Venigalla and Chimalakonda
Smart Learning Environments (2023) 10:9
https://doi.org/10.1186/s40561-023-00228-y

Smart Learning Environments

RESEARCH Open Access

TAnnotator: Towards Annotating Programming E-textbooks with Facts and Examples

Akhila Sri Manasa Venigalla* and Sridhar Chimalakonda

*Correspondence:
cs19d504@iittp.ac.in

Research in Intelligent Software & Human Analytics (RISHA) Lab, Department of Computer Science & Engineering, Indian Institute of Technology Tirupati, Tirupati, India

Abstract

E-textbooks are one of the commonly used sources to learn programming, in the domain of computer science and engineering. Programming related textbooks provide examples related to syntax, but the number of examples are often limited. Thus, beginners who use e-textbooks often visit other sources on the internet for examples and other information. Adding dynamic information to programming related e-textbooks such as additional information about topics of discussion and real-world programming examples could enhance readers' experience, and improve their learning. Hence, towards enhancing user experience with programming-based e-textbooks, we present *TAnnotator*, a web-based portal that dynamically annotates computer-programming based e-textbook, *The C++ Tutorial*, with related programming examples and *tooltips*. The *tooltips* aim to provide further knowledge to the readers about various concepts being discussed in textbooks by providing related facts adjacent to the text of the topic in the e-textbook. *TAnnotator* has been evaluated to assess the usefulness, user experience and complexity using UTAUT2 model through a user survey with 15 volunteers. The results of the survey indicated that *TAnnotator* was useful in providing additional knowledge on top of the e-textbook.

Keywords: E-textbooks, Dynamic annotation, LDA, Facts, Examples

Introduction

Use of online resources has increased exponentially with widely available and accessible internet. Novice programmers often rely on various online knowledge sources such as MOOC courses, e-textbooks,[1] interactive and intelligent online tutors, blogs, crowd sourced platforms and so on to learn programming (Keuning et al., 2021; Venigalla & Chimalakonda, 2020; Vinaja, 2014). Many emerging technologies are being used in the classrooms to improve both teaching and learning (Almiyad et al., 2017; Dutta et al., 2022; Kim et al., 2016; Weber & Brusilovsky, 2016). Augmented reality, virtual reality, collaborative environments are being integrated into teaching to improve information retention by learners (Holstein et al., 2018; Kao & Ruan, 2022; Mystakidis et al., 2021).

[1] e-books and e-textbooks are used interchangeably.

Among the available online sources for learning, textbooks are observed to comprise of authentic information (Oates, 2014). E-textbooks are observed to be used on a large scale in the field of computer science and engineering (Fischer et al., 2015; Tang, 2021). This extensive use of textbooks also resulted in various innovative ways to be incorporated into the basic e-textbooks (Hori et al., 2015; Rockinson-Szapkiw et al., 2013; Weng et al., 2018). E-textbooks are now being developed to be interactive, including animation features, simulations and emotions (Chang & Chen, 2022; Lee et al., 2013; Sun et al., 2012; Weng et al., 2018). Dictionaries are also being added to the current e-textbooks to reduce the user efforts of browsing through separate platforms for clear understanding of some words in the textbooks (McGowan et al., 2009; Rockinson-Szapkiw et al., 2013). Several other features including quick search, note making and low printing costs have resulted in e-textbooks having an edge over physical textbooks (Davis & Song, 2020; Sun et al., 2012; Weng et al., 2018). Textbooks are also integrated with content from community Q&A forums to provide information that is deficient in the e-textbook (Ghosh, 2022; Venigalla & Chimalakonda, 2020). Other features to support easy navigation across the e-textbook based on various frequent jump-back behaviours of students using e-textbooks are being explored (Ma et al., 2022).

Programming related e-textbooks have also been integrated with video and audio tutorials, visualizations, code executions and so on (Ericson et al., 2015; Miller & Ranum, 2012; Solcova, 2016; Venigalla & Chimalakonda, 2020; Weber & Brusilovsky, 2016). Code execution platforms that show the run time execution of code snippets presented in the textbook are observed to contribute to better and clear understanding (Solcova, 2016; Weber & Brusilovsky, 2016). Researchers have also attempted to enhance existing information platforms by integrating information available on multiple external sources (Venigalla & Chimalakonda, 2020).

