7. **Webcam window**

   The webcam window captures and displays a live stream of image data from a USB webcam device mounted on top of the PC monitor. The webcam is pointed directly at the user such that the image displays the head and shoulders of the learner, with the face roughly centred, as they read from the Hendrix 1.0 chat and content windows and respond to questions.

7.4.2 Domain

The Hendrix 1.0 domain model implements expert knowledge as a graph of interconnected entities. Table 7.1 defines the types of knowledge entities in the graph. A full specification of Hendrix 1.0 pilot knowledge domain can be found in A.

A graph-based approach allows for novel use of *shortest-path* queries (Neo4J, b) to dynamically structure tutorial plans based on directional relationships between concepts. The directional relations between knowledge entities (figure 7.3) let Hendrix 1.0 know which concepts develop upon which, which definitions relate to which concepts and where a code snippet can demonstrate an idea or

can be used to examine a learner's understanding. Encoding knowledge in this way improves upon static Chatbot scripting, such as AIML and PatternScript, as it allows Hendrix 1.0 to plan ahead in the conversation based on the current context of the conversation without explicit instruction needing to be encoded.

Table 7.1 Domain entities in Hendrix 1.0 knowledge graph

Entity	**Description**
Concept	A *Concept* node defines a single concept or topic within the knowledge domain and contains a title, a canned introduction and a set of synonym words for search optimisation. Introduction text is displayed in conversational dialogue when a new concept is introduced
Hand-out	The *Hand-out* node contains canned information for the learner to read. The handout is displayed in the secondary content window
Example	The *Example* node contains canned examples of a concept. The example is displayed in the secondary content window
Definition	The *Definition* node contains a canned definition of a concept. The definition text is displayed in conversational dialogue
Question	The *Question* node contains question text, a set of required answer patterns, and an explanation of the solution. Questions have related Guidance nodes. The question and solution texts are displayed in conversational dialogue as part of the assessment and feedback interactions
Code	The *Code* node contains a link to a code excerpt for a question or guidance step. The code excerpt is displayed in the secondary content window
Guidance	The *Guidance* node contains a simplified sub-question text, a set of required answer patterns, and an explanation of how the sub-question relates to the *Question*. The sub-question and explanatory texts are displayed in conversational dialogue as part of the guidance interactions

The graph (figure 7.3) is populated with tutorial content, with emphasis placed on the relationships between concepts, examples, definitions, questions and applied programming problems (see section 7.4.2).

In addition to graph database, Lucene indexes are automatically populated from the graph. While the graph provides a structure for Hendrix 1.0 to follow, a framework for deploying content into the tutorial, full-text Lucene indexes allow for efficient full-text searching of content. The Lucene full-text

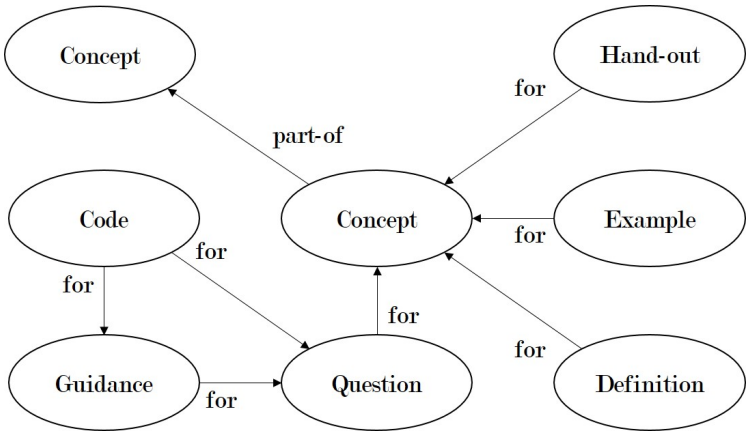


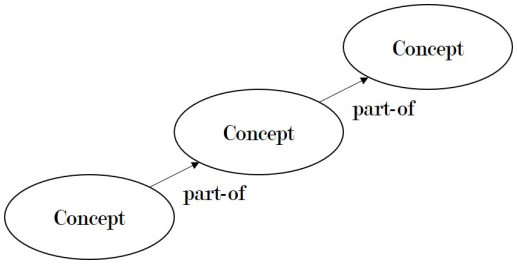
Figure 7.3 Knowledge graph structure

indexes allow for advanced searching and ranking of content by keywords using 'term-frequency - inverse document frequency' (TF-IDF) (Lucene) as a content relevance ranking metric.

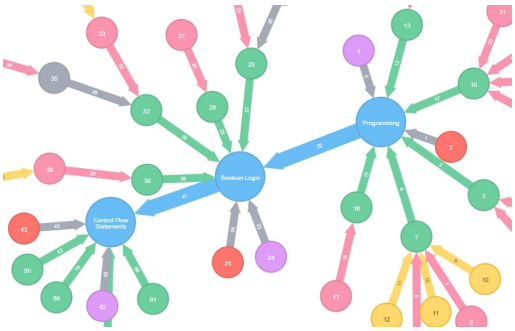
The structure of a tutorial

The graph is structured such that any given concept is a *part of* one or many parent concepts and acts as a hub for content entities including definitions, examples, questions and feedback guidance. The '*part of*' relationship in the graph structure shown in Figure 7.3 is used for structuring a conversational tutorial. The *part of* relationship gives a hierarchical structure to the concepts within the knowledge domain, allowing for reasoning of questions such as '*which concepts are part of X?*' (figure 7.4a) or a predicate assertion such as '*to understand X a learner must pass assessments on all part-of relations, the set of entities Y, for X*'.

Modelling the tutorial knowledge domain in this way represents the nature of knowledge sharing between a tutor and a student. The student has some learning objective within the knowledge domain they wish to master and Hendrix 1.0 has the entire knowledge domain with which it can reason, direct and support learning. However, if the ontology is truly to represent a human tutor's knowledge domain it must encode not just conceptual information but materials to support learning.



(a) Concept path



(b) Implementation of concept path in Neo4J

Figure 7.4 Hendrix 1.0 wireframe diagrams

The proposed graph structure (section 7.4.2) has many of the desired characteristics of a cognitivist learning environment (section 2.5), such as a hierarchy of concepts with associations and relationships. Further, the graph contains nodes for definitions, demonstrations, questions, guidance and examples of applied concepts.

Figure 7.5 shows the process of deploying knowledge in a tutorial. The cognitivist philosophy (section 2.2.1) requires that assessment is followed by corrective feedback. The Hendrix 1.0 knowledge graph contains pre-canned text for a correct answer, incorrect answer and partially correct answer. In addition, the knowledge graph contains *guidance* steps associated with questions. Guidance steps are follow-up questions that can be deployed into conversation to support the development of understanding when a learner does not provide a complete or fully correct answer to the question. Guidance steps allow the learner to explore the concept in a structured way with additional prompts from the tutor.

Figure 7.6 shows a question with two guidance entities associated. Relationships within the graph can be assigned attributes. The directional relationship between *Guidance* and *Question* entities are assigned a numeric attribute to indicate precedence. Precedence is important in relation to guidance entities as these can be seen as *steps* towards a complete solution.

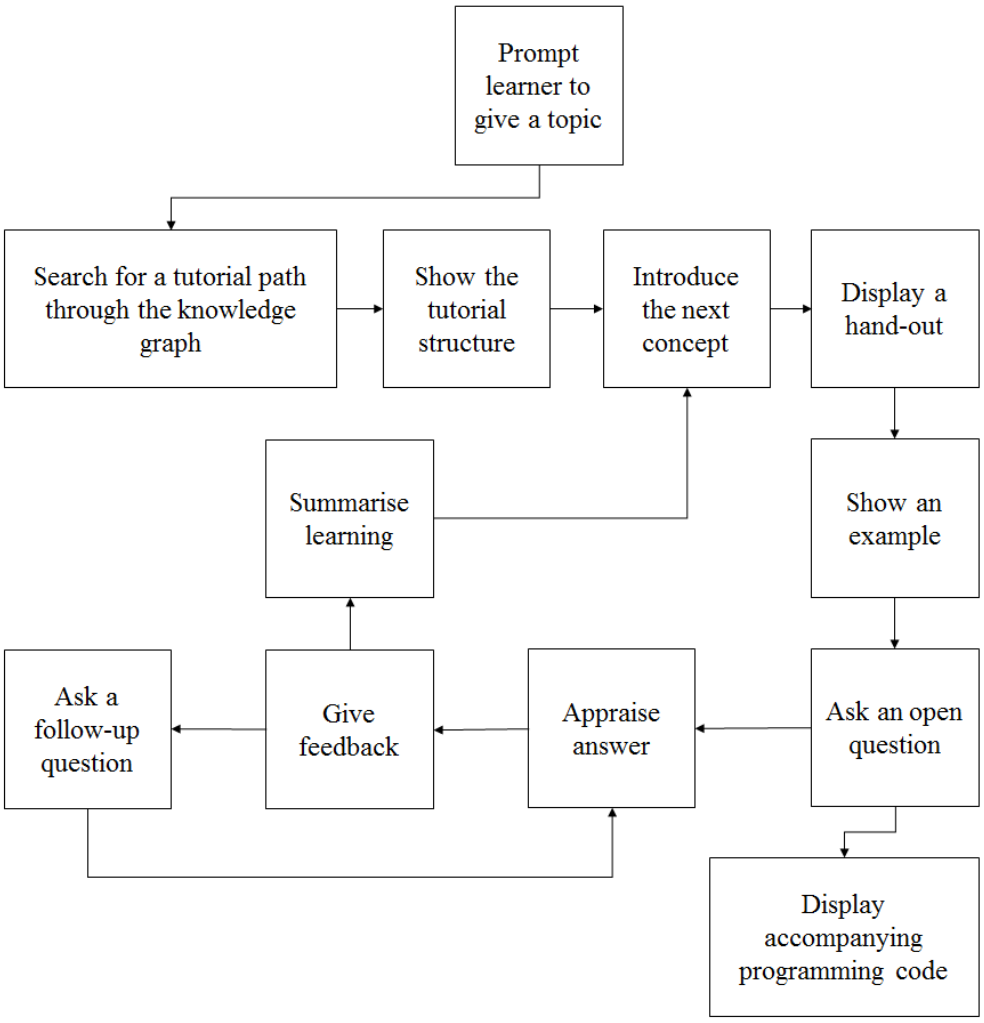


Figure 7.5 Hendrix 1.0 tutorial process

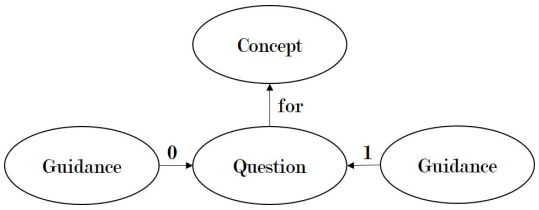


Figure 7.6 Hendrix 1.0 ontology of relationship between concept, question and guidance steps with order attributes

The design of the knowledge domain requires that relationships have properties on which to predicate the most accurate entity selection. As a basic example, adding an integer attribute to the connecting relationship would allow the graph structure to represent the correct order of guidance steps, in a linear feedback construction process, expressing that one guidance step must come before another.

7.4.3 Tutor

The Hendrix 1.0 tutor model encapsulated the *intelligent* behaviour of the system. The Hendrix 1.0 tutor model contains algorithms for conversational interaction, reasoning and assessment of understanding. The intelligent agent has four primary functions:

1. Structure a tutorial for a given learning objective
2. Identify acts of speech in dialogue and model the conversation
3. Ask questions, mark answers and give guidance
4. Answer questions from learners

Planning tuition

Hendrix 1.0 implements a two loop algorithm (Vanlehn, 2006). The outer loop provides the macro-level goal orientation for the conversation – the learning objective for the tutorial – while the inner loop allows for micro-level transactions – the conversational moves which support goal attainment. Algorithm 1 shows how a stated learning objective is used to create and populate a tutorial content structure, a learning scaffold, dynamically from the knowledge graph. Algorithm 1 lines 2 - 5 show the process by which a tutorial path is found by keyword relevance.

As shown in Algorithm 1, when presented with a new learning objective, Hendrix 1.0 first locates a matching node within the graph. Hendrix 1.0 constructs the tutorial macro structure dynamically by finding the shortest path between a root starting node and the destination node((Neo4J, b). Hendrix 1.0 then walks the tutorial concept path, pulling in handouts, definitions, demonstrations, questions, code and guidance by searching for inward relations on each concept and question node.

Algorithm 1: Create tutorial from graph database

```
Data: input utterance from learner S
Result: tutorial
convert S to lower-case as l(S);
parse l(S) for keywords as k(l(S));
perform full text search on indexes for nodes matching k(l(S)) as
  candidates;
select top node in candidates by rank order by tf − idf score as O;
create tutorial T by shortest path graph search as
  shortest_path(root, O);
for each concept as C in tutorial T do
    C[handouts] = search graph (handout) − [for]− > C;
    C[definitions] = search graph (Definition) − [for]− > C;
    C[demonstrations] = search graph (Demonstration) − [for]− > C;
    C[questions] = search graph (Question) − [for]− > C;
    for each question as Q in tutorial C[questions] do
        Q[code] = search graph (Code) − [for]− > Q;
        Q[guidance] = search graph (Guidance) − [for]− > Q;
        for each guidance as G in tutorial Q[guidance] do
            G[code] = search graph (Code) − [for]− > G;
        end
    end
end
return populated tutorial structure T;
```

When Hendrix 1.0 processes a node, the node type informs Hendrix 1.0 what type of event to raise in conversation, for example, asking the learner for confirmation before showing an example or presenting a question to the learner.

Figure 7.7 shows an example dialogue in which Hendrix 1.0 introduces a tutorial plan.

Speech act classification and conversation modelling

Hendrix 1.0 models the conversation as a process of speech act (Searle, 1969) fulfilment. A speech act could entail asking a question, expressing an opinion for scrutiny, giving a confirmation or any other *action* within the conversation that elicits a force upon the response of the other actors. Speech acts are *fulfilled* by a closing response. Table 7.2 shows the 7 speech acts Hendrix 1.0 can use in conversation.

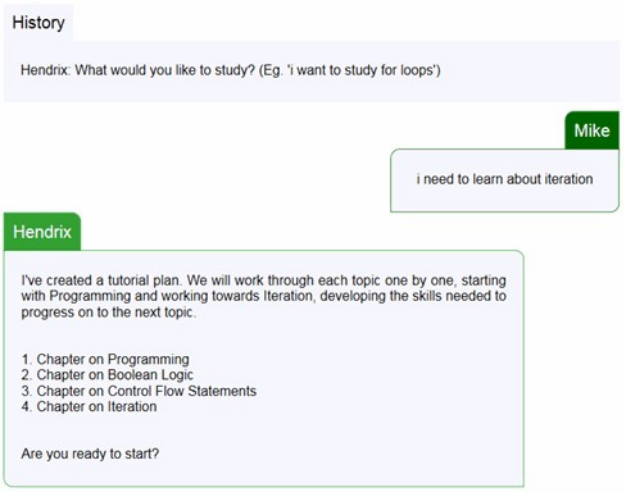


Figure 7.7 Example dialogue demonstrating Hendrix 1.0 constructing a tutorial path

Table 7.2 Overview of formative tutorial question data

Speech Act	**Description**
Confirmation	Binary agreement or disagreement statement
Request Definition	Learner is requesting the definition of a concept
Request Demonstration	Learner is requesting the demonstration of a concept
Smalltalk	Learner is moving off-topic
Abuse	Learner is using abusive or offensive language
Answer*	Learner is answering a question
Unknown	If no speech act can be identified, the *Unknown* act is assigned

** There are no patterns for this speech act type but it can be identified by a combination of the presence of other speech acts and context in relation to previous speech acts.*

Hendrix 1.0 keeps track of the order of speech acts by pushing new speech acts to the head of a *first in, last out* (FILO) stack. A new speech act is placed at the head of the stack and is the focus of fulfilment attempts until either a response satisfies the conditions of the speech act, removing it from the stack, or a new speech act is placed on top, relegating the act to a lower priority within the conversation.

The conversation is turn based and either actor, Hendrix 1.0 or learner, can raise new speech acts at any time. The conversation continues until

there are no remaining speech acts on the stack. Hendrix 1.0 raises speech acts to the head of the stack when a novel concept, question, example or demonstration is encountered in the tutorial path (section 7.4.3). Working in this way ensures that the conversation can be free-flowing, with either party raising new contextually relevant speech acts to the head of the conversation. In doing so, either party can temporarily divert the conversation to explore information and knowledge. Maintenance of the FILO stack ensures that all speech acts are ultimately fulfilled. When a temporary conversational diversion runs out of new speech acts, Hendrix 1.0 will simply return to the last most recent unfulfilled act within the conversation.

Algorithm 2, part of the Speech Act Classification Service (figure 7.1:2), shows how Hendrix 1.0 recognises speech acts in dialogue with the learner. Algorithm 2 lines 3 to 9 show how the input dialogue text is parsed against regular expression patterns for each speech act class. Line 13 shows how the weighting for each class, a count of pattern matches, is used to rank the classes. The match with the highest weighted score is returned.

Algorithm 2: Speech Act Classification by Weighted Matching

Data: input utterance from learner S
Result: speech act type
1 convert S to lower-case as $l(S)$;
2 create set of Speech Acts as SA containing each speech act type;
3 **for** *each speech act class type as c in* SA **do**
4 | load speech act class regular expressions for c as $P(c)$;
5 | set bias weight $SA[c][bias]$ between 1 and 2;
6 | **for** *each pattern as p in set* $P(c)$ **do**
7 | | $SA[c][weight]$ += match ($l(S)$,p) ? 1 : 0;
8 | **end**
9 **end**
10 **if** *current context is a question* **then**
11 | $SA[Answer][weight]$ += 1;
12 **end**
13 **return** $SA[c]$ where ($SA[c][weight] * SA[c][bias]$) is highest;

Defining separate functions in accordance with intent, or act of speech, allows Hendrix 1.0 to decouple function from content. Unlike conventional

scripted conversational agents (ConvAgent) and CITS (Latham et al., 2014), Hendrix 1.0 chooses *how* to use content based on the conversational context. The result is to simplify the tutorial scripting process and remove all conditional logic from the content layer.

Hendrix 1.0 is able to use task-specific logic to formulate a response to a known speech act type. Unlike conventional scripted conversational agents (ConvAgent) and CITS (Latham et al., 2014), which simply load text from a script, Hendrix 1.0 can have different algorithms to respond to different types of conversational contexts. For example, Hendrix 1.0 can perform a range of complex tasks, such as searching text indexes, retrieving multi-media and marking answers.

While the approach is currently limited to six speech acts, the framework is extensible. New speech acts requiring different functionality can be integrated without needing to change the underlying tutorial content. For example, it would be possible to create a new *find* speech act for use in requests such as '*find me a web page for x*', which could integrate a search API to find web pages relevant to *x*.

Ask questions, mark answers and give feedback guidance

Marking learners' answers during conversational tutoring is one of the most important tasks for Hendrix 1.0. The domain knowledge for the tutorial contains questions clustered around concepts. Hendrix 1.0 asks the learner each question and parses the answer for correctness. If the answer is correct, Hendrix 1.0 gives positive feedback and an explanation of the answer. If the answer is incorrect, Hendrix 1.0 selects any guidance questions which are related to the question (figure 7.4a) and guides the learner through those sub-questions. The process is intended to reflect the pedagogy of a human tutor using a scaffold learning framework.

Hendrix 1.0 marks answers using a set of pre-defined syntactic patterns (algorithm 3). Each question node contains a set of syntactic answer patterns encoded as regular expressions, along with a minimum number of matches required to assert correctness.

Algorithm 3: Marking learner's answers

```
   Data: text input from learner as S
 1 set lower-case S as l(S);
 2 classify_speech_act(l(S)) as SA;
 3 if SA is Answer and current event is Question then
 4 |   fetch required answer patterns as P from current_question[patterns];
 5 |   for each pattern as p in pattern set P do
 6 |   |   correct_matches += match(l(S), p) ? 1 : 0;
 7 |   end
 8 |   set score equals correct_matches/length(P);
 9 |   if score is 1 then
10 |   |   display canned positive feedback response;
11 |   |   display solution from current_question[solution];
12 |   else
13 |   |   if score is greater than 0 then
14 |   |   |   display canned partial positive feedback response;
15 |   |   else
16 |   |   |   display canned incorrect answer feedback response;
17 |   |   end
18 |   |   fetch guidance step from current_question[guidance];
19 |   |   set new event Guidance;
20 |   end
21 end
```

As shown in algorithm 3, when a response utterance is classified as an answer, the response text is parsed against a set of required answer patterns defined on the current question. The number of correct answer pattern matches is counted and a score is devised by dividing the match count by the pattern set length.

If the answer utterance does not meet the required number of matches, guidance steps can then be deployed into the conversation to fill the gaps in knowledge where patterns were not matched. Guidance steps contain a sub-question for the learner to answer, prompting the learner to consider a specific detail of the overall question. Figure 7.8 shows an example dialogue

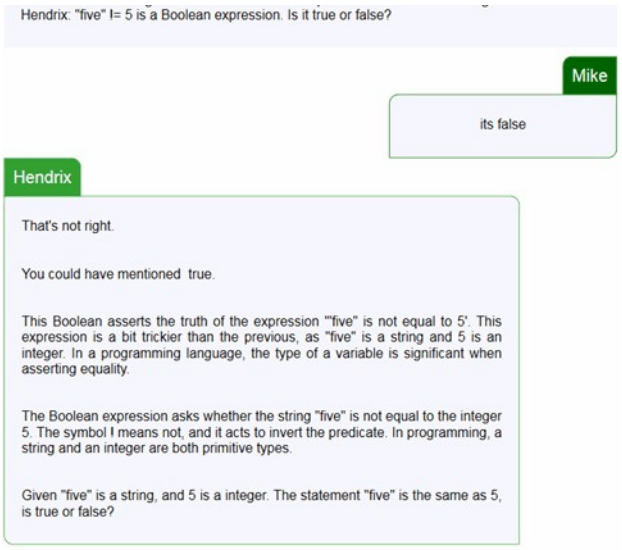


Figure 7.8 Example dialogue showing a conversation in which Hendrix 1.0 asks a question, marks the answer and provides feedback

where a question is posed, marked and feedback is given. Guidance step answers are marked using the same pattern matching algorithm as Answers (algorithm 3).

Answering learners' questions

Similar to the (Figa and Tarau, 2004) conversational agent, Hendrix 1.0 can answer questions from learners on any topic which is contained within the knowledge domain. Using the event classification algorithm, Hendrix 1.0 is able to recognise requests for both explanations and demonstrations. When Hendrix 1.0 identifies a request for information, either explanation or demonstration, Hendrix 1.0 parses the utterance for subject keywords and searches for the best matching knowledge domain entity. Algorithm 4 shows how a user's input text is parsed to identify a request type and the knowledge graph is searched to provide a definition or demonstration of a concept.

Hendrix 1.0 uses a Lucene index (Lucene; Neo4J, a) to allow for rapid searching. The entities contained within the knowledge domain graph are automatically indexed into the TF-IDF (Lucene) based index. The index matches entities based on the rarity of the keyword match over the entire

Algorithm 4: Algorithm for providing definitions and demonstrations of concepts

```
   Data: text input from learner as S
 1 set lower-case S as l(S);
 2 classify_speech_act(l(S)) as SA;
 3 if SA is Definition or Demonstration then
 4 |   fetch keywords k(l(S)) from l(S) as set K;
 5 |   search full-text indexes for Concept nodes matching words in K as set
   |    candidates;
 6 |   if any candidates then
 7 |   |   select top 1 candidates by rank order according to tf − idf score
   |   |    as C;
 8 |   |   if SA is Definition then
 9 |   |   |   fetch d from graph search d(Definition) − [for]− > C;
10 |   |   |   display d[text];
11 |   |   end
12 |   |   if SA is Demonstration then
13 |   |   |   fetch e from graph search e(Example) − [for]− > C;
14 |   |   |   display e[text];
15 |   |   end
16 |   else
17 |   |   display not found text;
18 |   end
19 end
```

knowledge domain. Using this approach ensures words that occur frequently are de-prioritised. De-prioritising frequent words is important as they contain the least information and are least likely to yield a valuable match. The method also naturally favours matches for noun phrases over individual words, as they have greater specificity and consequently greater rarity.

The Hendrix 1.0 knowledge domain is designed to support semantic search by the inclusion of a synonym field on definition entities. Adding synonyms to the definitions allows for a broader range of phraseology to be used in conversation with Hendrix 1.0, while still being able to match and return content that is relevant. Once a match is found the index returns to Hendrix 1.0 the Node ID for the knowledge domain entity within the graph and Hendrix 1.0 is able to select a definition or a demonstration, using the relations encoded within the graph.

Using speech act classification Hendrix 1.0 is able to detect implicit and explicit requests for information. Figure 7.9 shows an example dialogue where Hendrix 1.0 recognises the learner's implicit request for definition and provides an answer. In the example, TF-IDF (Lucene) correctly matches the incomplete search term 'markup' to the node for 'Markup Languages'.

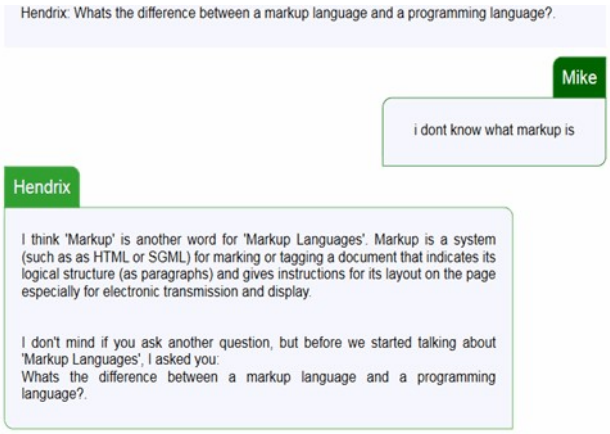


Figure 7.9 Example dialogue demonstrating Hendrix 1.0 identifying and responding to a request for definition

7.4.4 Student

The Hendrix 1.0 student model keeps track of a learner's conversation, tutorial position and progress. The student model is used to store attributes of the student including name, gender and ethnicity of the learner, tutorial conversation dialogue, tutorial progress and answer scores.

Hendrix 1.0 has been designed to gather visual behavioural data during tutorials. Hendrix 1.0 uses a web camera (figure 7.2c) to record learners during each question-answer interaction. As shown in algorithm 5, Hendrix 1.0 saves the image data for each answer period on disk, in a folder structure mapped to the question-answer. As a step towards the overall research objective of developing a comprehension-responsive CITS, during the pilot study of Hendrix 1.0 image data and answer scores will be collected to form a pilot data set for analysis of learner non-verbal behaviour during on-screen learning.

Algorithm 5: Pseudo-code algorithm for saving sequential image sets against learner answer response periods

```
1  create Dictionary as D for holding bitmaps for questions;
2  while web camera is on do
3      if current event is Question then
4          if D contains key current_question[id] then
5              get bitmap from webcamera stream as b;
6              add b to L where L[key] equals current_question[id];
7          else
8              fetch list of folder names from image storage location as dir;
9              if not all composite keys session_id + L[key] in dir list then
10                 create folder in image storage location as session_id +
                    L[key];
11                 for each bitmap b in L[key] values do
12                     save b in session_id + L[key] as index of b in L[key]
                        values;
13                 end
14             end
15             add new tuple (current_question[id], bitmap container)
16         end
17     end
18 end
```

7.5 Conclusion

This chapter has presented an overview of the Hendrix 1.0 CITS (section 7.3), detailing the novelties, challenges and requirements of the system. Architecture has been presented (section 7.4) and domain knowledge structures detailed (section 7.4.2). Key functions of the software have been highlighted, discussed and presented in pseudo-code algorithms (sections 19, 7.4.2 and 7.4.3).

The contribution of research engineering work discussed in this chapter, is Hendrix 1.0's ability to structure tutorial plans dynamically using shortest path queries over a directional graph of concepts, reducing the overhead of pre-planning and encoding specific tutorial progressions in static scripting files, and its ability to parse discursive, mathematical and programmatic content in conversations.

The contribution of research discussed in this chapter is to implement the micro-adaptive behaviours discussed in section 4.8.2, producing a system with

more micro-adaptive behaviours than previous CITS discussed in literature VanLehn et al. (2017).

Hendrix 1.0 micro-adaptations include hints and feedback on solutions, hints and feedback on approach to problem-solving, decomposition of tasks and answering students' questions.

The system will be used to evaluate empirically whether a CITS can be used to tutor computer programming effectively using natural language interactions and whether learners can develop knowledge through directed tuition. The study procedure, method and results are detailed in Chapter 8.

Chapter 8

Study: Evaluation of Hendrix 1.0 CITS

8.1 Introduction

This chapter presents a pilot study carried out to evaluate the conversational and educational ability of Hendrix 1.0 in a real-world learning environment, a university computer lab, with enrolled undergraduate and post-graduate students.

In the pilot study 15 students from Manchester Metropolitan University undertook a tutorial on Java programming with Hendrix 1.0. Hendrix 1.0 is evaluated using a combination of statistical performance measures, accuracy and error rates and user feedback on coherence of discourse, perception of benefit and usability. In the study, Hendrix 1.0 classified correctly the utterance type of 91% of input sentences, marked 94.5% of question answers correctly and was rated 4 out of 5 for user satisfaction.

Section 8.2 presents the research questions for the study. An overview of the study is presented in section 8.5, with method and results for each research question detailed in sections 8.7.1 and 8.8. A discussion of the pilot study results is presented in section 8.9 and conclusions are detailed in section 8.10.

8.2 Research questions

The study presented in this section evaluates the effectiveness of Hendrix 1.0 CITS in terms of conversational functionality and ability to teach.

1. Does Hendrix 1.0 converse effectively?

2. Does Hendrix 1.0 facilitate learning?

Figures 8.1 and 8.2 (see sections 8.7.1 and 8.8.1) define the GQM (see section 4.9) for evaluation of Hendrix 1.0, integrating both objective and subjective measures.

8.3 Contribution

The contribution of this research is to evaluate whether an intelligent CITS designed to deliver syllabus material through discourse using adaptive and micro-adaptive behaviours (see literature in section 4.8) is successful in conversing to facilitate learning via computer mediated interaction.

The research in this chapter explores not only whether the technology is reliable and accurate in conversation, but also how learners feel about interacting with and learning from the virtual agent. The outcome of this experiment, performance metrics and user feedback, will inform the development of a second prototype system.

8.4 Tutorial content

The tutorial focused on concepts and applied skills relating to a basic iterative '*for*' loop. The 'For' loop was chosen as a tutorial topic based on the first year computing syllabus, while tutorial dialogue (hints, feedback and tone) were derived by observation of first semester classroom tutorials at Manchester Metropolitan University in which tutors and students iterated a topic through

question and answer group discussion. The tutorial content for the pilot study is detailed in full in Appendix A.

8.5 Study overview

In this study 15 students from Manchester Metropolitan University volunteered to complete a tutorial on the construction and application of 'For' loops. The experiment consisted of four steps:–

1. Participants completed a 10 question MCQ on Java programming;

2. Participants were instructed to take a tutorial on 'For' loops using the Hendrix 1.0 CITS, during which metrics for evaluation (figure 8.2) of Hendrix 1.0 were recorded in log files;

3. Participants repeated the 10 question MCQ on Java programming from *step 1*;

4. Participants completed a user experience survey to rate Hendrix 1.0 performance and provide feedback on their experience of using the system.

8.5.1 Ethics

Participation was voluntary and not compensated for participation. Participants were required to sign a consent form prior to participating in the experience. The consent form detailed the data collected during the experiment, how that data would be analysed and the intention to distribute or publish data from the experiment. Data collected during the experiment was not anonymous and as such personally identifiable information such as names, email addresses and demographic information has been stored on a physically secured computer with data fully encrypted. All information used for redistribution or publication purposes is anonymised or aggregated so as not to be disclosive.

8.6 Participant information

The participant group consisted of 15 adult student volunteers from the School of Computing, Mathematics and Digital technology at Manchester Metropolitan University. The participant group consisted of 11 male and 4 female participants, with 7 of participants studying at undergraduate level and 8 studying at postgraduate level.

8.7 Does the intelligent conversational agent converse effectively?

Figure 8.1 shows the four questions that will be used to evaluate whether the conversational intelligent tutoring system, Hendrix 1.0, can converse effectively. In this section the method, results, discussion and conclusion are presented for each question.

8.7.1 Method

Four questions have been identified using the GQM method (see section 4.9). The goal, questions and metrics are shown in figure 8.1.

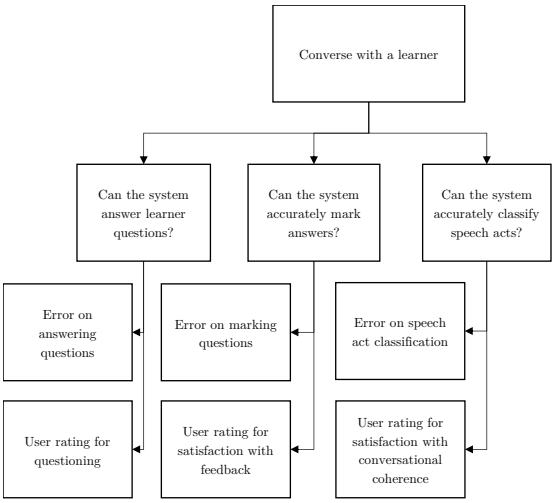


Figure 8.1 GQM for evaluation of conversational function

System errors are calculated by human analysis of the conversational log files. Each response from Hendrix 1.0 has been labelled as correct or incorrect given the speech act (section 4.7) and information content. The errors in performing specific tasks are used to analyse the performance of Hendrix 1.0.

1. Error answering a learner's question. Three types of error are scrutinised: 1) the utterance was not a question; 2) no answer was found for question; 3) an incorrect answer was provided.

2. Error marking a learner's answer. Three types of error are scrutinised: 1) Fully correct and marked as either partially correct or incorrect; 2) Partially correct and marked as either fully correct or incorrect; 3) Incorrect and marked as either partially or fully correct.

3. Error classifying an act of speech. One type of error is scrutinised: 1) Incorrect speech act identified.

Subjective metrics are assessed based on participant feedback, provided via a satisfaction survey. Each participant was asked to complete a user feedback survey after completing the tutorial and post-tutorial multiple-choice quiz. Participants were asked to show agreement with statements using a 5 point Likert scale. Table 8.1 shows the feedback statements relevant to the GQM as defined in section 8.7.1.

Table 8.1 User feedback questions

Question id	**Statement text**
1	I was able to get relevant answers to questions I asked
2	I would use this software to learn
3	The conversational feedback the tutor gave me helped me to learn
4	The conversation I had with the tutor was coherent
5	Overall I am satisfied with learning from this system

Each metric for each question will be analysed to give insight into the overall success of the system in relation to the stated goal. Where appropriate,

a one- or two- tail T test will be used to evaluate whether mean statistics are significantly different between groups.

8.7.2 Can the system answer learner questions?

To evaluate the accuracy of Hendrix 1.0 in answering questions from learners, a benchmark of accuracy has been found in Smith et al. (2014). The study reports that SEEKER, a natural language system database interface for goal-oriented conversation, had a 12% error rate on answering user queries from the database. When Hendrix 1.0 answers questions, there is a similar information retrieval problem. In this experiment Hendrix 1.0 is benchmarked against the 12% error rate found in Smith et al. (2014).

Log files contained a total of 72 participant utterances that Hendrix 1.0 had classified as the 'Learner Question' speech act. 54 classifications were correctly identified queries and 18 were misclassified non-query speech acts. For the 18 misclassified speech acts, Hendrix 1.0 could not answer the query as the utterance content was not a question. For the purposes of evaluating question answering, the misclassified utterances were excluded. Speech act misclassification errors are analysed in section 8.7.4.

Hypothesis

- H0: The error rate on responding to learners' questions is $> 12\%$ or mode average user satisfaction with asking questions is $\leq 3/5$
- H1: The error rate on responding to learners' questions $\leq 12\%$ and mode average user satisfaction with asking questions is $\geq 4/5$

Results and discussion

Table 8.2 shows a breakdown of the errors in correctly classified learner questions for each participant, along with the satisfaction score given for the statement 'I was able to get relevant answers to questions I asked'.

Table 8.2 Errors and satisfaction scores for learners asking questions

Partici-pant	**Number of questions**	**Error Count**	**Satisfaction Score**
1	8	3	2
2	7	0	5
3	2	2	3
4	0	0	3
5	4	0	4
6	4	1	4
7	4	2	3
8	7	3	4
9	3	0	3
10	1	0	4
11	2	0	4
12	5	1	5
13	2	1	2
14	1	0	4
15	7	1	3
Totals	57	14	
Mode			4

The results in Table 8.2 show that a total of 14 errors occurred leading to incorrect responses being given for 24.56% of learner questions.

While user satisfaction does meet the lower limit of 4 out of 5 support for the 'I was able to get relevant answers to questions I asked' the error rate is almost twice the benchmark target for accuracy. The disparity between high user satisfaction and high error appears due to the distribution of errors among participants. The mode user satisfaction for learners who did not experience any errors is 4, while the mode for learners who did experience errors is 3.

Given the results in Table 8.2 the null hypothesis cannot be rejected.

8.7.3 Can the system accurately mark learners' answers provided in discourse?

From the 15 participants a total of 494 question answers were given. Hendrix 1.0 marked each answer automatically in accordance with the marking schema defined within the knowledge graph. To evaluate marking accuracy, Hendrix

1.0 is benchmarked against the error rate of human tutorial marking reported in Mitchell et al. (2003). The study reports that human tutorial answer-marking has an error of 5.5%.

Hypothesis

- H0: The error rate on marking learners' answers is $\geq 5.5\%$ or mode user satisfaction with answer marking $\leq 3/5$
- H1: The error rate on marking learners' answers is $< 5.5\%$ and mode user satisfaction with answer marking $\geq 4/5$

Results and discussion

Table 8.3 shows the error count and error rate for marking. The results show that Hendrix 1.0 does not exceed the marking error rate of human tutors as reported in Mitchell et al. (2003) and exceeds the accuracy of computer-based marking for free text answers reported in Mitchell et al. (2002).

Table 8.3 Error rate for marking answers

Count of answers	**Count of errors**	**Error rate (%)**
494	27	5.5

Table 8.4 shows the marking errors and satisfaction scores for questions 2 - 5 (table 8.1 in the section 8.7.1). As marking of answers is fundamental to the structure and progression of the tutor-led conversation, perception of the answer marking performance of Hendrix 1.0 is exposed in satisfaction levels for ability to learn using the system (Q2), accuracy of tutor feedback (Q3), conversational coherence (Q4) and overall usability of the system (Q5).

As the error rate for marking is equal to the human marking benchmark (5.5%) and reported mode user satisfaction for relevant qualitative user experience questions are 4 (agree) then the null hypothesis is rejected. The results in

Table 8.4 Error rate and satisfaction scores for marking answers

Participant	**Number of errors**	**Q2**	**Q3**	**Q4**	**Q5**
1	1	4	4	4	4
2	2	2	4	4	4
3	1	4	5	4	5
4	0	4	4	4	4
5	2	4	4	4	4
6	5	4	5	4	4
7	2	4	3	4	4
8	0	3	4	3	4
9	2	5	5	4	4
10	8	2	3	2	4
11	1	4	4	2	4
12	0	4	4	5	5
13	1	4	3	4	4
14	0	4	4	4	4
15	2	4	3	3	4
Mode	**1.8**	4	4	4	4

Tables 8.3 and 8.4 suggest that Hendrix 1.0 is able to mark tutorial answers accurately according to the marking schema.

