Cognitive Training Assistants for Cost Estimators (Incubate Phase I, WRT-1049.8.1)

White Paper

August 15, 2022

PI: Daniel Selva (Texas A&M University)

Co-Is: Theodora Chaspari (TAMU) and Alejandro Salado (University of Arizona)

Period of performance: Sep-1-2021 to Jun-30-2022

Budget: $100k

Table of Contents

1. Overview

This paper describes the results of our AIRC Incubator project titled "Cognitive Training Assistant for Cost Estimators".

The goal of this Incubator project was to demonstrate **proof of concept** for a cognitive assistant to support training of new cost estimators in the Department of Defense (DOD). A Cognitive Assistant (CA) is defined here as an Artificial Intelligence (AI) tool, usually with a natural language interface, that augments human intellect in a specific task by retrieving and processing relevant information from multiple information sources and providing it to the user at the right time. It also has the capability to learn and adapt to the user and problem at hand.

Cost estimation is a complex iterative process consisting of various steps: gathering the required information, selecting an overall strategy and one or more existing models, developing new models if needed (including calibration and validation), performing the estimate, and conducting sensitivity analyses as appropriate. There are challenges for beginner cost estimators in each of those steps, including dealing with incomplete datasets, appropriately assessing the performance of new models, projecting beyond historical ranges of validity, adequately reporting the level of uncertainty around a point estimate, understanding how to use joint cost-schedule distributions, etc.

Currently, the training of new cost estimators is done primarily through traditional instruction in live classrooms, and thus it is a time-consuming process. Traditional instruction typically implies reduced opportunities for hands-on learning opportunities, which are known to improve learning. This type of instruction is also not tailored to each individual, so the pace can be too fast for some trainees and too slow for others. The use of CAs can allow for more interactive and tailored instruction for each individual and area, as demonstrated with intelligent tutoring systems in other areas of education (Corbett et al., 1997).

The idea of using AI tools to enhance the learning of trainees is not new and has been studied for decades (Ong & Ramachandran, 2003). However, in the DOD Acquisition context, we are still in the early stages of incorporating advanced AI tools into workflows and, in particular, CAs have not been adopted yet as training tools. Previous attempts to adopt this technology in the workplace failed because of a combination of insufficient performance of the underlying machine learning models and lack of familiarity of the users with this mode of interaction. With CAs now being ubiquitous in our daily lives, and the significant recent advances we have seen in machine learning, the time is now ripe for infusion of this technology in the workplace.

The remainder of this white paper describes the work done in this project.

A review of the relevant literature in cognitive assistants and intelligent tutoring systems was performed. The results are summarized in Section 2.

The use case for this CA as defined initially at a high level was to provide trainees with interactive hands-on opportunities to learn the concepts, methods, and best practices related to estimating the lifecycle cost of a complex system, namely a space mission. Throughout the project, we worked with stakeholders in the DOD to refine this use case. The resulting use case is defined in more detail in Section 3.

The approach to develop the prototype with the limited time and resources was to leverage the existing Daphne CA developed by the PI's. This proved successful and we were able to develop a functioning prototype adapted to the use case. The architecture and design of the prototype are discussed in Section 4.

One of the main aspects of the CA is that it should provide individualized training and adapt to the user's individual needs. In this Incubator project, this was demonstrated in the context of selecting questions for the various learning assessments and learning opportunities that best address the user's needs (e.g., reinforcing weaker areas). However, we explored other ways of adapting to the user's cognitive state as well (e.g., mental workload, level of engagement). Section 5 describes the preliminary efforts that were done in this area, particularly with respect to using biosensors such as eye trackers to estimate those parameters.

The validation of the assistant is briefly discussed in Section 6. Roughly biweekly meetings with the stakeholders were done to obtain feedback about the prototype and guide development. The demonstrations that occurred during these meetings also served as informal validation of our efforts. The primary goal of the project was successfully achieved.

A primary concern of the stakeholders was to estimate what it would take to develop and maintain such a system within their organization. Section 7 provides an initial rough estimate of the cost and resources that would be needed, based on several rough assumptions.

Finally, Section 8 of this document contains the main conclusions of the work, limitations, and opportunities for future work.

2. Literature review

There is relevant literature in the use of cognitive assistants and other intelligent agents for educational purposes. Much of this literature is contained under the umbrella term of intelligent tutoring systems (ITS). ITS are intelligent systems that help students master a subject by providing them with learning opportunities that are tailored to their specific needs.

Following the success of expert systems and other kinds of intelligent decision support systems in the 1980s, ITS were proposed as a method that could radically improve student outcomes in education by providing unprecedented ability to adapt to individual differences (Corbett et al., 1997).

Key to this adaptation was the ability of these systems to estimate the skill level of student for a number of areas based on the student performance in some learning opportunities provided by the system, using Bayesian algorithms among others (Mayo, 2001). These skill levels could then be used to select the next learning opportunity to provide to the student given some goal, such as to reinforce the weaker areas. Theoretical frameworks and algorithms were developed and successfully deployed based on Partially Observable Markov Decision Processes (Folsom-Kovarik et al., 2013) among others.

Educators were especially excited about the potential of this technology to democratize education and improve student outcomes for populations that needed it the most (Nye, 2015). They were deployed in various educational centers with some success (Koedinger et al., 1997). Specifically, it was observed that using ITS, student learning outcomes and student engagement could be improved (Kim et al., 2020).

The basic rigid systems developed in the 1990s evolved into more advanced systems including mixed-initiative interfaces with question answering systems (A.C. Graesser et al., 2005) and affective computing technologies (D'Mello et al., 2005).

While the initial emphasis of ITS was on K-12 education, the technology has also been applied to adult education (Cheung et al., 2003) and training in the workplace (Ong & Ramachandran, 2003). In the latter case, it was found that using ITS could improve training performance and return on investment.

While the potential of these technologies is important, some barriers have also been identified for their implementation and widespread adoption. These include limitations in their performance (Sarrafzadeh et al., 2008) and high development and maintenance costs among others (Nye, 2014).

3. Use cases for the Cognitive Assistant

Three main use cases were defined for the CA: 1) tutor; 2) assistant; 3) tradespace exploration.