Crowd sourced Question and Answer platforms such as StackOverflow are being integrated with code snippets from crowd sourced code sharing platforms such as GitHub and other code hosting platforms such as JExamples (Reinhardt et al., 2018; Venigalla et al., 2019). GitHub is also being integrated with information from StackOverflow platform (Pletea et al., 2014). Several developers post information and observations about various programming concepts on multiple view-sharing platforms such as blogs[2] and developer forums.[3] Integrating e-textbooks with information available on multiple sources could help in learning through different integrated resources, at one place, rather than spending time to explore different resources in finding the desired related information (Almansoori et al., 2021; Seeling, 2016). For example, Almansoori et al. have observed lack of security related discussions in the introductory programming textbooks and suggested augmenting security related information to the contents in the textbook (Almansoori et al., 2021). Patrick Seeling has also observed that integrating online programming textbooks with self-check exercises and feedback has improved the learners' performance (Seeling, 2016). These integrations help users to gain a wider range of knowledge and exposure. The current e-books, are an authentic source of information, however, they contain only

[2] https://en.cppreference.com/w/cpp/language.

[3] https://moodle.org/mod/forum/s://moodle.org/mod/forum/.

static data. Though e-textbooks are not the only learning materials in a course, they could be used as a reference. Moreover, e-textbooks could support enthusiast learners, though not enrolled in a course, in learning concepts of programming.

The existing research on improving e-textbooks mostly focuses on integration with augmented reality, virtual reality, audio and video tutorials and so on (Rockinson-Szapkiw et al., 2013; Sun et al., 2012; Weber & Brusilovsky, 2016). However, most of these enhancements are static in nature, indicating that they are predefined for a given textbook. Also, the code execution platform augmented with e-textbooks implements only the code presented in the textbook. Only interaction and intelligent tutor-based e-textbooks exhibit a scope to present information apart from that available in the textbook. It has also been observed that existing programming language textbooks do not suffice in well explaining the concepts of programming languages, motivating the need to improve these textbooks (Mazumder et al., 2020). However, to the best of our knowledge, except for the intelligent tutor-based textbooks, we are not aware of any research that aims to dynamically integrate an e-textbook with further conceptual information from other sources.

Though *DynamiQue* (Venigalla & Chimalakonda, 2020) aims to integrate e-textbook with information from StackOverflow, it is only in the form of question and answers and no programming examples or conceptual information is presented. Researchers have also observed that additional examples help readers to construct better mental models, which could further help in improving the readers understanding on a topic (Gerjets et al., 2006).

Readers of programming oriented e-textbooks spend time and effort to visit external sources to explore examples and other additional information about a concept, owing to limited content in the e-textbook. Hence, we propose *TAnnotator* to reduce these efforts by dynamically augmenting e-textbook with examples and tooltips containing additional information, extracted from external sources.

We are primarily interested to explore the following two research questions in this work:

- *RQ1:* What is the possibility of integrating external knowledge to e-textbooks?
- *RQ2:* Is the annotation of additional content to e-textbooks useful from user perspective?

Hence, towards integrating external knowledge to e-textbooks, *TAnnotator*, a web-based portal aims to annotate programming-based textbooks with various facts about the content being discussed in the textbook. It also augments the e-textbook with code-snippet examples, as an add-on to already specified examples on the textbook. Figure 1 presents an example of a fact (Fig. 1B) and code snippet example (Fig. 1A) displayed on *TAnnotator* for the topic—*#include*. To analyse the usefulness of this annotation from user perspective, we evaluated the usefulness of *TAnnotator* through a qualitative user survey based on UTAUT2 model (Rondan-Catalun˜a et al., 2015) with 15 participants. The results of the survey indicated that *TAnnotator* was

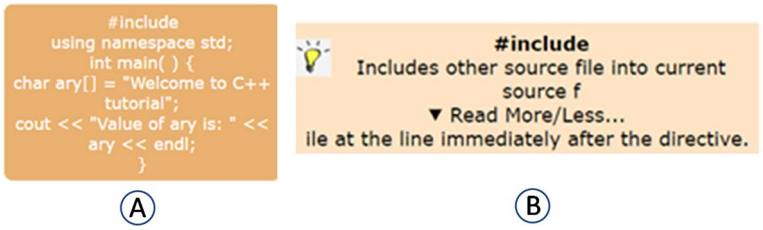

Fig. 1 Sample tooltip with code snippet example in (**A**) and factual information in (**B**)

found to be useful and easy to use. The questionnaire and results of the user survey are presented here.[4]

Related work

Several technologies have been introduced to improve and teaching and learning (Bailey & Zilles, 2019; Benotti et al., 2018; Zavala & Mendoza, 2018). *Mumuki* has been proposed as a web-based tool, to reduce work load of teachers and to avoid biased evaluations by assessing code snippets submitted by students and providing feedback to teachers about the performance of students (Benotti et al., 2018).