8.7.4 Can the system accurately classify acts of speech?

To evaluate the accuracy of Hendrix 1.0 in classifying speech acts, a benchmark of accuracy has been found in Smith et al. (2014). The study reports that SEEKER, a natural language system database interface for goal-oriented conversation, gave an incorrect response to the conversational move 14% of time. Hendrix 1.0 decides the correct response by a process of speech act classification detailed in section 19. In this experiment Hendrix 1.0 is benchmarked against the 14% error rate reported in Smith et al. (2014).

To evaluate user satisfaction with speech act classification, participants were asked to rank their agreement with Q4 (table 8.1), 'The conversation I had with the tutor was coherent'. Table 8.6 shows the participants' responses, where 1 is strongly disagree and 5 is strongly agree.

Hypothesis

- H0: The error in classifying speech acts is $> 14\%$ or mode user satisfaction with conversational coherence $\leq 3/5$
- H1: The error rate on classifying speech acts is $\leq 14\%$ and mode user satisfaction with conversational coherence $\geq 4/5$

Results and discussion

A total of 1003 utterances were provided by the 15 participants. Table 8.5 shows the error count and error rate for marking.

Table 8.5 Error rate for marking answers

Count of utterances	Count of errors	Error rate (%)
1003	92	9.2

Table 8.5 shows that the error rate in classifying speech acts was 9.2%. The results show that Hendrix 1.0 also performs favourably when compared to examples of free text speech act classification found in literature Kang et al. (2010); Moldovan et al. (2011); Smith et al. (2014).

Table 8.6 shows the user satisfaction scores for conversational coherence. The mode user satisfaction for coherence is 4, indicating satisfaction with speech act classification.

Given the results in Tables 8.5 and 8.6 the null hypothesis is rejected.

However, the chat logs indicate that some problems occurred more frequently than others. An example is that of identifying questions. Statements were misclassified as questions as a result of learners using ambiguously interrogative words. For example, in the statement 'x *where* y is equal to z' the presence of keywords *where* and *is* within the sentence were incorrectly taken to denote that the learner was asking a question (e.g. '*where is* x?'). The errors made in classifying interrogative utterances highlight the issue that parsing individual words or simple word combinations is inadequate for requisite specificity. The

Table 8.6 Satisfaction with conversational coherence

Participant	**Q4 satisfaction**
1	4
2	4
3	4
4	4
5	4
6	4
7	4
8	3
9	4
10	2
11	2
12	5
13	4
14	4
15	3
Minimum	2
Maximum	5
Mode	4

use of more complex regular expressions, which allow for a greater depth of syntactic and semantic definition, would eliminate these errors.

8.8 Does the CITS facilitate learning?

Figure 8.2 shows the GQM which will be used to evaluate whether the CITS, Hendrix 1.0, facilitates learning. In this section the method, results, discussion and conclusion are presented for each question.

8.8.1 Method

To investigate whether the intelligent conversational agent can facilitate learning, two questions have been identified using the GQM method (see section 4.9). The goal, questions and metrics are shown in figure 8.2.

Learning gain is calculated using the equation from Colt et al. (2011) detailed in equation 2.2.

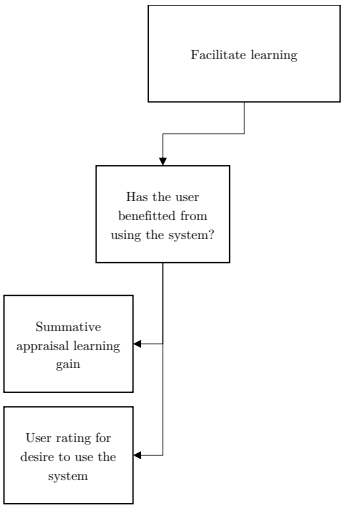


Figure 8.2 GQM for evaluation of learning facilitation

User satisfaction is assessed from surveyed participant feedback. Each satisfaction metric is measured using a Likert-like scale between 1 and 5, where 1 is strongly disagree and 5 is strongly agree.

8.8.2 Has the user benefitted from using the system?

Hypothesis

Learning gain is measured in this study by subtracting post-tutorial test scores from pre-tutorial test scores. The increase in test performance is the learning gain.

- H0: Learning gain (eq 2.2) is $\leq 0\%$ or mode user desire to learn from the system $\leq 3/5$
- Learning gain (eq 2.2) is $\geq 0\%$ and mode user desire to learn from the system $\geq 3/5$

Results and discussion

The results in Table 8.7 show the overall learning gain within the participant group.

Table 8.7 Learning gain

	Population	**Standard Deviation**	**Mean MCQ Score (%)**
Pre-tutorial	15	2.22	68.00
Post-tutorial	15	1.92	75.33
Increase (%)			**7.33**
Learning gain (%)			**22**

Table 8.8 Two-sample one-tail T-tests comparing pre- and post-tutorial increase

Test	μ	σ	n	σ^2	$\mu_1 - \mu_2$	t	critical t value for $p = 0.2$
Pre	68.00	22.19	15	32.84			
Post	75.30	19.24	15	24.68	-7.30	-0.96	-0.85

Results in Table 8.7 show that learning gain was positive. The one-tail two-sample T test in Table 8.8 shows the increase between pre- and post-tutorial testing with 80% confidence ($p = 0.2$).

Table 8.9 Learning gain by academic level

Academic level	**Level label**	**Pre-tutorial average MCQ score (%)**	**Post-tutorial average MCQ score (%)**	**Increase (%)**	**Learning Gain (%)**
8	PhD	80	90	10.0	50
7	Masters	50	62	12.0	24
6	3rd year bachelors	65	70	5.0	14
5	2nd year bachelors	-	-	-	
4	1st year bachelors	80	84.2	4.2	21

Table 8.10 Learning gain by prior subject expertise

Course	**Pre-tutorial average MCQ score (%)**	**Post-tutorial average MCQ score (%)**	**Increase (%)**	**Learning Gain (%)**
Non-computing	40	53	13.0	39.11
Computing	75	80.8	5.8	23.2

Table 8.10 shows that non-computing students benefitted the most from taking the introductory tutorial on computer programming.

In response to the user satisfaction survey statement 'I would use this software to learn' the mode average agreement was 4 out of 5, indicating that across demographic groups participants felt the software helped them to learn. These results suggest that the alternative hypothesis can be accepted and that study participants benefitted from learning using the Hendrix 1.0 CITS.

The results in Tables 8.7 and 8.10 suggest that the null hypothesis can be rejected.

8.9 Discussion

Hendrix 1.0 has been evaluated in this study using a goal-question-metric framework. The high-level goals of the system were to:

- Converse effectively
- Facilitate learning

The system has shown good performance in classifying speech acts during tutorial conversation, allowing for a coherent discussion to be composed around the tutorial content. User satisfaction with coherence of discourse is high and errors are lower than comparison benchmarks from literature.

The system has also shown good performance in accurately marking students answers. Hendrix 1.0 performs the task with equivocal accuracy to that reported in literature for human tutor marking and improves on reported accuracy for unsupervised computerised marking of free text answers. User satisfaction with answer-marking is high.

Hendrix 1.0 has underperformed in answering learners' questions. The system failed to meet the 14% benchmark error rate taken from literature. However, the system was able to answer 76% of learner questions correctly.

Analysis of log files show that improvements to keyword extraction could make improvement to accuracy in the next iteration of the software.

Hendrix 1.0 has satisfied the metrics for two out of three questions for the goal of conversing effectively, with the third achievable with minor revisions to the scripts used to identify query utterances.

The participant group showed a positive learning gain, with non-computing students benefitting most from the tutorial. For non-computing students the learning gain exceeded the 30% normalised average learning gain benchmark used in Colt et al. (2011). User satisfaction with learning from Hendrix 1.0 was also high.

Overall, Hendrix has performed well. However, improvements are needed. Analysis of utterances where error occurred has highlighted three key areas for improvement:

- Add definitions to the knowledge base to help with question answering
- Update the part-of-speech models to include domain specific words and acronyms to aid with question answering
- Expand speech act pattern sets to include noun-phrases such as 'where is *', 'what is *', 'how does *', 'show me *' and 'tell me *'

8.10 Conclusions

The contribution of research in this Chapter has been to evaluate whether the intelligent CITS, Hendrix 1.0, which was designed to deliver syllabus material through discourse using adaptive and micro-adaptive behaviours identified in literature (section 4.8) has been successful at conversing with e-learners and whether the conversational system has facilitated learning.

Participants across demographic subgroups showed positive learning gains of 22% ($p = 0.2$) (table 8.7) for the cohort and up to 39.11% (table 8.10) for the non-computing student group, indicating that Hendrix 1.0 can teach Java

programming effectively. Learners also reported high levels of satisfaction for usability and conversational coherence, indicating that they would like to use the system to learn. The results presented in this chapter suggest that Hendrix 1.0 is a suitable and effective e-learning platform with which to evaluate the benefit that can be added by incorporation of comprehension based adaptation.

The research discussed in this chapter suggests that the selected combination of technologies (Chapter 7) selected from literature (Chapter 4), and the adaptive and micro-adaptive coaching behaviours from literature (section 4.8) and encoded within the system behaviour (section 7.4.3), does provide a minimal viable product for further development.

Chapter 9

Image pre-processing and comprehension classifier training

9.1 Introduction

This chapter details two software tools designed and developed to aid initial exploratory experimentation on image data sets gathered during the pilot study of Hendrix 1.0 (Chapter 8). A behavioural indexer was created to pre-process image data so that non-verbal behaviour could be extracted using a bank of pre-trained artificial neural networks.

Two stages of data processing are required for experimentation as detailed in Chapter 10. The first is an indexing process which will extract a model of non-verbal behaviour from a stream of sequential images. The second is an artificial neural network (Chapter 5) which will first learn and then classify the descriptive vectors of aggregate non-verbal behaviour.

Artificial neural networks have been chosen to fulfil all aspects of classification - locating faces and features, classifying non-verbal behaviour and classifying comprehension. The approach has been chosen for experimentation because of positive results reported in literature and discussed in sections 3.4.3 and 6.3.1).

Section 9.2 details the indexing tool and section 9.3 the artificial neural network training and testing tool.

9.2 Behavioural indexing tool

An indexer (algorithm 6) has been developed to convert the raw source image data into a structured data set containing rows of behavioural descriptions and ground-truth comprehension labels. Each row represents a measurement of behavioural channels over a given period of time. The period of time over which behaviours are aggregated is an experimental variable discussed in Chapter 10. The data processing pipeline used to convert images into behavioural descriptors is based on that described in section 6.4 of the literature review.

The behavioural model extracted from images is detailed in section 9.2.1 of this chapter. In this section the indexing algorithm (algorithm 6) is discussed.

Tutorial conversation logs, answer scores and image data were gathered during the pilot study of Hendrix 1.0 (Chapter 8). As shown in algorithm 6, the indexer first selects all tutorial users for which data is held. The indexer loads the user's tutorial log file and extracts all *answer* entries. Each answer entry contains a unique identifier and a score for the answer given.

Using the collection of answer unique identifiers, the indexer loads each question's image set in turn. The image set for each answer contains a chronologically ordered set of 360 by 240 pixel images recorded from a front-facing web camera attached to the PC terminal on which Hendrix 1.0 was used during the pilot study (Chapter 8).

For each image set the indexer groups the images into batches representing a given time period. The number of images in each batch, or time window, is representative of a specified time over which behavioural observations will be aggregated. Buckingham et al. (2014); Rothwell et al. (2007) showed positive results when aggregating behaviours over a one-second time window, equivalent

Algorithm 6: Pseudo-code algorithm for indexing non-verbal behaviour

```
1  Create a matrix to hold data as results;
2  Get all users as users;
3  for each user in users do
4  |  Extract (gender, ethnicity) from tutorial log file as demographics;
5  |  Extract all (answer_id, answer_score
6  end
7  ) pairs from tutorial log file as answers;
8  for each answer in answers do
9  |  Read images from directory 'image_data/answer_id' in chronological
   |   order as images;
10 |  Group images by selecting n images and skipped i images to create
   |   batches for images for time windows of observation;
11 |  for each window in windows do
12 |  |  for each image in window do
13 |  |  |  Create new matrix to hold non-verbal behaviour;
14 |  |  |  Search image for a face using an artificial neural network;
15 |  |  |  if face is found then
16 |  |  |  |  Search image for eyes using an artificial neural network;
17 |  |  |  |  if eyes is found then
18 |  |  |  |  |  Make measurements of behaviours in face and features
   |  |  |  |  |   as nvb;
19 |  |  |  |  |  Add nvb to matrix;
20 |  |  |  |  end
21 |  |  |  end
22 |  |  end
23 |  |  Average matrix columns to produce summary;
24 |  |  Add demographic + summary + answer_score to results;
25 |  end
26 end
27 Randomise row order of results;
28 Write each vector and label in results to CSV file;
```

to 15 sequential images. The optimal time window to use is a parameter for experimentation which is discussed in Chapter 10.

Once images are grouped into time windows, the indexer uses an artificial neural network to search the image for faces fitting within a 50 by 50 pixel square. If a face is found, the face region is scanned using a second artificial neural network for 15 by 15 pixel regions containing eyes. While literature discussed in section 6.3 has highlighted a potential weakness of artificial neural

networks with scale variance, the approach was selected for initial trial due to reported success in similar applications (section 6.3.1).

Should face and features be located within an image, a bank of pre-trained classifiers is used to make measurement of non-verbal behaviours. The non-verbal behaviours measured are detailed in section 9.2.1. Measurements are made by first reducing the dimensionality of the pixel data for a feature using principal component analysis (PCA), as discussed in section 6.3.1), and they are then passed into a specifically trained artificial neural network to produce a binary true or false classification, a +1.0 or -1.0 output, indicating the truth of a given behaviour. For example, a measurement will be made by selecting the left eye pixel data, reducing using PCA and passing to a classifier which will give a true or false value indicating if the eye gaze is directed forward.

Geometries, movement and skin tone changes are measured by calculating the change in the location of features and by sampling the colour values of pixel data contained within facial features.

True or false questions are asked in sequence for each feature and each relevant behaviour, resulting in a vector of +1.0 or -1.0 observations.

The vector of observed behaviour for each image is added to a matrix for the time window and then summarised using cumulative and average statistics to produce a single 39 variable descriptive vector, as discussed in section 6.4 of the literature review.

Finally, the demographic data for the user, behavioural vector and answer score are written as a single row in a comma separated value file. The process is repeated for each answer given by each user.

To facilitate the experimentation discussed in Chapter 10, the indexing process is repeated with a range of time window durations and intervals.

9.2.1 Model of non-verbal behaviour

The model of behaviour is based on a survey of features used in similar applications (section 6.2). Table 9.1 details the behavioural channels under observation.

Table 9.1 Behavioural data model

Type	Channels	Examples
Learner	2	Gender, Ethnicity
Eyes	17	Eye openness, Gaze direction, Blink
Geometries	18	Position, Rotation, Movement
Physiological	2	Blush, Blanche

9.3 Classifier training and testing tool

A comprehension classifier has been developed based on an artificial neural network (Chapter 5). An artificial neural network has been chosen as the classifier due to the successes reported in literature for similar applications (discussed in section 6.5).

The artificial neural network (ANN) will be trained on data produced by the indexer. The topology of the artificial neural network has been designed to match the inputs generated by the indexer, with 39 input nodes to accept the 39 variables in each behaviour vector. The ANN has a single output node with a threshold output function to produce a classification of either +1.0 for comprehension or -1.0 for non-comprehension.

The network is trained using back-propagation of errors (section 5.4.1), and tested using 10-fold cross validation Haykin (1994).

While the input layer and output layer of the network are defined by the shape of the data and the desired classification output, the precise topology of the network's hidden layers are a parameter for the experimentation discussed in Chapter 10.

The classifier training tool (algorithm 7), a command-line application, has been developed to facilitate rapid evaluation of a range of topological parameters. The command-line trainer accepts arguments to define the structure of the network and instantiates, trains and tests a network of the specified topology.

Algorithm 7: Pseudo-code algorithm for training artificial neural network

```
1  Load CSV as results;
2  Split results into 10 each sized partitions as folds;
3  for each fold in folds do
4  |  Set Te = fold;
5  |  Set Va = next fold;
6  |  Set Tr = all folds except Te and Va;
7  |  Create new MLP neural network as network;
8  |  Train network on Tr;
9  |  Evaluate network on Va;
10 |  Test network on Te;
11 |  Save network and performance statistics to log file;
12 end
```

The trainer allows the network configuration to be determined empirically by testing each combination of parameters in Table 10.1.

9.4 Conclusion

This chapter has presented two tools designed and developed to facilitate a pilot study of computational analysis of non-verbal behaviour and classification of comprehension from image data using artificial neural network. The tools pre-process the image data into descriptive vectors of non-verbal behaviour for a given time window and then use the pre-processed data to train and test a comprehension classifier based on artificial neural networks. In Chapter 10 the command-line tools presented in this chapter will be used to evaluate the parameters for data pre-processing, window duration and interval, and for network topology.

Chapter 10

Study: Exploring the relationship between reading comprehension and learner non-verbal behaviour

10.1 Introduction

This chapter presents an initial exploratory study intended to identify, from data, the feasibility, performance, requirements and constraints of modelling learner comprehension from non-verbal behaviour, using an artificial neural network classifier. An artificial neural network has been chosen for initial experimentation as the classifier has performed well on related problems discussed in section 6.5.

Using a dataset of comprehension labelled web camera images gathered during the pilot of Hendrix 1.0 CITS (Chapter 8), the effectiveness of a neural networks based approach identified in literature (Chapter 3.4) is empirically evaluated in terms of classification accuracy.

This exploratory study aims to answer several high-level questions which will provide guidance on experimental design, data treatment and technical implementation of a real-time comprehension classification system (section 10.2).

Section 10.2 details the research questions the study aims to answer and section 10.3 explains the motivation for conducting this study. Section 10.4 outlines the study procedure. The method is detailed in section 10.5 and results in section 10.6. This aim to identify the optimum parameters for data pre-processing and classifier topology. Conclusions are shared in section 10.6.

10.2 Research questions

The research questions explored in this experiment are necessary to define the treatment of image data in extracting and measuring non-verbal behaviour over time and in selecting the optimum topology for an artificial neural network comprehension classifier.

1. When and for how long does comprehension indicative behaviour occur?
2. Can an MLP achieve binary comprehension classification at above chance levels? (>50%)
3. What set of classifier hyper parameters produce the highest accuracy?

10.3 Motivation

A core pedagogic device in cognitivist educational practice is scaffolding (Chapter 2.2). Scaffolding involves the introduction and fading of support for learners during problem-solving. Literature suggests that successful scaffolding should allow the learner to first attempt problem-solving without explicit intervention or instruction, promoting the demonstration of competency. If a learner experiences difficulty, the tutor should then introduce increasing levels of support

to help develop task and subject understanding. When competency is again demonstrated, the explicit support and instruction should be withdrawn, or *faded*, allowing the learner to take control of problem-solving processes again.

Intelligent tutoring systems aim to mimic the flexibility of human tuition by adapting to a variety of real-time learner feedback channels. However, a generalised, affordable, practical and non-intrusive method of identifying e-learner comprehension automatically in near real-time is absent, as established in Chapter 3.

The Hendrix 1.0 CITS discussed in Chapters 7 and 8 is able to introduce and fade support in response to answers a learner provides to direct questions. However, observation of tutor practice in the classroom (Chapter 2 section 3.3) suggests that human tutors enact timely interventions, to scaffold the learning experience, by estimating accurately the learner's degree of comprehension from non-verbal behaviour.

To develop the ability of the Hendrix CITS to perform human-like cognitive apprenticeship, the platform must be equipped with the functionality to model and classify e-learner comprehension in real-time from coarse grained analysis of non-verbal behaviour. Accurate real-time classification of e-learner comprehension would provide a feedback channel for intelligent adaptation of materials, user interface elements and discourse, without depending solely on incorrect answers as a trigger for the introduction of scaffold.

Literature (section 3.4) has suggested that a neural networks based approach (Rothwell et al., 2006, 2007) of behavioural summary and classification can be used to classify learner comprehension in dyadic verbal interactions under interview conditions (Buckingham et al., 2012, 2014).

10.4 Study procedure

15 undergraduate, masters and PhD students at Manchester Metropolitan University undertook the experiment. The participant size was based on

participant numbers for similar pilot studies of CITS identified in literature (Cai et al., 2011; Serón and Bobed, 2016).

To assess the effectiveness of the method discussed in literature (Buckingham et al., 2012, 2014; Rothwell et al., 2006, 2007), the author has attempted to recreate the data treatment and classification process. A new dataset of labelled images was collected during the pilot study of Hendrix 1.0 CITS (Chapter 8). Each answer period during the on-screen tutorial was recorded using a front-facing web camera attached to the PC. Each recording was stored as a sequential set of images, along with the score assigned for the answer provided. Answer scores are used as a ground-truth measure of comprehension.

Answers were marked automatically by the Hendrix 1.0 CITS during the pilot study discussed in Chapter 8. Correct answers are labelled as 'comprehension', while incorrect answers are labelled as 'non-comprehension' based on the tutorial log files.

This pilot study consisted of the following steps:

1. Participants took a seat at a computer in a computer laboratory at Manchester Metropolitan University

2. Participants read and signed a consent form for the experiment and reviewed a set of instructions detailing their tasks for the experiment

3. Participants started the Hendrix 1.0 CITS application by double clicking the executable on the desktop

4. Participants were instructed to take a tutorial on 'For loops' using the Hendrix 1.0 CITS

5. Participants were recorded using a front-facing web camera while answering each question

6. Participants answers to tutorial questions were recorded in log files

7. Participants were recorded, subject to ethics, during the tutorial using a front-facing web camera

8. Participant non-verbal behaviour was analysed for each question

9. A bank of artificial neural networks were trained to classify comprehension from observed e-learner non-verbal behaviour using an array of hyper-parameters

10.4.1 Ethics

Under ethical approval from Manchester Metropolitan University study participants consented to undertake an on-screen tutorial with a conversational intelligent tutoring system called Hendrix 1.0. During the tutorial participants' dialogue with Hendrix 1.0 was recorded. In addition, image data captured from a front-facing web camera was recorded and saved to encrypted portable media.

As data collected during the experiment was not anonymous and contained personally identifying information including names, email addresses and demographic information, the data collected was transferred to and stored on a physically secured computer with an encrypted hard drive. In accordance with ethical approval and the terms of consent for the experiment, all information used for redistribution or publication purposes was anonymised or aggregated so as not to be personally identifying.

10.4.2 Participant information

The participant group consisted of 15 student volunteers from the School of Computing, Mathematics and Digital technology at Manchester Metropolitan University. The participant group consisted of 11 male and 4 female participants, with 7 of participants studying at under-graduate level and 8 studying at post-graduate level.

10.5 Method

In this experiment a grid search is performed to evaluate the optimum set of parameters for both data pre-processing and artificial neural network topology.

Data pre-processing parameters are temporal, determining the period of analysis, the way in which the total answer period is segmented and summarised and the intervals between summaries (Table 10.1). ANN topology is tested with 10, 15, 20 and 25 hidden layer nodes.

In a grid search, each combination of parameters is trained and tested. The full search grid of 120 parameter combinations is detailed in Table B.1. The exhaustive search aims to identify the set of parameters which produce the highest possible classification accuracy.

Image data and ground truth comprehension scores were collected during the pilot study of Hendrix CITS (Chapter 8). Image sets have been pre-processed using the behavioural indexing process detailed in section 9.2. Varying the temporal parameters of pre-processing has produced 30 data sets for evaluation. The parameter combinations are shown in Table 10.1.

Each row in Table 10.1 shows the parameters for indexing of the non-verbal behaviour. Each row represents a separate data set of labelled non-verbal behaviour, generated using the temporal variables detailed in Table 10.1.

For each data set in Table 10.1, four configurations of network topology have been tested - 10, 15, 20 and 25 hidden neurons within a single hidden layer. Testing classification accuracy with each configuration of the artificial neural network will identify the best network topology to use for classification of behavioural data.

The full results table is shown in Appendix B. Please refer to the parameter Table 10.2 for descriptions of the column headings in the results table.

Table 10.1 Table of data sets with different temporal configurations

Analysis duration (seconds)	**Window duration (seconds)**	**Interval between windows (seconds)**	**Delay before analysis (seconds)**
30	5	2.66	0
30	4	2	0
30	3	1.66	0
30	2	1	0
30	1	0.66	0
10	5	2.66	0
10	4	2	0
10	3	1.66	0
10	2	1	0
10	1	0.66	0
10	5	2.66	5
10	4	2	5
10	3	1.66	5
10	2	1	5
10	1	0.66	5
10	5	2.66	10
10	4	2	10
10	3	1.66	10
10	2	1	10
10	1	0.66	10
10	5	2.66	15
10	4	2	15
10	3	1.66	15
10	2	1	15
10	1	0.66	15
10	5	2.66	20
10	4	2	20
10	3	1.66	20
10	2	1	20
10	1	0.66	20

10.6 Results and discussion

Using ANOVA with backwards elimination it is possible to model the effect of duration, window and delay on the test set classification accuracy. Figure 10.1 confirms that parametric testing can be used as the residuals fall within a normal distribution.

Table 10.2 Table of experimental parameters

Abbr.	Full	Description
Dur	Segment Duration	The number of seconds selected for analysis
Win	Analysis Window	The number of seconds included in each analysis window
Int	Window Interval	The number of seconds between analysis windows
Del	Delay	The number of seconds to delay analysis before selecting the duration period
Nodes	Nodes	The number of hidden layer nodes in the neural network

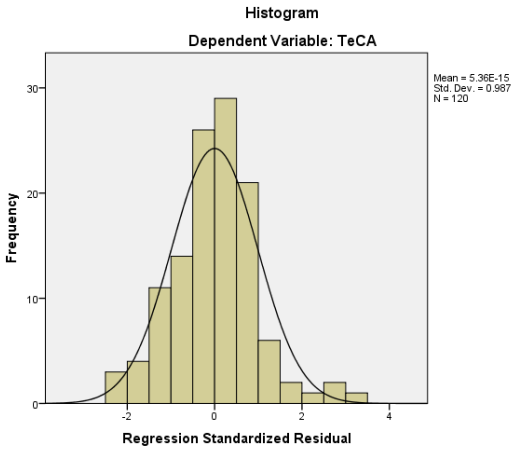


Figure 10.1 Histogram of regression residuals for the model

ANOVA was used to test the significance of temporal variables Duration, Window, Interval and Delay on test set classification accuracy (TeCA) for all network configurations shown in appendix B.

Model pruning using backward elimination by analysis of significance of Coefficients shows that Window length in seconds, the length of the temporal window for which each comprehension classification is made, is the least significant (sig=.820) factor in classification accuracy. Removing Window from the model gives a significant ANOVA model (sig=.000), with Duration (sig=.002), Interval (sig=.000) and Delay (sig=.000) all highly significant variables.

Partial regression plots show the coefficient between an independent and dependent variable in the model. Figures 10.2, 10.3 and 10.4 show the partial regression plots, scatter splots of the coefficients between the parameter and

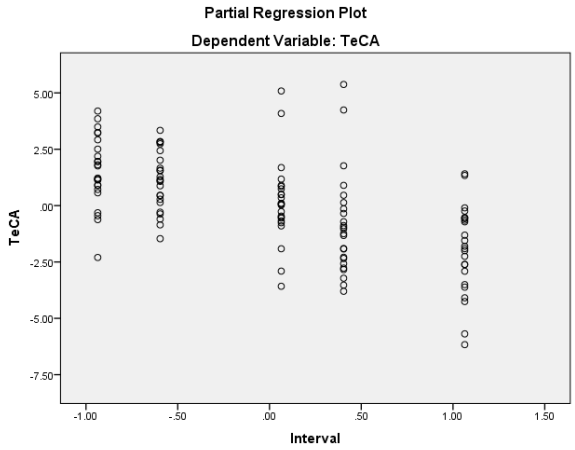


Figure 10.2 Partial regression plot for Interval variable against Test set Classification Accuracy (TeCA)

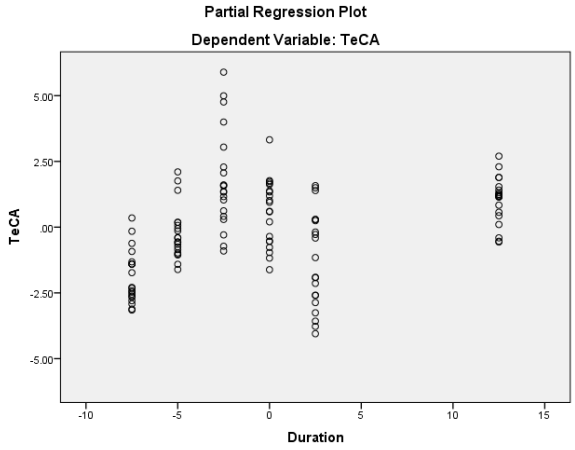


Figure 10.3 Partial regression plot for Duration variable against Test set Classification Accuracy (TeCA)

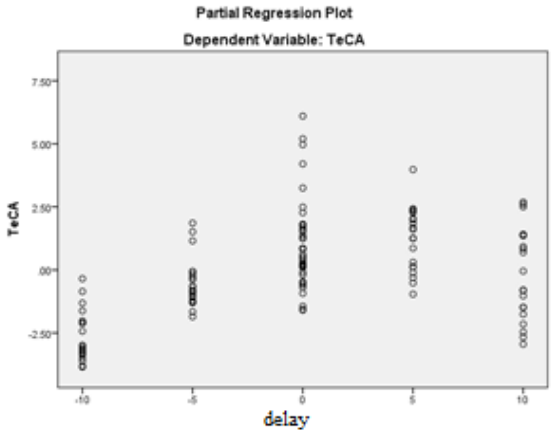


Figure 10.4 Partial regression plot for Delay variable against Test set Classification Accuracy (TeCA)

classifier accuracy, for interval, window duration and delay. Each point on the scatter plot shows the coefficient with classifier accuracy (TeCA) given a distinct configuration of all other parameters in the model. The higher the coefficient with TeCA, the more likely the independent variable will increase classifier accuracy (TeCA).

The interval variable appears to have a linear correlation (figure 10.2) against the TeCA, confirming that shorter intervals produce better TeCA outcomes. However, the same cannot be said for duration or delay. A closer investigation of the duration (figure 10.3) and delay (figure 10.4) variables show that there may be diminishing returns after the 10 second point.

While the window variable was excluded from the multivariate ANOVA model due to low significance, using linear regression against the test set classification accuracy (TeCA) suggests that window duration could have a suppressive effect on TeCA (beta=-.542). Analysis (figure 10.5) of the average TeCA by window length from 10-fold cross validation results (Table B.1) supports a depressive effect.

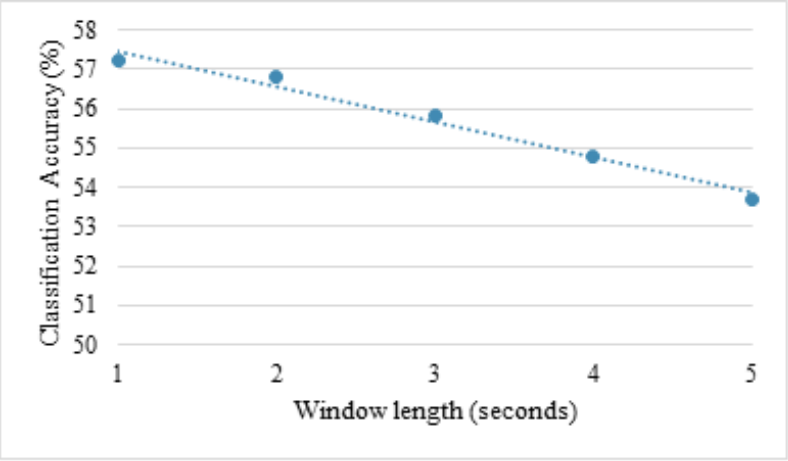


Figure 10.5 Average TeCA by window length

The correct topology of the network can allow for greater generalisation, but larger networks take longer to train and can suffer from over-fitting. As literature is absent a method of predicting the optimum topology for a network, in this experiment several hidden layer node configurations have been tested,

Table 10.3 Table of data sets with different variable configurations

ANOVA					
Model	**Sum of squares**	**df**	**Mean Square**	**F**	**Sig.**
1 Regression	1.917	1	1.917	.342	.560b
Residual	661.102	118	5.603		
Total	663.019	119			

a. Dependent Variable: TeCA
b. Predictors: (Constant), Nodes

Table 10.4 Table of data sets with different variable configurations

Number of hidden layer nodes	**Average TeCA %**
10	54.9
15	54.4
20	57.4
25	55.8

with 10 the lowest and 25 the highest. Linear regression is used to analyse the relationship between the number of hidden layer nodes and the test set classification accuracy. The regression model shows no significant relationship between the number of nodes in the hidden layer and the TeCA (sig=.560). However, configurations with 20 nodes achieved average TeCA 1.6% higher than the next best configuration (Table B.1 in appendix B).

The exploratory analysis presented in this chapter highlights which parameters for data pre-processing result in the highest classification accuracy. The results of this experiment have informed the design of a comprehension classifier.

While none of the 120 10-fold cross validation outputs showed worse than chance levels of accuracy on the test set, they all fall short of the 70 – 80% accuracy benchmarked from literature (Buckingham et al., 2012, 2014).

Despite the disappointing classification accuracies achieved, the results have highlighted some interesting findings. The regression coefficients indicate that three significant conclusions can be drawn from the analysis:

1. Longer durations yield higher classification accuracy

2. Shorter intervals between windows yield higher classification accuracy

3. Comprehension associative non-verbal behaviour is least evident in the first 10 seconds of the answer period

In addition to these findings, the following suggestions could be made regarding optimum configuration:

1. 20 nodes is a suitable number of hidden layer nodes

2. Shorter window lengths yield higher classification accuracy

To test whether these conclusions are correct a second experiment is required. The experiment will use a longer duration, with a short window length and short interval. Based on the results in Table 10.4 the network will be configured with 20 hidden layer nodes.

10.7 Further refining parameters for data treatment

Based on the findings highlighted in Chapter 10, detailed in section 10.6, it is anticipated that generating data sets with long durations (60 seconds) and short intervals (.66 seconds) will yield the highest classification accuracies.

In this experiment the delay and window length variables are to be examined again, taking into account the conclusions from experiment 1 (section 10.6).

The delay variable is to be extended, excluding data recorded in the first 10, 20 and 30 seconds of each recorded answer period. The window length variable is to be evaluated for each delay value at 1 and 2 seconds.

10.7.1 Method

Grid search is used to evaluate the effect of the analysis duration and interval parameters of data treatment on classification accuracy using image data and

comprehension ground truths gathered during the Hendrix CITS pilot study (Chapter 8).

Six data sets have been created using the process detailed in section 9.2. The parameter combinations for grid search are detailed in Table 10.5.

Table 10.5 Table of data sets with different variable configurations

Duration (seconds)	**Interval (seconds)**	**Delay (seconds)**	**Window (seconds)**
60	0.66	10	1
60	0.66	10	2
60	0.66	20	1
60	0.66	20	2
60	0.66	30	1
60	0.66	30	2

10.7.2 Results and discussion

The ANN classification accuracy results presented in Table 10.6 show the outcome of 10-fold cross validation for each data set in Table 10.5 with network configurations shown as successful in section 10.6.

Table 10.6 Results of network training

Duration (seconds)	**Interval (seconds)**	**Delay (seconds)**	**Window (seconds)**	**Nodes**	**TeCA**
60	0.66	10	1	20	58.36
60	0.66	10	1	25	58.77
60	0.66	10	2	20	60.55
60	0.66	10	2	25	60.81
60	0.66	20	1	20	60.79
60	0.66	20	1	25	59.32
60	0.66	20	2	20	63.65
60	0.66	20	2	25	63.56
60	0.66	30	1	20	62.98
60	0.66	30	1	25	63.01
60	0.66	30	2	20	67.20
60	0.66	30	2	25	68.02

The results shown in Table 10.6 support the conclusions of experiment 1 (section 10.8). A partial regression plot for the delay variable (figure 10.7)

visualises the relationship, indicating that an increased delay yields higher test set classification accuracy.

ANOVA[a]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	99.751	3	33.250	46.233	.000[b]
	Residual	5.753	8	.719		
	Total	105.505	11			
2	Regression	99.751	2	49.875	78.017	.000[c]
	Residual	5.754	9	.639		
	Total	105.505	11			

a. Dependent Variable: TeCA

b. Predictors: (Constant), Nodes, Window, Delay

c. Predictors: (Constant), Window, Delay

Figure 10.6 ANOVA table for classification accuracy with dependent variables window and delay

Figure 10.7) shows that comprehension associative non-verbal behaviour is more evident in the 30 – 90 second region of the answer period.

The regression residuals plotted in figure 10.3 suggest that the classifier's ability to learn discriminant patterns of non-verbal behaviour indicating comprehension and non-comprehension states varies depending on the length of time analysed in each learner answer period.

The results suggest that the most distinct and discriminative patterns of behaviour are captured between 30 and 90 seconds into the answering of a question. This is the period in which most answers are submitted by learners. Results in Table 10.6 show that classifier accuracy reaches 68.02% when trained and tested on data from the 30 - 90 second period of each answer.