Tutor use case: The tutor use case refers to the traditional use of the CA as an ITS. The user uses a web-based tool to navigate a number of learning modules. After each learning module, the user completes a short quiz to assess their progress. The user can also do longer tests that serve the dual purpose of a learning opportunity for the user and helping the CA estimate the user's skill level across various areas. During these tests, the CA selects the next question to show to the user based on those skill levels, to maximize some objective such as reinforcing weaker areas. At any point during the learning modules (or even during the tests if enabled), the user can ask questions to Daphne, such as "*What is Delta V*"? Daphne answers this question by providing a succinct answer as well as a link to the relevant slides from the learning materials where the concept is discussed.

Assistant use case: The assistant use case is inspired by a learning-by-doing instructional approach and "in-game tutorials". The agent helps the user while performing an actual cost estimation task, responding to user questions like in the tutor use case, but also providing suggestions and notifications that include pointing out mistakes, providing feedback, or suggesting relevant training materials on a topic, for example.

Tradespace Exploration use case: The tradespace exploration use case is inspired by the existing Daphne CA, which was developed to support the tradespace exploration process for Earth observation satellite systems. Daphne already has some capabilities related to cost estimation, and it particular it can answer questions about why the cost of an architecture is what it is. It was thought that comparing the cost of a wide range of architectures very quickly could be a useful way of helping cost estimators new to the space commodity learn about space cost drivers.

While functionality for each of the three use cases was developed, it was decided in agreement with the stakeholders to develop the Tutor use case in more detail than the others. The rest of the report focuses on the Tutor use case.

4. Design of the Cognitive Assistant

A fully functioning prototype of the cognitive assistant has been developed. The overall architecture is represented in Figure 1.

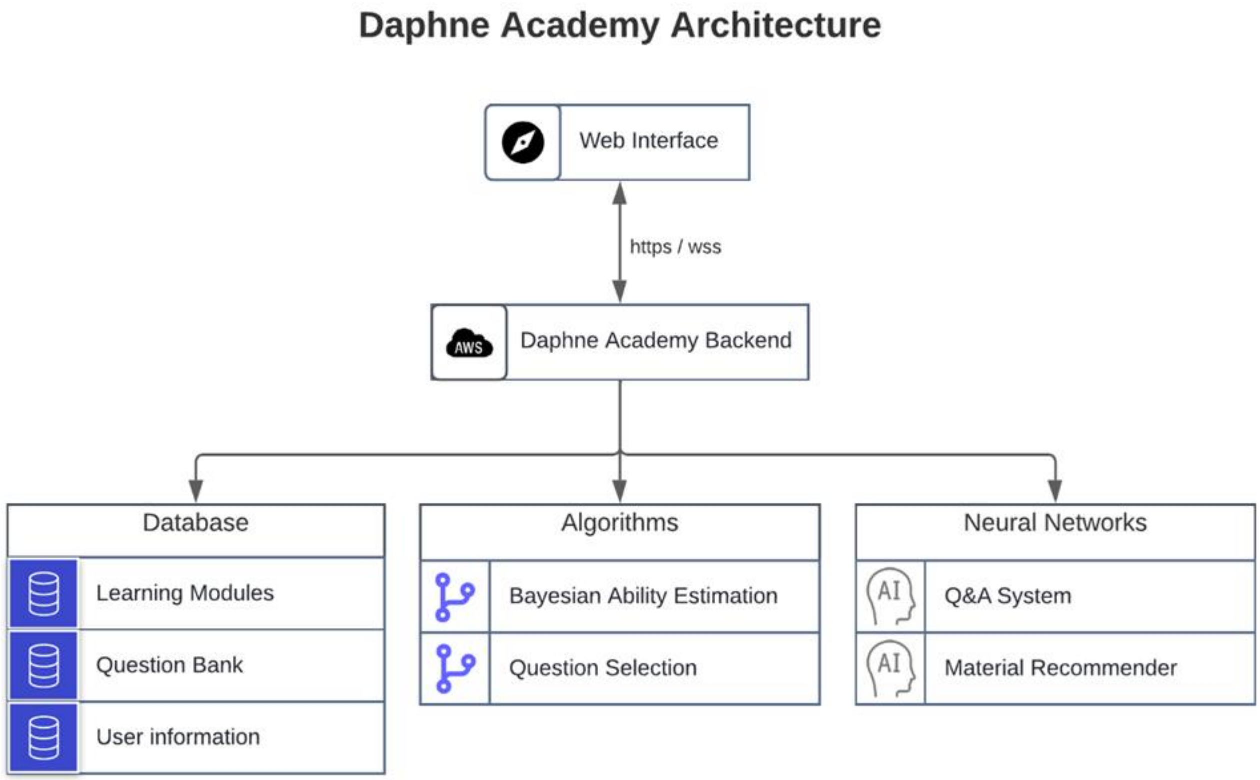

Figure 1: Overall architecture of the Daphne Academy tutoring agent

The system consists of a web-based front end that communicates with a back end server through http requests and Web Sockets. The back-end consists of databases, user adaptation algorithms, and neural network models.

There are three databases: 1) the Learning Module database, which contains the instructional materials (i.e., Power Point slides and module questions for now) organized in a number of learning modules; 2) the Question Bank database, which contains both questions the agent uses for tests and different topics that questions and learning modules can inherit; and 3) the User Information database, which contains user specific information such as assigned learning modules, estimated user ability levels across different topics, user communication with the cognitive assistant, and user test history.

Daphne Academy uses two main algorithms to adapt to the user: 1) an algorithm to estimate the user's skill level based on their answers to various questions and some question parameters; and 2) an algorithm to select the next question in an exam based on the estimated skill levels.

Finally, the back end contains two deep learning models that the agent uses to answer questions. These two models are 1) the QA system, which is trained to respond to user questions such as "*What is focal length?*"; and 2) The Material Recommended, which is used to complement the answers to those questions with links to instructional materials that are relevant.

Some more details on the front and back end are provided in the following sections.

Front end

The front-end is web-based to facilitate portability. Therefore, it can run on a browser on any device. A few snapshots of the front end are shown below. Figure 2 shows an overview of the front-end. On the left, we see the main menu where the user can navigate through the different learning modules, view their current estimated mastery levels for the various areas, and select various learning opportunities. In the center, we see a slide of a current learning module on the basics of lifecycle cost estimation. The user can go back and forth in this presentation at their own pace. The right side of Figure 2 shows the chatbox. The user can ask questions there at any point during the interaction.