Emotions have been integrated into the e-textbooks to reduce the cognitive load on the students and support better learning achievements. It has been observed that e-textbooks with emotions helped students in learning and retention when compared to e-textbooks without emotions integrated and also paper textbooks (Chang & Chen, 2022).

Content in the e-textbooks is analysed to identify deficient areas that have a scope for inclusion of more information (Ghosh, 2022). These identified deficient content types are explored in the community Q&A forums and relevant information is extracted and integrated to the e-textbook at the deficient area. Such integration was observed to enhance the learning interest among students (Ghosh, 2022).

Zavala et al., have proposed to automatically generate programming assignments based on a specific template mentioned by teachers. This idea has been demonstrated by integrating it with existing programming practice tool (Zavala & Mendoza, 2018). *uAssign* has been proposed as a software, that can be embedded into any Learning Management System (Bailey & Zilles, 2019). It aims to generate assignments based on unix terminal and also facilitates teachers to understand the skills of students on unix terminal.

Krusche et al., have proposed *ArTemis* to support automatic assessment of programming solutions and provide feedback to the students based on their performance instantly, that could constructively improve their programming skill (Krusche & Seitz, 2018).

Improving conceptual learning has become the primary goal of learning in the present day. A study conducted on different types of collaborative learning indicated better conceptual skills in students who followed a feedback and discussion based collaborative

[4] http://bit.ly/35PT9lC.

learning, rather than those who followed collaborative learning with minimal discussions (Harsley et al., 2017).

Integrating information across various sources also supports learners to gain both knowledge and insights of other developers. StackOverflow has been integrated with definitions of APIs, extracted from JDK and examples from a code hosting platform-JExamples (Venigalla et al., 2019). This integration of information from multiple sources improves knowledge of users about various APIs and also supports the users to be aware of multiple ways in which a specific API can be used (Venigalla et al., 2019). Reinhardt et al. have integrated information from multiple platforms to provide insights on API usage patterns to users (Reinhardt et al., 2018). They have integrated source code available in public repositories on GitHub with code snippets present in questions and answers of StackOverflow. Code snippets that misuse APIs are detected from StackOverflow and are presented with multiple examples from GitHub that display the correct usage of APIs (Reinhardt et al., 2018).

Also, linking information present on multiple platforms might provide valuable insights on issues present in code snippets. *DynamiQue* has been developed to integrate e-textbooks with questions and answers present on StackOverflow (Venigalla & Chimalakonda, 2020). Relevant questions were identified based on topics on the page, extracted using LDA topic modelling technique. However, no other information except question and answers from StackOverflow was augmented with e-textbooks (Venigalla & Chimalakonda, 2020). E-textbooks have also been integrated with programming practice sessions and automatic grading to ease and improve learning in a programming course (Ellis et al., 2019).

Several approaches and tools have aimed to improve learning of programming concepts by making learning interactive, interesting and simple (Berns et al., 2019; Harsley et al., 2017; Krusche & Seitz, 2018; Weber & Brusilovsky, 2016). Various tools have also been augmented to e-textbooks to ease the learning of students (Rockinson-Szapkiw et al., 2013; Sun et al., 2012; Venigalla & Chimalakonda, 2020). Considering the advantages of multiple source integration, linking e-textbooks with information from multiple sources could be of a great advantage to learners (Venigalla & Chimalakonda, 2020). Though there is adequate research in integrating various forms of information present across multiple programming-based platforms, there is hardly any research that attempts to integrate multiple source conceptual information with textbooks, other than integration of question and answers.