The finding supports the technical approach taken, suggesting that a computational analysis of learner non-verbal behaviour can be used to classify comprehension states. However, the results highlight a flaw in the study design. The Hendrix 1.0 platform allows users a high degree of freedom in task sequencing, and a broad range of materials to refer to while answering questions. This freedom makes it impossible to identify the specific tasks being undertaken

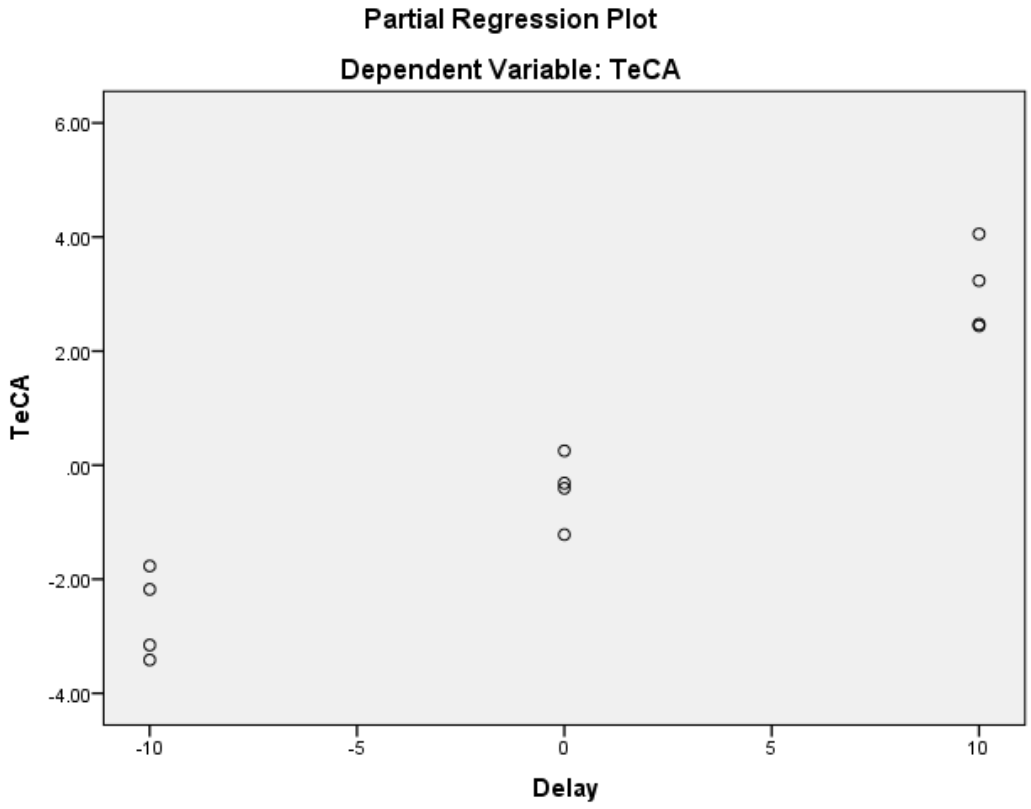


Figure 10.7 Partial regression plot for delay variable

at any given time during an answer period. It is not known whether at any given time the learner is watching a video, reading text or typing an answer. The result showing higher classification accuracy around the time of answer submission may suggest that stress response non-verbal behaviour, as induced by increased cognitive load, is most apparent immediately prior to answer submission.

To test more accurately whether NVB indicative of non-verbal behaviour is most evident at the point of answer formulation and submission, a second experiment is needed. The second experiment will reduce the freedom of the learner and focus on capturing only responses to question information, answer formulation and answer submission.

10.8 Conclusion

This chapter has presented a pilot study exploring the viability of a neural networks based approach to near real-time comprehension assessment by modelling of learner non-verbal behaviour.

The contribution of this research is to bring together technologies for NVB modelling and behavioural classification, as discussed in Chapters 5 and 6, and exploring whether NVB is expressive of comprehension in a computer mediated e-learning context in such a way that a machine learning algorithm can distinguish between comprehension and non-comprehension (Chapter 3) states in learners reading question and answer tutor dialogue.

The results of the study suggest that the approach has potential, but that the complexity of the environment, learners working with multiple information sources and multiple on-screen windows, made the behavioural data difficult to interpret and learn from.

The experiments conducted as part of the exploratory study presented in this chapter have highlighted six findings relating to the questions posed in section 10.1:

1. Classification accuracy at 68.02% can be achieved using the proposed method of image analysis and classification by artificial neural network

2. Classification accuracy fails to meet the benchmarks from literature

3. The data collection tool (Hendrix) allows too much freedom in task switching during recording, making it difficult to understand the presence and meaning of behaviours demonstrated in training data

4. A network with 25 hidden layer nodes provided the highest classification accuracy

5. There is little difference in classification accuracy between 20 or 25 nodes; however, a larger network will be slower to train

6. A short analysis window is optimal, with one and two second windows yielding best results

7. A shorter interval analysis windows is optimal, suggesting continuous analysis rather than staggered analysis performs best in terms of classification accuracy

It is unclear where behaviours relate to an information source or where they might relate to navigating the user interface. A simpler experiment is required to gather data for classifier training which better isolates comprehension indicative behaviours.

Chapter 11

COMPASS

11.1 Introduction

This chapter details the design and development of COMPASS (Holmes et al., 2017b), a novel near real-time comprehension classification system for use in e-learning environments. This chapter outlines the motivation for developing the system and the contributions of the system (section 11.2), an overview of noteworthy system functionality (section 11.4), an overview of the software architecture (section 11.5), the behavioural data model (section 11.6) and NVB analysis process (section 11.7), and the technical approach for classifying comprehension by analysis of non-verbal behaviour (section 11.8). Conclusions are presented in section 11.9.

11.2 Motivation

Developing from the findings of the initial pilot study (Chapter 10.1), this chapter presents a system designed to capture, model and classify comprehension indicative e-learner non-verbal behaviour during short on-screen problem-solving interactions.

COMPASS is a novel system for estimating e-learner comprehension in real-time by automatic analysis of non-verbal behaviour. COMPASS uses non-

intrusive measurement of multiple channels of non-verbal behaviour, including feature state, facial movement, posture and physiological change, to model and classify e-learner comprehension in real-time. COMPASS requires only a computer and a web camera and has been developed for use in real-world classroom environments. COMPASS does not rely on *a priori* psychological models of behaviour or recognition of posed expressions but uses supervised machine learning to learn the discriminative patterns of non-verbal behaviour indicative of comprehension and non-comprehension states.

This chapter presents COMPASS, a near real-time Comprehension Assessment and Scoring System. COMPASS has been developed as part of an on-going project to equip an intelligent e-learning platform with human-like understanding of comprehension indicative learner behaviour, so as to allow the system to enact timely and appropriate interventions in the learning process. COMPASS addresses the gap in the literature for a generalised, practical, non-intrusive, real-time classifier designed to give feedback on learners' comprehension levels during mental processing for a variety of on-screen information types including discursive text, numeric and algebraic equations, programming code and diagrams.

The COMPASS image processing algorithm and comprehension classifier have been developed and evaluated using a large dataset of comprehension class-labelled web camera footage, generated by student volunteers at Manchester Metropolitan University using a bespoke on-screen quiz system.

COMPASS overcomes problems identified with previous NVB analysis techniques (Chapter 3.4) by combining low cost non-intrusive hardware, robust image processing techniques and the ability to learn from real-life learner behaviour. COMPASS has been developed as a .NET library which can be included in any .NET application. COMPASS is intended to provide a simple interface allowing training and evaluation of cognitive classifiers, automatic

near real-time analysis of non-verbal behaviour from image data streams and cognitive state estimation and classification.

11.3 Contributions

The contributions of COMPASS are:-

- Demonstration of extracting multi-channel NVB from standard resolution web camera images;
- Enhancement of the behavioural model found in literature (section 6.2), expanded to include meta-data relevant to learners;
- Demonstration that artificial neural networks are capable of learning discriminant patterns as represented by the behavioural model design;
- Demonstration of a near real-time reading comprehension classification algorithm.

11.4 Key functionality

- **Capture images from a stream of web camera image data**
  COMPASS accepts an ordered collection of images from a standard resolution USB web camera. Each collection of images contains a temporally linear tranche of data, a window, for the overall response period.
- **Locate face and facial features using object recognition techniques**
  COMPASS uses Haar cascades to locate the face and facial features in each image. Feature states such as gaze direction are classified using trained multilayer perceptron networks.
- **Extract physiological data from images using pixel data sampling**

Physiological behaviours such as blushing or blanching are extracted from the raw image data using pixel data sampling.

- **Summarise behaviours over an arbitrary period of time**
  Behaviours from a given analysis window are summarised using average or cumulative statistics, producing a vector of 42 NVB channels normalised to real-number values between -1.0 and +1.0.

- **Read in spreadsheets of behavioural data**
  COMPASS can be trained by loading in pre-formatted CSV files containing behavioural summary vectors and class labels.

- **Train a multilayer perceptron network and produce evaluation statistics**
  COMPASS trains the MLP using back-propagation of errors, with early stopping provided by a look-ahead algorithm. COMPASS produces and logs classifier training and testing statistics in the form of confusion matrices.

- **Classify comprehension based on input behavioural vectors**
  COMPASS incorporates a multilayer perceptron network and threshold function to classify input behavioural feature vectors as either comprehension or non-comprehension indicative.

11.5 Software architecture

The COMPASS comprehension assessment and scoring system is a .NET library for the analysis of non-verbal behaviour from image data and the estimation of comprehension by classification of behavioural patterns. The logical structure of COMPASS decouples processing logic from statistical models, meaning classification techniques can be changed without significant re-engineering. The COMPASS library provides a simple set of interfaces, allowing for easy

integration into any .NET framework application. Figure 11.1 shows the logical architecture of the COMPASS system, along with an indication of inputs and outputs for each system interface.

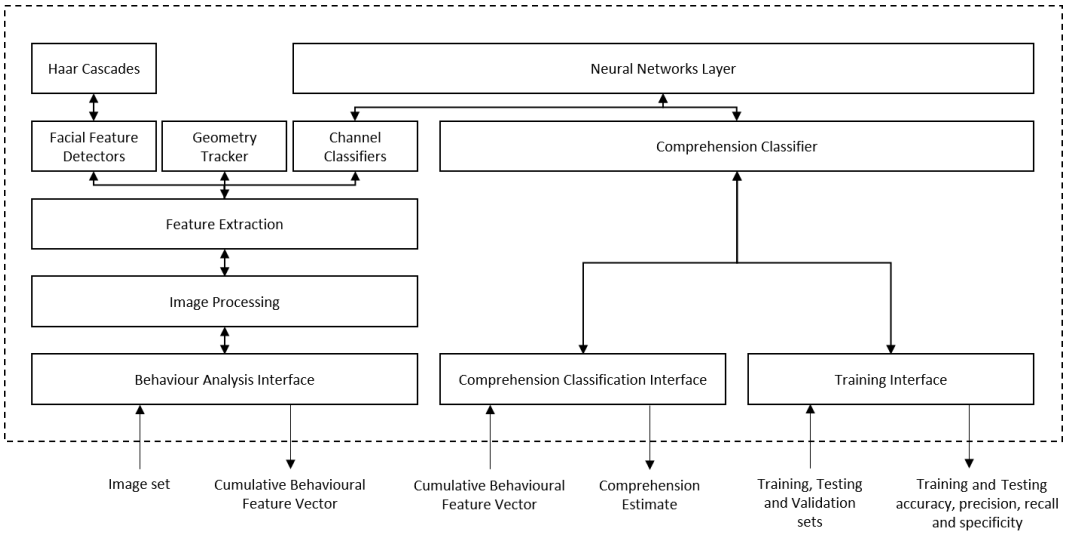


Figure 11.1 Logical architecture of COMPASS

As shown in figure 11.1, COMPASS has an application programming interface (API) which provides core interfaces for integration into software. Each of the interfaces provides access to important functions of COMPASS - behaviour tracking, comprehension classification and classifier training and testing.

- **Behavioural Analysis Interface**

  The behavioural analysis interface provides methods for extracting behavioural patterns from image stream data. An image set for a given period of time is passed in and a cumulative behavioural feature vector (CBFV), representing the description of behaviour, is returned.

- **Comprehension Classification Interface**

  The comprehension classification interface provides methods for converting a CBFV, a description of non-verbal behaviour, into a real-number comprehension estimate between -1.0 (non-comprehension) and +1.0 (comprehension). Classification of comprehension can then be made separately by applying a threshold function to the comprehension estimate.

- **Training Interface**

  The training interface allows a developer to train the comprehension classifier from labelled inputs. The API accepts a collection of class labelled CBFV and returns 10-fold cross validated training and testing accuracy scores.

11.6 Behavioural data model

The COMPASS behavioural data model is based on features commonly included in similar behaviour modelling methods. A survey of commonly used features was produced in Chapter 6 section 11.6 Table 6.1. Building upon literature, a 42 variable descriptive model has been produced (Table 11.1).

A single variable in the model is referred to as a *channel*. As shown in literature (section 3.4), behaviours can be captured by observation of current behaviour. However, as behaviours are *transient*, currently observable behaviour must also be related to previous behaviours. By including channels for the current object state as well as channels for object state change, both *state* and *activity* can be represented in the model. For example, in addition to channels for the *states* 'left eye open' and 'left eye closed', an *activity* channel is included for 'blink'.

The model is populated by surveying the state of individual behavioural channels within each image of the web camera image stream and combining this with meta-data about the learner. A behavioural channel is a single observed behaviour, such as 'left eye gaze right' or 'head rotated left'. Each behavioural channel is a true or false question, which is represented as either +1.0 or -1.0. Each *state* channel can be observed in each individual image, while each *activity* channel can be computed by comparing the current observed channel state to its previous values within a chronological series of behavioural models.

The data model defined in Table 11.1 is populated from each web camera stream image over a specified time period, incorporating normalised values for

Table 11.1 Behavioural data model

Type	Channels	Examples
Learner	5	Gender, Ethnicity, Academic level, Specialism, Experience
Eyes	17	Openness, Gaze direction, Blink
Geometries	18	Position, Rotation, Movement
Physiological	2	Blush, Blanche

the behaviour expressed by the learner and meta-data about the learner and system state. The behaviour modelling process for set time-period produces a matrix of dimensions 42 by n-1, where n is the number of images in the time-period. Finally, the matrix is summarised to produce a single 42 variable cumulative behavioural feature vector (CBFV) (Table 11.2), representative of the normalised behaviour for each channel over the time-period.

Table 11.2 Illustrative example of behavioural data model and cumulative behavioural feature vector

	Channel 1	Channel 2	...	Channel 42	Channel 42
Image 1	0.5	-1	...	1	1
Image 2	0.5	-1	...	1	1
...					
Image 13	0.5	-1	...	-1	1
Image 14	0.5	-1	...	-1	1
CBFV	0.5	-1	..	0	1.0

11.7 Extracting and analysing non-verbal behaviour

Haar cascades (section 6.3.2) were used to identify the face and facial features within each web camera stream image. Using Haar cascades allowed for effective face detection in an uncontrolled scene, more akin to a real-world learning environment. Images are grey scaled and faces scaled down, to reduce

processing overheads, before being decomposed further into component features (for example, left and right eyes).

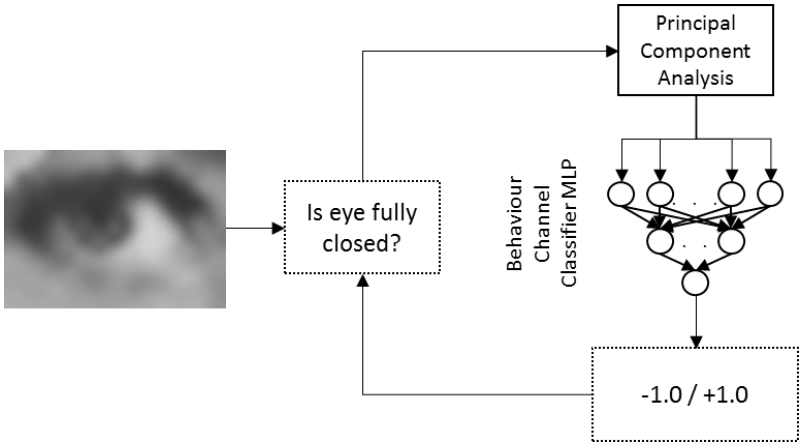


Figure 11.2 Channel classification method for 'eye fully closed'

Figure 11.2 shows how a behavioural channel is classified from an image region of interest (ROI). The ROI pixel data is reduced using principal component analysis (PCA) before being classified by a feed-forward multilayer perceptron network. Each channel classifier network is trained to satisfy a single channel variable in the model (Table 11.1), outputting a binary true or false classification (+1.0/-1.0). Pixel sampling and geometries are also used to calculate change between image frames.

The extraction process is repeated for each image in the given time window, producing a matrix of feature vectors which is then summarised to a Cumulative Behavioural Feature Vector (CBFV), as described in section 11.6.

11.8 Estimating comprehension by analysis of non-verbal behaviour

Comprehension estimation is performed using a machine learning algorithm. Literature shows that affect and cognitive states have been successfully classified from non-verbal behaviour using a number of classification algorithms. While (Whitehill et al., 2008) has some success with regression modelling on

action units, (Karran et al., 2015) and (Fairclough et al., 2015) used Support Vector Machines to classify detailed physiological data streams. However, the most relevant literature comes from (Rothwell et al., 2006) and (Buckingham et al., 2014), both of whom use Multilayer Perceptron Networks to classify behaviourally descriptive numeric patterns extracted from image data.

A comparison of machine learning algorithms 11.3, trained using pilot study data, showed that the multilayer perceptron network achieved the highest accuracy with 10-fold cross validation.

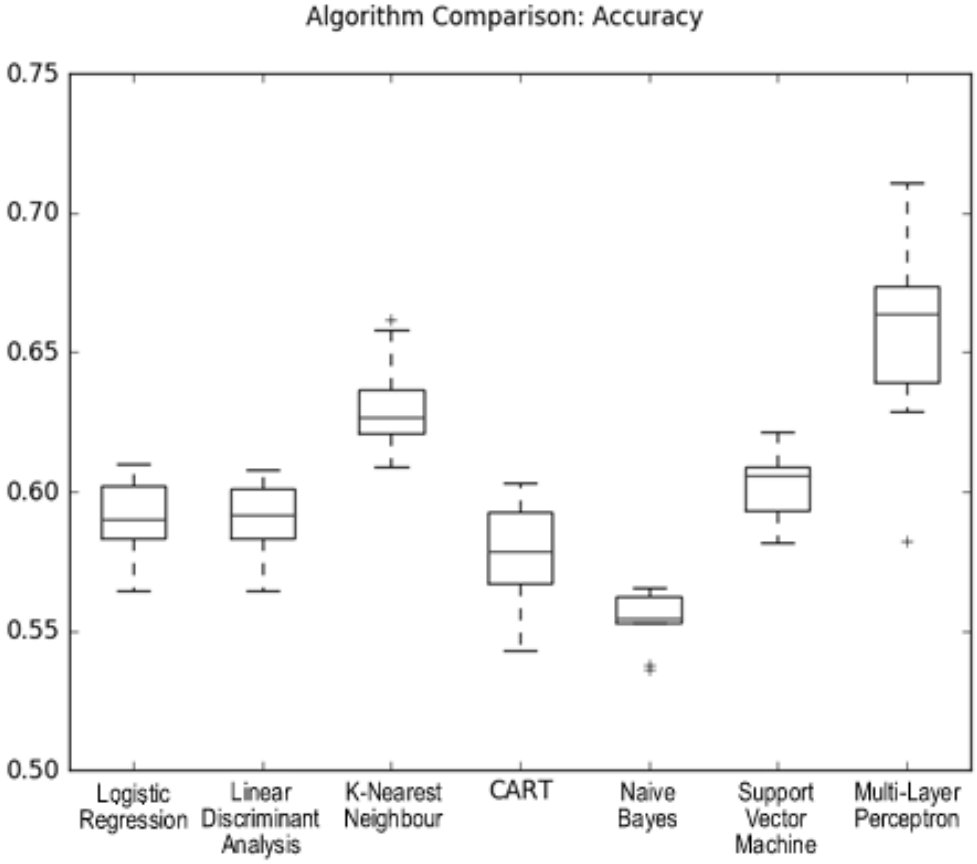


Figure 11.3 Comparison of accuracy achieved by machine learning algorithms on pilot study data

COMPASS has followed the established design pattern used in works by Rothwell et al. (2006) and Buckingham et al. (2014). COMPASS, having generated a CBFV (section 11.6) from a set of chronologically sequenced images, uses a trained MLP to estimate comprehension on a continuous scale from -1.0 to +1.0. The MLP output represents the strength of association between the

input CBFV and the binary 'comprehension' and 'non-comprehension' training labels, where -1.0 is polar 'non-comprehension' and +1.0 is polar 'comprehension'. 0.0 represents an unknown state. Classification of the continuous scale comprehension estimate can be made by applying a threshold function such as equation 11.1. The classifications boundaries, accuracy, precision and recall can all be tuned by altering the *-threshold* and *+threshold* variables.

$$comprehension = \left\{ \begin{array}{rl} \text{-1.0,} & tanh(a) \leq -threshold \\ 0, & +threshold > tanh(a) < -threshold \\ 1.0 & tanh(a) \geq +threshold \end{array} \right\} \quad (11.1)$$

11.9 Conclusion

This chapter has presented an overview of the COMPASS comprehension assessment and scoring system and its contribution to the field (section 11.1). Sections 11.6, 11.7 and 11.8 have detailed the technical approach to behavioural modelling and comprehension classification.

The contribution of research engineering work discussed in this chapter is to provide definition of a technical method for near real-time comprehension state assessment by coarse analysis of facial and upper body non-verbal behaviour expressed during reading comprehension within e-learning.

The aim of COMPASS is to provide, for the first time, a means of near real-time comprehension assessment in e-learning. COMPASS does so by providing a stream of comprehension classifications, based on live observation of e-learner non-verbal behaviour. The system is the first to apply near real-time non-verbal behaviour tracking to classifying comprehension of on-screen information. COMPASS will be trained, tested and evaluated using data gathered from students undertaking on-screen question and answer interactions.

Chapter 12

Study: Training and evaluating COMPASS

12.1 Introduction

This chapter presents an empirical evaluation of COMPASS (Chapter 11).The chapter discusses a study conducted at Manchester Metropolitan University in which 44 students were recorded whilst undertaking on-screen e-learning activities. The study examines whether a computerised analysis of e-learner non-verbal behaviour can be used to train and test an artificial neural network classifier to produce 'comprehension' and 'non-comprehension' classifications at above chance levels of accuracy. The COMPASS classifier was trained and tested to assess the accuracy of comprehension and non-comprehension classifications. The trained comprehension classifier achieved normalised classification accuracy of 75.8%.

Section 12.2 presents the research question investigated in the study and section 12.3 the study procedure, ethics and participant demographics. The study method is presented in section 12.4, with results and discussion presented in section 12.6. Conclusions are shared in section 12.7.

This study has been detailed in Holmes et al. (2017b).

12.2 Research question

The study presented in this chapter investigates one research question:

1. Is the normalised accuracy of COMPASS binary comprehension classifications above 50%?

12.3 Study procedure

The study uses data collected from 44 undergraduate students at Manchester Metropolitan University (MMU) undertaking e-learning activities within a bespoke e-learning environment. Students were asked to complete a 21-question multiple choice quiz on Java programming, logic and information systems diagrams, while being recorded using a front-facing web camera attached to one of the pre-configured laptops provided for the experiment.

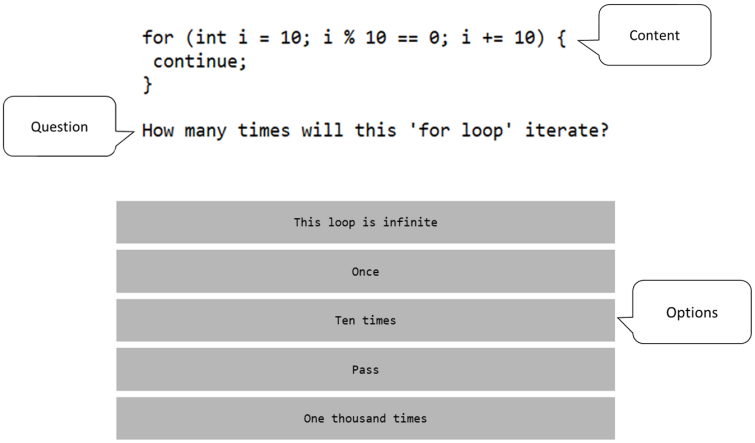


Figure 12.1 Screen shot of quiz system

The bespoke quiz system developed for the experiment (Figure 12.1) displayed a series of multiple choice questions in a random order. During each question answer response period the learner was recorded using a front-facing web camera positioned on top of the monitor (Figure 12.2). The image stream from the web camera was saved against the question answer provided by the

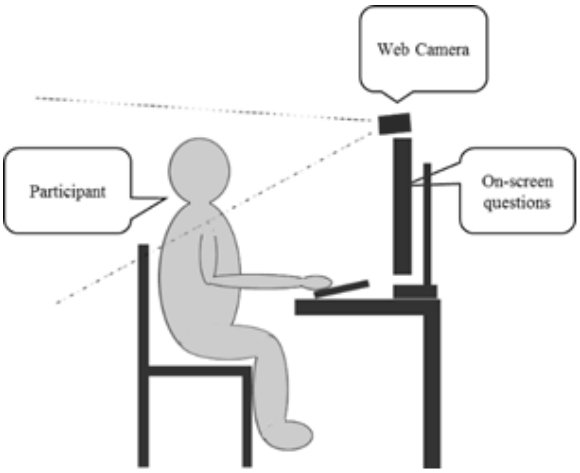


Figure 12.2 Physical layout of experiment

learner. The experiment yielded a large data set of learners, question answers and associated temporally sequential images.

The image stream data for each question-answer response period was then broken into 1 second tranches and analysed to extract a cumulative behavioural feature vector (CBFV) of 42 variables containing 37 statistical behavioural observations, 5 learner meta-data constants and 1 system state input variable for each 1 second of observation. The CBFV for each 1 second was saved to disk along with the answer score for the response period it came from. A multi-layer perceptron artificial neural network was then trained to classify comprehension from CBFV by back-propagation of errors (Haykin, 1994). The network was trained and evaluated using 10-fold cross validation.

12.3.1 Ethics

Participants were required to sign a consent form prior to participation. The consent agreement detailed the information collected, its research use, and how the information would be securely stored and distributed. Personally identifiable information about participants, including images, will not be made public. In this experiment, participation was compensated by means of a retail voucher.

12.3.2 Participant information

44 student volunteers were randomly selected from the Science and Engineering Faculty at Manchester Metropolitan University (MMU). The participant group was diverse, with a mix of ages, ethnicities and programming experience, so as to reflect the student body in computing related subjects. Participants' ages ranged from 18 to 38, with a mean average age of 21 years old. Of the 44 participants, 40 were enrolled on computing or computer science related courses, while 4 were enrolled on other science, mathematics or engineering courses. 39 participants identified that they had prior programming experience, while 5 identified no prior experience.

To represent the inherent diversity of the student body, the participation is ethnically diverse. The ethnic demographics of the group are shown in Table 12.1. Ethnicity may be of particular importance as literature (Bond et al., 1990; Buckingham et al., 2012; Rothwell et al., 2006) suggests that ethnicity, and culture, can play important role in mediating subconscious non-verbal behaviour. Literature suggests that the classifier may learn slightly different patterns of behaviour for each demographic group.

Table 12.1 Ethnic demographic groups

Group	Label	Count	Percentage
1	Asian or British Asian	15	34
2	Black or Black British	1	2
3	Mixed or multiple ethnic groups	1	2
4	Other ethnic group	0	0
5	White or White British	26	59
6	Undisclosed	1	2

12.4 Method

12.4.1 Data collection

As a consequence of findings detailed in Chapters 10 and 10.7, a more concise training and testing data set was required. Findings highlighted in section 10.8 indicated that behaviours in the training set needed to be constrained and that the classifier trained and tested using only behaviours immediately resultant from the comprehension or non-comprehension of on-screen information. To achieve this, the data set should be collected using a simpler on-screen system with fewer activities. The participant should simply be shown some information, their response recorded and a ground truth comprehension measurement taken. A bespoke multiple choice quiz system was developed to collect the data required.

The participant learners completed a 21-question multiple-choice quiz covering topics related to Java programming, logic and information systems diagramming. For each question period the learner was recorded using a front-facing web camera. As in figure 12.2, the camera was positioned on top of the PC monitor and directly facing the learner. Participants undertook the experiment individually and without interruption but were situated in a semi-public space within the university foyer. Learners used a bespoke quiz system (figure 12.1), designed for the experiment. The quiz system began by capturing information about the learner including age, gender, ethnicity and academic level, whether they were wearing glasses, had prior programming experience or were enrolled on a computing related degree course.

The quiz started by asking a control question to establish a baseline for positive comprehension. The question 'how old will you be in 4 years time?' was included to ensure that the dataset contains some authentic comprehension results. The multiple-choice options for the question were generated based on information captured during set up. The 20 on-topic questions were presented

in a random order so as not to allow fatigue to bias responses to any individual question. For each question the list of answer options were presented in a random order to prevent on-screen layout biasing behaviour for specific correct or incorrect options. A 3-second countdown was displayed between each question, to prevent behavioural overlap in different answer periods. Each question was presented with two or more answer options and a 'pass' option. Pass was included to try to minimise guessing behaviour. For each answer a score of -1 (incorrect) or +1 (correct) was recorded in the database. A pass selection recorded a -1 score, with a flag to denote no attempt at answering was made. A recording of the learner was made for each question asked. For each question the web camera began recording as soon as the question-content was presented on-screen. Recording stopped when an answer option was selected. A unique answer ID was used to link the recording to the database record for the learner's given answer.

Questions within the quiz were based on the first year undergraduate computer science syllabus and were evaluated by programming course lecturers at MMU. Questions fell into three categories on Bloom's revised taxonomy (**?**) - *Remembering*, *Understanding* and *Analysing*. Questions were designed to provide more or less challenge, in line with Bloom's cognitive domain. Examples of quiz questions and taxonomic categories are shown in Table 12.2.

Table 12.2 Example of questions in Bloom's revised taxonomy of the cognitive domain

Question	Category
In relation to Java programming, what does the acronym JVM stand for?	Remember
$\forall x(P(x) \wedge Q(x)) \equiv \forall x P(x) \wedge \forall x Q(x)$ Is this statement true or false?	Understand
Given the constructor $for(inti = 10; i\%10 == 0; i+ = 10)$ how many times will this loop iterate?	Analyse

12.4.2 Creating a training data set

The data collection phase of the experiment (section 12.4.1) produced a large data set of web camera recordings and answer scores. To develop a training data set for the COMPASS classifier it was necessary to extract the non-verbal behaviours defined in the COMPASS behavioural data model (section 11.6) from each one second of web camera footage, for each answer response period, for each participant. Each behavioural data model for each one second of analysed web camera footage is referred to as the Cumulative Behavioural Feature Vector (CBFV) (section 11.6).

A console application was developed to process the answers and web camera footage collected during the data collection phase of the experiment (section 12.4.1). Figure 12.3 shows the process for indexing the CBFV for answer periods. The indexing process outputs a single comma separated value (CSV) file with each row containing a single 42 variable CBFV plus a comprehension target label.

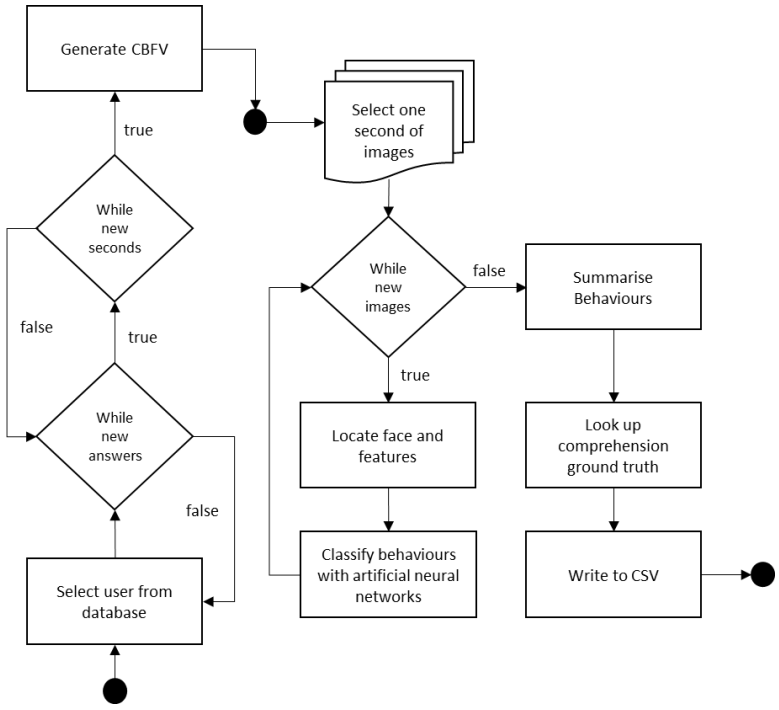


Figure 12.3 CBFV indexing process diagram

12.4.3 Training and evaluating the classifier

A multilayer perceptron artificial neural network was coded, tuned, trained, and tested. The network was configured with 42 input nodes, a single hidden layer containing 20 fully connected nodes (equation 12.1), and a single output node (equation 12.2). Network weights were initialised to 0±1/fan-in where fan-in is the number of inputs to the neuron. Binary comprehension classification is performed by application of a threshold function to the network output (equation 12.3, where t is a threshold value in the range 0.0 to 1.0).

$$h_i = \sigma \sum_{j=1}^{n=42} w_{ij} x_j + b_i \tag{12.1}$$

$$out = \sum_{i=1}^{n=20} w_i h_i + b \tag{12.2}$$

$$class = \left\{ \begin{array}{ll} 1.0 & tanh(out) \geq +t \\ \text{-1.0,} & tanh(out) \leq -t \\ 0 & \text{otherwise} \end{array} \right\} \tag{12.3}$$

The dataset of comprehension labelled CBFV was split into two sets, training (Tr = 90%) and testing (Te = 10%). The classifier was initially trained using 10-fold cross validation, with each CBFV appearing once in the test set. For each fold the classifier was trained by back-propagation of errors for a maximum of 2000 epochs (Haykin, 1994). Training was halted early if training set mean square error (MSE) did not reduce over 200 epochs. The MSE was checked at 100 epoch intervals. If early stopping was triggered, the last best configuration of weights was saved to disk. The performance of the classifier has been evaluated by the average classification accuracy and precision over the 10 folds.

Literature (Bond et al., 1990; Rothwell et al., 2006) suggests that ethnicity plays an important role in mediating subconscious non-verbal behaviour. To establish the effect ethnicity has on classifier accuracy, the training and evaluation was repeated using only data for the predominant (59%) ethnic class group 5 (Table 12.1).

12.5 Data

The 44 participants completed 869 questions and generated 185,075 web camera video stream images for analysis. Not all participants answered the full 21 questions. In two cases volunteers' other commitments prevented completion and in one case the application crashed. Incomplete quiz data was included in the dataset as the random ordering of questions prevented per-question bias. Table 12.3 shows a breakdown of the data collected in both correct and incorrect answer classes. Learners can opt to *pass* on a question. *Pass* responses are treated as incorrect answers.

Table 12.3 Overview of learners' question answer data

	Correct answer given	Incorrect answer given	Total
Answers	500	369	869
Footage (seconds)	6,836	5,503	12,336
Images	102,535	82,540	185,075
Class %	55.42	44.58	100.00

Web camera images were captured at 15 frames per second. If a full feature set was found in each image for the 1 second period, it would produce a matrix in the dimensions 42 by 14 for each 1 second of footage. However, not all frames yield a full feature set. For example, the face or features may become obscured by an object or by rotation of the head. When the behavioural model

cannot be populated due to missing behavioural data, processing on the frame is ended and the matrix is discarded.

Table 12.4 Breakdown of cumulative behavioural feature vectors extracted from question response periods

Comprehension	CBFV extracted	% of data
+1.0	3,551	53.86
-1.0	3,041	46.13
Total	6,592	100.00

The extraction process is repeated for each image in the given time window, producing a matrix of feature vectors which is the summarised to a Cumulative Behavioural Feature Vector (CBFV), as described in section 11.6. Table 12.4 shows the number of CBFV successfully created in each comprehension class.

The +1/-1 class distribution for CBFV (Table 12.4) is close to that of the source (Table 12.3), indicating consistent behaviour extraction across comprehension classes. The CBFV produced for each second will be the input vectors for comprehension classifier training and testing (section 12.6).

12.6 Results and discussion

Table 12.5 shows the training (Tr), validation (va) and test set (Te) mean square error (MSE) for each of the 10 folds, along with the epoch at which the minima was found. The high average MSE suggests that there is a degree of noise within the model. The authors anticipated the model would be noisy, as the training data set would contain many weak examples of the label. The threshold function (equation 12.3) allows for noise, weakly indicative patterns, to be excluded.

Table 12.6 shows performance statistics (equations in 12.4) for true positive (TP), true negative (TN) and normalised classification accuracy (CA), as

Table 12.5 10-fold cross validation training

	All data			**Ethnic group 5 only**		
Fold	Epoch	Tr MSE	Te MSE	Epoch	Tr MSE	Te MSE
0	1100	0.74	0.95	800	0.77	1.00
1	600	0.75	0.93	1300	0.72	0.91
2	400	0.80	0.97	800	0.73	1.03
3	2000	0.65	1.07	1900	0.64	1.06
4	800	0.76	1.01	1600	0.70	1.11
5	1000	0.74	0.94	200	0.84	0.92
6	400	0.77	0.93	1200	0.70	1.03
7	800	0.74	0.99	700	0.74	0.96
8	400	0.82	0.98	1500	0.66	0.97
9	1500	0.67	1.04	1500	0.70	0.94

Table 12.6 Classifier performance

	Training			Testing		
Threshold	TP (%)	TN (%)	CA (%)	TP (%)	TN (%)	CA (%)
All data						
±0.6	88.8	78.1	83.5	79.4	62.2	71.7
±0.8	91.3	81.5	86.4	81.3	64.7	73.9
±0.9	92.3	81.2	86.7	81.0	65.3	74.1
Only data from ethnic group 5						
±0.6	88.2	81.3	84.7	74.2	64.7	69.9
±0.8	90.5	84.8	87.6	77.2	68.0	73.1
±0.9	91.7	85.6	88.6	78.8	72.2	75.8

percentages, averaged over 10 folds with selection thresholds ranging ±0.6−±0.9. The results in Table 12.6 show that accuracy increases as the threshold is raised.

$$\begin{aligned} TP &= \frac{tp}{tp+fn} \\ TN &= \frac{tn}{tn+fp} \\ CA &= \frac{tp+tn}{tp+fp+tn+fn} \end{aligned} \tag{12.4}$$

The results in Table 12.6 show that when the classifier is trained using data from different ethnic groups the test set classification accuracy for non-

comprehension (TN) behaviour is weaker than when trained on a single group. The results support the suggestion from literature (Bond et al., 1990; Rothwell et al., 2006) that stress response NVB differs by demographic grouping. Literature also suggests that gender should be considered; however there were an insufficient number of female participants in this experiment to isolate the variable.

Table 12.7 Test set confusion matrices for group 5 classifier

Threshold ±0.8		Prediction		
		Positive	Negative	
Observation	Positive	602	178	77.2%
	Negative	199	422	68.0%
		75.2%	70.3%	
Threshold ±0.9		**Prediction**		
		Positive	Negative	
Observation	Positive	304	82	78.8%
	Negative	86	223	72.2%
		77.9%	73.1%	

The trade-off in application therefore relates to accuracy versus frequency of classification. With a lower threshold, the network classifies a greater number of the CBFV but at the expense of accuracy, or vice versa. This effect is evident when comparing the count and accuracy of classifications shown in the confusion matrices 12.7.