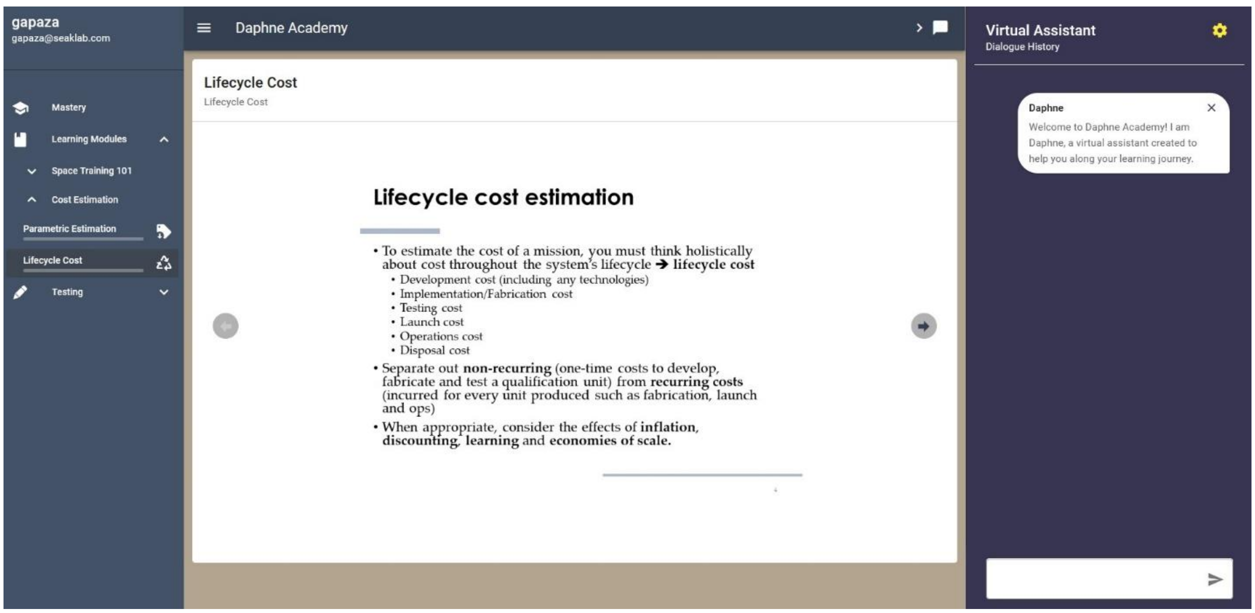

Figure 2: Overview of the Daphne Academy front end.

Figure 3 shows an example of the user asking a question "which spectrum do radars operate on" which is answered by Daphne with a succinct definition using the QA System. In addition, Daphne uses the Material Recommender model to identify the instructional materials that are relevant for this question and provide the user with a link to the corresponding slide set. For each recommended material, it also provides a relevancy score from 0% to 100%.

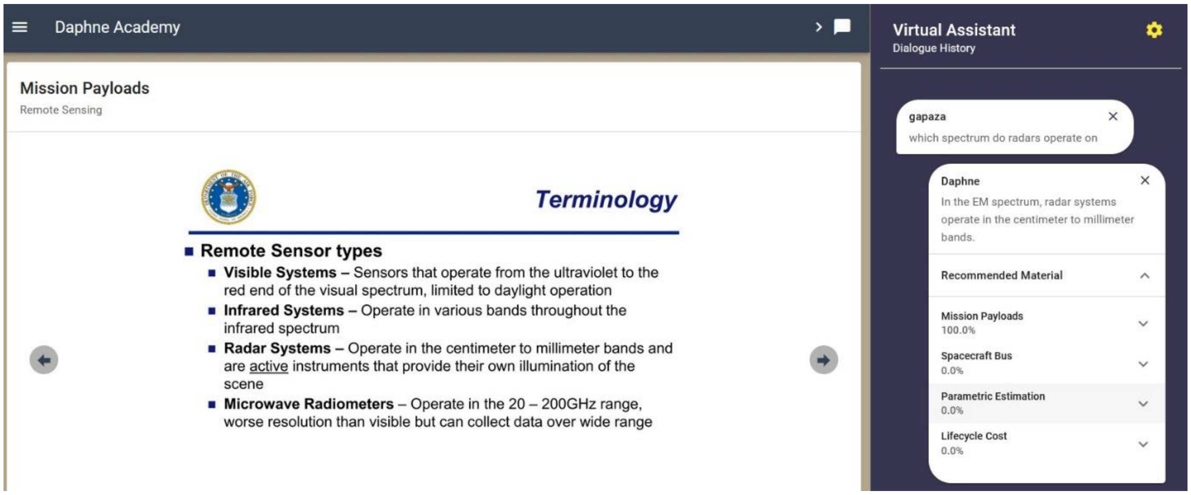

Figure 3: Daphne Academy responds to a user question and recommends relevant instructional materials.

If the user selects a particular recommended material, Daphne Academy can also identify the most relevant slides within the slide set and provide a similar recommendation and relevancy score for a specific slide (See Figure 4 shows). (This recommendation could be combined with the previous one in future versions of the agent.)