Design and development of *TAnnotator*

TAnnotator augments e-textbook in the domain of programming, with facts and examples from various discussion forums. The current version of *TAnnotator* is a web portal and augments only one specific e-textbook—*The C++ Tutorial*, with facts related to some keywords in the textbook content. Each page of the e-book is rendered onto the web portal and facts are displayed adjacent to the text in the e-book, page-by-page, as shown in Fig. 3. Currently, we demonstrate the idea of augmenting e-book with facts and examples, with six pages of the e-book, which could be easily extended to all pages in the e-textbook.

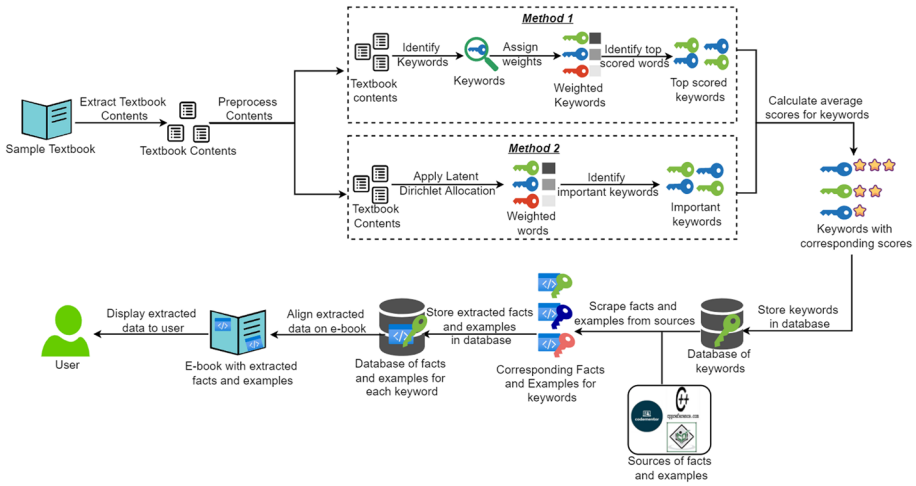

Fig. 2 Approach followed in designing *TAnnotator*

We used *Flask*[5] framework to design and implement the web portal of *TAnnotator*. Implementation of *TAnnotator* requires identification of useful keywords and rendering of facts and examples for these keywords. Useful keywords can be obtained from identifying frequent words and through topic modelling. In a textbook, the words that occur with more frequency and those that occur with least frequency may not be useful. For example, the word *good* might have highest frequency in a text that describes usefulness of a data structure, but, this does not imply towards any useful topic. Hence, the words with medium frequencies could be identified as useful keywords, because the words with high frequencies might sometimes be insignificant (Choi & Park, 2019). A plenty of approaches such as Latent Semantic Analysis (LSA), Latent Dirichlet Allocation (LDA), and so on can be used to identify topics in a text through topic modelling. Among these approaches, LDA facilitates processing of long texts that occur in textbooks, and hence is used in many studies that involve topic modelling (Onan et al., 2016; Venigalla & Chimalakonda, 2020). For the implementation of *TAnnotator*, we chose an approach that includes identification of the keywords through LDA and also based on frequency of the keyword, to ensure identification of most useful keywords.

The approach followed in development of *TAnnotator* is displayed in Fig. 2 and presented below.

- *Step 1: Extract Textbook Content*- The content in the textbook is extracted page-by-page in '*.pdf*' format and are stored in '*.html*' format for further processing and rendering. The content in '*.pdf*' format is converted to '*.html*' format using the *pyPdf*[6] package.
- *Step 2: Process Content*- This content is then processed for lemmatization, stemming and stop word removal. We created a set of stopwords that are to be removed, which

[5] https://flask.palletsprojects.com/en/1.1.x/.

[6] https://pypi.org/project/pyPdf/.

are considered during stopword removal process. This set of stopwords is based on the stopwords in English language listed in 'wordcloud'[7] and 'nltk'[8] libraries. The process of stemming involves excluding prefixes and suffixes of the words and eliminating punctuations from the text. These words are morphologically analysed and further converted into their dictionary base forms, which is termed as lemmatization (Plisson et al., 2004; Porter et al., 1980). Count vectorizer was used for further cleaning and conversion of words into machine readable format. After preprocessing the text, we employ two methods to identify the keywords that are required to extract facts.