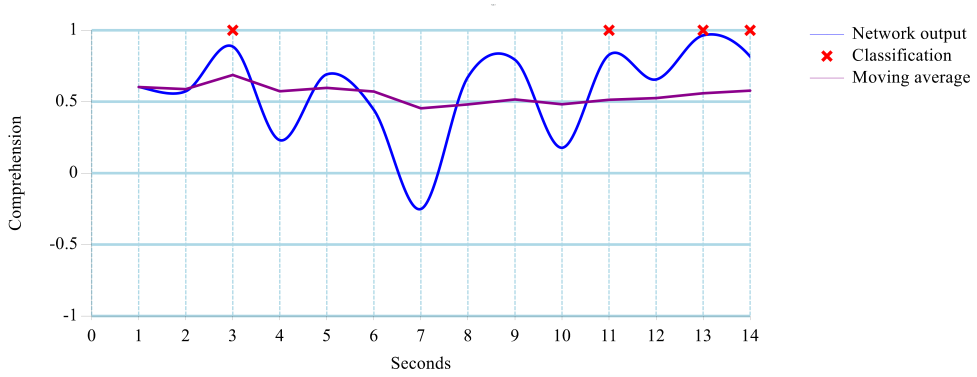


Figure 12.4 COMPASS time-series for a correct answer period (Holmes et al., 2017b)

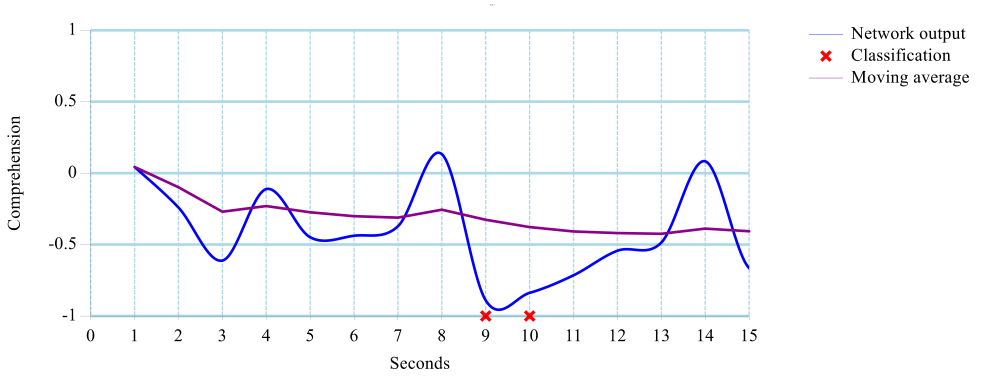


Figure 12.5 COMPASS time-series for an incorrect answer period (Holmes et al., 2017b)

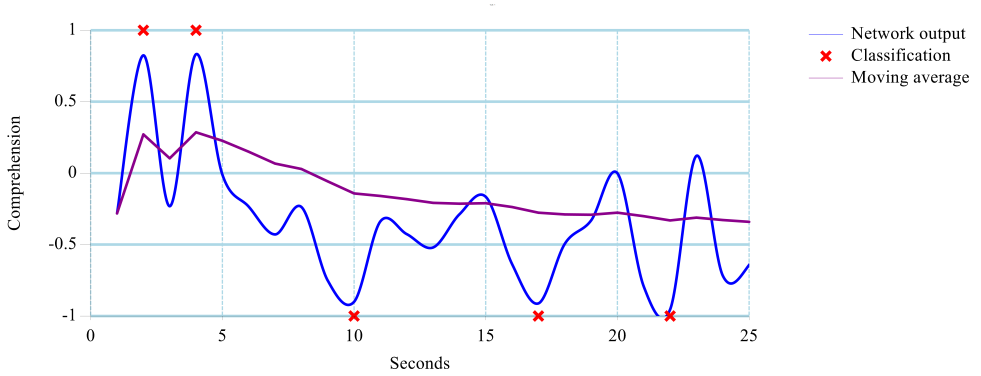


Figure 12.6 COMPASS time-series for an incorrect answer period with change of behaviour (Holmes et al., 2017b)

Figures 12.4, 12.5 and 12.6 show COMPASS real-time comprehension time-series data for the group 5 only data subset. While the time-series for figures 12.4 and 12.5 show consistent comprehension indicative behaviour, matching the outcome of the interaction, figure 12.6 shows how COMPASS can track a learner's comprehension indicative behaviour changes as they process information from the screen. Figure 12.6 highlights the value of the system, as COMPASS is able to identify the point at which the learner shifts from comprehension to non-comprehension during an answer period. The strong change in behaviour shown in figure 12.6 at 10 seconds could be used to trigger an appropriate intervention in the learning process.

12.7 Conclusion

This chapter has presented the development and evaluation of a computational approach to real-time learner comprehension analysis during on-screen information processing, using a combination of image processing and artificial neural networks.

The contribution of the research discussed in this chapter is to demonstrate that non-verbal behaviour, as drawn from literature in section 6.2 and discussed in implementation in section 11.6, can be used as an effective indicator of comprehension and non-comprehension states (defined in section 3.2) during reading of on-screen information in an e-learning environment, without the need for specialist equipment or body-attached sensors used by researchers in literature (Bednarik and Tukiainen, 2006; Chen et al., 2014; Copeland et al., 2014; D'Mello and Graesser, 2010; Gerjets et al., 2014; Yusuf et al., 2007) (see discussion in section 3.4.2).

The chapter has presented methods for extracting a behavioural data model from web camera image streams using a combination of Haar cascades, artificial neural networks, and geometries. The paper has presented methods for training and evaluating a multilayer perceptron network (MLP) to classify behavioural patterns as indicative of comprehension or non-comprehension.

The results presented in this paper show that the methods are effective in extracting a data model of non-verbal behaviour from web camera image streams, and that a MLP is an effective tool for classification of the behavioural model. The results suggest, in support of literature, that comprehension indicative non-verbal behaviour does differ for demographic groups.

The results show that the application of a logistic function to the MLP output allows for tuning of classification accuracy, by exclusion of patterns which are weakly indicative. The results identify a threshold of ± 0.9 as optimal for classification accuracy where the classifier is trained on individual demographic groupings, achieving a test set normalised classification accuracy of 75.8% and

average precision of 75.5%. Analysis of the COMPASS time-series for question-answer periods demonstrates how the classification system could be used as a real-time feedback channel for an adaptive e-learning platform, enabling timely and appropriate interventions in the learning process.

However, there are a number of limitations to this study. The demographic make-up of the participant group made it difficult to fully assess the effect that demographic variables have on indicative patterns of behaviour. There were too few female participants to evaluate the effect of gender on behaviour. The study also does not address how question type (Table 12.2) might promote differing behaviour. In future work these two questions should be addressed.

Chapter 13

Hendrix 2.0

13.1 Introduction

Hendrix 2.0 is an experimental adaptive CITS that uses near real-time modelling of e-learner non-verbal behaviour to classify comprehension states during conversational question-answer interactions and increase the specificity of guidance offered to learners in response to perceived *non-comprehension*.

The innovative conversational intelligent tutoring system, Hendrix 2.0, uses comprehension classification as a decision making feedback channel to enact an important pedagogic tenet of cognitive apprenticeship - demonstration of competence. The system withholds explicit guidance on overcoming challenges to encourage self-directed problem solving. Hendrix 2.0 can then intervene in the learning process if the student expresses behaviour associated with non-comprehension or impasse.

By combining automatic comprehension classification and conversational intelligent tutoring, the system is able to effectively mimic human tutors ability to adapt discourse, instruction and pedagogy in response to experiential understanding of learner behaviour.

This chapter presents an overview of the Hendrix 2.0 system, discussing integration and improvement of the Hendrix 1.0 (Chapter 7) and COMPASS

(Chapter 11). The algorithm for comprehension assessment and pedagogic intervention during conversational tuition is detailed (section 13.8).

13.2 Contributions

The contributions of this system are:

1. Near real-time modelling and classification of e-learner comprehension during conversational virtual tutoring

2. Adaptation of user interface to show *recommendations* based on near real-time comprehension classification

3. Adaptation of tutorial discourse based on near real-time comprehension classification

13.3 Comprehension classifications

Hendrix 2.0 performs two types of micro-adaptation (section 4.8) in response to classifications of 'non-comprehension'. As discussed in Chapter 12, comprehension classification accuracy is tuned using the minimum threshold value. In this research two thresholds. In this work the terms 'weak' ($\pm$ 0.4) and 'strong' ($\pm$ 0.8) are used to describe the *certainty* of classification.

13.4 Key functionality

Hendrix 2.0 is a conversational intelligent tutoring system designed to tutor computer programming. Hendrix 2.0 advances work reported in Chapter 7 by integrating near real-time e-learner comprehension modelling and classification (Chapter 11) as part of an adaptive algorithm. Hendrix 2.0 inherits the functional specifications of Hendrix 1.0 (section 7.3.2) and COMPASS (section 11.4) in addition to new functionality designed to enhance the contextual

relevance of micro-adaptive behaviours (section 4.8) by incorporating non-verbal behaviour as an indicator of reading comprehension.

- **Model e-learner NVB during conversational tutoring**

  A discrete summary of e-learner non-verbal behaviour is produced by analysis of live web camera stream data during conversational question and answer interactions between the tutor and learner.

- **Classify comprehension in near real-time during conversational tutoring**

  A comprehension classification is produced for each discrete behavioural summary over the time period of each question and answer interaction between tutor and learner, producing a time-series of comprehension measurements.

- **Adapt tutorial conversation based on comprehension classification**

  When set to adaptive mode, Hendrix 2.0 will monitor comprehension classifications in real-time. If a strong non-comprehension classification is made, Hendrix 2.0 will adapt the planned dialogue to include an additional hint dialogue. See section 13.8.2.

- **Adapt the tutorial interface to show *recommendations* based on comprehension classification**

  When set to adaptive mode, Hendrix 2.0 will monitor comprehension classifications in real-time. If a weak non-comprehension classification is made, Hendrix 2.0 will display a set of 'recommended questions' for a learner to ask. The recommendations are automatically generated from the current conversational context and will prompt Hendrix 2.0 to provide either a definition or an example of a word or concept. See section 13.8.1.

- **Capture participant demographic information**

  To allow for evaluation of effectiveness across tranches of the participant group, the system must capture demographic information including gender, ethnicity and academic level.

- **Validate face and feature location in web camera stream**

  To ensure that comprehension data is collected during tutorial sessions, Hendrix 2.0 must validate face and feature detection during initialisation of the software.

- **Record dialogue, questions, answer scores, full behavioural model data and classification time-series in a database**

  The system must record full accounts of each participants data including all chat dialogue, questions, answer scores, full behavioural model data and both raw comprehension classifier output and classifications.

13.5 Requirements

Hendrix 2.0 will be deployed for experimental purposes on PC hardware within the university building. The system inherits the technical specification described in section 7.3.2, in addition to:

- **Use a configuration file to enable adaptation**

  The software must run in both adaptive and non-adaptive modes: 1) non-adaptive mode that produces and records comprehension estimates and classifications but does not adapt; 2) adaptive mode that produces and records comprehension estimates and classifications and enacts pedagogic adaptation based on classifications. Configuration switches will allow for a single Hendrix 2.0 solution to be used for both control and experimental groups.

- **Use a configuration file to set the neural network to use**

  The software must be able to load in a trained artificial neural network

to produce estimates and classifications of e-learner comprehension from a local file location. This functionality allows the classifier to be easily selected prior to experimentation without code modification.

- **Use a configuration file to set classification thresholds**

  The software must be able to call upon configurable classification thresholds. This functionality allows comprehension classification boundaries to be easily configured without code modification, prior to experimentation.

13.6 User Interface

The Hendrix 2.0 CITS uses a chat style interface to deliver conversational tutoring. The wireframe diagram (figure 13.1) shows how screen real-estate is used for the various components of the main Hendrix 2.0 interface. List numbering relates to numbering in wireframe diagrams 13.1 and 13.2.

1. **Menu**

   The menu will allow for easy configuration of Hendrix 2.0 during experimentation including options to toggle video recording, comprehension classification and adaptation. These options allow for creation of control and experimental versions of the software for experimentation.

2. **Logo**

   The Hendrix 2.0 logo will be presented in the top right of the chat window.

3. **Chat history**

   The chat history is presented in chronological order within a scrollable panel.

4. **Current dialogue**

   The current dialogue shows the latest message from Hendrix 2.0. This message is always the dialogue to which the learner will respond.

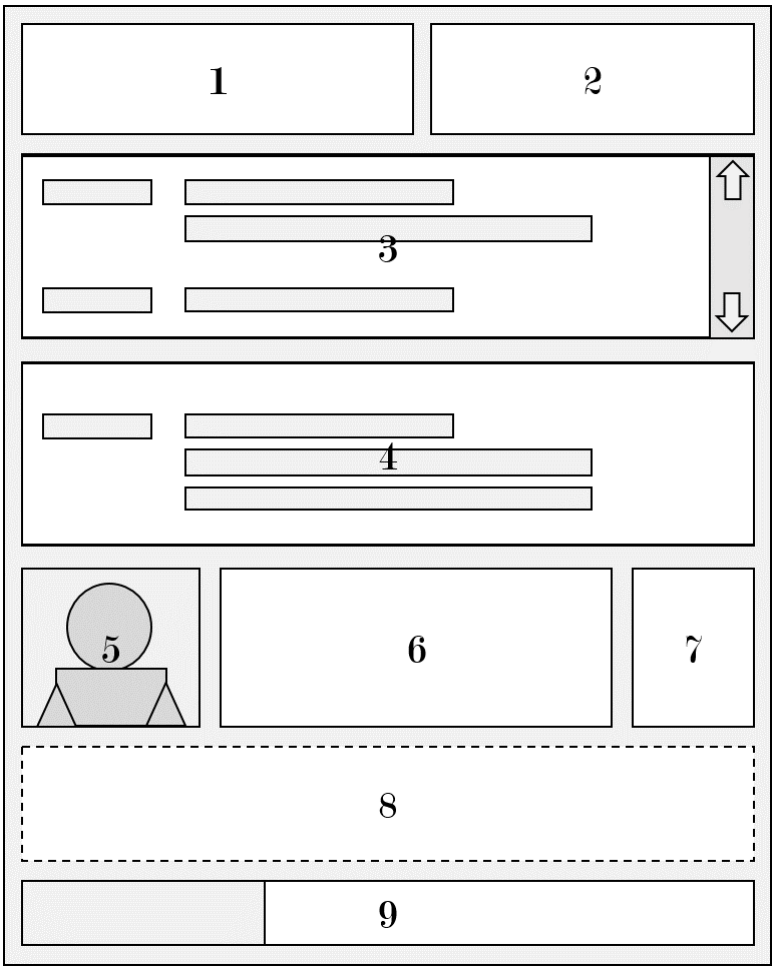


Figure 13.1 Wireframe for the Hendrix 2.0 chat interface

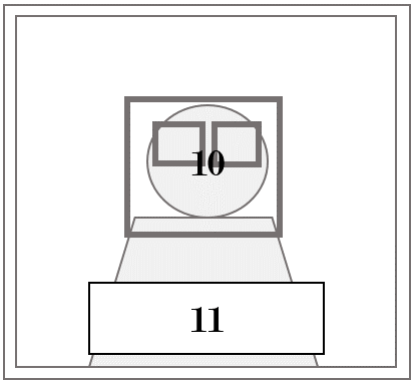


Figure 13.2 Wireframe for the Hendrix 2.0 camera verification

5. **Web camera**

   The web camera image stream is shown in real-time next to the text input field.

6. **Text input field**

   The text input field allows a learner to type dialogue to send to Hendrix 2.0 by pressing the return key or clicking on the submit button.

7. **Submit button**

   The submit button sends the text in the text input field to Hendrix 2.0.

8. **Recommendations area**

   If *WEAK non-comprehension* is detected a set of *recommendations* will be displayed below the chat interface. These *recommendations* are click-able phrases which will automatically prompt Hendrix 2.0 to provide information. Recommendations are generated automatically using keyword analysis of recent discourse.

9. **Tutorial progress bar**

   The tutorial progress bar displays the learner's progression through the tutorial as a percentage completed.

10. **Camera preview and facial feature highlighting**

    A set-up window allowing the learner to configure their web camera and verify face and feature detection is functioning.

11. **Accept face and feature detection and proceed to tutorial**

    A proceed button which is enabled once a face and both eyes are highlighted.

Figure 13.3 shows a screen-shot of the Hendrix 2.0 chat interface. The implemented desktop application follows the wireframe specification (figure 13.1).

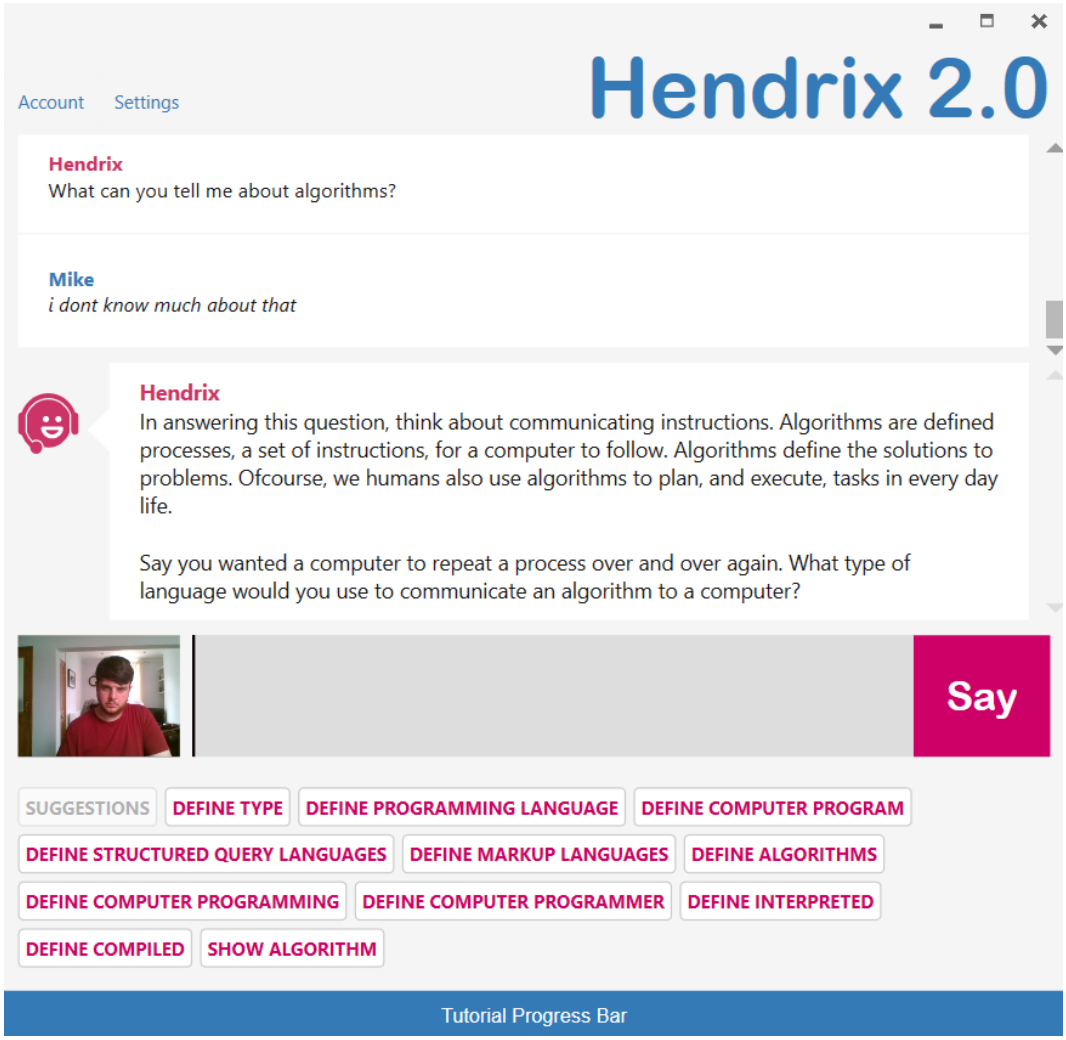
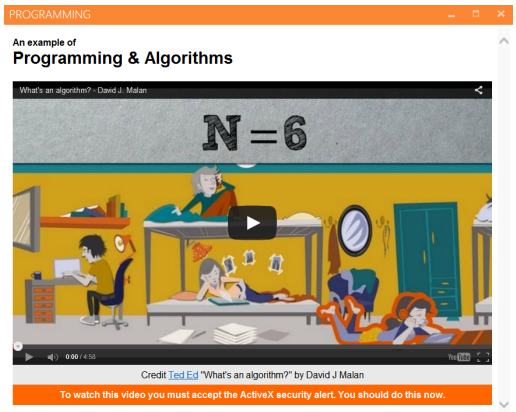
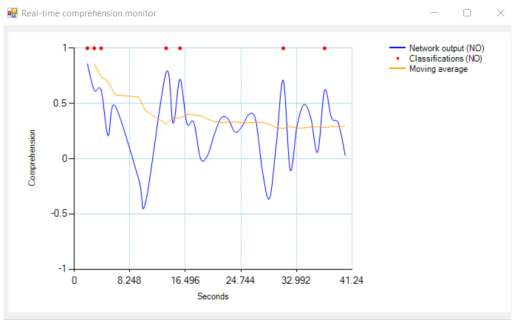


Figure 13.3 Screen-shot: example of the Hendrix 2.0 chat interface during a tutorial

In addition to the main chat interface the Hendrix 2.0 desktop application has two additional secondary windows. The first is a browser window, able to load and display supporting content such as code samples and multimedia (figure 13.4a). The browser window can be opened by Hendrix 2.0 in response to a user request or to support a question. The second additional window is the comprehension monitor window (figure 13.4b). The comprehension monitor window is a debug tool to give visibility on the performance of COMPASS during testing and experimentation. The comprehension monitor window is not intended to be visible to the learner during the experiment.



(a) Screen-shot: example of the Hendrix 2.0 supporting content window



(b) Screen-shot: example of the Hendrix 2.0 real-time comprehension monitor during tutorial questioning

Figure 13.4 Hendrix 2.0 tertiary windows

13.7 Software architecture

Hendrix 2.0 has a modular architecture, integrating four major system modules - Hendrix 2.0 CITS, COMPASS comprehension modelling and classification, a persistent data context and an integration layer to mediate between Hendrix 2.0 and COMPASS. Figure 13.5 shows the system architecture, detailing the relationship between Hendrix 2.0 and COMPASS. Each major component in figure 13.5 will now be described.

13.7.1 Hendrix 2.0

This section details the major changes in function between Hendrix 1.0 and Hendrix 2.0.

Conversation

Building on the Hendrix 1.0 platform (section 7.4) the Hendrix 2.0 system components offer very similar functionality to that described in Chapter 7, concerned with structuring and coordinating the conversational interactions necessary to deliver goal-oriented tuition. In this subsection the author discusses the improvements and changes made to the system.

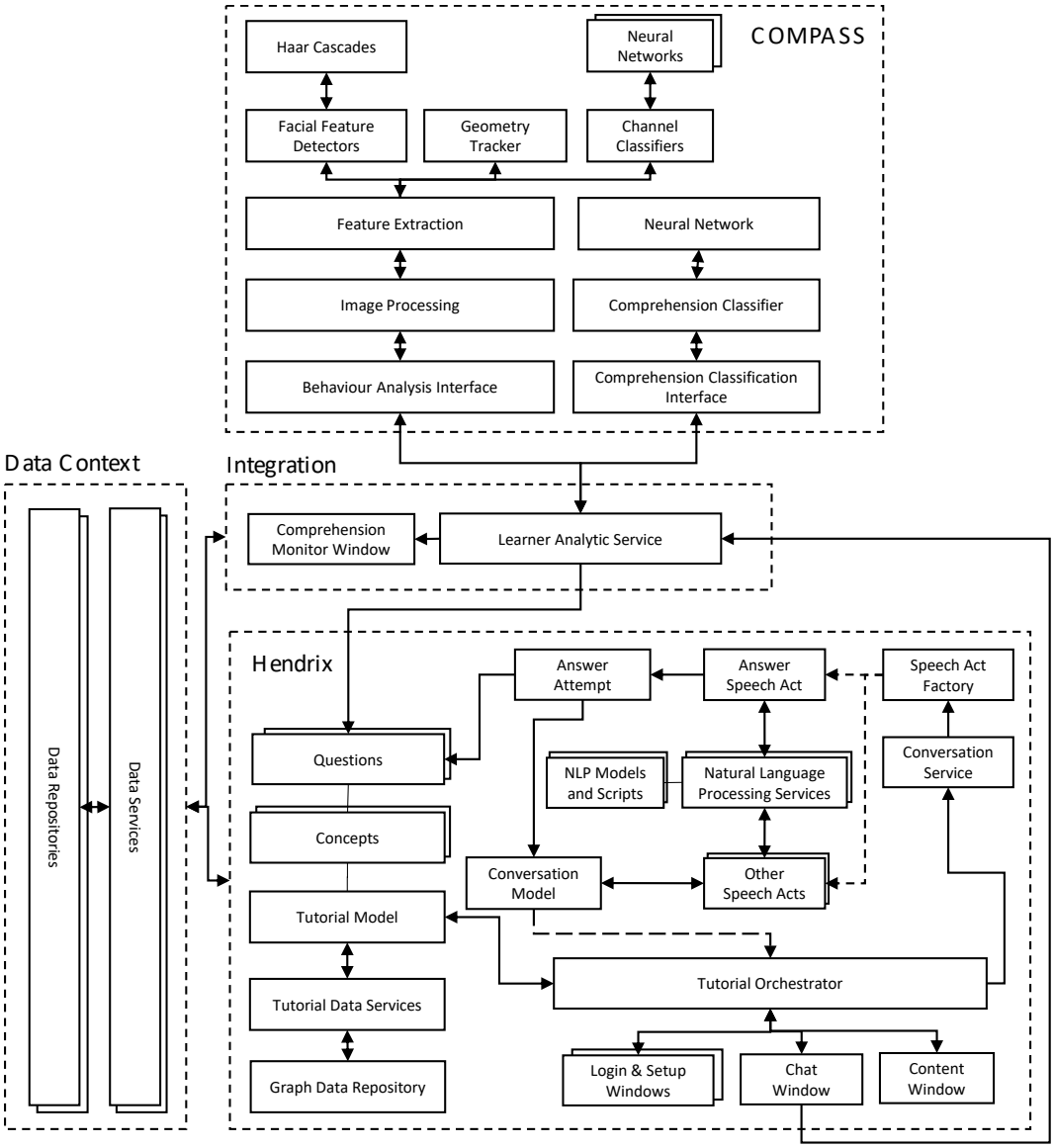


Figure 13.5 Hendrix 2.0 system architecture

As shown in wireframes (section 13.6) Hendrix 2.0 uses a number of graphical user interfaces to capture and present tutorial content and conversational interactions. The primary user interface is a chat-style incorporating a web camera feed.

Improving on the procedural algorithms of Hendrix 1.0 (section 7.4), Hendrix 2.0 uses speech act classification to model conversation dynamically. The flow of conversational behaviour for Hendrix 2.0 is not pre-defined, but determined by the sequence of speech acts, events, raised in conversation.

As shown in figure 13.5 Hendrix 2.0 directs learner dialogue through from the chat window, via the tutorial orchestrator and conversational service, to the speech act factory. The speech act factory classifies the speech act using the algorithm defined in section 19.

Each speech act contains functionality for that specific act. Hendrix 2.0 processes each speech act in sequence to follow the context of the conversation. The shift from procedural programming to event-driven programming (Petrusha, 2017) reduces the complexity of dialogue processing algorithms by avoiding combinatorial complexity in conditional statements and encourages separation of concerns.

A benefit of decoupling implementation of speech act processing and dialogue flow, is the ability to easily extend the conversational functionality. In Hendrix 2.0, adding a new speech act and associated functionality can be done without changing any existing code. A new script file of patterns is added along with a new speech act class containing the relevant functionality. Hendrix 2.0 is immediately able to identify and deploy the new speech act.

Knowledge

As with Hendrix 1.0, the domain model (see section 7.4.2) is represented in a graph and index searches. As with Hendrix 1.0, tutorials are structured by calculating the *shortest-path* (section 7.4.2).

An improvement has been made to the knowledge representation in Hendrix 2.0 by adding a new graph entity *Misconception*. In addition to each question having *Answers*, they also have *Misconceptions*. *Misonceptions* contain answer patterns representing anticipated mistakes, along with feedback appropriate to correct the mistake. Mistakes were anticipated by observation of conversational logs from the first pilot study (Chapter 8) where questions remained unchanged, by detailing the inverse of answers, or from personal experience discussing programming techniques with novice programmers. Including this

new knowledge entity in the graph allows Hendrix 2.0 to provide more specific support when learners answer incorrectly and is anticipated to improve the effectiveness of the tuition delivered.

Comprehension awareness

To support adaptation in response to comprehension states, the *Question* tutorial entity has been updated to access the *Learner Analytic Service*. When a *Question* entity is awaiting an *Answer Attempt*, the *Question* entity polls the *Learner Analytic Service* to view the current comprehension state of the learner. If the comprehension state of the learner is *strong non-comprehension* then an additional support dialogue is queued. The additional support dialogue is displayed only once the learner starts to type their answer, so as not to interrupt ongoing problem-solving.

13.7.2 COMPASS

COMPASS (Chapter 11) is a novel system for estimating e-learner comprehension in real-time by automatic analysis of non-verbal behaviour. COMPASS uses non-intrusive measurement of multiple channels of non-verbal behaviour, including feature state, facial movement, posture and physiological change, to model and classify e-learner comprehension in real-time. COMPASS allows Hendrix 2.0 to model and classify learner comprehension in real-time, during conversational interactions. Hendrix 2.0 is able to feed COMPASS image data captured from a USB web camera and receive, in return, access to time-series of comprehension estimates and classifications. Hendrix 2.0 uses COMPASS as a decision making feedback channel on which to based intelligent adaptations to the user interface and conversational content used during tutoring.

13.7.3 Integration

The Integration layer, shown in the architecture diagram (figure 13.5), performs maintenance and brokering of communications between the two systems, Hendrix 2.0 and COMPASS. The *Learner Analytic Service* contains novel algorithms for maintaining information on the learner's comprehension state during question-answer interactions, as described in section 13.8. The *Learner Analytic Service* makes comprehension state information available to Hendrix 2.0 so that adaptation can be performed.

When a new question-answer interaction is initiated, the *Learner Analytic Service* starts to accept web camera image data from Hendrix 2.0 one second at a time. The *Learner Analytic Service* then feeds the raw image data through the COMPASS *Behavioural Analysis Interface* to produce a Cumulative Behavioural Feature Vector (CBFV) (see sections 11.6 and 11.7). The CBFV is then classified using the COMPASS *Comprehension Classification Interface*, returning a comprehension estimate between -1.0 and +1.0 and a comprehension classification based on threshold values. The comprehension classification algorithm is detailed in section 11.8.

Each CBFV and associated comprehension classification is assigned to a time-series for the current question-answer interaction time period. At the end of the question-answer period the time-series is saved to the current *data context*.

13.7.4 Data context

The *data context* provides a consistent data layer maintaining objects and states of data entities within the system. Using .NET Entity Framework data objects created, modified or destroyed are automatically updated or saved within the database. Using this approach allows the Hendrix 2.0 database to show a complete representation of system state, facilitating greater visibility of experimental data.

13.8 Adaptation

As discussed in review of literature (section 4.8.2), (VanLehn et al., 2017) highlights six micro-adaptive behaviours. While their ITS Dragoon implements just three of the behaviours, Hendrix 2.0 implements five of the six, replacing the concrete articulation strategy (Heffernan and Croteau, 2004) of Dragoon (VanLehn et al., 2017) with a verbal articulation strategy (Heffernan and Croteau, 2004) more similar to that of OSCAR (Latham et al., 2012b).

1. **Feedback and hints on the model**

   Hendrix adapts to give hints on the learner's mental model by searching answers for both missing words (gaps in knowledge) and incorrect words (misconceptions).

2. **Feedback hints on the learner's process (meta-tutoring):** Comprehension adaptive hints are written to help the learner adapt how to approach the current problem. See (XX) for an example dialogue.

3. **Reflective debriefing**

   Hendrix summarises what the learner got right and what was wrong in the discussion.

4. **Decomposition of tasks**

   Hendrix models tutorials as a graph of concepts, breaking concepts into problems a learner can explore. Each problem is decomposed into conversational steps, with dialogue steps extended where solutions exclude relevant information or contain incorrect information.

5. **Answering student questions**

   Hendrix can answer questions based on its own knowledge graph.

The key contribution of Hendrix 2.0 is the ability to enact adaptive discursive interventions based on real-time classification of e-learner comprehension. As identified in literature (section 3.4), no current research addresses estimation and

classification of comprehension states during processing of on-screen information in an e-learning platform.

Extending the successful implementations of the Hendrix 2.0 CITS (Chapter 7) and the COMPASS real-time comprehension classification system (Chapter 11), Hendrix 2.0 integrates the two systems to monitor e-learner comprehension during conversational tutoring and enact timely interventions in the learning process, based on real-time comprehension classifications.

While COMPASS (Chapter 11) tracks both comprehension and non-comprehension, pedagogic intervention is based on real-time identification of *non-comprehension* events. Hendrix 2.0 intends to help learners overcome non-comprehension by enacting interventions of increasing specificity. COMPASS classifies two levels of non-comprehension, *weak non-comprehension* and *strong non-comprehension* (Chapter 11).

1. **Weak non-comprehension adaptation**

   When non-comprehension classifications are *Weak*, Hendrix 2.0 updates the user interface to show a list of suggested questions that a learner could ask of Hendrix 2.0 to further their understanding of the current topics under discussion. The intervention is non-intrusive and does not act to forcefully change the current conversational objectives.

2. **Strong non-comprehension adaptation**

   When non-comprehension classifications are *Strong*, Hendrix 2.0 intervenes directly in the conversation to offer a helpful supporting dialogue. In Hendrix 1.0, support dialogue was deployed when an incorrect answer was provided. However, in Hendrix 2.0, learner non-comprehension is anticipated and an additional supporting dialogue is introduced before an answer is given. The intention is to help the learner overcome non-comprehension and avoid *impasse* which leads to negative affect and loss of motivation over time (see section 3.4).

Both interventions are triggered by the same process during tutorial question-answer periods, as shown in figure 13.6. Adaptive algorithm steps are highlighted in green.

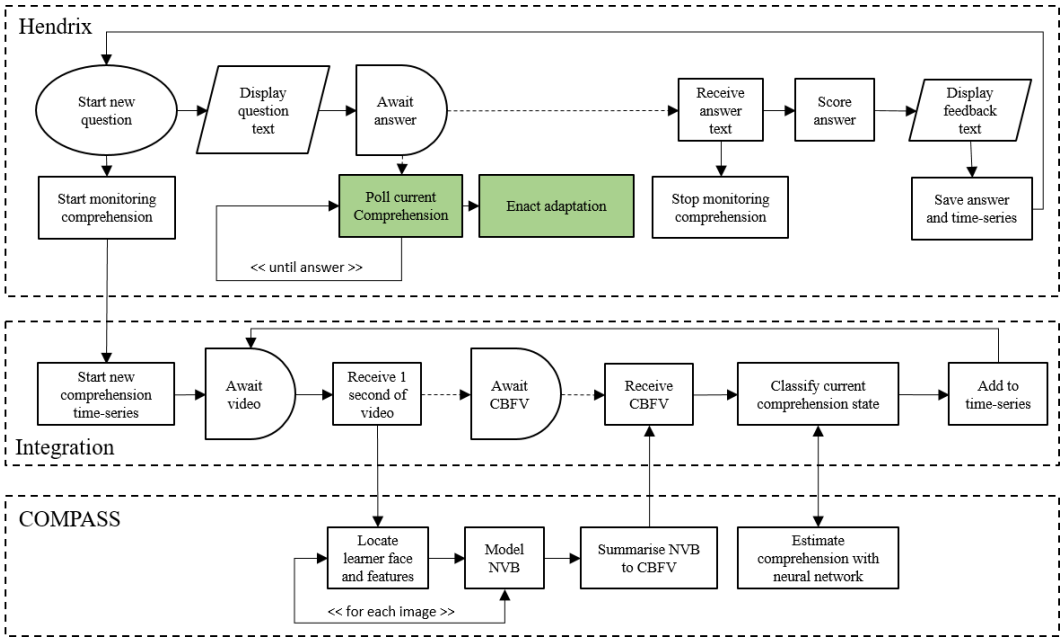


Figure 13.6 Hendrix 2.0 adaptation process

13.8.1 Adapting to weak non-comprehension classifications

When a weak non-comprehension classification occurs during a question-answer response period (figure 9), Hendrix 2.0 adapts the user interface to display a set of suggested requests a learner could make to help explore the topics under discussion. Recommendations are displayed as buttons underneath the text input field, as shown in figure 13.1. Each button will prompt Hendrix 2.0 to provide a definition or example of a keyword or phrase found within the current question dialogue. Buttons are automatically generated by keyword analysis of the most current Hendrix 2.0 dialogue. The adaptive algorithm 8 begins by fetching the current comprehension classification for the learner. If the current comprehension state is *weak non-comprehension* then adaptation is enacted. The most recent dialogue string sent to the learner by Hendrix

2.0 are parsed using customised part-of-speech models to identify noun and noun-phrase chunks. Each noun or noun-phrase chunk is then used as search keywords in a TF-IDF (Lucene). Results, or *recommendation* buttons, are presented in ranked order.

Algorithm 8: Algorithm to trigger UI adaptation based on comprehension classification

```
1  for each one second awaiting answer do
2      fetch comprehension state as comprehension;
3      if comprehension == weak non-comprehension then
4          fetch last hendrix dialogue as dialogue;
5          parse dialogue for keywords set as K;
6          search content indexes using tf-idf(K) for result set R;
7          for result in R do
8              create button for result;
9              add button to chat window;
10         end
11     end
12 end
```

13.8.2 Adapting to strong non-comprehension

When a *strong* non-comprehension classification is detected during a question-answer response period (algorithm 9), Hendrix 2.0 pushes a single, question specific, support dialogue into the conversation at the point non-comprehension occurs. A splash screen is displayed, forcefully intervening, if a learner begins answering while in a non-comprehension state (as shown in figure 13.8). Once the splash screen has been acknowledged the learner is once again free to type without further intervention from the tutor.

13.9 Examples of tutorial conversation

In this section the author presents selected excerpts from a tutorial conversation transcript, along with screen captures of the user interface.

Algorithm 9: Algorithm to trigger dialogue adaptation based on comprehension classification

```
for each one second awaiting answer until adapted == true do
    fetch comprehension state as comprehension;
    if comprehension == strongnon-comprehension then
        fetch support dialogue from current Question as hint dialogue;
        push hint dialogue into Conversation Model for display;
    end
    adapted = true
end
await typing;
if (learner begins typing and
  comprehension == strongnon-comprehension) then
    open new splash screen with hint dialogue;
end
```

13.9.1 Setting a learning objective

Table 13.1 shows the beginning of a tutorial conversation in which the learner expresses a learning objective and Hendrix 2.0 automatically generates a tutorial plan.

Table 13.1 Setting a learning objective

Speech Act	Actor	Content
Confirm Want To Study	Hendrix 2.0	What would you like to study?
Objective	Student	i need to better understand conditional statements
Information	Hendrix 2.0	I've created a tutorial plan. We will work through each topic one by one, starting with Programming and working towards Control Flow Statements. In each topic you will be asked questions, allowing you to demonstrate your knowledge.
Information	Hendrix 2.0	1. Chapter on Programming
Information	Hendrix 2.0	2. Chapter on Boolean Logic
Information	Hendrix 2.0	3. Chapter on Control Flow Statements

Speech Act	**Actor**	**Content**
Information	Hendrix 2.0	At times during this tutorial you might see some suggestions appear at the bottom of the chat window. These are a prompt for you to ask a question about the topic of discussion. You can click the buttons at the bottom of the window and I'll give you additional information to help with the question.
Information	Hendrix 2.0	You will have a number of attempts to answer each question. If you get an answer partly right, I'll remember which bit you got right - so you don't need to repeat yourself.
Confirm Start Concept	Hendrix 2.0	Are you ready to start?