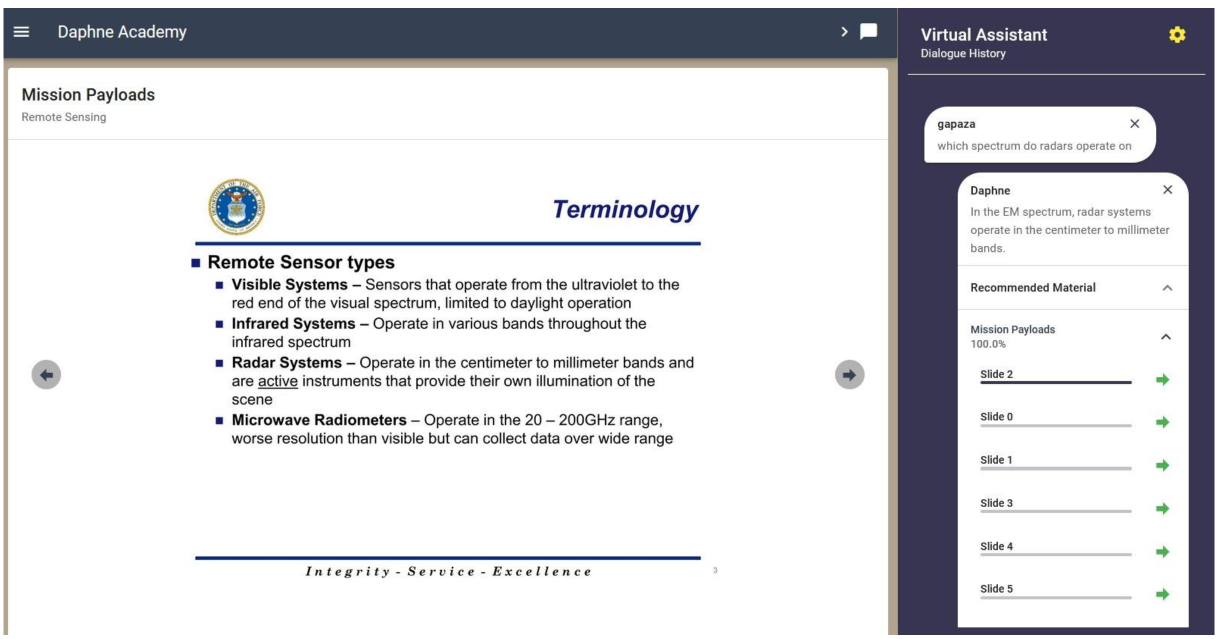

Figure 4: If the user selects a particular recommended material, Daphne Academy can also identify the most relevant slides within the slide set.

At the end of each learning module, the user can take a quiz to practice their newly acquired knowledge and skills, and assess their level of mastery of the content in the learning module. Figure 5 shows an example multiple-choice question during one of these quizzes. When the user submits the answer, it gets immediate feedback on not only whether they got the correct answer, but also why. These questions are selected from the Question Bank using the algorithm described in the Back end subsection. The user answers are used by the agent to estimate the Mastery level.

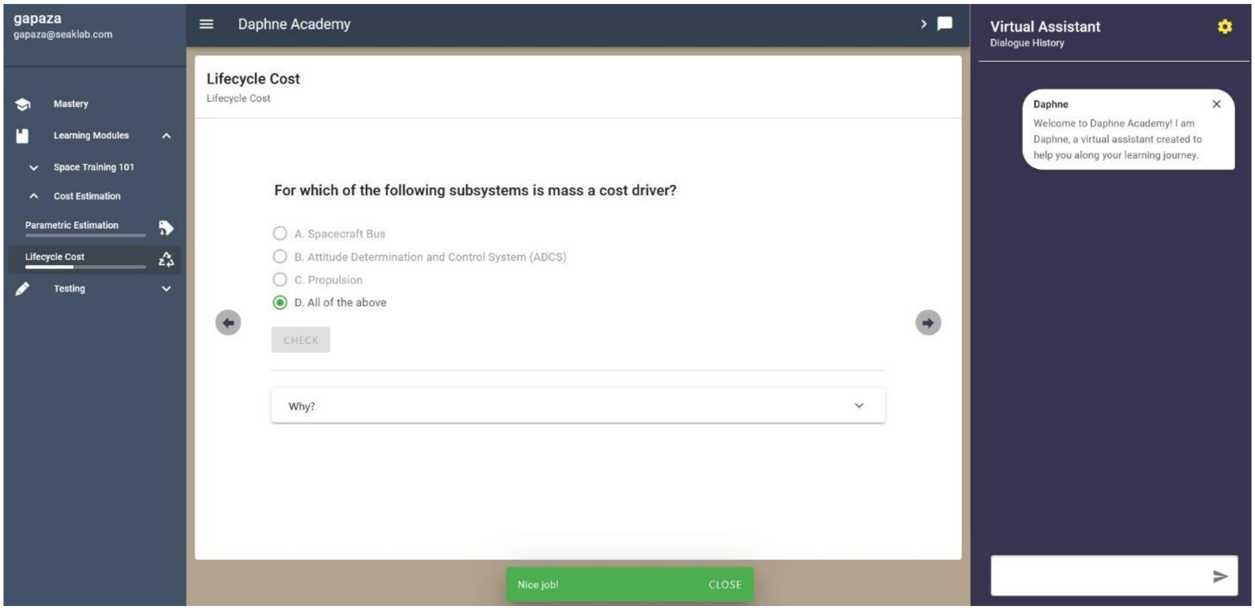

Figure 5: Example test question

In addition to the end of module quizzes, there are also on-demand exams that can be taken at any point to assess or certify student learning, or simply as learning opportunities (see Figure 6).

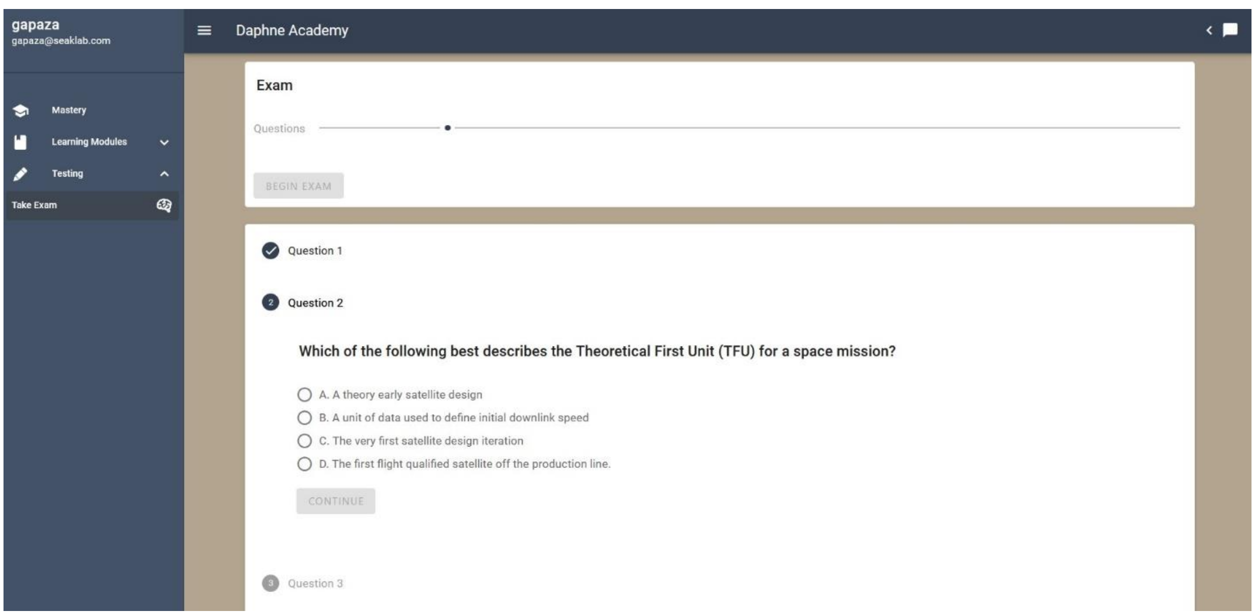

Figure 6: Example of exam with multiple questions.

An important feature of Daphne Academy is that these exams are tailored to an individual. Specifically, the questions a user gets in this exam depend on the mastery level of the user in the different relevant areas. This is determined and generated automatically by Daphne Academy, without requiring user interaction.

Initial estimates of mastery levels are available if the user has completed the corresponding end of module quizzes. Initial estimates of mastery level can also be set by default based on level of expertise. These initial estimates are then updated during the test, using the algorithm described in the back end section.

Back end

The main components of the back end are the two models (QA system and Material Recommender), the two adaptive algorithms (user skill estimation and question selection), and the two databases (Learning Module DB and Question Bank DB).

QA system: The model consists of two classifier convolutional neural networks (CNNs): one for classifying the high-level sentence intent (a.k.a. *role*), and one for classifying the specific question type. Daphne Academy can have multiple roles specializing in different types of questions (e.g., different areas of cost estimation or different kinds of task). This first prototype, however, only has a Tutor role that englobes all question types.

Concerning the neural network architecture (see Figure 7), each CNN has three independent convolutional layers preceded by an input layer and an embedding layer. For each convolutional layer, a ReLU activation function is applied, followed by a max pooling layer. Next, the feature maps produced by each max pooling layer are concatenated, flattened, and fed to a dropout layer with 0.5 dropout rate. Finally, a fully connected dense layer with a softmax activation function is followed as the output layer. The training dataset contains ~2000 items for each CNN.

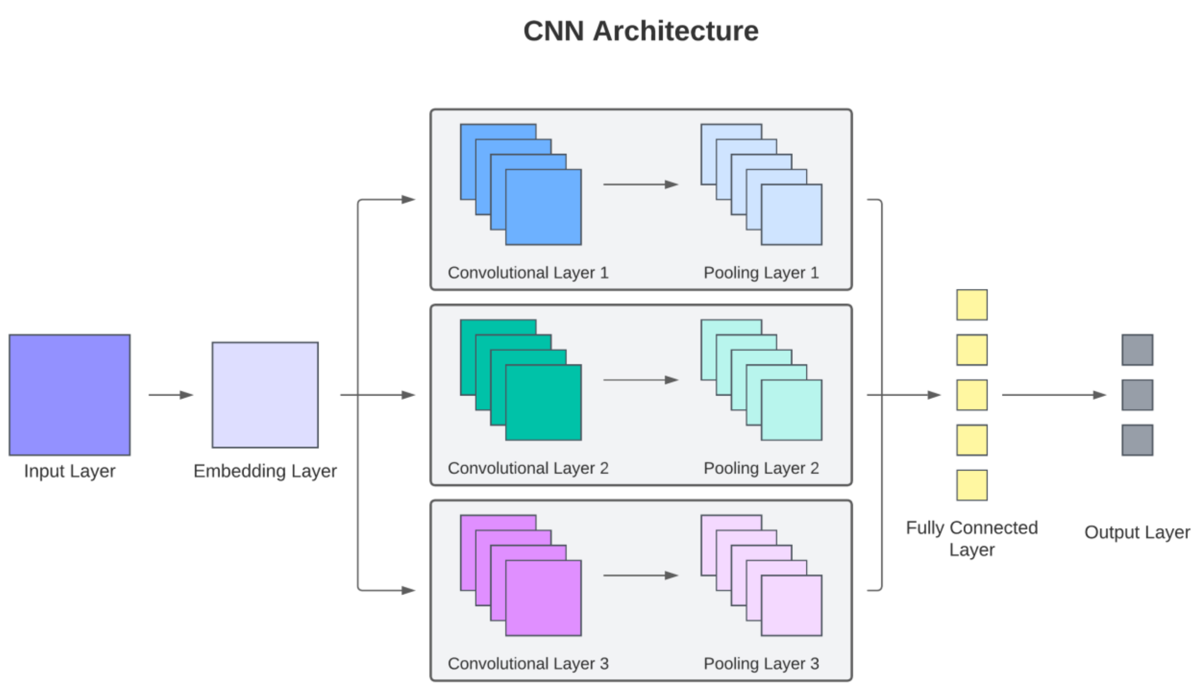

Figure 7: Daphne Academy CNN Architecture

Material Recommender: It uses the same CNN-based architecture as the QA system. (When applied to slide classification: one for classifying learning module, one for classifying slides in that module).

Learning Module Database: The learning Module database contains the instructional materials used in the learning modules. Theoretically this could include documents, presentations, videos, and other media. For now, it only contains learning module slides, where a slide can either be informational or question based. Specifically, the current database contains 4 learning modules on various aspects of cost estimation for space commodities. Two of these learning modules are adapted from a preexisting D.O.D. Space Training 101 course (provided by the stakeholders, see Figure 3), whereas the other two were produced by the PI team.

Question Bank Database: The Question Bank database contains questions that are used to create the learning module quizzes and the on-demand exams. It could potentially contain questions and problems of various kinds, as long as some feedback can be provided to the student. For now, however, it only contains multiple-choice and true false questions. Specifically, the current prototype contains 42 different questions.

User Information Database: The User Information database stores user specific information including estimated ability levels for each assigned topic, assigned learning modules, testing history, and any communication made with the cognitive assistant. This allows users to view their past performance, visualize their progress in different topic areas, and identify weak areas in the material assigned to them.