13.9.2 Introducing a concept

Table 13.2 shows the continuation of the tutorial conversation, in which Hendrix 2.0 introduces the first concept of the tutorial, displays the *hand-out* sheet in a secondary window and offers to show the learner an example. Figure 13.7 shows the user interface during the interactions shown in the transcript.

Table 13.2 Introducing a new concept

Speech Act	**Actor**	**Content**
Information	Hendrix 2.0	First we'll take a look at Programming.
Information	Hendrix 2.0	While there are many programming languages, this tutorial will focus on learning the Java programming language.
Information	Hendrix 2.0	I've put up some information for you to refer to. Have a read, and get back to me when you have finished.

Speech Act	Actor	Content
Confirm Show Demonstration	Hendrix 2.0	Would you like to see an example of Programming before we go further?

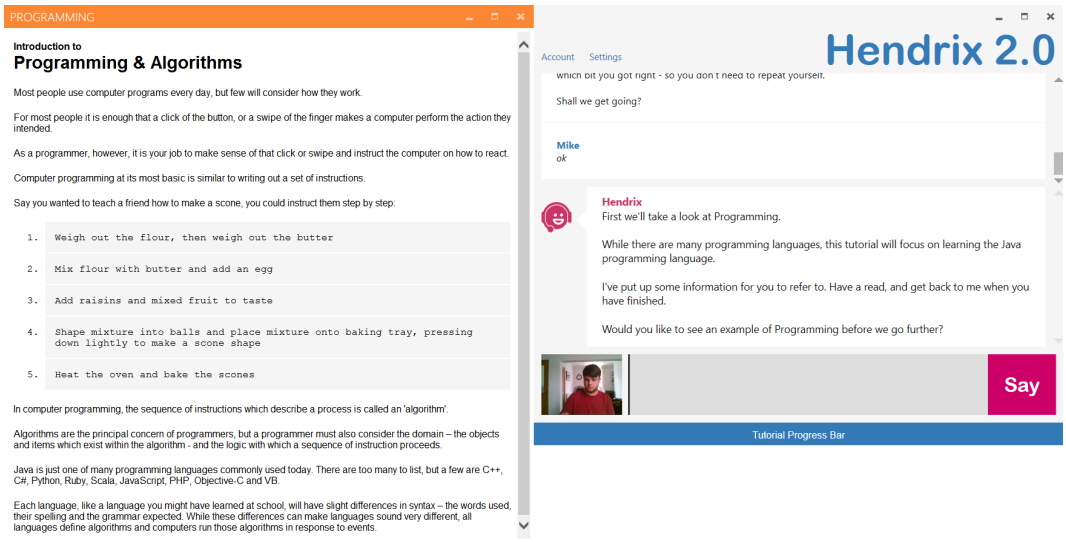


Figure 13.7 Introduction to a concept and display of *hand-out* sheet

13.9.3 Adaptation and intervention

Table 13.3 shows a transcript excerpt in which Hendrix 2.0 asks the learner an open question to assess their understanding of a topic. Hendrix 2.0 adapts the conversation to the learner's responses, offering more specific structured questioning to prompt the learner. During the transcript Hendrix 2.0 detects strong non-comprehension and enacts an adaptive conversational *Intervention* - pushing an additional support dialogue into the chat window. The intervention splash screen for the interaction is shown in figure 13.3.

Table 13.3 Exploring learner knowledge and enacting interventions based on comprehension classification

Speech Act	Actor	Content
Tutor Question	Hendrix 2.0	What can you tell me about algorithms?
Answer	Student	i dont know much about that

Speech Act	**Actor**	**Content**
Feedback	Hendrix 2.0	Now might be a good time to ask a question about a word or concept you dont understand.
Feedback	Hendrix 2.0	Why not ask me a question? You can ask me questions at any time, and we can come back to our tutorial afterwards You could ask me to define a word, or look for an example of a concept.
Tutor Question	Hendrix 2.0	Say you wanted a computer to repeat a process over and over again. What type of language would you use to communicate an algorithm to a computer?
Intervention	Hendrix 2.0	In answering this question, think about communicating instructions. Algorithms are defined processes, a set of instructions, for a computer to follow. Algorithms define the solutions to problems. Of course, we humans also use algorithms to plan, and execute, tasks in every day life.
Answer	Student	a programming language
Feedback	Hendrix 2.0	You've managed to answer 50% of this question.
Feedback	Hendrix 2.0	At a basic level all software consists of algorithms. Algorithms are sets of instructions, defined by a programmer, which a computer can follow.
Feedback	Hendrix 2.0	Excellent work, your answer shows you understand how algorithms are encoded for computers, but there is a little more to the question.

13.9.4 Responding to learner requests

Table 13.4 shows a tutorial transcript excerpt in which Hendrix 2.0 asks a question and is met with a counter question. Hendrix 2.0 identifies that the

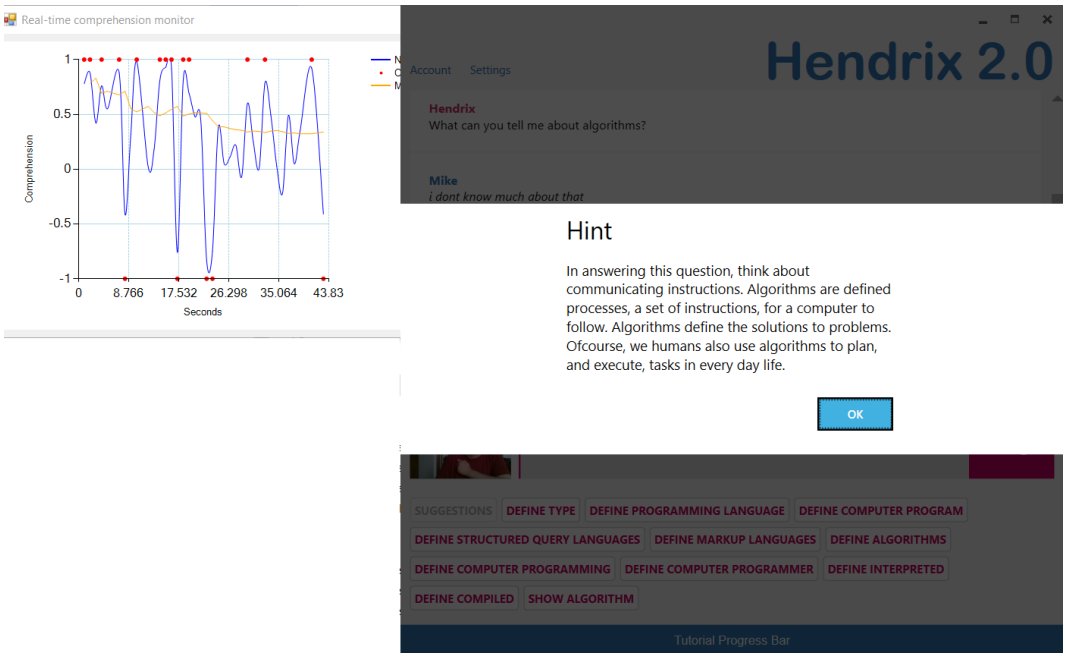


Figure 13.8 Splash screen for conversational intervention during questioning

response utterance contains a request for definition and searches the knowledge graph for a definition entity with closely matching keywords. Figure 13.9 shows a screen shot of the user interface for the interaction.

Table 13.4 Introducing a new concept

Speech Act	**Actor**	**Content**
Tutor Question	Hendrix 2.0	Not all programming languages need to be compiled. Can you think of a programming language which runs without compilation?
Request Definition	Student	tell me some languages?

Speech Act	**Actor**	**Content**
Information	Hendrix 2.0	A programming language is a language which defines instructions for a computer to execute. A programming language is primarily concerned with three types of instruction - algorithms, logic and domain modelling. This is in contrast to markup languages which are concerned only with the display and presentation of information, and query languages which are concerned with the definition and relations in a database. There are too many programming languages to list, but some you might come across are PHP, Python, Ruby, Perl, Cron, Javascript, Cobol, Fortran, Pascal, C, C++, C#, F#, VB (or Visual Basic) and ASP.
Tutor Question	Hendrix 2.0	Not all programming languages need to be compiled. Can you think of a programming language which runs without compilation?
Intervention	Hendrix 2.0	In answering this question, think about interpreted programming languages. An interpreted language is one which is not compiled in advance, but interpreted at run time. These languages are often used for Web applications. Which languages would you use to build a dynamic website?
Answer	Student	php isnt compiled
Feedback	Hendrix 2.0	You've managed to answer 100% of this question.

Speech Act	Actor	Content
Feedback	Hendrix 2.0	Well done, you have demonstrated good understanding of interpreted programming languages.

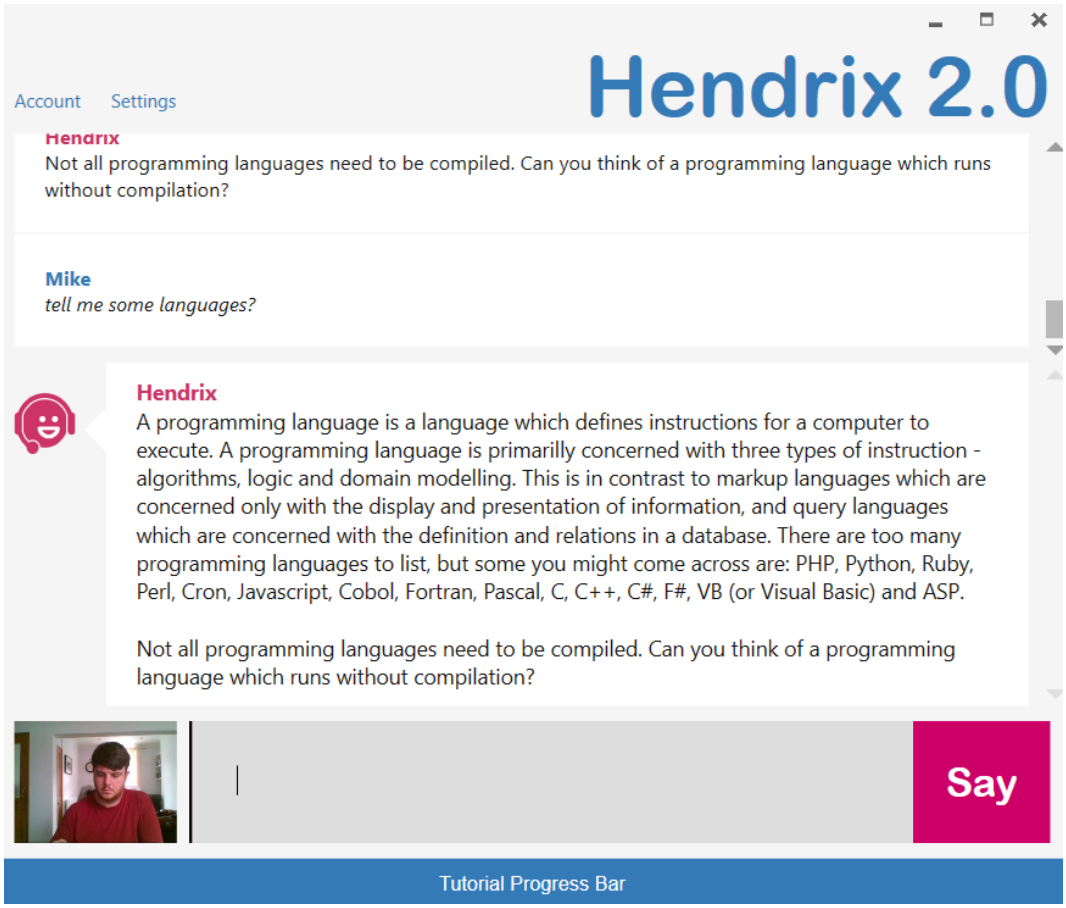


Figure 13.9 Hendrix 2.0 answering a learner's question

13.10 Conclusion

This chapter has presented an overview of the Hendrix 2.0 comprehension based adaptive CITS, detailed the contribution of the system (section 13.1), architecture (section 13.7) and highlighted key functionality of the adaptive system (section 13.8).

The contribution of the research engineering work discussed in this chapter is to define a technical method for development of comprehension based micro-adaptive behaviour within a conversational intelligent tutoring system.

The system is the first of its kind, integrating a CITS with near real-time comprehension modelling from non-verbal behaviour. The system has been developed to help bridge the gap in effective practice between human tuition and e-learning by equipping Hendrix 2.0 with the human-like ability to perceive comprehension and non-comprehension from e-learners' non-verbal behaviour and adapt both user interface and discourse to support learning.

The system will be used to evaluate empirically whether comprehension based adaptation within a CITS can improve learning outcomes and tutorial performance.

Chapter 14

Study: Comprehension classification accuracy with a CITS

14.1 Introduction

This chapter presents a study undertaken to evaluate the accuracy of COMPASS (Chapter 11) comprehension classifications during on-screen conversational tutoring with Hendrix 2.0 (Chapter 13)). The study presented in this chapter aims to demonstrate that the COMPASS system is able to reliably identify comprehension and non-comprehension indicative behaviours in the context of a CITS.

Section 14.3 presents the research question investigated in this study. Section 14.4 provides an overview of the study procedure, ethics and participant demographics. Section 14.5 details the method, with results in section 14.5.1. Discussion and conclusions follow in sections 14.6 and 14.7.

14.2 Motivation

This study is motivated by the need to evaluate the reliability of COMPASS as a real-time feedback channel when integrated into the Hendrix 2.0 CITS. In addition, the successful transfer of COMPASS from the training environment (section 12.3) to a more complex pedagogy.

14.3 Research question

1. Can COMPASS accurately classify e-learner comprehension of on-screen information during conversational interactions with Hendrix 2.0?
    - H0: COMPASS achieves $< 70 \pm 5\%$ normalised classification accuracy
    - H1: COMPASS achieves $>= 75\%$ normalised classification accuracy

The hypotheses are based on matching or exceeding the classification accuracy achieved during training and testing of the classifier.

14.4 Study overview

The study had 51 undergraduate students from Manchester Metropolitan University (MMU) take an on-screen conversational tutorial using the Hendrix 2.0 CITS. The size participant group was chosen to fall within that identified in literature Latham et al. (2014); Lin et al. (2014). During tutorial question and answer interactions, Hendrix 2.0 captured real-time video footage of the learner and used COMPASS to classify comprehension and non-comprehension states. Study participants answered a total of 1269 appraised tutorial questions during which COMPASS generated 7,027 comprehension classifications. The accuracy of the 7,027 classifications is evaluated in this study by comparing the classification with the ground-truth answer score provided for the associated response period. Results are presented in the form of confusion matrices with classification accuracies.

14.4.1 Study procedure

During the experiment student study participants will be asked to undertake an on-screen conversational tutorial on programming concepts, logic, computing mathematics and coding. Questions in the tutorial are designed to elicit a variety of discursive, mathematical and programming code solutions and cover three tiers of the revised Bloom's taxonomy (section 2.3.2). Questions are also categorised as either simple or complex, as defined by the complexity of the answers required.

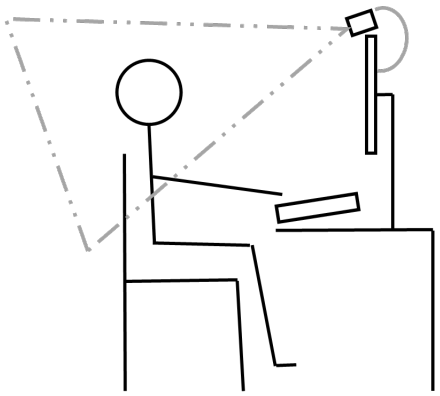


Figure 14.1 Physical layout of experiment

As indicated by the Hendrix 2.0 wire frame diagram (figure 13.1), the system captures a real-time web camera feed during the tutorial. As shown in figure 14.1, the physical layout of the experiment puts the learner's face directly in front of the web camera, to facilitate capture of facial and behavioural feature data. For each answer response period Hendrix 2.0 and COMPASS produce a time-series of comprehension classifications (as shown in figure 13.6) based on analysis of learner non-verbal behaviour (NVB).

For each 1-second of time spent responding to a question Hendrix 2.0 has asked, COMPASS (Chapter 11) produces a comprehension estimate and classification. The comprehension estimate is the raw output of the comprehension classifier, a real-number value between -1.0 and +1.0 (see section 11.8). The comprehension classification is made by applying a logistic function to the

comprehension estimate. As described in section 11.8, the classification is based on the estimate meeting or exceeding a defined threshold.

The optimum thresholds for classification are a variable to be investigated in this experiment; however, from previous findings (section 12.6) classification accuracy will be evaluated at intervals ±0.4, ±0.6 and ±0.8. For this experiment the ground-truth division of answer periods demonstrating 'comprehension' and 'non-comprehension' is set at 50% answer correctness. To reduce complexity, answer scores have been rounded to the nearest 25%, leaving scores at 0, 25, 50, 75 and 100% correctness.

To evaluate classifier performance the sensitivity (equation (14.2)), specificity (equation (14.1)) and normalised classification accuracy (equation (14.3)) are calculated for comprehension, non-comprehension and accuracy respectively.

$$\frac{tn}{tn + fp} \tag{14.1}$$

$$\frac{tp}{tp + fn} \tag{14.2}$$

$$\frac{tp + tn}{tp + fp + tn + fn} \tag{14.3}$$

Symbols are:

- True Positive (tp)

  Answer period ground-truth is 'comprehension' and comprehension classification is 'comprehension'

- False Positive (fp)

  Answer period ground-truth is 'non-comprehension' and comprehension classification is 'comprehension'

- True Negative (tn)

  Answer period ground-truth is 'non-comprehension' and comprehension classification is 'non-comprehension'

- False Negative (fn)

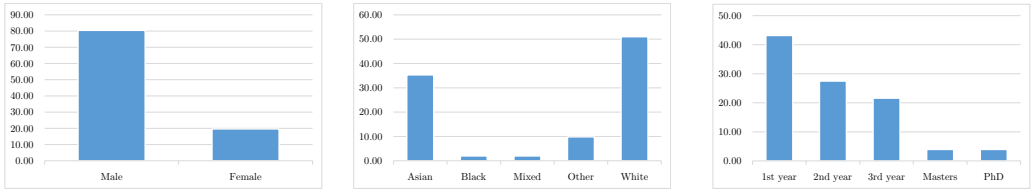
  Answer period ground-truth is 'comprehension' and comprehension classification is 'non-comprehension'

14.4.2 Ethics

Participants were asked to sign a consent form prior to undertaking the experiment. The consent form detailed the purpose of the experiment, the activities required of the participant, the data collected during the experiment and the future uses and publication of the data collected. During the experiment, sensitive personal information such as names, email addresses, age, gender and ethnicity were collected. In addition, information identifying the learner including their course of study and university ID were also collected. Due to the sensitive nature of the data collected, information has been stored on a fully encrypted hard drive on a computer stored in a secure location on university premises. As per the consent agreement, personally identifiable information will not be made public.

14.4.3 Participant information

51 student participants volunteered to take part in the experiment. All participants were over the age of 18. 80% of participants are male, 20% female. Over 50% of participants identify as 'white or white british', with the second largest ethnographic group identifying as 'asian or british asian' (35%). 92% of participants are undergraduate students, with 43% in first year, 27.5% in second year and 21.5% in third year; 8% of participants were postgraduate students.



(a) Gender (b) Ethnicity (c) Academic level

Figure 14.2 Demographics for participant group

14.5 Method

51 student participants from the MMU Faculty of Science & Engineering undertook a tutorial on computer programming, using the Hendrix 2.0 system. The experiment procedure required each participant to:

1. Read and sign the consent form

2. Complete on-screen conversational tutorial on programming concepts, logic, computing maths and coding using the Hendrix 2.0 CITS

The resultant tutorial data and comprehension classifications were then analysed to establish the normalised classification accuracy for comprehension and non-comprehension states across the participant group, for demographic subgroups and by question type.

14.5.1 Results and discussion

Overview of data collected

All 51 participants undertook the on-screen tutorial using Hendrix 2.0. Participants answered a total of 1,269 questions, an average of 26 questions per participants. The number of questions answered by each participant depended on the completeness of the answers provided for each question.

The facial feature extraction and behavioural encoding algorithm produced cumulative behavioural feature vectors (CBFV) for 77.38% of response periods. The feature extraction algorithm performed well on average for both male and female participants. The algorithm successfully created full CBFV for 58% of

response periods for males, and 62% of response periods for females. Although the success rate for both groups is equivocal, due to the gender imbalance in the participant group over 77% of CBFV are from male participants, while only 22% yield from female participants.

Figure 14.3 shows that NVB extraction underperforms for participants identifying as 'black' or 'mixed' ethnicity.

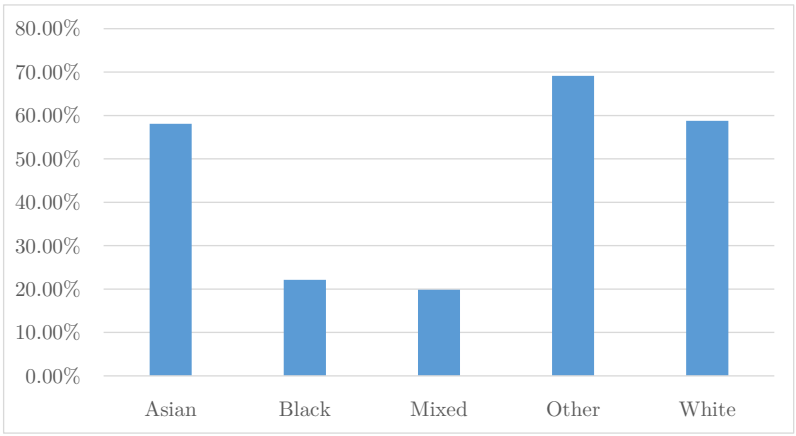


Figure 14.3 NVB extraction performance by ethnicity

Due to a combination of participant demographics (figure 14.2 and NVB extraction performance (figure 14.3), as with prior research in developing the classifier (section 12.5), the majority (58%) of classifiable NVB comes from white participants (figure 14.4).

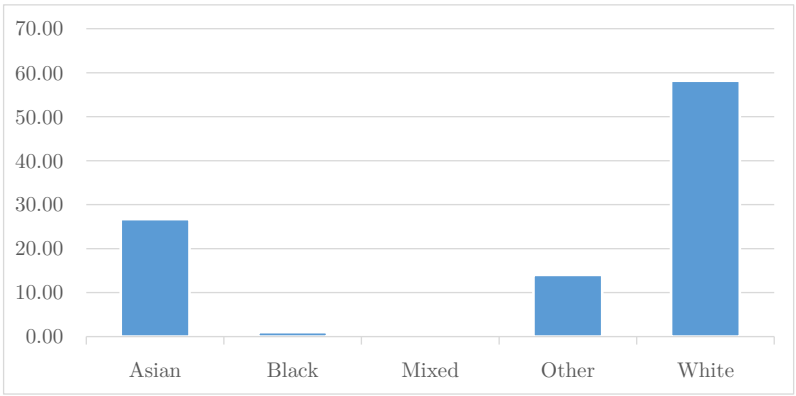


Figure 14.4 Makeup of CBFV dataset by ethnicity

The imbalance in population demographics for comprehension classifier training data, as shown in Chapter 12 sections 12.3.2 and 12.5, is expected

to affect classification accuracy adversely for those demographic groups which have been underrepresented in the training data.

Explanation of results presentation

Table 14.1 is included at the beginning of this chapter to illustrate and explain the layout and content of the results presented in this chapter. In Table 14.1 the headings are indicative as follows:

- Threshold: $\pm Threshold$ value of the logistic step function applied for binary classification
- Cases: The ground-truth cases (NC = Non-comprehension, C = Comprehension)
- NC: A count of the number of non-comprehension classifications within each case
- C: A count of the number of comprehension classifications within each case
- Accuracy: Classification accuracy as defined by the equations 14.1 and 14.2, respectively
- Normalised Accuracy: Classification accuracy as defined by the equation 14.3

Table 14.1 Table demonstrating the layout and content of results tables

		Classifications		
Threshold	**Cases**	**NC**	**C**	**Accuracy (%)**
$\pm threshold$	NC	tn	fp	see eq 14.1
	C	fn	tp	see eq 14.2
	Normalised Accuracy (%)			see eq 14.3

Overall classifier accuracy

Table 14.2 shows the accuracy of comprehension classification at three experimental thresholds - $\pm 04, \pm 0.6$ and ± 0.8 - for all participants. The results show that classifier accuracy (equation 14.3) does not exceed chance levels ($50 \pm 5\%$).

Table 14.2 Binary classifier accuracy for all participants

		Classifications		
Threshold	**Cases**	**NC**	**C**	**Accuracy (%)**
±0.4	NC	2359	2994	44.07
	C	1713	2550	59.82
	Normalised Accuracy (%)			**51.05**
±0.6	NC	1748	2190	44.39
	C	1204	1881	60.97
	Normalised Accuracy (%)			**51.67**
±0.8	NC	1117	1120	49.93
	C	706	1009	58.83
	Normalised Accuracy (%)			**53.80**

Classification accuracy by gender

Tables 14.3 and 14.4 show classifier performance by gender. The results in tables 14.3 and 14.4 show that the classifier has a bias towards overestimation of female comprehension. The small number of female participants in the training data set (section 12.3.2) may have contributed to over-fitting the female demographic variable to competent female participants.

Classification accuracy by ethnicity

Having demonstrated in section 14.5.1 that the classifier performs best for males, this section focuses on further analysis of results for this demographic group. Tables 14.5, 14.6 and 14.7 show classifier accuracy results for male participants by ethnicity. The results show that classifier performance is not even across demographic groups.

Table 14.3 Binary classifier accuracy for Male participants

		Classifications		
Threshold	**Cases**	**NC**	**C**	**Accuracy (%)**
±0.4	NC	2001	2119	48.57
	C	1444	1949	57.44
	Normalised Accuracy (%)			**52.58**
±0.6	NC	1546	1528	50.29
	C	1031	1422	57.97
	Normalised Accuracy (%)			**53.70**
±0.8	NC	1044	743	58.42
	C	623	755	54.79
	Normalised Accuracy (%)			**56.84**

Table 14.4 Binary classifier accuracy for Female participants

		Classifications		
Threshold	**Cases**	**NC**	**C**	**Accuracy (%)**
±0.4	NC	358	875	29.03
	C	269	601	69.08
	Normalised Accuracy (%)			**45.60**
±0.6	NC	202	662	23.38
	C	73	459	72.63
	Normalised Accuracy (%)			**44.18**
±0.7	NC	73	377	16.22
	C	83	254	75.37
	Normalised Accuracy (%)			**41.55**

Table 14.5 Binary classifier accuracy for Asian Male participants

		Classifications		
Threshold	**Cases**	**NC**	**C**	**Accuracy (%)**
±0.4	NC	619	363	63.03
	C	657	268	28.97
	Normalised Accuracy (%)			**46.51**
±0.6	NC	486	279	63.53
	C	324	78	19.40
	Normalised Accuracy (%)			**45.95**
±0.8	NC	315	149	67.89
	C	324	78	19.40
	Normalised Accuracy (%)			**45.38**

Table 14.6 Binary classifier accuracy for Other Male participants

		Classifications		
Threshold	**Cases**	**NC**	**C**	**Accuracy (%)**
±0.4	NC	92	609	13.12
	C	60	221	78.65
	Normalised Accuracy (%)			**39.15**
±0.6	NC	69	408	14.47
	C	324	78	19.40
	Normalised Accuracy (%)			**38.26**
±0.8	NC	36	205	14.94
	C	28	104	78.79
	Normalised Accuracy (%)			**37.53**

Table 14.7 Binary classifier accuracy for White Male participants

		Classifications		
Threshold	**Cases**	**NC**	**C**	**Accuracy (%)**
±0.4	NC	1290	1147	52.93
	C	698	1324	65.48
	Normalised Accuracy (%)			**58.62**
±0.6	NC	991	841	54.09
	C	463	1018	68.74
	Normalised Accuracy (%)			**60.64**
±0.8	NC	693	389	64.05
	C	271	573	67.89
	Normalised Accuracy (%)			**65.73**

Table 14.5 shows that the classifier underestimated comprehension for participants identifying as Asian or British Asian (19.40% CA on *comprehension*). Table 14.6 shows that the classifier overestimates comprehension for participants identifying as Other (14.94% CA on *non-comprehension*). As with the evident bias identified for gender, similarly, for participant demographics with low representation in the training data the classifier has over-fit the demographic variable to a small number of specific learners.

However, for the White Male demographic group (table 14.7), which makes up 58% of the overall data gathered, the results show that classifier accuracy was strong (65.73% normalised CA).

Classification accuracy by question complexity

As discussed in section 2.3.3, guessing behaviour may result from questions requiring simple response formulations, such as true or false statements. Guessing behaviour could result in 'strong non-comprehension' behaviour becoming associated with a high answer score, reducing the evident accuracy of the classifier. Complex questions, which cannot be so easily guessed, may produce a more accurate association between expressed non-verbal behaviour and resultant response correctness.

Section 14.5.1 demonstrated that the classifier performs best for White Males. Further analysis of performance in this section focuses on the White Male demographic subgroup. 37.6% of answer periods were responses to simple questions, while 62.4% were responses to complex questions. Tables 14.8 and 14.9 show classifier accuracy for White Males, broken down by question complexity.

Table 14.8 Binary classifier accuracy for simple questions

		Classifications		
Threshold	**Cases**	**NC**	**C**	**Accuracy (%)**
±0.4	NC	299	534	35.89
	C	257	272	51.42
	Normalised Accuracy (%)			**41.92**
±0.6	NC	209	380	35.48
	C	187	210	52.90
	Normalised Accuracy (%)			**42.49**
±0.8	NC	117	203	36.56
	C	125	129	50.79
	Normalised Accuracy (%)			**42.86**

The results presented in tables 14.8 and 14.9 show that classifier accuracy is markedly different for simple and complex questions. While accuracy on responses to simple questions sits at around chance levels, accuracy for responses to complex questions achieves normalised accuracy of 75.44%.

Table 14.9 Binary classifier accuracy for complex questions

Threshold	Cases	Classifications NC	C	Accuracy (%)
±0.4	NC	991	613	61.78
	C	441	1052	70.46
	Normalised Accuracy (%)			**65.97**
±0.6	NC	782	461	62.91
	C	276	808	74.54
	Normalised Accuracy (%)			**68.33**
±0.8	NC	576	186	75.59
	C	146	444	75.25
	Normalised Accuracy (%)			**75.44**

14.6 Discussion

While results presented in 14.5.1 do not support the alternative hypothesis (section 14.3) that comprehension classification above chance levels can be achieved during conversational tutoring, further analysis presented in sections 14.5.1 and 14.5.1 demonstrate that given constraints the classifier can achieve above 75% normalised classification accuracy.

The investigation has identified and empirically demonstrated the affect of a number of factors on classifier performance. Results in section 14.5.1 support literature (Bond et al., 1990; Rothwell et al., 2006) which suggests NVB indicative of cognitive function is mediated in part by demographic factors such as gender and ethnicity. The results presented in this chapter support the argument for training separate comprehension classifiers for each gender and ethnicity separately. Results presented in section 14.5.1 additionally highlight how learner behaviour is inhibited or exacerbated by the complexity of the answer they must construct. The results demonstrate that learner NVB in response to simple closed questioning is less distinct than the more identifiable behaviour expressed by learners in response to complex open questioning.

The high classification accuracy shown in results in section 14.5.1, above 75%, suggest that mean average comprehension score could be indicative of the grade

assigned to the answer provided by the learner. Figure 14.5 shows the mean average comprehension score for each answer score (0, 25, 50 and 100%) by the threshold applied. The trend-lines shown in figure 14.5 highlight the relationship between COMPASS classifications and learner attainment. The trend-lines indicate that lower average comprehension estimates are strongly associated with lower answer scores while higher average comprehension estimates are strongly associated with higher answer scores, supporting the hypothesis that COMPASS is able to accurately recognise comprehension indicative patterns of non-verbal behaviour.

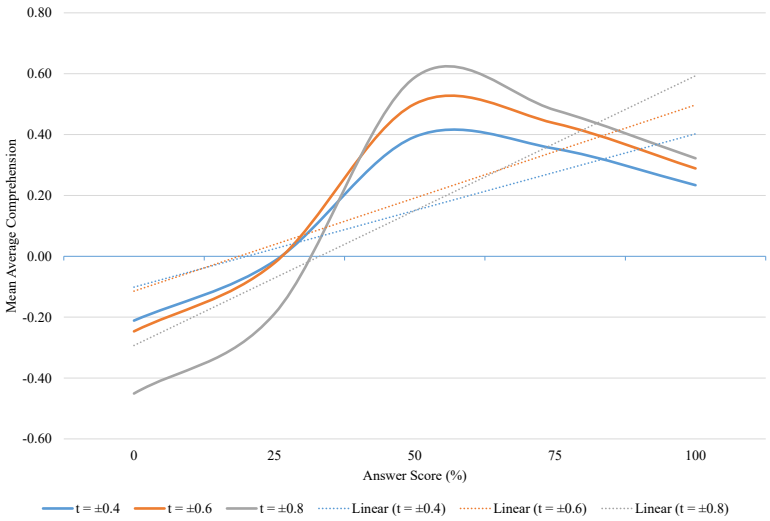


Figure 14.5 Average comprehension score by answer score for complex questions answered by White Males

14.7 Conclusion

This chapter has presented an empirical studying of the transferability of COMPASS (Chapter 11) to a conversational tutoring style.

The contribution of the research discussed in this chapter has been to empirically evaluate whether a trained machine learning classifier, in this instance a multi-layer perceptron artificial neural network, can be transferred to classify comprehension and non-comprehension (as defined in section 3.2) from indicative patterns of learner non-verbal behaviour (drawn from literature

in section 6.2 and discussed in implementation in section 11.6) within complex conversational interactions with an intelligent virtual tutor.

The study has tested empirically whether patterns of comprehension indicative behaviour learned by the COMPASS comprehension classifier are generalised and transferable across e-learning environments, students and pedagogies.

The results presented in this study show that COMPASS has effectively generalised, such that in a new e-learning environment, with different students, questions and pedagogy, the COMPASS classifier is still able to produce normalised classification accuracy of (75.44%). The normalised CA achieved in this study, for White Males answering complex questions, is in-line with the benchmark set during classifier training and testing. Plotting average comprehension estimates against grade boundaries (figure 14.5) shows the relationship between average estimated comprehension and achieved answer scores.

The results demonstrate that COMPASS is capable of providing an accurate real-time feedback mechanism on learner comprehension during on-screen conversational tutoring, given limitations relating to question complexity and training set demographic imbalance.

Having established that comprehension and non-comprehension can be accurately classified within a CITS tutorial, further research and analysis is required to evaluate the effect of conversational adaptation to non-comprehension events on outcomes for learners. In future work supplementary data sets should be collected to allow for training and evaluation of separate classifiers for each demographic subgroup.

An unanticipated area of future research, outside the immediate scope of this project, emerges from plotting the average comprehension score by answer score. The results suggest a relationship between comprehension score and

answer score. In future research, it may be possible to establish whether a grade forecasting model can be created from mean average comprehension scores.

Chapter 15

Study: The effect of comprehension based adaptation on learning outcomes

15.1 Introduction

This chapter presents a study undertaken to evaluate the effect of comprehension based adaptation on formative (answer scores) and summative (learning gain) measures of learner knowledge during tuition with Hendrix 2.0. In this study two comprehension based adaptations are trialled at *weak* and *strong* levels of *non-comprehension* (Chapter 13 section 13.8). The study detailed in this chapter explores the effect of micro-adaptation (see section 4.8.2), by means of timely introduction of hints and feedback, effects the learning outcomes of learners using the Hendrix 2.0 software.

In section 15.3 the research questions are defined; experimental design was detailed in section 15.4 and method in section 15.5. Results for each research question are presented in section 15.6. Discussion and conclusions follow in sections 15.7 and 15.8.

15.2 Motivation

The study aims to evaluate whether the introduction of recommendations and hints during conversation with Hendrix 2.0 can help learners overcome *non-comprehension* and facilitate effective learning, therefore progressing the research by demonstrating that human-like comprehension based adaptation can benefit e-learners.

15.3 Research questions

In this experiment the authors evaluate improvement to formative and summative assessment scores made by graphical and conversational adaptation based on real-time comprehension classification. Two questions are investigated:

1. Does comprehension based adaptation improve formative assessment scores during tuition?

   - H1.0: $\mu_1 - \mu_2 \geq 0$
   - H1.1: $\mu_1 - \mu_2 < 0$

   where μ_1 and μ_2 are the mean question scores for control and experimental groups respectively, when strong non-comprehension was detected.

2. Does comprehension based adaptation improve summative assessment scores in post-tuition tests?

   - H2.0: $\mu_1 - \mu_2 \geq 0$
   - H2.1: $\mu_1 - \mu_2 < 0$

   where μ_1 and μ_2 are the mean learning gains for control and experimental groups respectively.

15.4 Study overview

For this study, 51 students from Manchester Metropolitan University (MMU) take an on-screen conversational tutorial using the Hendrix 2.0 CITS. Learners were assigned at random into either control or experimental groups. The control group used a version of Hendrix 2.0 which classified comprehension but did not adapt to comprehension. The experimental group used a version of Hendrix 2.0 which classified comprehension and enacted comprehension based adaptation.

15.4.1 Study procedure

51 undergraduate students from Manchester Metropolitan University volunteered to participate in this experiment. Participants undertook the experiment in computer labs within the University using facilities and computers available for student use. The experiment procedure was:

1. Read and sign consent form
2. Participant was assigned a computer and a pen drive containing either the control or experimental version of the Hendrix 2.0 CITS
3. Complete 15 MCQ questions on computer programming
4. Undertake a tutorial on computer programming using the Hendrix 2.0 CITS
5. Repeat the 15 MCQ questions on computer programming
6. Complete a feedback survey

15.4.2 Ethics

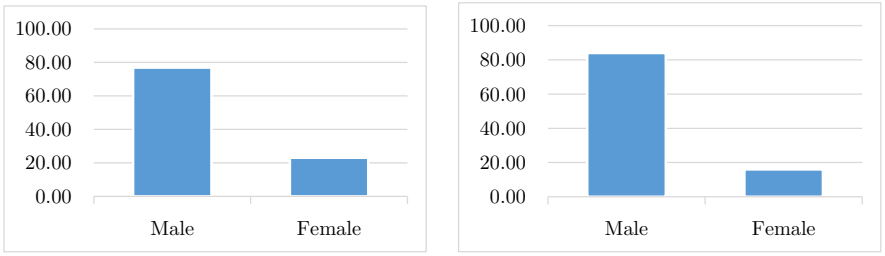
Participants signed a consent form prior to undertaking the experiment. The consent form detailed the purpose of the experiment, the activities required of the participant, the data to be collected during the experiment and the future uses and publication of the data collected.