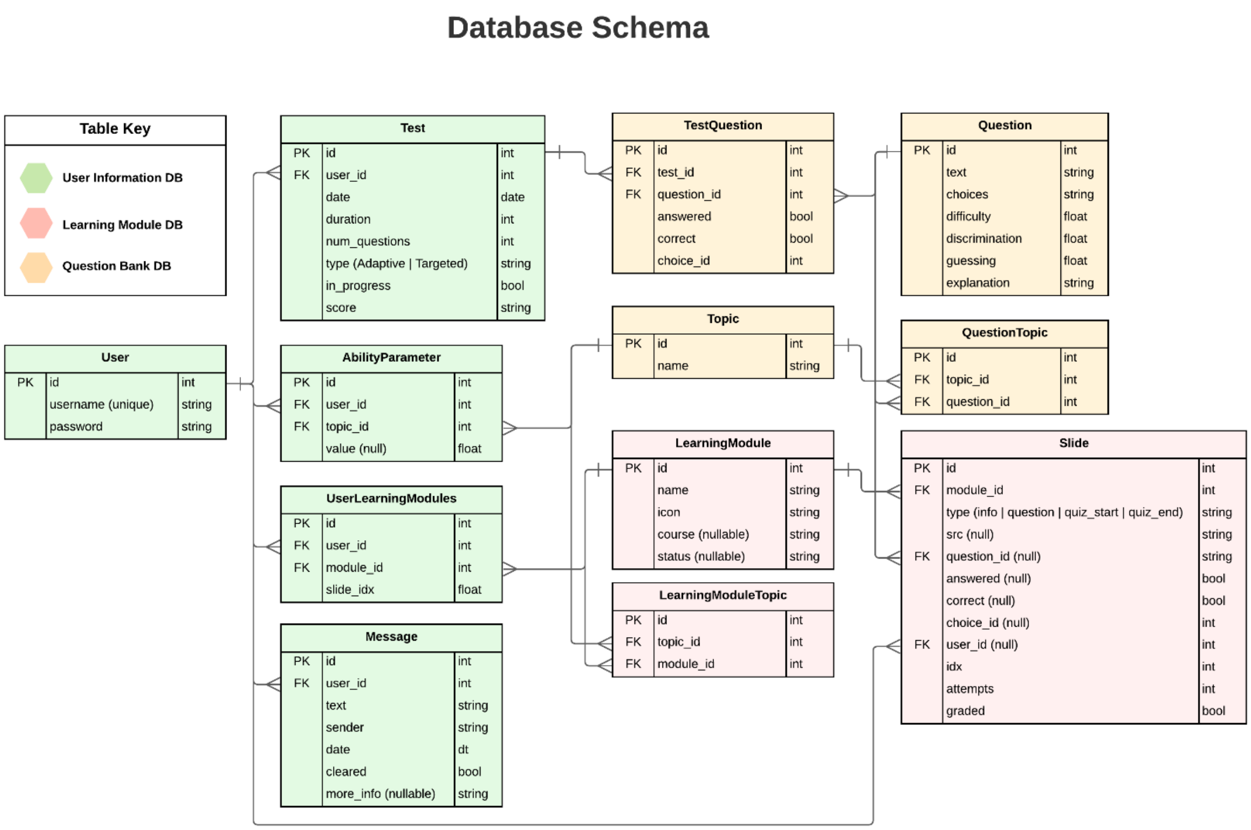

Figure 8: Daphne Academy Database Schema

Skill level estimation algorithms: Daphne Academy estimates the skill level of the user for various purposes. To do this, it uses a 3-parameter Logistic Model to model the probability that the user will correctly answer a particular question as a function of the user's skill level plus 3 other parameters: the difficulty of the question; the discriminatory power of the question; and the probability of guessing the correct answer. Specifically, the model is as follows:

$$p(\theta) = c + (1 - c)\frac{\exp[a(\theta - b)]}{1 + \exp[a(\theta - b)]}$$

where

- a: discrimination parameter
- b: difficulty parameter
- c: guessing parameter
- θ: user skill level
- p: Probability user answers correctly

Using this model, Daphne Academy estimates the user's skill level across several areas by performing Bayesian updates of the prior (initial) estimates with the information coming from the responses provided by the user to different questions. Intuitively, if the user answers a question correctly, this will increase Daphne's estimate of the skill level of the user in the areas relevant to that question. It will do so by an amount that depends on the probability that the user answers the question correctly based on the equation above. Mathematically:

maximize: $p(\theta|u) = [\prod_{j=1}^{K} p_j(\theta)^{u_j}[1 - p_j(\theta)]^{1-u_j}] * p(\theta)$ over $\theta \in (0, 1)$ where

- u: Data on user answered questions: $u_j = 0$ or $u_j = 1$ depending if the question was answered correctly for $j = 1, \ldots, K$
- $p_j(\theta)$: Probability user answered question j correctly

Question selection algorithm: A key feature of Daphne academy is that it intends to provide individualized training. This could be done in many different ways, but in this prototype, the functionality that was demonstrated is adaptive question selection in the exams. The idea is that the next question a user sees in an exam can be optimally selected on the basis of some objective function which could relate to reinforcing the user's weaker areas, or simply obtaining as accurate an estimate as possible of the user's skill levels. Note that there can be multiple conflicting objectives.

For this prototype, the question selection algorithm selects the next question in two steps: 1) determine the user's weakest topic area, and 2) select a question from the determined topic area so as to maximize information gained with respect to the user's ability level. Step one is completed by comparing the user's ability levels across topics, selecting the topic with the lowest corresponding ability level, and querying the Question Bank database for questions in this topic. Step two is completed by calculating the information gained for each question using the Item Information Function (see Figure 9) and selecting the question that produces the highest value.

$$I(\theta) = \left[a_i^2 \frac{1 - P_i(\theta)}{P_i(\theta)}\right]\left[\frac{(P_i(\theta) - c_i)^2}{(1 - c_i)^2}\right]$$

Figure 9: Item Information Function

5. Adapting to the user's cognitive state

Tracking the underlying mechanisms that affect learner control (e.g., experienced cognitive load, working memory capacity) can assist Intelligent Tutoring Systems to better understand the student and respond to his/her personal needs. Prior work on intelligent tutoring systems has attempted to model the learner at the behavioral, cognitive (e.g., cognitive load), meta-cognitive (e.g., ability to self-assess thought process and learning outcome), and affective (e.g., motivation) levels, and integrate the learner characteristics (e.g., expertise, skills, perceptions) to the content delivered and decisions made by the tutoring system.

Cognitive Load Theory (CLT) is a well-known theory, which has been empirically confirmed in numerous studies (Bannert 2002). According to the CLT, information during learning must be held in one's working memory until it has been processed sufficiently to pass into their long-term memory. Since the working memory's capacity can be very limited, when too much information is presented at once, the learning task becomes cognitively too demanding and much of the corresponding information is lost. Effective intelligent tutoring systems can incorporate elements of the CLT by tracking the learner's cognitive load and adapting the instruction accordingly. Potential ways to achieve that would be to reduce the problem space by breaking problems down into parts, and by using partially completed problems and worked examples; merging together multiple sources of visual information whenever possible; or extending the capacity of working memory by appropriate scaffolding of the considered problem.

Motivated by the above, we worked on an unobtrusive way to capture cognitive load via physiological measurements. Prior work indicates that eye tracking measures are viable alternatives toward measuring and tracking one's cognitive load. For example, Ikehara et al. found that eye movement is correlated with the level of cognitive load, measured via an easy and difficult cognitive task conducted by 34 male volunteers in the U.S. Air Force Academy (Ikehara & Crosby, 2005). Chen et al. found that pupil diameter increases with task difficulty in a set of 20 participants (9 female, 11 male) who conducted 7 cognitive tasks distributed over 5 difficulty levels (Chen et al., 2011). Krejtz et al. found that pupil diameter and microsaccade magnitude can discriminate difference levels of task difficulty (i.e., easy v.s. difficult), and hence cognitive load, in data collected by seventeen Psychology students (Krejtz et al., 2018). These findings can be leveraged toward designing adaptive tutoring systems that can track the user's cognitive load levels using pupil dilation measures collected (e.g., via the Tobii eye tracker). During times of elevated cognitive load, the tutoring system can appropriately relief the user's cognitive load via administering easier problems, breaking down the problem space, or providing additional scaffolding to the problem. This approach can be evaluated via a within-subject user study, in which the adaptive learning process can be compared to a control condition (i.e., no adaptation) and a self-adaptive learning condition (i.e., the user chooses which parts of the tutorial to follow). A tentative experimental setup of the proposed study appears in the following table.

Page	**Activity**		**Time**	**Cumulative Time**
--	1	Informed Consent	5 min	5 min
--	2	Tobi Eye-Tracker X5 setup (blinks, pupil diameter, gaze position etc.)	5min	10 min
--	3	Individual Differences Measures *Demographic questionnaire (including education and work experience)*	5 min 10 min	38 min

		General mental ability test [or IQ test] *Five Factor Model of Personality (IPIP)* *Daily experience survey*	10 min 3 min	
--	4	Psychological Relaxation Video	10 min	48 min
--	5	Training Overview	5 min	53 min
--	6	*Condition 1 Training (Control)* Training NASA TLX Survey Engagement Survey *Condition 2 Training (Self-Adaptive Learning)* Training NASA TLX Survey Engagement Survey Training Usability Survey *Condition 3 Training (Adaptive Learning via Cognitive Load Tracking)* Training NASA TLX Survey Engagement Survey Training Usability Survey	30 min 30 min 30 min	2 hour 23 minutes
--	8	Exit interview & surveys	10 min	2 hour 33 min
	TOTAL TIME			~ 3 hr

6. Validation of the Cognitive Assistant

A thorough validation effort based on experiments with human subjects was not compatible with the scope and resources of the incubator project. Therefore, various demonstrations of different versions of the prototype were done to the stakeholders to obtain feedback and gain some level of validation.

7. Cost of developing and maintaining the Cognitive Assistant

One of the concerns expressed by the stakeholders was the cost of implementing, deploying, and maintaining such a system in their organization. An initial effort was done to get a sense of the cost and resources that would be required.

Four development and maintenance cases have been defined and costed:

1) *Incorporation of training material for a new application domain to Daphne Academy*. Application domain here refers to a type of mission or platform of interest to the DoD.
2) *Update of cost estimation models*. When the DoD updates their cost models to reflect new datasets, technology evolution, or different modeling approaches, these are incorporated into Daphne Academy.
3) *Fine tuning/optimization of learning algorithms*. Advances in instructional design are anticipated throughout the life of Daphne Academy. Furthermore, usage experience may reveal areas of improvement to increase learning effectiveness and efficiency. This case covers updating Daphne Academy to reflect those advances and improvements.

4) *General maintenance of Daphne Academy infrastructure and user base*. Being a software tool, it is expected that there will be general maintenance activities necessary to support the Daphne Academy's user base. Furthermore, advances in user interfaces, natural language processing, etc. become available, Daphne Academy might be updated to improve its performance.

Costing for each case is described next, after presenting the cost estimation process that has been used.

Cost estimation process

Sizing was performed based on Lines of Code (LOC) and cost estimation was performed using COCOMO II as the cost model (as implemented in http://softwarecost.org/tools/COCOMO/). Costing assumes that Daphne Academy is a complete software product. It does not cover the development of Daphne Academy to become an operational product. In addition to traditional development and certification activities, operational activities not included in this estimation could include implementation of diverse assessment instruments, user management, user interface, or implementation of diverse information navigation and presentation mechanisms, among others. The estimation focuses on the scenarios identified in the previous section, which are the driving components behind the cost to deploy Daphne across the sponsor.

LOCs were directly counted and/or estimated by Daphne Academy's developer for each case identified above. This was based on the experienced gained during this Incubator project and the actual code of Daphne Academy prototype. Effects due to scaling, that is, increase in the complexity of Daphne as it incorporates large amounts of learning material, were not considered. The effect of this kind of variables needs to be explored in a future, more detailed analysis.

The number of concepts in a slide was used as a proxy for the learning information that needs to be captured in Daphne Academy. Specifically, we assumed that a slide contains in average 3 concepts. As with scaling, increasing in complexity resulting from higher orders of indexing of such information was not considered and is left for a future, more detailed analysis.

Labor rate was notionally set at $20K/person-month. COCOMO II drivers were set to nominal, with the following exceptions:

- Product complexity: very low.
- Required software reliability: low.
- Developed for reusability: very high.
- Multisite development: very low.