During the experiment, sensitive personal information such as names, email addresses, age, gender and ethnicity were collected, in addition to information identifying the learner, including their course of study and university ID.

Due to the sensitive nature of the data collected, information has been stored on a fully encrypted hard drive on a computer stored in a secure location on university premises. In accordance with the consent agreement, personally identifiable information will not be made public.

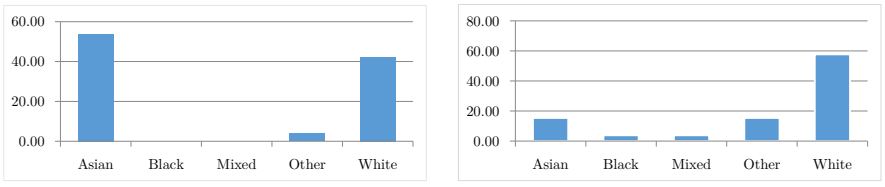
15.4.3 Participants

51 participants took part in the experiment, 26 assigned to the control group and 25 assigned to the experimental group. Participants were assigned to either control or experimental groups at random. Neither the author nor any participant knew which group was being assigned to each participant when assignment was made. Figures 15.1, 15.2 and 15.3 show the make-up of both control and experimental groups by demographics.



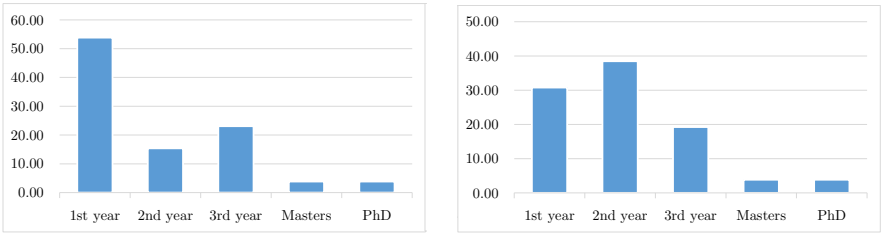
(a) Control (b) Experimental

Figure 15.1 Gender demographics for control and experimental groups



(a) Control (b) Experimental

Figure 15.2 Ethnicity demographics for control and experimental groups



(a) Control (b) Experimental

Figure 15.3 Academic level demographics for control and experimental groups

15.5 Method

51 student participants from the Faculty of Science & Engineering at Manchester Metropolitan University took part in the experiment. Participants, randomly assigned into control and experimental groups, provided a total of 1,269 answers to 604 questions. This section presents the methods for investigating the research questions detailed in section 15.3.

Can comprehension based adaptation improve learner formative assessment scores?

H1 (section 15.3)) tests whether adaptation to strong non-comprehension improves the mean average question score of participants.

The alternative hypothesis $\mu_1 - \mu_2 < 0$, where μ_1 is the control group and μ_2 is the experimental group, is true only if the mean average question score for the experimental group is higher than that of the control group.

To test the hypothesis the participants were split into control and experimental groups. Group membership was decided by assignment of a pen drive containing either the adaptive or non-adaptive version of the software. Neither the participants nor the experimenter knew which group was being assigned. The control group contained 25 participants and the experimental group contained 26 participants.

All participants undertook a tutorial on computer programming, using the Hendrix 2.0 CITS. For both groups comprehension and non-comprehension classifications were generated during question-answer interactions and answer

scores, cumulative question score and comprehension classification time-series recorded in a database.

An answer score was defined as the mark between 0 and 100% for information contained in a single conversational response. A question score (equation 15.1) is the sum of scores for all answers given to a single question over multiple conversational turns. As Hendrix 2.0 does not double count scores for answers, the maximum question score was also 100%.

$$\sum_{n=attempt}^{attempts} f(answer) = \text{question score} \tag{15.1}$$

During question-answer interactions, Hendrix 2.0 monitored the comprehension classification time-series data returned from COMPASS. For the experimental group Hendrix 2.0 performed adaptation based on detection of non-comprehension events during question-answer response periods. The adaptation algorithm has three levels of adaptation:

- Level 0: 'No non-comprehension classification'
- Level 1: 'Weak non-comprehension classification'
- Level 2: 'Strong non-comprehension classification'

At level 0 no adaptation was enacted. At level 1, Hendrix 2.0 introduced a set of recommended question buttons (see 13.8.1). At level 2, Hendrix 2.0 intervened in the conversation to offer additional support to the learner (see section 13.8.2).

Can comprehension based adaptation improve learner summative assessment scores?

H2 (section 15.3)) tests whether adaptation to strong non-comprehension increases learning gain (see section 2.3.1).

The alternative hypothesis $\mu_1 - \mu_2 < 0$, where μ_1 is the control group and μ_2 was the experimental group, is true only if the mean average learning gain for the experimental group was higher than that of the control group.

To test the hypothesis the participant group was split into control and experimental groups. Group membership was decided by assignment of a pen drive containing either the adaptive or non-adaptive version of the software. Neither the participant nor the experimenter knew which group was being assigned. The control group contained 25 participants and the experimental group contained 26 participants.

Participants completed a 15 question multiple choice quiz on computer programming before undertaking a tutorial on computer programming using Hendrix 2.0 CITS. Following the tutorial, participants repeated the 15 question computer programming Multiple Choice Quiz (MCQ).

To test hypothesis H2, whether adaptation increases learning gain, pre-tutorial and post-tutorial test scores were compared between the control and experimental groups.

15.6 Results

15.6.1 Overview of data collected

Tables 15.2 and 15.1 show average participant scores for both groups at the question and the answer level.

Table 15.1 Overview of tutorial answer data

Group	**Total number of answers provided**	**Average answers per question**	**Average answer value (%)**
Control	674	2.07	37.04
Experimental	595	2.13	33.42

Table 15.2 Overview of tutorial question data

Group	Number of questions attempted	Average question score (%)
Control	325	54.90
Experimental	279	51.45

During the tutorial, participants answered three types of question: programming, discursive and mathematical. Table 15.3 shows average question scores for each type of question. Table 15.4 shows average question scores for each topic: algorithms, programming concepts and terminology, computing mathematics and logic, and applied Java programming.

Table 15.3 Overview of formative tutorial question data by type

Group	Type	Average question score (%)
Control	Programming	54.37
	Discursive	46.19
	Mathematical	70.27
Experimental	Programming	54.50
	Discursive	40.54
	Mathematical	70.37

Table 15.4 Overview of formative tutorial question data by topic

Group	Topic	Average question score (%)
Control	Basic algorithms	62.50
	Programming concepts & Terminology	23.14
	Computing Maths & Logic	65.57
	Applied Java programming	49.58
Experimental	Basic algorithms	75.41
	Programming concepts & Terminology	19.33
	Computing Maths & Logic	54.55
	Applied Java programming	44.42

Tutorial questions covered three stages of cognitive mastery on Bloom's taxonomy: remember, create and analyse (Anderson et al., 2000). The three

stages were selected so as provide a broad variation in degree of challenge, from the simplest (remembering) to the most complex (creating). Table 15.5 shows average question scores across the three cognitive stages.

Table 15.5 Overview of formative tutorial question data by Bloom's taxonomy category

Group	Category	Average question score (%)
Control	Analyse	65.57
	Create	60.26
	Remember	43.13
Experimental	Analyse	54.55
	Create	59.00
	Remember	39.79

Participants completed pre- and post-tutorial multiple choice quizzes. Table 15.6 provides an overview of learner MCQ performance in both control and experimental groups.

Table 15.6 Overview of summative pre- and post-tutorial MCQ scores

Group	Number of MCQ answers	Average pre-tutorial MCQ score (%)	Average post-tutorial MCQ score (%)
Control	330	67	70
Experimental	315	70	67

15.6.2 The effect of adaptation on learner performance under formative assessment

Table 15.7 shows one-tail two sample t-tests comparing the mean question scores of control and experimental groups at each adaptation level in the system. Hypothesis H1 (section 15.3) tests whether adaptation to strong non-comprehension improves average learner attainment. While the results in Table 15.7 show that the absolute mean average question score for users of the adaptive system was higher than that of the non-adaptive, the t statistic

shows that the difference was insignificant ($p < 0.05$). With $t = -0.28$ for the samples, there was low confidence that population means would reflect this finding. For H1, it is therefore not possible to reject the null hypothesis for the entire population.

Table 15.7 Two sample t-tests comparing control and experimental group question scores

Group	**Level**	μ	σ	n	σ^2	$\mu_1 - \mu_2$	t
Control	No non-comprehension	57.09	45.52	127	16.32		
Experimental	No non-comprehension	50.07	44.98	151	13.40	7.01	1.2
Control	Weak non-comprehension	54.35	44.95	31	65.18		
Experimental	Weak non-comprehension	45.21	44.57	24	82.77	9.15	0.75
Control	Strong non-comprehension	53.35	43.76	167	11.47		
Experimental	Strong non-comprehension	54.88	44.08	104	18.68	-1.54	-0.28

Comprehension classifier accuracy will affect the performance of the adaptation algorithm, as false positive or false negative classifications cause incorrect behaviour. As highlighted in previous work (Chapter 14), classifier accuracy was highest for the White Male subgroup. Table 15.8 shows average question and answer scores by adaptation level for White Males.

Table 15.8 shows one-tail two sample t-tests comparing the mean question scores for the White Male subgroup of both control and experimental groups, again at each adaptation level. The results in Table 15.8 show that the absolute mean average question score for users of the adaptive system was higher than that of the non-adaptive when adaptation was enacted in response to strong non-comprehension. With $t = -0.83$ for the samples with an 80% confidence ($p = 0.2$), the result suggests that where classification accuracy was highest direct intervention in the learning process may improve question scores.

Table 15.8 Two sample t-tests comparing control and experimental group question scores for White Male subgroup

Group	Level	μ	σ	n	σ^2	$\mu_1 - \mu_2$	t
Control	No non-comprehension	66.75	43.12	55.00	33.80		
Experimental	No non-comprehension	57.45	44.40	73.00	27.00	9.29	1.19
Control	Weak non-comprehension	65.00	45.00	10.00	202.50		
Experimental	Weak non-comprehension	45.94	43.63	16.00	118.99	19.06	1.06
Control	Strong non-comprehension	50.58	40.45	26.00	62.94		
Experimental	Strong non-comprehension	58.46	42.10	65.00	27.27	-7.88	-0.83

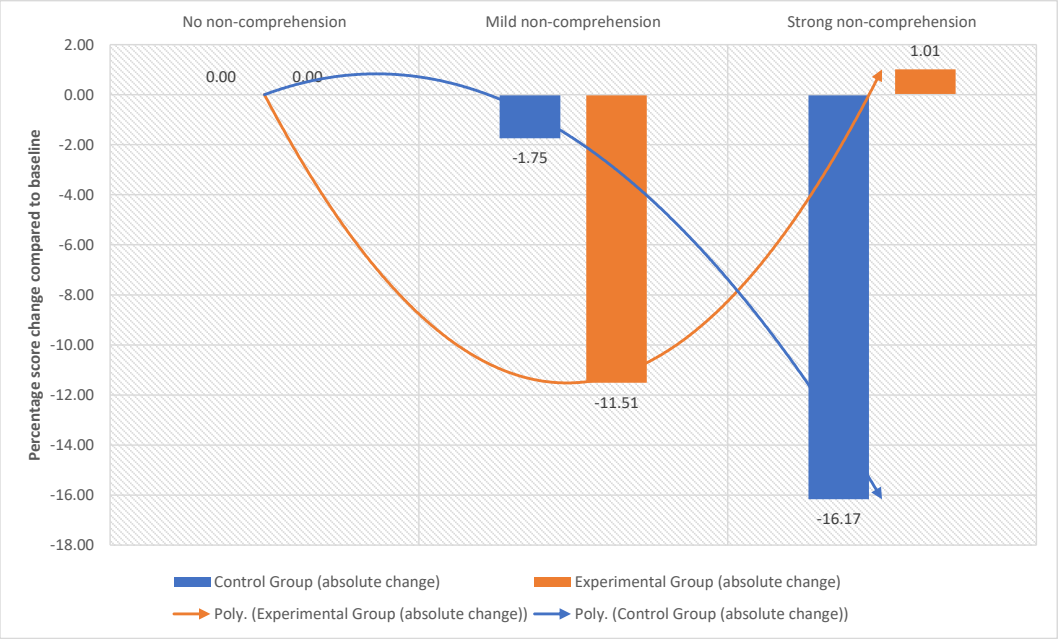


Figure 15.4 Effect of adaptation as percentage change in question score for White Males

15.6.3 The effect of adaptation on learner performance under summative assessment

To investigate H2 (see section 15.3), learning gain (see section 2.3.1) has been calculated for each participant. 22 participants from the control group and 21 participants from the experimental group completed both pre- and post-tutorial

MCQ tests. Results for participants who did not complete both tests or missed questions on either test are excluded. Table 15.9 presents results for the control and Table 15.10 for the experimental group.

Table 15.9 Control group pre- and post-tutorial test scores and normalised average learning gain

Participant	**Pre**	**Post**	**Learning gain (%)**
1	40.00	40.00	0
2	33.33	40.00	7
3	86.67	86.67	0
4	93.33	93.33	0
5	86.67	86.67	0
6	73.33	93.33	20
7	46.67	53.33	7
8	40.00	66.67	27
9	73.33	80.00	7
10	80.00	86.67	7
11	93.33	93.33	0
12	86.67	80.00	-7
13	60.00	53.33	-7
14	73.33	66.67	-7
15	60.00	73.33	13
16	53.33	33.33	-20
17	53.33	60.00	7
18	46.67	46.67	0
19	53.33	53.33	0
20	73.33	80.00	7
21	93.33	100.00	7
22	80.00	80.00	0

Table 15.11 shows a two sample t-test for the alternative hypothesis that $\mu_1 - \mu_2 < 0$ where μ_1 is the mean normalised average learning gain shown in Table 15.9 and μ_2 is the mean normalised average learning gain shown in Table 15.10. The results in Table 15.11 show that the null hypothesis cannot be rejected as μ_2 was lower than μ_1.

With $t = 1.93$ (table 15.11) falling into the 90% confidence region of the upper tail, the results suggest that the control group learning gain was significantly higher than the experimental group. However, table 15.12 shows that the learning gain of the control group is insignificant.

Table 15.10 Experimental group pre- and post-tutorial test scores and normalised average learning gain

Participant	**Pre**	**Post**	**Learning gain (%)**
1	80.00	80.00	0
2	66.67	53.33	-13
3	93.33	93.33	0
4	100.00	100.00	0
5	73.33	73.33	0
6	53.33	46.67	-7
7	100.00	93.33	-7
8	73.33	73.33	0
9	60.00	66.67	7
10	53.33	40.00	-13
11	73.33	66.67	-7
12	53.33	46.67	-7
13	66.67	73.33	7
14	73.33	80.00	7
15	53.33	40.00	-13
16	53.33	53.33	0
17	100.00	100.00	0
18	80.00	80.00	0
19	53.33	66.67	13
20	53.33	53.33	0
21	66.67	33.33	-34

Table 15.11 Two tail two sample t-test for comparison of mean learning gain between control and experimental groups

Group	μ	σ	n	σ^2	$\mu_1 - \mu_2$	t
Control (μ_1)	3.03	9.77	15	6.37		
Experimental (μ_2)	-3.17	7.71	15	3.96	6.21	1.93

Table 15.12 One tail two sample t-test for comparison of mean learning gain within the control group

Test	μ	σ	n	σ^2	$\mu_1 - \mu_2$	t
Pre (μ_1)	67.27	18.85	22.00	16.14		
Post (μ_2)	68.57	18.30	22.00	15.22	-1.30	-0.23

15.7 Discussion

While the results in Table 15.7 were not significant above chance levels, where classifier accuracy improves (Table 15.8) the effect of adaptation when *string non-comprehension* was detected with 80% confidence. Results for the exper-

imental group of White Male participants (Table 15.8) show an 80% confidence that comprehension based intervention increases tutorial answer scores compared to the non-adaptive control group. For the experimental group, introducing a hint dialogue when *strong non-comprehension* (level 2 adaptation) was classified results in an average answer score increase of up to 17.08% ($p = 0.2$) over the control group average score when *strong non-comprehension* was detected. The performance between the control and experimental group students demonstrates the effect a timely intervention can have for learners who were struggling.

In this experiment learning gain was not significant above chance levels. While Table 15.11 shows that the control group had a higher mean normalised average learning gain, Table 15.12 shows that the gain was not significant.

The lack of evident learning gain contradicts the results of the pilot study, in which learning gain was detected with 80% confidence (Chapter 8). However, the results chime with criticisms found in literature (section 2.3.1) which suggest the more complex the interactions between tutor and student, the harder it is to assess the effectiveness of a system through a learning gain metric because of compounding factors and many independent variables.

A difference between this study and the pilot (Chapter 8) is the time taken to complete. This experiment had a longer pre-tutorial test, longer tutorial and longer post-tutorial test. It is possible that fatigue contributed to deflating post-tutorial test scores. Fatigue may also explain why 9 out of 51 participants failed to complete the final MCQ.

In this experiment no testing was done using short or long tutorials, and no repetition of tutorials was possible. A future study would be required to evaluate whether the nullified learning gain effects are due to fatigue and whether time duration or repetition of interaction has a relative effect on learning outcomes, as suggested in literature (VanLehn et al., 2017).

One improvement to the methodology for this experience would be to leave a resting period between the tutorial and the post-tutorial MCQ, allowing all participants an equal time to recover from the effects of fatigue.

15.8 Conclusion

This chapter has presented a study of the effect of comprehension based adaptation on learning outcomes for students studying with the aid of a CITS.

The contribution of the research discussed in this chapter was to evaluate empirically the effect of comprehension-state relevant micro-adaptive behaviours (hints on models and process) on the learning outcomes of students. The research in this chapter puts to the test the hypothesis, influenced by the findings of VanLehn et al. (2017) (see discussion in section 4.8), that *context and timeliness* of coaching behaviour micro-adaptations (see discussion of literature in section 4.8.2) are important to the overall measurable benefit a learner receives from using a tutoring system.

The study has demonstrated successfully the benefit that comprehension based adaptation in e-learning can offer to students experiencing difficulties comprehending on-screen information.

An insignificant ($p = 0.2$) effect of 17% was found between learner intra-tutorial scores for the adaptive intervention group and the non-adaptive control group. It is possible that the effect is insignificant due to the small sample size (51 students) for the study.

Despite being inconclusive, the experiment shows potential for future development in which research is required to assess the most effective intervention design. In addition, future experiments should address the effect of participant fatigue and interaction duration or repetition on learning outcomes.

Chapter 16

Conclusions and future work

The aim of this research has been to design and develop a conversational intelligent tutoring system (CITS) which improves learning outcomes by making timely pedagogic interventions based on assessment of learner comprehension and improves the learning outcomes for students by adapting tutorial discourse and interaction in response to near real-time comprehension classification.

The research had aimed to answer three questions: 1) can a CITS tutor computer programming effectively; 2) can comprehension and non-comprehension states be classified automatically in real-time during on-screen learning activities; 3) can comprehension states be used effectively to adapt pedagogy, content and materials in an intelligent e-learning environment.

16.1 Summary of research

Literature (Chapter 3) has shown that experienced human tutors are able to scaffold the learning experience by adapting discourse and pedagogy in response to non-verbal behavioural indicators of learner comprehension and non-comprehension. The literature highlights the importance of pedagogy and adaptation and the gap in effective practice between human tuition and e-learning. This research has been motivated by the need to enhance the effective pedagogy of e-learning platforms, bridging the gap in practice between

human tuition and digital tuition, by incorporating human-like comprehension based adaptation.

Literature (Chapter 3) highlighted that there was no demonstrated method for modelling e-learner comprehension from non-verbal behaviour which could feasibly be deployed at scale in a real-world classroom environment. Literature left two important questions unanswered: with what degree of accuracy can e-learner comprehension of on-screen information be classified by observation of non-verbal behaviour in a real-world classroom environment and can comprehension based adaptation in e-tuition improve learning outcomes for students?

The aim of this research has been to design, develop and evaluate novel computational methods for accurately classifying e-learner comprehension of on-screen information by near real-time observation of non-verbal behaviour using consumer grade computer hardware and peripherals, and furthermore to demonstrate that comprehension classification can be used for adaptation of on-screen tutorial content and pedagogy to improve the effectiveness of e-tuition.

This thesis has presented an overview of relevant literature on educational theory and instructional design for e-learning (Chapter 2), conversational intelligent tutoring systems (Chapter 4), comprehension assessment (Chapter 3), machine learning with artificial neural networks (Chapter 5) and computational modelling and classification of non-verbal behaviour using computer vision and artificial neural networks (Chapter 6).

The thesis has presented an iterative process of software design, development and evaluation, each study progressing towards a comprehension based adaptive learning system. Hendrix 1.0, a CITS for tutoring computer science and programming, was presented in Chapter 7 and evaluated in Chapter 8. The research questions addressed in Chapter 8 were:

1. Does the intelligent conversational agent converse effectively?

2. Does the intelligent conversational agent facilitate learning?

Hendrix 1.0 CITS was able to converse effectively with learners, making relatively few objective errors in marking or conversational construction and receiving high user satisfaction scores, while learners' post-tuition test scores showed a 22% ($p = 0.2$) normalised average learning gain.

Chapter 10 presented the first pilot study on analysis of non-verbal behaviour by computational means. The research questions in Chapter 10 were broad, seeking to survey the effect of data selection, temporal grouping and classifier hyper-parameters. The result of the pilot study suggested the technical approach was viable but highlighted problems in the method. The results showed that data used in classifier training needed to be recorded in a simple on-screen environment devoid of user interface and process complexity, an environment in which a single on-screen stimulus could account for the resultant display of non-verbal behaviour.

Chapter 11 presented the COMPASS comprehension classification system. COMPASS is trained and tested in a study presented in Chapter 12. The research question in Chapter 12 was:

1. Can computational analysis and classification of NVB predict comprehension and non-comprehension of on-screen information at above chance levels?

The study results showed that when the classifier was trained on individual demographic groupings it achieved test set normalised classification accuracy of 75.8% and average precision of 75.5%.

Chapter 13 presented the Hendrix 2.0 comprehension adaptive CITS, which integrates conversational tutoring with near real-time comprehension classification and comprehension based adaptation. Hendrix 2.0 is evaluated in Chapters 14 and 15.

The research question investigated in Chapter 14 was:

1. Is the trained COMPASS comprehension classifier transferable into a more complex e-learning environment?

The results of the study showed that COMPASS had generalised effectively, such that in a new e-learning environment, with different students, questions and pedagogy, the COMPASS classifier was able to maintain normalised classification accuracy of (75.44%).

The research questions investigated in Chapter 15 were:

1. Does comprehension based adaptation improve formative assessment scores during tuition?

2. Does comprehension based adaptation improve summative assessment scores in post-tuition tests?

The study shows that adaptation to strong non-comprehension, by means of a discursive intervention, increased ($p = 0.2$) learner answer scores by 17% compared to the control group (non-adaptive system users) also displaying strong non-comprehension. The study did not highlight that learning gain measured in post-tutorial tests was increased by use of the adaptive system.

16.2 Summary of findings

Through the series of design, development and evaluation phases summarised in section 16.1 the thesis findings answer the overarching research questions laid out in Chapter 1 section 1.3.

The pilot study of Hendrix 1.0 demonstrated the algorithms designed for facilitation of conversation performed well under experimental conditions, with few systematic errors made in understanding or appraising learner conversational input. The mean learning gain effects of 22% ($p = 0.2$) were not statistically significant for the small pilot group (15 learners). User feedback on the system showed overall satisfaction with the system performance, coherence of conversation and that users felt the system helped them to learn.

The COMPASS comprehension classifier achieved 75.8% normalised classification accuracy during testing and 75.4% accuracy when integrated into Hendrix 2.0. Both the initial COMPASS training and testing study, and the subsequent study in a conversational context, show that there is some degree of common facial non-verbal behaviour indicative of reading comprehension states (as defined in section 3.2) expressed during reading comprehension interactions with on-screen information in both multiple choice question-answer and typed free-text conversational interactions with a virtual agent.

The Hendrix 2.0 study found a positive 17% mean difference in question-level appraisal score in favour of the comprehension adaptive group over the control non-adaptive group, in line with the alternative hypothesis that comprehension responsive discourse micro-adaptation helps learners to overcome impasse. However, the findings are not statistically significant and therefore the results are inconclusive.

16.3 Novelty and contribution

The research has advanced literature on application of computer vision and machine learning techniques in an educational context. The contribution of the research is both technical and educational. The research has focused on designing, developing and evaluating software systems capable of advanced micro-adaptive behaviour.

The technical contributions of this research are:-

1. A CITS for tutoring computer programming (Hendrix 1.0)
2. A model of learner comprehension based on NVB (COMPASS)
3. A near real-time NVB analysis algorithm using both Haar cascades, PCA and a bank of artificial neural networks (COMPASS)
4. A artificial neural network based comprehension classifier (COMPASS)

5. A comprehension based adaptation algorithm for micro-adaptation within a conversational intelligent tutoring system (Hendrix 2.0)

The research presented in this thesis has developed upon knowledge in the field of adaptive CITS in tutoring computer sciences, and contributes five key findings in relation to comprehension based adaptive algorithms. The educational contributions of this research to the field of e-learning and learning analytics are:

1. Demonstration that a CITS can tutor computer programming effectively using natural language

2. Demonstration that learner non-verbal behaviour can be automatically modelled from image data in near real-time within a CITS

3. Demonstration that patterns of non-verbal behaviour can be used to estimate comprehension with greater than 75% accuracy in multiple on-screen learning environments, including a CITS

4. Demonstration of a technique for modelling and classifying comprehension in near real-time without use of specialist hardware or body attached sensors

5. Findings that suggest that conversational adaptation within a CITS in response to comprehension estimates improves learner tutorial scores ($p = 0.2$)

The contribution of the work has been appreciated, with papers accepted to conference (Holmes et al., 2015a, 2017a) and for publication (Holmes et al., 2017b).

16.4 Limitations

1. Due to demographic imbalance in the participant groups used for classifier training and testing it is not possible to demonstrate that the classifier is equally successful across gender or ethnic subgroups.

2. The behaviour modelling approach adapted from literature makes the assumption that specific facial features and geometries will express the totality of significant non-verbal behaviour. Switching to unsupervised learning of behavioural patterns, for example using recurrent convolutional networks, may expose novel and unanticipated behaviours.

3. The experiments on tutorial scores and on learning gain have not presented significant ($p = 0.05$) results. It is therefore not possible to conclude with desired certainty that the system does increase tutorial scores or learning gain when compared to a non-adaptive system.

4. The research has not explored the effect of different types of adaptive interventions, therefore the comprehension based adaptation used in experimentation may not be optimally effective.

16.5 Future work

In future work it would be beneficial to address the unresolved problem of cross-cultural analysis of NVB. To assess the effectiveness of Hendrix 2.0 across the entire undergraduate cohort it is necessary to train the facial detection classifiers, behaviour channel classifiers and comprehension classifier on data for each demographic subgroup. Doing so would optimise the accuracy of classifications throughout the behaviour modelling and classification pipeline and is anticipated to improve accuracy across the cohort. However, retraining each classification step will require substantial volumes of data to be gathered for each demographic subgroup.

Advances in the application of recurrent neural networks (Graves et al., 2007; Lopes et al., 2017; Mayya et al., 2016; Pinheiro and Collobert, 2014; Teng and Yang, 2016; Tompson et al., 2014) to automatic scene labelling in images presents an interesting opportunity to reduce the computational cost of NVB analysis. The current process of de-constructing images into discrete behaviours could be replaced by an automatic analysis of behaviours.

In future work there is an opportunity to better integrate comprehension estimation with knowledge structure, search and selection by weighting the relationships between knowledge entities (nodes in the graph) by a function of the learner's current comprehension state. In doing so, the selected learning path is always contextualised to the current comprehension state.

COMPASS is a system which has the potential for cross-over into other fields of research. In future work it would be interesting to explore how COMPASS fits into learning analytics and to what extent COMPASS could help instructional designers assess and improve on-screen information. Further, COMPASS may be able to play a role in advancing user experience testing for websites or software.

Bibliography

Zone of Proximal Development. In J. L. Longe, editor, *The Gale Encyclopedia of Psychology*, volume 2, page 1233. Gale, Farmington Hills, MI, 3 edition, 2016.

Martha Wagner Alibali, Lucia M. Flevares, and Susan Goldin-Meadow. Assessing knowledge conveyed in gesture: Do teachers have the upper hand? *Journal of Educational Psychology*, 89(1):183, 1997.

Adela-Diana Almási, Stanisław Woźniak, Valentin Cristea, Yusuf Leblebici, and Ton Engbersen. Review of advances in neural networks: Neural design technology stack. *Neurocomputing*, 174:31–41, January 2016. ISSN 09252312. doi: 10.1016/j.neucom.2015.02.092. URL http://linkinghub.elsevier.com/retrieve/pii/S0925231215010358.

Diego Raphael Amancio, Cesar Henrique Comin, Dalcimar Casanova, Gonzalo Travieso, Odemir Martinez Bruno, Francisco Aparecido Rodrigues, and Luciano da Fontoura Costa. A Systematic Comparison of Supervised Classifiers. *PLoS ONE*, 9(4):e94137, April 2014. ISSN 1932-6203. doi: 10.1371/journal.pone.0094137. URL http://dx.plos.org/10.1371/journal.pone.0094137.

John Anderson and Kenneth Koedinger. Cognitive Tutors: Lessons Learned. *The Journal of Learning Sciences*, 4, 1995.

Lorin W. Anderson, David R. Krathwohl, Peter W. Airasian, Kathleen A. Cruikshank, Richard E. Mayer, Paul R. Pintrich, James Raths, and Merlin C. Wittrock. *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*. Pearson, New York, 1st revised edition edition edition, December 2000. ISBN 978-0-8013-1903-7.

Richard C. Anderson. How to Construct Achievement Tests to Assess Comprehension. *Review of Educational Research*, 42(2):145, 1972. ISSN 00346543. doi: 10.2307/1170014.

J. L. Austin. *How to Do Things with Words*. Oxford University Press, 1962.

Imran S. Bajwa, M. Shahid Naweed, M. Nadim Asif, and S. Irfan Hyder. Feature based image classification by using principal component analysis. *ICGST Int. J. Graph. Vis. Image Process. GVIP*, 9:11–17, 2009.

Urvashi Bakshi and Rohit Singhal. A survey on face detection methods and feature extraction techniques of face recognition. *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)*, 3(3):233–237, 2014. URL http://www.mihantarjomeh.com/wp-content/uploads/2015/01/A-SURVEY-ON-FACE-DETECTION.pdf.

Brock E. Barry, Jonathan Bodenhamer, and James J. O'Brien Jr. Student nonverbal communication in the classroom. In *American Society for Engineering Education*. American Society for Engineering Education, 2011. URL https://pdfs.semanticscholar.org/0c80/7f665e4785a58fc499bd6f9b96a43469bb52.pdf.

Marian Stewart Bartlett, Joseph C. Hager, Paul Ekman, and Terrence J. Sejnowski. Measuring facial expressions by computer image analysis. *Psychophysiology*, 36(2):253–263, 1999. URL http://onlinelibrary.wiley.com/doi/10.1017/S0048577299971664/full.

Victor Basili and Dieter Rombach. The Goal Question Metric Paradigm. In *Encyclopedia of Software Engineering*, volume 2. John Wiley & Sons, Inc., 1994.

Roman Bednarik and Markku Tukiainen. An eye-tracking methodology for characterizing program comprehension processes. In *Proceedings of the 2006 symposium on Eye tracking research & applications*, pages 125–132. ACM, 2006. URL http://dl.acm.org/citation.cfm?id=1117356.

Ph Besnard, M.-O. Cordier, and Yves Moinard. Ontology-based inference for causal explanation. In *International Conference on Knowledge Science, Engineering and Management*, pages 153–164. Springer, 2007. URL http://link.springer.com/chapter/10.1007/978-3-540-76719-0_18.

Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python*. O'Reilly, Beijing ; Cambridge [Mass.], 1st ed edition, 2009a. ISBN 978-0-596-51649-9.

Steven Bird, Ewan Klein, and Edward Loper. Accessing Text Corpora and Lexical Resources, 2009b. URL http://www.nltk.org/book/ch02.html.

Charles F. Bond, Adnan Omar, Adnan Mahmoud, and Richard Neal Bonser. Lie detection across cultures. *Journal of Nonverbal Behavior*, 14(3):189–204, 1990. ISSN 1573-3653. doi: 10.1007/BF00996226. URL http://dx.doi.org/10.1007/BF00996226.

Luca Botturi. Instructional design & learning technology standards. Technical report, Università della Svizzera italiana, 2003. URL https://doc.rero.ch/record/5154/files/1_icefq09.pdf.

Kristy Elizabeth Boyer, Robert Phillips, Michael Wallis, Mladen Vouk, and James Lester. Balancing cognitive and motivational scaffolding in tutorial dialogue. In *International Conference on Intelligent Tutoring Systems*, pages 239–249. Springer, 2008. URL http://link.springer.com/chapter/10.1007/978-3-540-69132-7_28.

Fiona J. Buckingham, Keeley A. Crockett, Zuhair A. Bandar, James D. O'Shea, Kathleen M. MacQueen, and Mario Chen. Measuring human comprehension from nonverbal behaviour using artificial neural networks. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pages 1–8. IEEE, 2012. URL http://ieeexplore.ieee.org/abstract/document/6252414/.

Fiona J. Buckingham, Keeley A. Crockett, Zuhair A. Bandar, and James D. O'Shea. FATHOM: A neural network-based non-verbal human comprehension detection system for learning environments. In *Computational Intelligence*

and Data Mining (CIDM), 2014 IEEE Symposium on, pages 403–409. IEEE, 2014. URL http://ieeexplore.ieee.org/abstract/document/7008696/.

H Burns, C. A. Luckhardt, J. W. Parlett, and C. L. Redfield. *Intelligent Tutoring Systems: Evolutions in Design*. Psychology Press, 1991.

Zhinqiang Cai, Danielle S. McNamara, Max Louwerse, Xiangen Hu, Mike Rowe, Arthur C. Graesser, Daniel Casasanto, Lera Boroditsky, Webb Philips, Jesse Greene, and others. NLS: A non-latent similarity algorithm. In *Proceedings of the Cognitive Science Society*, volume 26, 2004. URL http://escholarship.org/uc/item/0sc5p977.pdf.

Zhiqiang Cai, A. C. Graesser, Carol Forsyth, Candice Burkett, Keith Millis, Patricia Wallace, Diane Halpern, and Heather Butler. Trialog in ARIES: User input assessment in an intelligent tutoring system. In *Proceedings of the 3rd IEEE international conference on intelligent computing and intelligent systems*, pages 429–433, 2011.

Kate Cain and Jane Oakhill. Assessment matters: Issues in the measurement of reading comprehension. *British Journal of Educational Psychology*, 76(4): 697–708, December 2006. ISSN 00070998. doi: 10.1348/000709905X69807. URL http://doi.wiley.com/10.1348/000709905X69807.

Rafael A Calvo and Sidney D'Mello. Affect Detection: An Interdisciplinary Review of Models, Methods, and Their Applications. *IEEE Transactions on Affective Computing*, 1(1):18–37, January 2010. ISSN 1949-3045. doi: 10.1109/T-AFFC.2010.1. URL http://ieeexplore.ieee.org/document/5520655/.

Modesto Castrillón, Oscar Déniz, Daniel Hernández, and Javier Lorenzo. A comparison of face and facial feature detectors based on the Viola–Jones general object detection framework. *Machine Vision and Applications*, February 2010. ISSN 0932-8092, 1432-1769. doi: 10.1007/s00138-010-0250-7. URL http://link.springer.com/10.1007/s00138-010-0250-7.

Modesto Castrillón-Santana, Oscar Déniz Suárez, Luis Antón Canalís, Lorenzo Navarro, and José Javier. Face and facial feauture detection evaluation: Performance evaluation of public domain haar detectors for face and facial feature detection. 2008.

Courtney B. Cazden and Vera P. John. *Functions of Language in the Classroom*. Waveland Pr Inc, Prospect Heights, Ill, March 1985. ISBN 978-0-88133-151-6.

Sheng-Chang Chen, Hsiao-Ching She, Ming-Hua Chuang, Jiun-Yu Wu, Jie-Li Tsai, and Tzyy-Ping Jung. Eye movements predict students' computer-based assessment performance of physics concepts in different presentation modalities. *Computers & Education*, 74:61–72, May 2014. ISSN 03601315. doi: 10.1016/j.compedu.2013.12.012. URL http://linkinghub.elsevier.com/retrieve/pii/S0360131513003369.

Kathleen F. Clark and Michael F. Graves. Scaffolding Students' Comprehension of Text. *The Reading Teacher*, 58(6):570–580, March 2005. ISSN 00340561. doi: 10.1598/RT.58.6.6. URL http://doi.wiley.com/10.1598/RT.58.6.6.

Allan Collins, John Seely Brown, and Susan E. Newman. Cognitive apprenticeship: Teaching the craft of reading, writing and mathematics. *Thinking: The Journal of Philosophy for Children*, 8(1):2–10, 1988. URL https://www.pdcnet.org/thinking/content/thinking_1988_0008_0001_0002_0010.

Henri G. Colt, Mohsen Davoudi, Septimiu Murgu, and Nazanin Zamanian Rohani. Measuring learning gain during a one-day introductory bronchoscopy course. *Surgical Endoscopy*, 25(1):207–216, January 2011. ISSN 0930-2794, 1432-2218. doi: 10.1007/s00464-010-1161-4. URL http://link.springer.com/10.1007/s00464-010-1161-4.

Ltd ConvAgent. ConvAgent. URL http://www.convagent.com/. [Online] [Accessed 2017-05-31].

Saman Cooray and Noel O'Connor. Facial feature extraction and principal component analysis for face detection in color images. *Image Analysis and Recognition*, pages 741–749, 2004.

Leana Copeland, Tom Gedeon, and Sumudu Mendis. Predicting reading comprehension scores from eye movements using artificial neural networks and fuzzy output error. *Artificial Intelligence Research*, 3(3), August 2014. ISSN 1927-6982, 1927-6974. doi: 10.5430/air.v3n3p35. URL http://www.sciedu.ca/journal/index.php/air/article/view/5085.

Lee Cronbach and Lita Furby. How we should measure "change," - or should we? *Psychological Bulletin*, 74(1):68 – 80, 1970.

K. Sidney D'Mello, Scotty D. Craig, Barry Gholson, Stan Franklin, Rosalind Picard, and Arthur C. Graesser. Integrating affect sensors in an intelligent tutoring system. In *Affective Interactions: The Computer in the Affective Loop Workshop at*, pages 7–13, 2005. URL http://www.academia.edu/download/12828307/iui-ai-05.pdf.

Sidney D'Mello and Art Graesser. Dynamics of affective states during complex learning. *Learning and Instruction*, 22(2):145–157, April 2012. ISSN 09594752.

Sidney K. D'Mello and Arthur Graesser. Multimodal semi-automated affect detection from conversational cues, gross body language, and facial features. *User Modeling and User-Adapted Interaction*, 20(2):147–187, June 2010. ISSN 0924-1868, 1573-1391. doi: 10.1007/s11257-010-9074-4. URL http://link.springer.com/10.1007/s11257-010-9074-4.