Integration effort and assessment were both set to 0% because the scenarios identified above mainly address the introduction of datasets and data structures to the product, not the development or modification of product capabilities or functionalities. Furthermore, only construction and transition costs are considered, since no need identification or architectural work is performed in any of the scenarios.

It should be noted that COCOMO II was considered convenient to meet the schedule constraints of the project, given the familiarity of the research team with it and its simplicity of use. However, its technical adequacy is questionable because, among others, the scenarios being estimated mainly address the introduction of datasets and data structures to the product, not modification of the product capabilities. This likely leads to an overestimation error in the estimates.

Case 1. Incorporation of training material for a new application domain to Daphne Academy.

Required effort to incorporate training material for a new application domain to Daphne depends on whether the new concepts are already available in Daphne Academy or not. If the concepts are already available, the effort is primarily limited to train the neural networks on the material. If the concepts are novel, additional effort to incorporate such concepts is necessary.

Scenario 1. Already available concepts.

Requires 20 LOC per slide. Corresponding effort is given in the following table for different sizes of the learning material to be incorporated.

N of slides	Person-month	Cost
1	<0.1	$563
10	0.3	$7,078
100	4.4	$89,043
1,000	56	$1,120,210

Scenario 2. Novel concepts.

Requires 20 + 15 x N of concepts LOC. Corresponding effort is given in the following table for different sizes of the learning material to be incorporated.

N of slides	Person-month	Cost
1	0.1	$2,056
10	0.9	$18,112
100	10.9	$218,279
1,000	136.7	$2,734,051

Case 2. Update of cost estimation models.

We have not been able to find reasonable LOCs that are needed to be updated for cost estimation models. Estimates assume this effort is similar in nature to maintenance. We provide estimates for notional sizes, which will have to be fine-tuned in a future model.

LOC	Person-month	Cost
100	0.2	$3,753
1,000	2.4	$47,214
10,000	29.7	$593,982

Case 3. Fine tuning/optimization of learning algorithms.

Assuming that future learning algorithms will be similar in complexity, 200 LOC required. Corresponding effort is 0.3 person-month, $7,078.

Case 4. General maintenance of Daphne Academy.

Maintenance will depend on the eventual size of Daphne. The current prototype is 3,400 LOC. Let us assume that the operational product will be one order of magnitude its size, for a total of 34,000 LOC, and that maintenance will be 10% of it. Assuming familiarity and software understanding, the estimated effort is 9.1 person-months/year, $181,359/year.

8. Conclusion

The capabilities of the current prototype of Daphne Academy are limited by the scope of this Incubator project. The following are some future work items to improve the CA:

- More roles could and should be added to the QA system
- The architecture of the neural network models could and should be optimized to improve the accuracy of the QA system and Material Recommender.
- Other objective functions could be explored for the question selection algorithm
- A diverse set of instructional mechanisms could be incorporated, integrating lecture notes, slides, and videos with assessment mechanisms such as open-ended responses, quizzes, and cost estimation assignments.
- Tutoring material that can help break down each problem and scaffold the answer to each problem can be designed.
- An adaptive tutoring system can be designed based on a personalized estimation of a user's cognitive load.

9. References

a.C. Graesser, Chipman, P., Haynes, B. C., Olney, a., Graesser, a. C., Chipman, P., Haynes, B. C., Olney, a., a.C. Graesser, Chipman, P., Haynes, B. C., Olney, a., Graesser, a. C., Chipman, P., Haynes, B. C., & Olney, a. (2005). AutoTutor: an intelligent tutoring system with mixed-initiative dialogue. *IEEE Transactions on Education*, *48*(4), 612–618. https://doi.org/10.1109/TE.2005.856149

Cheung, B., Hui, L., Zhang, J., & Yiu, S.-M. (2003). SmartTutor: An intelligent tutoring system in web-based adult education. *Journal of Systems and Software*, *68*(1), 11–25.

Corbett, A. T., Koedinger, K. R., & Anderson, J. R. (1997). Intelligent tutoring systems. In *Handbook of Human-Computer Interaction* (pp. 849–874). Elsevier Science B. V. https://doi.org/10.1126/science.228.4698.456

D'Mello, S. K., Craig, S. D., Gholson, B., & Franklin, S. (2005). Integrating affect sensors in an intelligent tutoring system. *Affective Interactions: The Computer in the Affective Loop Workshop, International Conference on Intelligent User Interfaces*, 7–13.

Folsom-Kovarik, J. T., Sukthankar, G., & Schatz, S. (2013). Tractable POMDP representations for intelligent tutoring systems. *ACM Transactions on Intelligent Systems and Technology*, *4*(2), 1–22. https://doi.org/10.1145/2438653.2438664

Kim, B., Suh, H., Heo, J., & Choi, Y. (2020). AI-Driven Interface Design for Intelligent Tutoring System Improves Student Engagement. *ArXiv Preprint ArXiv:2009.08976*.

Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). *Intelligent Tutoring Goes To School in the Big City*. 30–43.

Mayo, M. J. (2001). *Bayesian Student Modelling and Decision-Theoretic Selection of Tutorial Actions in Intelligent Tutoring Systems*. http://ir.canterbury.ac.nz/bitstream/10092/2565/1/thesis_fulltext.pdf

Nye, B. D. (2014). Barriers to ITS adoption: A systematic mapping study. *International Conference on Intelligent Tutoring Systems*, 583–590.

Nye, B. D. (2015). Intelligent tutoring systems by and for the developing world: A review of trends and approaches for educational technology in a global context. *International Journal of Artificial Intelligence in Education*, *25*(2), 177–203. https://doi.org/10.1007/s40593-014-0028-6

Ong, J., & Ramachandran, S. (2003). Intelligent Tutoring Systems: Using AI to Improve Training Performance and ROI. *Networker Newsletter*, *19*(6), 1–6.

Sarrafzadeh, A., Alexander, S., Dadgostar, F., Fan, C., & Bigdeli, A. (2008). “How do you know that I don’t understand?” A look at the future of intelligent tutoring systems. *Computers in Human Behavior*, *24*(4), 1342–1363. http://ovidsp.ovid.com/ovidweb.cgi?T=JS&PAGE=reference&D=psyc6&NEWS=N&AN=2008-05428-005