Gwyneth Doherty-Sneddon and Fiona G. Phelps. Teachers' Responses to Children's Eye Gaze. *Educational Psychology*, 27(1):93–109, February 2007. ISSN 0144-3410, 1469-5820. doi: 10.1080/01443410601061488. URL http://www.tandfonline.com/doi/abs/10.1080/01443410601061488.

Paul Ekman and Wallace V. Friesen. *Unmasking the Face: A Guide to Recognizing Emotions from Facial Clues*. Malor Books, Cambridge, MA, 2003. ISBN 978-1-883536-36-7.

K. Emmorey, R. Thompson, and R. Colvin. Eye Gaze During Comprehension of American Sign Language by Native and Beginning Signers. *Journal of Deaf Studies and Deaf Education*, 14(2):237–243, July 2008. ISSN 1081-4159, 1465-7325. doi: 10.1093/deafed/enn037. URL https://academic.oup.com/jdsde/article-lookup/doi/10.1093/deafed/enn037.

Peggy A. Ertmer and Timothy J. Newby. Behaviorism, cognitivism, constructivism: Comparing critical features from an instructional design perspective. *Performance Improvement Quarterly*, 26(2):43–71, 2013. URL http://onlinelibrary.wiley.com/doi/10.1002/piq.21143/full.

Stephen H. Fairclough, Alexander J. Karran, and Kiel Gilleade. Classification Accuracy from the Perspective of the User: Real-Time Interaction with Physiological Computing. pages 3029–3038. ACM Press, 2015. ISBN 978-1-4503-3145-6. doi: 10.1145/2702123.2702454. URL http://dl.acm.org/citation.cfm?doid=2702123.2702454.

Elizabeth Figa and Paul Tarau. Conversational Agents as Web Services. In Elizabeth Figa and Paul Tarau, editors, *Proceedings of the International Conference on Internet Computing*, 2004.

F. A. Fonte, Juan Carlos, Burguillo Rial, and Martín Llamas Nistal. TQ-Bot: An AIML-based Tutor and Evaluator Bot. *Journal of Universal Computer Science*, 15(7):1486–1495, 2009. URL http://jucs.org/jucs_15_7/tq_bot_an_aiml/jucs_15_07_1486_1495_fonte.pdf.

Michael Geis. *Speech Acts and Conversational Interaction*. Cambridge University Press, 1995.

Peter Gerjets, Carina Walter, Wolfgang Rosenstiel, Martin Bogdan, and Thorsten O. Zander. Cognitive state monitoring and the design of adaptive instruction in digital environments: lessons learned from cognitive workload assessment using a passive brain-computer interface approach. *Frontiers in Neuroscience*, 8, December 2014. ISSN 1662-453X. doi: 10.3389/fnins.2014.00385. URL http://journal.frontiersin.org/article/10.3389/fnins.2014.00385/abstract.

Arthur C. Graesser, Natalie K. Person, and Joseph P. Magliano. Collaborative dialogue patterns in naturalistic one-to-one tutoring. *Applied Cognitive Psychology*, 9(6):495–522, 1995. ISSN 1099-0720. doi: 10.1002/acp.2350090604. URL http://dx.doi.org/10.1002/acp.2350090604.

Arthur C. Graesser, Peter Wiemer-Hastings, Katja Wiemer-Hastings, Derek Harter, Tutoring Research Group Tutoring Research Group, and Natalie Person. Using latent semantic analysis to evaluate the contributions of students in AutoTutor. *Interactive learning environments*, 8(2):129–147, 2000. URL http://www.tandfonline.com/doi/abs/10.1076/1049-4820(200008)8:2;1-B;FT129.

Arthur C. Graesser, Kurt VanLehn, Carolyn P. Rosé, Pamela W. Jordan, and Derek Harter. Intelligent tutoring systems with conversational dialogue. *AI magazine*, 22(4):39, 2001. URL http://www.aaai.org/ojs/index.php/aimagazine/article/view/1591.

Arthur C. Graesser, Kristen Moreno, Johanna Marineau, Amy Adcock, Andrew Olney, and Natalie K. Person. AutoTutor improves deep learning of computer literacy: Is it the dialog or the talking head? *Artificial intelligence in education: Shaping the future of learning through intelligent technologies*, 97:47, 2003.

Arthur C. Graesser, Shulan Lu, George Tanner Jackson, Heather Hite Mitchell, Mathew Ventura, Andrew Olney, and Max M. Louwerse. AutoTutor: A tutor with dialogue in natural language. *Behavior Research Methods, Instruments, & Computers*, 36(2):180–192, 2004.

Alex Graves, Santiago Fernández, editor="de Sá Joaquim Marques Schmidhuber, Jürgen", Luís A. Alexandre, Włodzisław Duch, and Danilo Mandic.

Multi-dimensional Recurrent Neural Networks, pages 549–558. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

Angela Grünberg. Saying and Doing: Speech Actions, Speech Acts and Related Events: Saying and Doing. *European Journal of Philosophy*, 22(2):173–199, June 2014. ISSN 09668373. doi: 10.1111/j.1468-0378.2011.00481.x. URL http://doi.wiley.com/10.1111/j.1468-0378.2011.00481.x.

Varsha Gupt and Dipesh Sharma. A Study of Various Face Detection Methods. *International Journal of Advanced Research in Computer and Communication Engineering*, 3(5), May 2014.

Kent L. Gustafson and Robert Maribe Branch. Revisioning models of instructional development. *Educational Technology Research and Development*, 45 (3):73–89, 1997. URL http://link.springer.com/article/10.1007/BF02299731.

Eija Haapalainen, SeungJun Kim, Jodi F. Forlizzi, and Anind K. Dey. Psychophysiological measures for assessing cognitive load. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pages 301–310. ACM, 2010. URL http://dl.acm.org/citation.cfm?id=1864395.

Tran Son Hai, Le Hoang Thai, and Nguyen Thanh Thuy. Facial Expression Classification Using Artificial Neural Network and K-Nearest Neighbor. *International Journal of Information Technology and Computer Science*, 7(3): 27–32, February 2015.

R. R. Hake. *Should we measure change? Yes.* in press, American Evaluation Association, 2010. URL https://www.researchgate.net/profile/Richard_Hake/publication/255649307_Should_We_Measure_Change_Yes/links/54886b1a0cf289302e30acb2/Should-We-Measure-Change-Yes.pdf.

John Harris, Jeffrey L. Hirst, and Michael Mossinghoff. Hamiltonian Paths and Cycles. In *Combinatorics and Graph Theory*, Undergraduate Texts in Mathematics. Springer-Verlag New York, 2 edition, 2008. URL http://www.springer.com/gp/book/9780387797106.

Hiyam Hatem, Zuo Beiji, and Raed Majeed. A Survey of Feature Base Methods for Human Face Detection. *International Journal of Control and Automation*, 8(5):61–78, May 2015. doi: 10.14257/ijca.2015.8.5.07. URL http://www.sersc.org/journals/IJCA/vol8_no5/7.pdf.

S Haykin. *Neural Networks: A Comprehensive Foundation.* Macmillan, 1994.

D Haylock and F Thangata. *Key Concepts in Teaching Primary Mathematics.* Sage Publications UK, Thousand Oaks, CA., 2007.

N.T. Heffernan and E.A. Croteau. Web-based evaluations showing differential learning for tutorial strategies employed by the ms. lindquist tutor. *Proceedings of 7th Annual Intelligent Tutoring Systems Conference*, 2004.

R Heinich, M Molenda, J. D Russell, and S Smaldino. *Instructional Media and New Technologies of Instruction.* Prentice Hall, Englewood Cliffs, N.J, 5 edition, 1995.

Mike Holmes, Annabel Latham, Keeley Crockett, James D. O'Shea, and Cathy Lewin. Hendrix: A conversational intelligent tutoring system for Java programming. Presented at the 15th annual UK workshops on Computational Intelligence, Exeter, UK, 2015a.

Mike Holmes, Annabel Latham, Keeley Crockett, and James D. O'Shea. Modelling e-learner comprehension within a conversational intelligent tutoring system. In *Proceedings of the 2017 World conference on Computers in Education*. IFIP, 2017a.

Mike Holmes, Annabel Latham, Keeley Crockett, and James D. O'Shea. Near real-time comprehension classification with artificial neural networks: decoding e-Learner non-verbal behaviour. *IEEE Transactions on Learning Technologies*, November 2017b.

Wayne Holmes, Manolis Mavrikis, Alice Hansen, and Beate Grawemeyer. Purpose and Level of Feedback in an Exploratory Learning Environment for Fractions. In Cristina Conati, Neil Heffernan, Antonija Mitrovic, and M. Felisa Verdejo, editors, *Artificial Intelligence in Education*, volume 9112, pages 620–623. Springer International Publishing, Cham, 2015b. ISBN 978-3-319-19772-2 978-3-319-19773-9. doi: 10.1007/978-3-319-19773-9_76. URL http://link.springer.com/10.1007/978-3-319-19773-9_76.

Kurt Hornik, maxwell Stinchcombe, and Halbert White. Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, 2:359–366, 1989.

Daniel Hrubes and Robert S. Feldman. Nonverbal Displays as Indicants of Task Difficulty. *Contemporary Educational Psychology*, 26(2):267–276, April 2001. ISSN 0361476X. doi: 10.1006/ceps.2000.1049. URL http://linkinghub.elsevier.com/retrieve/pii/S0361476X0091049X.

Michael Hüsken and Peter Stagge. Recurrent neural networks for time series classification. *Neurocomputing*, 50:223–235, 2003. URL http://www.sciencedirect.com/science/article/pii/S0925231201007068.

Fei Hu, Li Li, Zi-Li Zhang, Jing-Yuan Wang, and Xiao-Fei Xu. Emphasizing Essential Words for Sentiment Classification Based on Recurrent Neural Networks. *Journal of Computer Science and Technology*, 32(4):785–795, July 2017. ISSN 1000-9000, 1860-4749. doi: 10.1007/s11390-017-1759-2. URL http://link.springer.com/10.1007/s11390-017-1759-2.

Thomas Huk and Stefan Ludwigs. Combining cognitive and affective support in order to promote learning. *Learning and Instruction*, 19(6):495–505, December 2009. ISSN 09594752. doi: 10.1016/j.learninstruc.2008.09.001. URL http://linkinghub.elsevier.com/retrieve/pii/S0959475208000911.

Chris Husbands and Jo Pearce. What makes great pedagogy? Nine claims from research. *National College for School Leadership*, 2012. URL http://universityofhullscitts.org.uk/scitts/site/downloads/Core3_Husbands.pdf.

IEEE. IEEE Reference Guide for Instructional Design, January 2015. URL http://www.ieee.org/education_careers/education/ref_guide/index.html.

Ryo Ishii, Yukiko I. Nakano, and Toyoaki Nishida. Gaze awareness in conversational agents: Estimating a user's conversational engagement from eye gaze. *ACM Transactions on Interactive Intelligent Systems*, 3(2):1, July 2013. ISSN 21606455. doi: 10.1145/2499474.2499480. URL http://dl.acm.org/citation.cfm?doid=2499474.2499480.

David Jonassen, Mark Davidson, Mauri Collins, John Campbell, and Brenda Bannan Haag. Constructivism and computer-mediated communication in distance education. *American Journal of Distance Education*, 9(2):7–26, January 1995. ISSN 0892-3647, 1538-9286. doi: 10.1080/08923649509526885. URL http://www.tandfonline.com/doi/abs/10.1080/08923649509526885.

Samira Ebrahimi Kahou, Xavier Bouthillier, Pascal Lamblin, Caglar Gulcehre, Vincent Michalski, Kishore Konda, Sébastien Jean, Pierre Froumenty, Yann Dauphin, Nicolas Boulanger-Lewandowski, Raul Chandias Ferrari, Mehdi Mirza, David Warde-Farley, Aaron Courville, Pascal Vincent, Roland Memisevic, Christopher Pal, and Yoshua Bengio. EmoNets: Multimodal deep learning approaches for emotion recognition in video. *Journal on Multimodal User Interfaces*, 10(2):99–111, June 2016.

Patrik Kamencay, Robert Hudec, Miroslav Benco, and Martina Zachariasova. Feature extraction for object recognition using PCA-KNN with application to medical image analysis. In *Telecommunications and Signal Processing (TSP), 2013 36th International Conference on*, pages 830–834. IEEE, 2013.

Sangwoo Kang, Harksoo Kim, and Jungyun Seo. A reliable multidomain model for speech act classification. *Pattern Recognition Letters*, 31(1):71–74, January 2010. ISSN 01678655. doi: 10.1016/j.patrec.2009.08.013. URL http://linkinghub.elsevier.com/retrieve/pii/S016786550900227X.

Alexander J. Karran, Stephen H. Fairclough, and Kiel Gilleade. A Framework for Psychophysiological Classification within a Cultural Heritage Context Using Interest. *ACM Transactions on Computer-Human Interaction*, 21 (6):1–19, January 2015. ISSN 10730516. doi: 10.1145/2687925. URL http://dl.acm.org/citation.cfm?doid=2722827.2687925.

S. P. Khandait, Ravindra C. Thool, and P. D. Khandait. Automatic facial feature extraction and expression recognition based on neural network. *International Journal of Advanced Computer Science and Applications*, 2011.

Kyoung-Min Kim, Jin-Hyuk Hong, and Sung-Bae Cho. A semantic Bayesian network approach to retrieving information with intelligent conversational agents. *Information Processing & Management*, 43(1):225–236, January 2007. ISSN 03064573. doi: 10.1016/j.ipm.2006.04.001. URL http://linkinghub.elsevier.com/retrieve/pii/S030645730600063X.

Walter Kintsch. *Comprehension: A paradigm for cognition*. Cambridge University Press, Cambridge, MA, 1998.

Mark L. Knapp and Judith A. Hall. *Nonverbal communication in human interaction*. Wadsworth Cengage Learning, Belmont, Calif., 8th, international edition, 2014. ISBN 1285083512;9781285083513;.

Youngjoong Ko. New feature weighting approaches for speech-act classification. *Pattern Recognition Letters*, 51:107–111, January 2015. ISSN 01678655. doi: 10.1016/j.patrec.2014.08.014. URL http://linkinghub.elsevier.com/retrieve/pii/S0167865514002803.

Chang-Yen Lai and Wen-Ching Liou. Rapid ADDIE Curriculums Design Model Based on the Heterogeneous Multimedia Information Integration. pages 485–490. IEEE, December 2007. ISBN 978-0-7695-3084-0. doi: 10.1109/ISM.Workshops.2007.87. URL http://ieeexplore.ieee.org/document/4476016/.

Thomas K. Landauer, Danielle S. McNamara, Simon Dennis, and Walter Kintsch, editors. *Handbook of Latent Semantic Analysis*. Psychology Press, 1 edition edition, June 2014. ISBN 978-1-138-00419-1.

Agnieszka Landowska. Affect-awareness framework for intelligent tutoring systems. In *Human System Interaction (HSI), 2013 The 6th International Conference on*, pages 540–547. IEEE, 2013.

Annabel Latham, Keeley Crockett, David McLean, and Bruce Edmonds. A conversational intelligent tutoring system to automatically predict learning styles. *Computers & Education*, 59(1):95–109, August 2012a. ISSN 03601315.

Annabel Latham, Keeley A. Crockett, David McLean, and Bruce Edmonds. Adaptive Tutoring in an Intelligent Conversational Agent System. In *Transactions on Computational Collective Intelligence*, volume VIII of *Lecture Notes in Computer Science*, pages 148–167. Berlin: Springer-Verlag, 2012b.

Annabel Latham, Keeley Crockett, and David McLean. An adaptation algorithm for an intelligent natural language tutoring system. *Computers & Education*, 71:97–110, February 2014. ISSN 03601315.

Annabel Marie Latham. *Personalising learning with dynamic prediction and adaptation to learning styles in a conversational intelligent tutoring system*. PhD thesis, Manchester Metropolitan University, 2011.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2324, 1998.

Carol D. Lee. A culturally based cognitive apprenticeship: Teaching African American high school students skills in literary interpretation. *Reading research quarterly*, pages 608–630, 1995. URL http://www.jstor.org/stable/748192.

Yuhua Li, David McLean, Zuhair A. Bandar, James D. O'shea, and Keeley Crockett. Sentence similarity based on semantic nets and corpus statistics. *IEEE transactions on knowledge and data engineering*, 18(8):1138–1150, 2006. URL http://ieeexplore.ieee.org/abstract/document/1644735/.

Hao-Chiang Koong Lin, Chih-Hung Wu, and Ya-Ping Hsueh. The influence of using affective tutoring system in accounting remedial instruction on learning performance and usability. *Computers in Human Behavior*, 41: 514–522, December 2014. ISSN 07475632. URL http://linkinghub.elsevier.com/retrieve/pii/S074756321400510X.

Diane J. Litman and James F. Allen. A plan recognition model for subdialogues in conversations. *Cognitive Science*, 11:163 – 200, 1987.

André Teixeira Lopes, Edilson de Aguiar, Alberto F. De Souza, and Thiago Oliveira-Santos. Facial expression recognition with Convolutional Neural Networks: Coping with few data and the training sample order. *Pattern Recognition*, 61:610–628, January 2017.

Yongzhong Lu, Jingli Zhou, and Shengsheng Yu. A survey of face detection, extraction and recognition. *Computing and informatics*, 22(2):163–195, 2012. URL http://www.cai.sk/ojs/index.php/cai/article/viewArticle/453.

Apache Lucene. TFIDFSimilarity (Lucene 4.0.0 API). URL http://lucene.apache.org/core/4_0_0/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html. [Online] [Accessed 2017-04-12].

Sandra Machida. Teacher Accuracy in Decoding Nonverbal Indicants of.pdf. *Journal of Educational Psychology*, 78(6):454–464, 1986.

Gwen C. Marchand and Antonio P. Gutierrez. The role of emotion in the learning process: Comparisons between online and face-to-face learning settings. *The Internet and Higher Education*, 15(3):150–160, June 2012. ISSN 10967516. doi: 10.1016/j.iheduc.2011.10.001. URL http://linkinghub.elsevier.com/retrieve/pii/S1096751611000571.

Veena Mayya, Radhika M. Pai, and M.M. Manohara Pai. Automatic Facial Expression Recognition Using DCNN. *Procedia Computer Science*, 93:453–461, 2016.

A Mehrabian. Communication Without Words. *Psychology Today*, 2:53–56, 1968.

Fernando A. Mikic, Juan C. Burguillo, Daniel A. Rodríguez, Eduardo Rodríguez, and Martín Llamas. T-Bot and Q-Bot: A couple of AIML-based bots for tutoring courses and evaluating students. In *Frontiers in Education Conference, 2008. FIE 2008. 38th Annual*, pages S3A–7. IEEE, 2008. URL http://ieeexplore.ieee.org/abstract/document/4720469/.

Fernando A. Mikic, Juan C. Burguillo, Martín Llamas, Daniel A. Rodríguez, and Eduardo Rodríguez. CHARLIE: An AIML-based Chatterbot which Works as an Interface among INES and Humans. In *EAEEIE Annual Conference, 2009*, pages 1–6. IEEE, 2009. URL http://ieeexplore.ieee.org/abstract/document/5335493/.

George A. Miller. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995. URL http://dl.acm.org/citation.cfm?id=219748.

Marvin Minsky. How Computer Science will change our lives. In *Artificial Life V: Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems*, Nara, Japan, 1996. MIT Press.

Marvin Minsky and Seymour A. Papert. *Perceptrons: An Introduction to Computational Geometry, Expanded Edition*. The M.I.T. Press, Cambridge, Mass, expanded edition edition edition, 1969.

Tom Mitchell, Terry Russell, Peter Broomhead, and Nicola Aldridge. Towards robust computerised marking of free-text responses. 2002. URL https://dspace.lboro.ac.uk/dspace/handle/2134/1884.

Tom Mitchell, Nicola Aldridge, and Peter Broomhead. Computerised Marking of Short-Answer Free-Text Responses. In *Manchester IAEA conference*, 2003.

Cristian Moldovan, Vasile Rus, and Arthur C. Graesser. Automated Speech Act Classification For Online Chat. In *Modern Artificial Intelligence and Cognitive Science Conference*, volume 710, pages 23–29, 2011.

Moodle. Moodle - Open-source learning platform, 2017. URL https://moodle.org/. [Online] [Accessed 2017-08-23].

Neo4J. The Neo4j Manual v2.3.2 11.1. Match, a. URL http://neo4j.com/docs/snapshot/query-match.html.

Neo4J. The Neo4j Manual v2.3.2 11.1. Shortest Path, b. URL http://neo4j.com/docs/snapshot/query-match.html#query-shortest-path. [Online] [Accessed 2017-04-12].

David J. Nicol and Debra Macfarlane-Dick. Formative assessment and self-regulated learning: A model and seven principles of good feedback practice. *Studies in higher education*, 31(2):199–218, 2006.

Jane Oakhill and Nicola Yuill. Pronoun Resolution in Skilled and Less-Skilled Comprehenders: Effects of Memory Load and Inferential Complexity. *Language and speech*, 29:25, 1986.

Karen O'Shea. An approach to conversational agent design using semantic sentence similarity. *Applied Intelligence*, 37(4):558–568, December 2012. ISSN 0924-669X, 1573-7497. doi: 10.1007/s10489-012-0349-9. URL http://link.springer.com/10.1007/s10489-012-0349-9.

Karen O'Shea. Natural language scripting within conversational agent design. *Applied Intelligence*, 40(1):189–197, January 2014. ISSN 0924-669X, 1573-7497. doi: 10.1007/s10489-012-0408-2. URL http://link.springer.com/10.1007/s10489-012-0408-2.

Karen O'Shea, Zuhair Bandar, and Keeley Crockett. A semantic-based conversational agent framework. In *Internet Technology and Secured Transactions, 2009. ICITST 2009. International Conference for*, pages 1–8. IEEE, 2009. URL http://ieeexplore.ieee.org/abstract/document/5402582/.

Michel Owayjan, Roger Achkar, and Moussa Iskandar. Face Detection with Expression Recognition using Artificial Neural Networks. In *Biomedical Engineering (MECBME), 2016 3rd Middle East Conference on*, pages 115–119. IEEE, 2016.

Charlotte J. Patterson, J. Michael Cosgrove, and Ralph G. O'brien. Nonverbal indicants of comprehension and noncomprehension in children. *Developmental Psychology*, 16(1):38, 1980. URL http://psycnet.apa.org/journals/dev/16/1/38/.

K Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2:559–572, 1901.

Ron Petrusha. Handling and Raising Events, 2017. URL https://docs.microsoft.com/en-us/dotnet/standard/events/.

Pedro O. Pinheiro and Ronan Collobert. Recurrent Convolutional Neural Networks for Scene Labeling. In *Proceedings of the 31 st International Conference on Machine Learning*, Beijing, China, 2014. JMLR: W&CP.

Kaska Porayska-Pomsta, Chris Mellish, and Helen Pain. Aspects of speech act categorisation: towards generating teachers' language. *International Journal of Artificial Intelligence in Education (IJAIED)*, 11:254–272, 2000. URL https://telearn.archives-ouvertes.fr/hal-00257108/.

Ramkumar Rajendran, Sridhar Iyer, Sahana Murthy, Campbell Wilson, and Judithe Sheard. A Theory-Driven Approach to Predict Frustration in an ITS. *IEEE Transactions on Learning Technologies*, 6(4):378–388, October 2013. ISSN 1939-1382. doi: 10.1109/TLT.2013.31. URL http://ieeexplore.ieee.org/document/6589592/.

Jeff Rickel and W. Lewis Johnson. Task-oriented collaboration with embodied agents in virtual worlds. *Embodied conversational agents*, pages 95–122, 2000. URL http://web.media.mit.edu/~cynthiab/Readings/rickel-taskoriented-00.pdf.

R Rojas. The Backpropagation Algorithm. In *Neural Networks*, pages 151–184. Springer, 1996a.

Raul Rojas. Perceptron learning. In *Neural Networks*, pages 77–98. Springer, 1996b. URL http://link.springer.com/chapter/10.1007/978-3-642-61068-4_4.

F. Rosenblatt. *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan Books, 1962.

Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958. URL http://psycnet.apa.org/journals/rev/65/6/386/.

Janet Rothwell, Zuhair Bandar, James O'Shea, and David McLean. Silent talker: a new computer-based system for the analysis of facial cues to deception. *Applied Cognitive Psychology*, 20(6):757–777, September 2006. ISSN 08884080, 10990720. doi: 10.1002/acp.1204. URL http://doi.wiley.com/10.1002/acp.1204.

Janet Rothwell, Zuhair Bandar, James O'Shea, and David McLean. Charting the behavioural state of a person using a backpropagation neural network. *Neural Computing and Applications*, 16(4-5):327–339, May 2007. ISSN 0941-0643, 1433-3058. doi: 10.1007/s00521-006-0055-9. URL http://link.springer.com/10.1007/s00521-006-0055-9.

Farzaneh Saadati, Rohani Ahmad Tarmizi, Ahmad Fauzi Mohd Ayub, and Kamariah Abu Bakar. Effect of Internet-Based Cognitive Apprenticeship Model (i-CAM) on Statistics Learning among Postgraduate Students. *PLOS ONE*, 10(7):e0129938, July 2015. ISSN 1932-6203. doi: 10.1371/journal.pone.0129938. URL http://dx.plos.org/10.1371/journal.pone.0129938.

Salisu Sani and Teh N.M. Aris. Computational Intelligence Approaches for Student/Tutor Modelling: A Review. pages 72–76. IEEE, January 2014. ISBN 978-1-4799-3858-2. doi: 10.1109/ISMS.2014.21. URL http://ieeexplore.ieee.org/document/7280882/.

Alan H Schoenfeld. Learning to think mathematically: Problem solving, metacognition, and sense-making in mathematics. In D Grouws, editor, *Handbook for Research on Mathematics Teaching and Learning*, pages 334–370. New York: MacMillan, 1992. URL http://cimm.ucr.ac.cr/ojs/index.php/eudoxus/article/download/177/312.

J. R. Searle. *Speech Acts: An essay in the Philosophy of Language*. Cambridge University Press, 1969.

Francisco J. Serón and Carlos Bobed. VOX system: a semantic embodied conversational agent exploiting linked data. *Multimedia Tools and Applications*, 75(1):381–404, January 2016. ISSN 1380-7501, 1573-7721. doi: 10.1007/s11042-014-2295-5. URL http://link.springer.com/10.1007/s11042-014-2295-5.

K. Dmello Sidney, Scotty D. Craig, Barry Gholson, Stan Franklin, Rosalind Picard, and Arthur C. Graesser. Integrating affect sensors in an intelligent tutoring system. In *Affective Interactions: The Computer in the Affective Loop Workshop at*, pages 7–13, 2005. URL http://www.academia.edu/download/12828307/iui-ai-05.pdf.

Stephanie Ann Siler and Kurt VanLehn. Investigating Microadaptation in One-to-One Human Tutoring. *The Journal of Experimental Education*, 83 (3):344–367, July 2015. ISSN 0022-0973, 1940-0683. doi: 10.1080/00220973.2014.907224. URL http://www.tandfonline.com/doi/full/10.1080/00220973.2014.907224.

Emma V. Smith, Keeley Crockett, Annabel Latham, Fiona Buckingham, and Mohammed Kaleem. SEEKER: A Conversational Agent as a Natural Language Interface to a Relational Database. In *World Congress on Engineering, WCE 2014*, World Congress on Engineering, WCE 2014. IAENG, London, 2014. ISBN 978-988-19252-7-5.

Catherine E. Snow. *Reading for understanding: toward an R&D program in reading comprehension*. Rand, Santa Monica, CA, 2002. ISBN 978-0-8330-3105-1.

Margaret Snowling, Kate Cain, Kate Nation, and Jane Oakhill. Reading comprehension: nature, assessment and teaching. 2009. URL http://eprints.lancs.ac.uk/50134/.

Michael Swan and Catherine Walter. Misunderstanding comprehension. *ELT Journal*, page ccw094, January 2017. ISSN 0951-0893, 1477-4526. doi: 10.1093/elt/ccw094. URL https://academic.oup.com/eltj/article-lookup/doi/10.1093/elt/ccw094.

John Sweller. *Instructional design in technical areas*. Camberwell, Australia: ACER Press, 1999.

Kathleen M. Swigger and Brigitte B. Holman. Intelligent tutoring systems for Air Force applications. In *Aerospace and Electronics Conference, 1988. NAECON 1988., Proceedings of the IEEE 1988 National*, pages 964–968. IEEE, 1988. URL http://ieeexplore.ieee.org/abstract/document/195125/.

Teng Teng and Xubo Yang. Facial expressions recognition based on convolutional neural networks for mobile virtual reality. pages 475–478. ACM Press, 2016.

Jing Tian, Andrzej P. Wierzbicki, Hongtao Ren, and Yoshiteru Nakamori. Constructing an ontology for a research program from a knowledge science perspective. In *International Conference on Knowledge Science, Engineering and Management*, pages 372–383. Springer, 2007. URL http://link.springer.com/chapter/10.1007/978-3-540-76719-0_37.

Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin. Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks. *ACM Transactions on Graphics*, 33(5):1–10, September 2014.

Marije van Amelsvoort, Bart Joosten, Emiel Krahmer, and Eric Postma. Using non-verbal cues to (automatically) assess children's performance difficulties with arithmetic problems. *Computers in Human Behavior*, 29(3):654–664, May 2013. ISSN 07475632. doi: 10.1016/j.chb.2012.10.016. URL http://linkinghub.elsevier.com/retrieve/pii/S0747563212002968.

Kurt Vanlehn. The behavior of tutoring systems. *International journal of artificial intelligence in education*, 16(3): 227–265, 2006. URL http://content.iospress.com/articles/international-journal-of-artificial-intelligence-in-education/jai16-3-02.

Kurt VanLehn. The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. *Educational Psychologist*, 46(4):197–221, October 2011. ISSN 0046-1520, 1532-6985. doi: 10.1080/00461520.2011.611369. URL http://www.tandfonline.com/doi/abs/10.1080/00461520.2011.611369.

Kurt VanLehn, Jon Wetzel, Sachin Grover, and Brett van de Sande. Learning How to Construct Models of Dynamic Systems: An Initial Evaluation of the Dragoon Intelligent Tutoring System. *IEEE Transactions on Learning Technologies*, 10(2):154–167, April 2017. ISSN 1939-1382. doi: 10.1109/TLT.2016.2514422. URL http://ieeexplore.ieee.org/document/7374728/.

Boban Vesin, Mirjana Ivanović, Aleksandra Klašnja-Milićević, and Zoran Budimac. Protus 2.0: Ontology-based semantic recommendation in programming tutoring system. *Expert Systems with Applications*, 39(15):12229–12246, November 2012. ISSN 09574174. doi: 10.1016/j.eswa.2012.04.052. URL http://linkinghub.elsevier.com/retrieve/pii/S0957417412006495.

Paul Viola and Michael J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004. URL http://link.springer.com/article/10.1023/B:VISI.0000013087.49260.fb.

Aldert Vrij, Katherine Edward, Kim P. Roberts, and Ray Bull. Detecting deceit via analysis of verbal and nonverbal behavior. *Journal of Nonverbal behavior*, 24(4):239–263, 2000. URL http://www.springerlink.com/index/V7141Q784W326Q21.pdf.

Aldert Vrij, Samantha A. Mann, Ronald P. Fisher, Sharon Leal, Rebecca Milne, and Ray Bull. Increasing cognitive load to facilitate lie detection: The benefit of recalling an event in reverse order. *Law and human behavior*, 32(3):253–265, 2008. URL http://link.springer.com/article/10.1007/s10979-007-9103-y.

Marilyn A. Walker, Diane J. Litman, Candace A. Kamm, and Alicia Abella. PARADISE: A framework for evaluating spoken dialogue agents. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 271–280. Association for Computational Linguistics, 1997. URL http://dl.acm.org/citation.cfm?id=979652.

Marilyn A. Walker, Diane J. Litman, and Candace A. Kamm. AT&T Labs| Research 180 Park Avenue Florham Park, NJ 07932-0971 USA walker, diane, cak, abella@ research. att. com. 1998.

Richard Wallace. AIML - The Artificial Intelligence Markup Language. URL http://www.alicebot.org/aiml.html.

Richard Wallace. The Anatomy of A.L.I.C.E. In *Parsing the Turing Test Part III*, pages 181–210. 2009.

Richard Wallace. Annotated A.L.I.C.E. AIML Files, 2017. URL http://www.alicebot.org/aiml/aaa/. [Online] [Accessed 2017-07-17].

Yinglin Wang, Jianmei Guo, Tao Hu, and Jie Wang. An ontology-based framework for building adaptable knowledge management systems. In *International Conference on Knowledge Science, Engineering and Management*, pages 655–660. Springer, 2007. URL http://link.springer.com/chapter/10.1007/978-3-540-76719-0_74.

Geoffrey I. Webb, Michael J. Pazzani, and Daniel Billsus. Machine learning for user modeling. *User modeling and user-adapted interaction*, 11(1):19–29, 2001. URL http://www.springerlink.com/index/RL172M3L46614737.pdf.

James M. Webb, Ellen M. Diana, Pamela Luft, Elizabeth W. Brooks, and Elizabeth L. Brennan. Influence of pedagogical expertise and feedback on assessing student comprehension from nonverbal behavior. *The Journal of Educational Research*, 91(2):89–97, 1997. URL http://www.tandfonline.com/doi/abs/10.1080/00220679709597526.

P. J. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, 1974.

Jacob Whitehill, Marian Bartlett, and Javier Movellan. Automatic facial expression recognition for intelligent tutoring systems. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*, pages 1–6. IEEE, 2008. URL http://ieeexplore.ieee.org/abstract/document/4563182/.

Jacob Whitehill, Zewelanji Serpell, Aysha Foster, Yi-Ching Lin, Brittney Pearson, Marian Bartlett, and Javier Movellan. Towards an optimal affect-sensitive instructional system of cognitive skills. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*, pages 20–25. IEEE, 2011. URL http://ieeexplore.ieee.org/abstract/document/5981778/.

Andrea Stevenson Won, Jeremy N. Bailenson, and Joris H. Janssen. Automatic Detection of Nonverbal Behavior Predicts Learning in Dyadic Interactions. *IEEE Transactions on Affective Computing*, 5(2):112–125, April 2014. ISSN 1949-3045. doi: 10.1109/TAFFC.2014.2329304. URL http://ieeexplore.ieee.org/document/6827904/.

Pornpit Wongthongtham and Behrang Zadjabbari. Signifying ontology complexity for knowledge sharing. In *Internet Technology and Secured Transactions, 2009. ICITST 2009. International Conference for*, pages 1–5. IEEE, 2009. URL http://ieeexplore.ieee.org/abstract/document/5402638/.

David Wood and Heather Wood. Vygotsky, Tutoring and Learning. *Oxford Review of Education*, 22(1):5–16, March 1996. ISSN 0305-4985, 1465-3915. doi: 10.1080/0305498960220101. URL http://www.tandfonline.com/doi/full/10.1080/0305498960220101.

Gary Woolley. Reading Comprehension. In *Reading Comprehension*, pages 15–34. Springer Netherlands, Dordrecht, 2011. ISBN 978-94-007-1173-0 978-94-007-1174-7. URL http://www.springerlink.com/index/10.1007/978-94-007-1174-7_2. DOI: 10.1007/978-94-007-1174-7_2.

Binjie Xiao. Principal component analysis for feature extraction of image sequence. In *Computer and Communication Technologies in Agriculture Engineering (CCTAE), 2010 International Conference On*, volume 1, pages 250–253. IEEE, 2010.

Ming-Hsuan Yang, David J. Kriegman, and Narendra Ahuja. Detecting faces in images: A survey. *IEEE Transactions on pattern analysis and machine intelligence*, 24(1):34–58, 2002. URL http://ieeexplore.ieee.org/abstract/document/982883/.

Shehnaaz Yusuf, Huzefa Kagdi, and Jonathan I. Maletic. Assessing the comprehension of UML class diagrams via eye tracking. In *Program Comprehension, 2007. ICPC'07. 15th IEEE International Conference on*, pages 113–122. IEEE, 2007. URL http://ieeexplore.ieee.org/abstract/document/4268246/.

Goranka Zoric, Karlo Smid, and Igor S. Pandi. Facial Gestures: Taxonomy and Application of Nonverbal, Nonemotional Facial Displays for Embodied Conversational Agents. In Toyoaki Nishida, editor, *Wiley Series in Agent Technology*, pages 161–182. John Wiley & Sons, Ltd, Chichester, UK, November 2007.

Appendix A

Pilot study tutorial content

Hendrix 1.0: pilot study tutorial content

In this section the structure, nodes and relationships of the ontology are specified. The objects and relations are based upon cognitivist devices including hierarchy and association, demonstration, definition, challenge and feedback. In this implementation metaphor and analogy are not included, as they are weaker relation to the knowledge domain of programming.

Node structure

1. Concept
    a. Example
    b. Definition
    c. Assessment
        i. Guidance
        ii. Code Sample

Node definitions

Node type	**Node fields**	**Field type**
Concept	Name Introduction Objectives	String String String Array
Example	Text Uri	String String
Definition	Text	String
Question	Text Answer Words Required Matches Max Attempts Feedback	String String Array Integer Integer String
Guidance	Information Question Answer Words Required Matches	String String String Array Integer
Code Sample	Uri	String

Relationship structure

1. Part of
    a. Definition
    b. Demonstration
    c. Assessment
        i. Guidance
        ii. Code

Relationship definitions

Relationship type	**Relationship fields**	**Field type**
Part of	Complexity	Double
Demonstration	Complexity	Double
Definition	Complexity	Double
Assessment	Order	Integer
Guidance	Order	Integer
Code Sample	-	-

Learning objectives

The ontology content is to represent the combined domains of knowledge, pedagogy and context. The ontology must therefore contain concepts, definitions, examples and questions to support the attainment of defined learning objectives.

The learning objectives to be considered for this implementation are:

Procedural / Apply

- Students will be able to construct loop constraints;
- Students will be able to construct if conditions;

Factual / Analyse

- Students will be able to identify different types of control constructs;
- Students will be able to identify scoping of variables within a loop;

Conceptual / Analyse

- Students will be able to evaluate the appropriateness of different types of control constructs;
- Students will be able to distinguish between for, while and do-while loops;
- Students will be able to distinguish between a true and false constraint condition;

Procedural / Analyse

- Students will be able to calculate loop iterations;
- Students will be able to evaluate variable values modified within loops constructs;

Content

In this section the content of the knowledge domain content is specified, using the nodes and relationships defined in the previous section, such as to meet the learning objectives stated above.

Concepts

For the initial experiment a single route through the ontology will be fully designed. The tree shown below indicates the tranche of the ontology to be designed. Nodes shown in grey will exist but will not have associations or content designed. Students will be asked to set ‘For Loops’ as a learning objective.

1. Programming
    a. Boolean logic
        i. Control flow statements
            1. Iteration
                a. For
                b. While
                c. Do
            2. Recursion
                a. Methods
                b. Classes
            3. Decision logic
                a. If

b. Switch

4. Branching

Programming

Concept : Programming	
Name	Programming
Introduction	While there are many programming languages, this tutorial will focus on learning the Java programming language. I have put a general definition of programming up, for you to read.
Objectives	In this tutorial you will learn about the fundamental concept of computer programming, as well as gaining awareness of some of the languages used to develop software.

Definition : Definition 1	
Text	Most people use computer programs every day, but few will consider how they work. For most people it is enough that a click of the button, or a swipe of the finger makes a computer perform the action they intended. As a programmer, however, it is your job to make sense of that click or swipe and instruct the computer on how to react. Computer programming at its' most basic is similar to writing out a set of instructions. Say you wanted to teach a friend how to make a scone, you could instruct them step by step: 1. *Weigh out the flour, then weigh out the butter.* 2. *Mix flour with butter and add an egg.* 3. *Add raisins and mixed fruit to taste.* 4. *Shape mixture into balls and place mixture onto baking tray, pressing down lightly to make a scone shape.* 5. *Heat the oven and bake the scones.* In computer programming, the sequence of instructions which describe a process is called an 'algorithm'. Algorithms are the principal concern of programmers, but a programmer must also consider the *domain* – the objects and items which exist within the algorithm - and the *logic* with which a sequence of instruction proceeds. Java is just one of many programming languages commonly used today. There are too many to list, but a few are C++, C#, Python, Ruby, Scala, JavaScript, PHP, Objective-C and VB. Each language, like a language you might have learned at school, will have slight differences in syntax – the words used, their spelling and the grammar expected. While these differences can make languages sound very different, all languages define algorithms and computers run those algorithms in response to events.
Relationship : Definition of (Programming)	
Complexity	0.0

Question : Question 1	
Text	Name three programming language other than Java.
Answer Words	C, C++, C#, Objective-C, Python, Ruby, Scala, Cobol, VB, VisualBasic, ASP, PHP, Fortran

Required Matches	3
Max Attempts	2
Feedback	There are many different programming languages. Java is just one. Many of the skills and techniques you will learn while mastering Java programming will be directly transferable into other programming languages. Like learning another human language, variations in spelling and grammar may make languages seem very different, but at their heart most programming languages are very similar to one another.
Relationship : Assessment for (Programming)	
Order	1

Question : Question 2	
Text	What are the three common concerns of a programming language?
Answer Words	'domain', 'algorithms', 'logic'
Required Matches	3
Max Attempts	2
Feedback	Algorithms are the principal concern of computer programming. Algorithms are a pre-defined set of instructions a computer must follow. Logic allows an algorithm to enact decision making. The domain is an important part of computer programming as the domain describes the objects contained within the algorithm.
Relationship : Assessment for (Programming)	
Order	2

Boolean logic

Concept : Boolean	
Name	Boolean Logic
Introduction	Boolean values are used in logic, which is the basis of Control Flow in executable code. I have put up a definition of which you might find useful.
Objectives	In this tutorial you will learn about Boolean operators and Boolean conditions. You will need to understand these concepts to progress onto decision logic, iterative loops or recursive methods.
Relationship : Part of (Programming)	
Complexity	0.0

Definition : Definition 1	
Text	Computer algorithms will often require the computer to make a choice about what action to take next. Imagine a simple system where-by we turn on a lamp if it is off, and turns off a lamp if it is on. 1. *If the lamp is on then turn off the lamp* 2. *If the light is off then turn on the lamp* In this simple algorithm we must make a logical assertion as to the current state of the light. The light example is useful to help you understand, as a light can only be in one of two states – the lamp is always either on or off. Similarly in computer programming, a Boolean can only in one of two states – it is either true or it is false.

	Boolean logic is used within computer programs to allow a program to make decisions within an algorithm based on an assertion about a variable or object, and that assertion will always be either true or false. Assertions made using Boolean values are called Boolean statements. Booleans conditions in programming code will often make use of operators. Common operators you will come across will be == (equal), != (not equal), < (less than) and > (greater than), <= (less than or equal to) and >= (greater than or equal to). All of these Boolean operators result in a true or false condition. For example, you may say: *If my age is greater than 18, allow me into the nightclub* And this could be presented programmatically as: *IF (age >= 18) { allowed_into_club = true }*
Relationship : Definition of (Boolean)	
Complexity	0.5

Definition : Definition 2	
Text	A Boolean value can represent two states, either true or false. These true or false values can then be used to modify the behaviour of a system within control flow statements such as decision logic or loops. Assertions made using Boolean values are called Boolean statements.
Relationship : Definition of (Boolean)	
Complexity	0.0

Example : Example 1	
Text	One of the most important uses of Boolean values is to determine a course of action.
Uri (content)	Example statements *if it is raining today, I will take my umbrella to university* The statement 'I will take my umbrella to university' is true if and only if the statement 'It is raining today' is also true. *if the bath water is cold, I will not get in* The statement 'I will not get in' is true if and only if the statement 'the bath water is cold' is also true. *if I have 10 apples and 10 is less than 20, then I have less than 20 apples* The statement 'I have less than 20 apples' is true if and only if the conditions 'I have 10 apples' and '10 is less than 20' are both true. This example is similar to the type of Boolean conditional statements you will commonly come across when writing programming code.
Relationship : Demonstration of (Boolean)	
Complexity	0.0

Question : Question 1	
Text	The Boolean statement 'If it is raining, I will take my umbrella to university', is true on what condition?
Answer Words	raining, rain
Required Matches	1
Max Attempts	2
Feedback	Thinking back to the definition, we know that a Boolean statement is only ever true or false, given a condition. In this Boolean statement the action 'take my umbrella' is dependent on the Boolean value of the condition 'it is raining'. In predicate logic the statement would be expressed as: *r (it is raining) and u (take my umbrella)* then $r \rightarrow u$.
Relationship : Assessment for (Boolean)	
Order	1

Question : Question 2	
Text	5 >= 10 is a Boolean conditional statement. Is it true or false?
Answer Words	false, not true, untrue
Required Matches	1
Max Attempts	1
Feedback	The statement shows a Boolean condition with the greater than or equal to operator. In this Boolean we are asserting the truth of the statement '5 is greater than or equal to 10'. In predicate formalism let *v (is equal to 5), c (is equal to 10),* and *t (v is greater than or equal to c)* then $v, c \rightarrow \bar{t}$.
Relationship : Assessment for (Boolean)	
Order	2

Guidance : Guidance	
Information	The Boolean statement asks whether 5 is greater than or equal to 10. The symbol > means greater than, and the symbol = means equal to.
Text	Is 5 >= 10?
Answer Words	no, false
Required Matches	1
Relationship : Guidance for (Question 2)	
Order	1

Question : Question 3	
Text	"five" != 5 is a Boolean conditional statement. Is it true or false?
Answer Words	True, not false, yes
Required Matches	1
Max Attempts	1
Feedback	This Boolean asserts the truth of the statement '"five" is not equal to 5'. This statement is a bit trickier than the previous, as "five" is a string and 5 is an integer. In a programming language, the *type* of a variable is significant when asserting equality. In predicate formalism let *v (is equal to "five"), c (is equal to 5) and t (v is not c)* then $v, c \rightarrow t$.

Relationship : Assessment for (Boolean)	
Order	3

Guidance : Guidance	
Information	The Boolean statement asks whether the string “five” is not equal to the integer 5. The symbol ! means not, and it acts to invert the predicate. In programming, a string and an integer are both primitive types.
Text	Given “five” is a string, and 5 is a integer. Is “five” the same as 5?
Answer Words	no, false
Required Matches	1
Relationship : Guidance for (Question 3)	
Order	1

Control Flow Statements

Concept : Control Flow	
Name	Control Flow Statements
Introduction	Control constructs break up the flow of execution by employing decision making, looping, and branching, enabling your program to conditionally execute particular blocks of code.
Objectives	In this tutorial you will learn about the different types of control flow statements you might use, and how they differ.
Relationship : Part of (Boolean Logic)	
Complexity	0.0

Definition : Definition 1	
Text	Program code is a linear set of instructions, forming the *algorithm*. By default, a computer will execute instructions in sequence, starting at the beginning of the instruction set and proceeding to the end. However, you may want to conditionally change the flow of logic, skip instructions or include optional instructions, based on some conditions. Control flow statements break up the flow of execution by employing decision making, looping, and branching behaviour. Control flow statements fall into three groups: decision-making statements (if-then and switch), loop statements (for, while, and do-while), and the branching statements (break, continue, and return). Conceptually, control flow statements allow a programmer to create programmatic representations of Boolean logic such as: *IF it is raining THEN take my umbrella ELSE take my sun glasses* *WHILE it is raining, keep my umbrella up* An interesting point to note is that a SWITCH statement, unlike any other type of control flow construct, can only assert equality (==) between two variables - all other control flow constructs can use the full range of conditional operators (==,<,>,!=,<=,>=).
Relationship : Definition of (Control Flow)	
Complexity	0.5

Definition : Definition 2	
Text	Control Constructs allow a programmer to manipulate the flow of code execution. Control Constructs can be grouped into three types: decision making logic, loops and branching. Decision making logic allow a programmer to switch between alternative code blocks, depending on a condition. Loops are a little more complex, allowing code to be repeated either a set number of times or until a condition is no longer true. Branching allows a programmer to control the flow of execution within a loop or decision making construct – for example, exiting a loop early.
Relationship : Definition of (Control Flow)	
Complexity	0.0

Example : Example 1	
Text	Control flow statements fall into three groups: decision-making statements (if-then and switch), loop statements (for, while, and do-while), and the branching statements (break, continue, and return).

Uri (content)	Example IF
	if (*variable operator comparator)* { *do this }*
	else { *do this }*
	
	Example SWITCH
	switch (*variable*) {
	case if *condition* true then: *do this* and *break;*
	case if *condition* true then: *do this* and *break;*
	case if no match then: *do this* and *break;*
	}
	
	Example FOR
	for (*initialization*; *termination*; *increment*) {
	do this
	}
	
	Example WHILE
	while (*variable operator termination*) {
	do something
	variable modifier;
	}
	
	Example DO-WHILE
	do {
	do something
	variable modifier;
	} while (*variable operator termination*);

Relationship : Demonstration of (Control Flow)	
Complexity	0.0

Question : Question 1

Text	Name two control constructs that are used for decision making.
Answer Words	if switch, if-else switch
Required Matches	2
Max Attempts	2
Feedback	The IF and SWITCH statements are the most basic forms of control flow statements, representing decision making logic for one (if), 2 (if-else) or many (switch) different cases.
Relationship : Assessment for (Control Flow)	
Order	1

Guidance : Guidance 1	
Information	The simplest type of decision making constructs allow a code block to execute if a condition is true, something like - IF a condition is true THEN do something.
Text	What is this statement called?
Answer Words	if, if-else, if else
Required Matches	1
Relationship : Guidance for (Question 1)	
Order	1

Guidance : Guidance 2	
Information	Decision making constructs allow a programmer to switch between different blocks of code.
Text	Other than an IF statement, what is another type of decision-making construct? Hint: this one allows for many cases.
Answer Words	switch
Required Matches	1
Relationship : Guidance for (Question 1)	
Order	2

Question : Question 2	
Text	Name three control constructs that are used for looping, or repeating code blocks.
Answer Words	for while do-while
Required Matches	3
Max Attempts	2
Feedback	The three forms types of loop are the For, While and Do-while loops. For is used where the programmer knows exactly how many times the loop must be repeated, and the do-while and while loops are used when the programmer wants the loop to continue until some condition is met.
Relationship : Assessment for (Control Flow)	
Order	2

Guidance : Guidance 1	
Information	Take a look at the example I\'ve put up.
Text	What type of loop is shown?
Answer Words	for
Required Matches	1
Relationship : Guidance for (Question 2)	
Order	1

Code Sample : For Loop Sample	

Uri (content)	**public static void** main(String[] args)
	{
	String numbers = "";
	for (**int** i = 0; i < 10; i++) {
	numbers = numbers + " " + i;
	}
	System.***out***.println(numbers);
	
	}

Relationship : Code for (Guidance 1)	
Order	1

Guidance : Guidance 2	
Information	I have put up another example of a loop.
Text	Which type of loop do you think this is?
Answer Words	while, do while
Required Matches	1
Relationship : Guidance for (Question 2)	
Order	2

Code Sample : While Loop Sample	

Uri (content)	**public static void** main(String[] args)
	{
	String apple = "apple";
	while (apple.length() % 2 != 0) {
	continue;
	}
	}

Relationship : Code for (Guidance 2)	

Guidance : Guidance 3	
Information	I have displayed the final code sample.
Text	Which type of loop do you think it shows?
Answer Words	do
Required Matches	1
Relationship : Guidance for (Question 2)	
Order	3

Code Sample : Do While Loop Sample	

Uri (content)	**public static void** main(String[] args)
	{
	String apple = "apple";
	do {
	break;
	} **while** (apple.length() % 2 != 0);
	
	}

Relationship : Code for (Guidance 3)	

Question : Question 3	
Text	Name three control constructs that are used for branching code.
Answer Words	break continue return
Required Matches	3
Max Attempts	2
Feedback	Branching code allows a programmer to manipulate the flow of execution within a control flow statement, for example, by exiting a loop or skipping to the next iteration. However, you are likely to come across branching code when using a Switch statement, as each case will often include a *break* clause.
Relationship : Assessment for (Control Flow)	
Order	3

Guidance : Guidance 1	
Information	The most basic type of branching construct allows a programmer to return a value, from the current code block, to the calling assignment.
Text	What is this branching construct called?
Answer Words	return
Required Matches	1
Relationship : Guidance for (Question 3)	
Order	1

Guidance : Guidance 2	
Information	The second type of branching construct allows a programmer to continue to the next step of an iteration, or Loop, skipping the remaining code within the loop. What is this construct called?
Text	What is this branching construct called?
Answer Words	continue
Required Matches	1
Relationship : Guidance for (Question 3)	
Order	2

Guidance : Guidance 3	
Information	The final type of branching construct is most often seen at the end of the cases in a switch statement. The construct allows a programmer to break execution and immediately exit the current code block.
Text	What is this branching construct called?
Answer Words	break
Required Matches	1
Relationship : Guidance for (Question 3)	
Order	3

Question : Question 4	
Text	You are designing a menu algorithm for a command line application. You have 10 options in your menu. To select an option from the menu, the user must enter a number from 1 to 10, each representing a different menu option. To decide which logic to execute given the menu selection, which type of construct would you use?
Answer Words	switch
Required Matches	1

Max Attempts	2
Feedback	A more complex type of construct allows a programmer to execute one of many 'cases', based on an input matching exactly with a case condition. This type of construct is particularly useful where a programmer has many distinct cases and need only execute one.
Relationship : Assessment for (Control Flow)	
Order	4

Iteration

Concept : Iteration	
Name	Iteration
Introduction	As a programmer you may find that most algorithms will involve repeating the same block of code, often many times. It is these situations that we will now try to identify and for which all programming languages offer further language constructs. The repeating of code is known as looping, or iteration.
Objectives	In this tutorial you will work towards identifying looping within an algorithm and identifying the most appropriate type of loop structure to use.
Relationship : Part of (Control Flow)	
Complexity	0.5

Definition : Definition 1	
Text	*Iteration* just means to repeat something, or in other words: a repetitive process. All programming languages have some mechanism for repeating stuff. You would use iteration if, say, you wanted to stick a label on each item of fruit on a grocer's stall. • *For each item of fruit, stick on a label* There are two main types of looping – deterministic and non-deterministic loops. These are jargon-y terms for pretty simple but important ideas. A 'Deterministic' loop just means that the programmer knows how many times the loop will need to repeat, or iterate. A deterministic loop can be expressed in pseudo-code as: • *FOR n iterations DO some action* In Java the deterministic loop type is called a 'for' loop. The for loop will repeat, or iterate, a set number of times. Unlike the deterministic 'for' loop, a non-deterministic loop will keep repeating indefinitely! Clearly we do not want to create a loop that repeats forever, as our program would never be able to finish. To define an stop point for the repeating code, we would use a Boolean condition – the same as those we met in the Boolean Logic tutorial. • WHILE some condition is true DO some action
Relationship : Definition of (Iteration)	
Complexity	0.5

Definition : Definition 2	
Text	There are two main types of looping. There are loops which will continue while-ever some Boolean statement is true, and those which will loop (or iterate) a defined number of times. A While or a Do-While loop will continue to iterate while-ever its condition is true. This can be referred to as non-deterministic, because the number of iterations is not specified within the loop condition. A For loop will iterate a determinable number of times. This is a deterministic loop, because the number of iterations are specified within the loop condition.
Relationship : Definition of (Iteration)	
Complexity	0.0

Example : Example 1	
Text	There are two main types of looping. There are loops which will continue while-ever some Boolean statement is true, and those which will loop (or iterate) a defined number of times.

Uri (content)	Example FOR
	for (*int i = 0*; *i <= 10*; *i++*) {
	do something until i is greater than 10
	}
	
	Example WHILE
	while (*i <= 10*) {
	do something while-ever i less than 11
	}
	
	Example DO-WHILE
	do {
	do something while-ever i less than 11
	} while (*i <= 10*);

Relationship : Demonstration of (Iteration)	
Complexity	0.0

Question : Question 1	
Text	You are designing an algorithm to print out the status (‘pass’ or ‘fail’) of each student in a list of students, getting each student from the list by their array index. In the code sample shown, the loop constructor is missing. Which type of loop would be best to use?
Answer Words	For, deterministic
Required Matches	1
Max Attempts	2
Feedback	Because the start and end of the iteration are known, a *Deterministic* loop, such as a For loop, is the most suitable type of loop.
Relationship : Assessment for (Iteration)	
Order	1

Code Sample : Counting Cars Sample	

Uri (content)	**public static void** main(String[] args)
	{
	ArrayList<Student> students = *GetStudents*();
	
	/* loop type */ (/***loop condition***/)
	{
	Student student = students.get(i);
	System.***out***.println(student.Status);
	}
	}
	
	private static ArrayList<Student> GetStudents()
	{
	ArrayList<Student> students = **new** ArrayList<Student>();
	
	students.add(**new** Student("pass"));
	students.add(**new** Student("pass"));
	students.add(**new** Student("pass"));
	students.add(**new** Student("fail"));
	students.add(**new** Student("fail"));
	students.add(**new** Student("fail"));
	
	return students;
	}
	
	public static class Student {
	public String Status;
	
	public Student(String status)
	{
	this.Status = status;
	}
	}

Relationship : Code for (Question 1)	

Guidance : Guidance 1	
Information	You will need to refer to the definition of Iteration, to answer this question. If you have closed the Iteration definition, just ask me to 'define iteration'. The loop start and end points are known, are numeric, and can be determined within the loop condition. This is known as a deterministic loop.
Text	What type of loop is suitable for a deterministic condition, where the start and end of a numeric range are known in advance?
Answer Words	for
Required Matches	1
Relationship : Guidance for (Question 1)	
Order	1

Question : Question 2	
Text	You are designing an algorithm which will simply print out the last string entered by a user, until they enter 'quit'. What type of loop would you use?
Answer Words	while, do while
Required Matches	1
Max Attempts	2

Feedback	Because you do not know how many iterations will occur, this is a *non-deterministic* type of loop, such as a While or Do-while.
Relationship : Assessment for (Iteration)	
Order	2

Guidance : Guidance 1	
Information	In this challenge, the number of iterations is unknown. You cannot define, in advance, how many times the user will enter a new string which is not equal to \'quit\'. This is known as a non-deterministic loop, because the number of iterations cannot be specified in the loop condition.
Text	What type of loop is suitable for a non-deterministic condition, where the number of iterations is unknown, or you simply want to repeat the code until a condition is met?
Answer Words	while, do-while
Required Matches	1
Relationship : Guidance for (Question 2)	
Order	1

The For Loop

Concept : For Loop	
Name	For Loop
Introduction	The For loop is the deterministic loop available in Java. As you know, it is used when it is known, on entry to the loop code, how many times to repeat the contained block of code.
Objectives	In this tutorial you will learn how to create For loop conditions, calculate iterations and use branching commands to manipulate the loop.
Relationship : Part of (Iteration)	
Complexity	0.5

Definition : Definition 1	
Text	The ‘for loop’ allows a programmer to create a repeating block of code which will repeat a set number of times. In pseudo-code a ‘for loop’ is represented as ‘FOR so many times DO these statements’. The condition of a For loop is a combination of three factors. The initialiser, the termination and the modifier. This allows a programmer to define the start point for the count, the end point for the count and the count modifier. *FOR initial state; given a condition; modify state DO these statements* The initial state is set once the loop starts. This is the starting state of our loop conditional variable and is commonly an integer. *FOR initial value of number is 0; given number is less than 10; increment number on each iteration* The condition is used to test whether the loop should be repeated again at the end of each execution of the block of code. This is a Boolean statement

	expressing some mathematical predicate on the variable, such as less than (<), greater than (>) or equal to (==). The updating of the value of the loop counter is performed as if it were the last line of the loop's block of code. This is commonly an increment (++), decrement (--) or other mathematical operation (such as /, * or %). Because the loop variable is modified by the *modifier*, it is considered bad practice to modify the loop variable inside the for loop code block.
Relationship : Definition of (For Loop)	
Complexity	0.0

Example : Example 1	
Text	There are two main types of looping. There are loops which will continue while-ever some Boolean statement is true, and those which will loop (or iterate) a defined number of times. The For loop will repeat an exact number of times.

Uri (content)	Example FOR
	
	In this example we have created a for loop which will repeat 11 times.
	
	The initial condition is i = 0. The condition is i <= 10, so while-ever i is less than or equal to 10.
	
	The modifier is i++. This means that for each iteration, add one to i. So, on the first iteration i is 0. On the second, i is 1, and so on.
	
	for (int i = 0; i <= 10; i++) {
	  System.out.println(i);
	}

Relationship : Demonstration of (For Loop)	
Complexity	0.0

Question : Question 1	
Text	Consider the code sample shown. How would you modify this For loop to count down to 0, rather than up to 10?
Answer Words	i=10, i--, decrement, subtract, take
Required Matches	2
Max Attempts	2
Feedback	The modification to count down from 10 to 0 involves decrementing, rather than incrementing the initial condition.
Relationship : Assessment for (For Loop)	
Order	1

Code Sample : Counting Down Sample	

Uri (content)	**public static void** main(String[] args)
	{
	  for (**int** i = 0; i <= 10; i++)
	  {
	  System.***out***.println(i);

	} }
Relationship : Code for (Question 1)	

Guidance : Guidance 1	
Information	The loop is initialised with an integer of value 0. The loop continues until the integer is equal to 10.
Text	In the sample code, what modification is made to the condition variable ‘i’ on each iteration?
Answer Words	increment one, add one, increment 1, add 1
Required Matches	2
Relationship : Guidance for (Question 1)	
Order	1

Guidance : Guidance 2	
Information	The modifier for the loop adds one to the condition variable with each iteration, counting up from 0 to 10.
Text	Given we changed the initial condition variable to i = 10, which modifier would allow us to count down from 10?
Answer Words	decrement, subtract one, -- operator
Required Matches	1
Relationship : Guidance for (Question 1)	
Order	2

Question : Question 2	
Text	A non-declarative loop\'s iteration count is?
Answer Words	Unknown, undefined
Required Matches	2
Max Attempts	2
Feedback	As you should know by now, a declarative loop is one where the start and end of iteration are known in advance. A non-declarative loop is one where the number of iterations is unknown.
Relationship : Assessment for (For Loop)	
Order	2

Question : Question 3	
Text	The start and end points for the loop should be well defined within the constructor of the loop. The modifying action being the last part of the constructor. Should you modify the initial condition variable inside the body of the loop?
Answer Words	no
Required Matches	1
Max Attempts	2
Feedback	It is bad practice to modify the condition variable of a For loop inside the loop. Many programming languages will not allow you to do so, as modification should only occur within the loop condition.
Relationship : Assessment for (For Loop)	
Order	3

Question : Question 4

Text	Take a look at the code sample and tell me which type of loop is shown?
Answer Words	enchanced for, for each
Required Matches	2
Max Attempts	2
Feedback	This is the enhanced For loop. This is another way of creating a declarative loop, but rather than iterating over a numeric range, it iterates over a set – or array.
Relationship : Assessment for (For Loop)	
Order	4

Code Sample : Enhanced For Sample	

Uri (content)	**public static void** main(String[] args)
	{
	String[] arrayOfNumbers = **new** String[] { "one", "two", "three" };
	for (String number : arrayOfNumbers)
	{
	System.***out***.println(number);
	}
	}

Relationship : Code for (Question 4)	

Guidance : Guidance 1	
Information	The loop shown is actually a For loop. However, unlike a normal For loop the condition is inferred from a variable. This can be viewed as a for each loop, where the string 'number' is an element of the array 'arrayOfNumbers'.
Text	How many times will the loop iterate?
Answer Words	3, three
Required Matches	1
Relationship : Guidance for (Question 4)	
Order	1

Question : Question 5	
Text	Think back to the discussion on iteration. We know that the initial condition variable is created when the loop is entered. I've put up a new code sample. Take a look. Will this code compile?
Answer Words	no nope wont error
Required Matches	1
Max Attempts	2
Feedback	The variable i only exists inside, or in the *scope*, of the For loop. The attempt to print the variable outside the loop will cause the code to fail to compile.
Relationship : Assessment for (For Loop)	
Order	5

Code Sample : Scope Sample	

Uri (content)	**public static void** main(String[] args)
	{
	for (**int** i = 0; i <= 10; i++)
	{
	System.***out***.println(i);
	}
	System.***out***.println(i);

	}
Relationship : Code for (Question 5)	

Question : Question 6	
Text	Take a look at the code sample. How many times will this loop iterate?
Answer Words	4, four
Required Matches	1
Max Attempts	2
Feedback	The initial condition for this loop is i = 1. The modifier is i *= 6, which means times the current value of i by six with each iteration.
Relationship : Assessment for (For Loop)	
Order	5

Code Sample : Complex Modifier Sample	

Uri (content)	**public static void** main(String[] args)
	{
	  for (**int** i = 1; i <= 216; i*=6)
	  {
	  System.***out***.println(i);
	  }
	}

Relationship : Code for (Question 6)	

Appendix B

Comprehension classification pilot study results

Table B.1 Exploration of parameters for training data selection

					MSE			CA		
Dur	Win	Int	Del	Nodes	Tr	Va	Te	Tr	Va	Te
30	5	2.66	0	10	0.96	0.98	1.01	56.39	55.57	52.43
30	4	2	0	10	0.93	0.97	0.99	59.34	56.21	56.07
30	3	1.66	0	10	0.93	0.97	0.98	59.67	56.78	55.99
30	2	1	0	10	0.92	0.96	0.97	59.67	56.80	55.57
30	1	0.66	0	10	0.94	0.96	0.97	59.51	58.16	57.16
30	5	2.66	0	15	0.92	0.97	1.00	59.71	56.34	55.30
30	4	2	0	15	0.94	0.97	0.99	57.79	56.12	54.62
30	3	1.66	0	15	0.93	0.97	0.99	59.29	57.62	55.96
30	2	1	0	15	0.93	0.96	0.98	59.36	57.90	56.08
30	1	0.66	0	15	0.93	0.96	0.96	60.06	58.26	57.90
30	5	2.66	0	20	0.92	0.98	1.01	60.23	56.25	55.70
30	4	2	0	20	0.92	0.98	1.00	59.32	55.31	53.65
30	3	1.66	0	20	0.93	0.97	0.99	59.84	57.83	56.04

Continued on next page

Table B.1 – continued from previous page

					MSE			CA		
Dur	Win	Int	Del	Nodes	Tr	Va	Te	Tr	Va	Te
30	2	1	0	20	0.91	0.96	0.98	61.11	58.39	57.22
30	1	0.66	0	20	0.93	0.96	0.97	60.34	58.21	58.13
30	5	2.66	0	25	0.93	0.99	1.01	58.90	55.56	54.14
30	4	2	0	25	0.93	0.97	0.99	59.50	56.65	55.59
30	3	1.66	0	25	0.92	0.97	0.98	60.46	57.51	56.68
30	2	1	0	25	0.92	0.97	0.98	60.30	57.11	56.82
30	1	0.66	0	25	0.92	0.97	0.98	60.87	58.16	57.75
30	5	2.66	0	10	0.96	0.98	1.01	56.39	55.57	52.43
30	4	2	0	10	0.93	0.97	0.99	59.34	56.21	56.07
30	3	1.66	0	10	0.93	0.97	0.98	59.67	56.78	55.99
30	2	1	0	10	0.92	0.96	0.97	59.67	56.80	55.57
30	1	0.66	0	10	0.94	0.96	0.97	59.51	58.16	57.16
30	5	2.66	0	15	0.92	0.97	1.00	59.71	56.34	55.30
30	4	2	0	15	0.94	0.97	0.99	57.79	56.12	54.62
30	3	1.66	0	15	0.93	0.97	0.99	59.29	57.62	55.96
30	2	1	0	15	0.93	0.96	0.98	59.36	57.90	56.08
30	1	0.66	0	15	0.93	0.96	0.96	60.06	58.26	57.90
30	5	2.66	0	20	0.92	0.98	1.01	60.23	56.25	55.70
30	4	2	0	20	0.92	0.98	1.00	59.32	55.31	53.65
30	3	1.66	0	20	0.93	0.97	0.99	59.84	57.83	56.04
30	2	1	0	20	0.91	0.96	0.98	61.11	58.39	57.22
30	1	0.66	0	20	0.93	0.96	0.97	60.34	58.21	58.13
30	5	2.66	0	25	0.93	0.99	1.01	58.90	55.56	54.14
30	4	2	0	25	0.93	0.97	0.99	59.50	56.65	55.59
30	3	1.66	0	25	0.92	0.97	0.98	60.46	57.51	56.68

Continued on next page

Table B.1 – continued from previous page

					MSE			CA		
Dur	Win	Int	Del	Nodes	Tr	Va	Te	Tr	Va	Te
30	2	1	0	25	0.92	0.97	0.98	60.30	57.11	56.82
30	1	0.66	0	25	0.92	0.97	0.98	60.87	58.16	57.75
10	5	2.66	0	10	0.99	0.99	1.05	54.65	52.85	50.67
10	4	2	0	10	0.94	0.98	1.02	56.49	53.36	51.52
10	3	1.66	0	10	0.92	0.97	1.01	59.51	56.56	55.14
10	2	1	0	10	0.94	0.97	1.00	57.53	55.68	53.69
10	1	0.66	0	10	0.94	0.97	0.99	56.89	54.52	53.67
10	5	2.66	0	15	0.94	1.00	1.01	56.95	52.31	52.38
10	4	2	0	15	0.91	0.99	1.02	59.01	53.93	51.72
10	3	1.66	0	15	0.94	0.97	1.02	58.28	55.18	53.39
10	2	1	0	15	0.94	0.97	1.00	56.88	55.15	52.83
10	1	0.66	0	15	0.92	0.97	0.98	59.62	55.64	54.86
10	5	2.66	0	20	0.94	1.00	1.02	58.50	53.24	51.68
10	4	2	0	20	0.94	0.99	1.03	57.94	52.24	51.07
10	3	1.66	0	20	0.90	0.97	1.01	60.55	55.82	54.64
10	2	1	0	20	0.92	0.98	0.99	59.07	55.11	54.57
10	1	0.66	0	20	0.94	0.98	1.00	57.48	55.03	53.97
10	5	2.66	0	25	0.89	1.01	1.06	61.33	53.23	50.20
10	4	2	0	25	0.89	0.99	1.01	60.81	54.07	53.26
10	3	1.66	0	25	0.90	0.99	1.02	60.19	54.98	52.38
10	2	1	0	25	0.91	0.97	1.01	59.81	54.70	53.44
10	1	0.66	0	25	0.93	0.97	0.99	57.88	56.25	55.19
10	5	2.66	5	10	0.95	1.00	0.99	56.73	52.42	52.70
10	4	2	5	10	0.91	0.97	1.00	61.67	56.07	54.06
10	3	1.66	5	10	0.93	0.97	0.99	59.43	55.90	54.68

Continued on next page

Table B.1 – continued from previous page

					MSE			CA		
Dur	Win	Int	Del	Nodes	Tr	Va	Te	Tr	Va	Te
10	2	1	5	10	0.90	0.96	0.98	61.94	58.63	56.61
10	1	0.66	5	10	0.91	0.94	0.96	60.89	58.90	56.90
10	5	2.66	5	15	0.91	0.99	1.01	58.96	54.23	53.40
10	4	2	5	15	0.92	0.98	1.00	59.02	55.37	54.23
10	3	1.66	5	15	0.90	0.97	1.01	60.71	57.68	54.27
10	2	1	5	15	0.89	0.95	0.98	60.58	58.27	56.02
10	1	0.66	5	15	0.91	0.95	0.96	61.85	58.92	58.44
10	5	2.66	5	20	0.90	0.99	1.03	60.60	53.57	53.64
10	4	2	5	20	0.91	0.98	1.00	59.53	55.16	53.03
10	3	1.66	5	20	0.89	0.96	1.03	62.18	58.95	54.39
10	2	1	5	20	0.90	0.95	1.00	60.53	57.78	55.40
10	1	0.66	5	20	0.85	0.94	0.98	64.56	60.07	59.14
10	5	2.66	5	25	0.85	0.97	1.04	64.30	56.27	52.04
10	4	2	5	25	0.87	0.99	1.00	61.84	54.33	53.98
10	3	1.66	5	25	0.91	0.97	1.02	60.04	56.80	54.19
10	2	1	5	25	0.85	0.95	0.99	63.86	58.88	56.50
10	1	0.66	5	25	0.85	0.95	0.97	65.85	59.93	58.80
10	5	2.66	10	10	0.93	0.99	1.05	60.23	55.37	53.00
10	4	2	10	10	0.86	0.94	0.99	64.60	59.57	57.37
10	3	1.66	10	10	0.88	0.94	1.00	63.16	59.37	56.11
10	2	1	10	10	0.90	0.95	0.97	61.99	57.59	58.04
10	1	0.66	10	10	0.89	0.95	0.97	63.15	59.89	58.53
10	5	2.66	10	15	0.90	0.99	1.01	61.85	55.25	55.51
10	4	2	10	15	0.84	0.95	0.98	66.10	59.86	60.98
10	3	1.66	10	15	0.88	0.98	0.98	63.38	59.13	57.29

Continued on next page

Table B.1 – continued from previous page

					MSE			CA		
Dur	Win	Int	Del	Nodes	Tr	Va	Te	Tr	Va	Te
10	2	1	10	15	0.90	0.95	0.96	61.84	58.22	58.45
10	1	0.66	10	15	0.90	0.95	0.97	62.17	59.10	58.85
10	5	2.66	10	20	0.82	0.96	1.03	67.16	59.43	56.94
10	4	2	10	20	0.77	0.93	1.02	70.20	62.79	59.85
10	3	1.66	10	20	0.82	0.95	0.98	67.10	60.82	59.69
10	2	1	10	20	0.87	0.95	0.99	64.31	59.99	58.94
10	1	0.66	10	20	0.88	0.95	0.97	63.17	59.58	58.11
10	5	2.66	10	25	0.84	0.97	1.03	65.09	57.32	53.61
10	4	2	10	25	0.84	0.96	1.04	65.19	58.27	54.36
10	3	1.66	10	25	0.83	0.95	0.97	67.26	60.65	60.69
10	2	1	10	25	0.84	0.96	1.00	66.00	60.05	57.18
10	1	0.66	10	25	0.87	0.95	0.97	64.01	59.64	58.83
10	5	2.66	15	10	0.92	0.99	1.04	60.07	57.20	57.68
10	4	2	15	10	0.90	0.98	1.04	63.33	57.08	56.12
10	3	1.66	15	10	0.89	0.97	1.03	62.35	57.26	56.75
10	2	1	15	10	0.89	0.95	0.98	63.73	60.10	59.01
10	1	0.66	15	10	0.90	0.96	0.99	63.38	59.95	56.98
10	5	2.66	15	15	0.86	0.98	1.09	64.48	58.89	55.69
10	4	2	15	15	0.90	0.97	1.00	62.49	57.47	53.92
10	3	1.66	15	15	0.91	0.95	1.01	62.49	58.88	57.16
10	2	1	15	15	0.88	0.93	0.98	64.74	60.20	59.07
10	1	0.66	15	15	0.90	0.97	1.00	63.11	59.08	57.18
10	5	2.66	15	20	0.87	0.98	1.01	65.43	56.93	55.73
10	4	2	15	20	0.89	0.96	1.03	63.62	58.00	54.36
10	3	1.66	15	20	0.80	0.99	1.06	68.36	59.12	56.35

Continued on next page

Table B.1 – continued from previous page

					MSE			CA		
Dur	Win	Int	Del	Nodes	Tr	Va	Te	Tr	Va	Te
10	2	1	15	20	0.81	0.94	1.02	68.65	61.88	58.28
10	1	0.66	15	20	0.90	0.96	0.99	62.83	58.46	57.41
10	5	2.66	15	25	0.87	0.97	1.05	65.19	58.65	55.54
10	4	2	15	25	0.83	0.95	1.04	66.88	59.81	57.16
10	3	1.66	15	25	0.85	0.97	1.06	66.31	57.82	55.80
10	2	1	15	25	0.80	0.94	1.00	68.81	61.86	59.10
10	1	0.66	15	25	0.90	0.98	0.98	62.22	57.48	57.41
10	5	2.66	20	10	0.93	1.03	1.09	58.99	52.70	52.67
10	4	2	20	10	0.86	0.97	1.06	64.65	58.36	53.12
10	3	1.66	20	10	0.93	0.99	1.03	59.93	54.72	54.02
10	2	1	20	10	0.88	0.98	1.03	63.81	59.51	56.63
10	1	0.66	20	10	0.90	0.97	1.01	62.25	59.04	54.62
10	5	2.66	20	15	0.93	1.05	1.05	60.28	51.48	51.23
10	4	2	20	15	0.83	0.97	1.05	66.88	57.78	53.40
10	3	1.66	20	15	0.81	0.97	1.05	68.08	58.65	58.10
10	2	1	20	15	0.85	0.98	0.99	65.82	58.83	58.03
10	1	0.66	20	15	0.86	0.98	1.01	64.85	58.90	56.47
10	5	2.66	20	20	0.87	1.03	1.04	64.15	54.79	56.39
10	4	2	20	20	0.82	0.98	1.03	66.75	59.42	57.38
10	3	1.66	20	20	0.61	0.95	1.13	76.55	63.12	56.41
10	2	1	20	20	0.86	0.98	0.99	65.67	58.59	57.37
10	1	0.66	20	20	0.84	0.96	1.00	66.93	59.88	58.68
10	5	2.66	20	25	0.85	1.03	1.12	66.28	57.18	50.75
10	4	2	20	25	0.81	1.00	1.08	68.46	56.96	54.09
10	3	1.66	20	25	0.76	1.01	1.12	71.39	57.22	53.34

Continued on next page

Table B.1 – continued from previous page

					MSE			CA		
Dur	Win	Int	Del	Nodes	Tr	Va	Te	Tr	Va	Te
10	2	1	20	25	0.79	0.96	1.04	68.93	59.64	58.08
10	1	0.66	20	25	0.86	0.97	1.01	65.65	59.28	58.12