- *Step 3: Method 1-* This method includes identification of keywords based on their frequency scores

    - *Identify frequent words-* Frequency of all the processed words is calculated and the words with highest, medium and lowest frequencies are identified. The frequency counts are listed in the descending order of their frequencies and the frequency count for top 15% of this list are considered highest, next 20% as medium and the remaining 65% as lowest frequencies. This decision on the thresholds for highest, medium and lowest is based on the observation of keyword lists for 10 excerpts of the textbook. These thresholds can be modified as per the content being discussed in the text and based on inputs from experts in the domain of text under discussion.
    - *Assign weights-* To ensure that the least frequent words are not considered as useful and to view relative importance of words, we used tf-idf method to up-weight more frequent terms and down-weight the least frequent terms.
    - *Identify useful words-* A sparse matrix has been generated based on these assigned scores. The words with highest and medium-value scores have been selected and stored.

- *Step 4: Method 2-* This method uses Latent Dirichlet Allocation approach of topic modelling to identify useful keywords in the text.

    - *Apply Latent Dirichlet Allocation-* Topics being discussed in the text are identified using LDA. We devised LDA to generate five topics, for the text on which it is applied. LDA is then applied on the text obtained after preprocessing, which resulted in top five topics being discussed in the text.
    - *Identify important keywords-* We considered the top three keywords in each of the five topics, along with their scores. This resulted in a set of fifteen keywords, which are stored for further processing in the subsequent steps.

- *Step 5: Calculate Average Scores for Keywords-* We fetched the keywords obtained during Method 1 and Method 2, and appended score as 1 to all the 15 keywords obtained through Method 2, as they did not include any specific scores. Method 1

[7] urlhttps://amueller.github.io/word cloud/generated/wordcloud.WordCloud.html.

[8] https://www.nltk.org/.

identifies important keywords based on the frequencies of the keywords. However, considering most frequent words alone might compromise identification of important, but not frequently occurring words. Hence, we apply LDA in the Method 2 to identify useful keywords. However, as we apply LDA only for five classes, the keywords obtained might miss out on the frequently occurring but useful keywords. This is because LDA assigns lower scores for words that occur across multiple classes. Hence, to retain both useful and frequent keywords and useful and not-frequent keywords, we employ two methods as discussed in Step 3 and Step 4. We then calculated the average scores of all the keywords obtained from Method 1 and Method 2. As we aimed to display 10 facts on each page of the e-book, we fetched top 10 keywords based on the averaged scores, for better results. However, using keywords only from Method 2 also results in useful topics, but compromises on frequency. The decision of displaying 10 facts is only to demonstrate the idea of annotating e-textbook. Based on the requirement of the users, the number of facts being displayed could be altered, with minimal changes in the source code of *TAnnotator*.

- *Step 6: Store keywords in database-* The obtained keywords are stored in the database to ease searching for facts in next phases.
- *Step 7: Scrape facts and examples from sources-* The factual data is scraped (fetched) dynamically from three discussion sources identified prior to the implementation of *TAnnotator*. These sources include *Tutorialspoint*,[9] *CPPReference*[10] and *code-Mentor*,[11] and were sufficient to obtain facts for the 10 keywords identified in each of the six pages. More number of sources can further be added if number of pages or the number of e-textbooks are increased. These facts are scraped by passing 'http' requests to the sources, appended with the identified keywords. These requests return factual data from the sources based on the keywords.
- *Step 8: Store extracted facts and examples in database-* The extracted facts and examples are stored in the database, along with their corresponding keywords in *.json* file.
- *Step 9: Align extracted data on e-book-* The scraped data stored in the database is extracted along with their keywords and is aligned adjacent to the text discussing about each keyword. The location of empty space in the margin, adjacent to the text discussing about the keyword is identified. A textbox containing the extracted data is placed in the space identified.
- *Step 10: Display extracted data to the user-* The facts are displayed in the form of *tooltips* to the user. All the keywords are highlighted on the text in the e-book with the help of *hilitor.js* library. When users hover on the tooltip, they are displayed with examples related to the facts that are annotated to the e-book.

This process of development answers RQ1, that while annotations to e-textbooks with additional knowledge could be done, this annotation is limited to availability of information, ease of extraction, associated license guidelines and so on. However, if the information is available for extraction, and is permitted to be extracted, while the content

[9] https://www.javatpoint.com/cpp-tutorial.

[10] https://en.cppreference.com/w/cpp/language.

[11] https://www.codementor.io/.

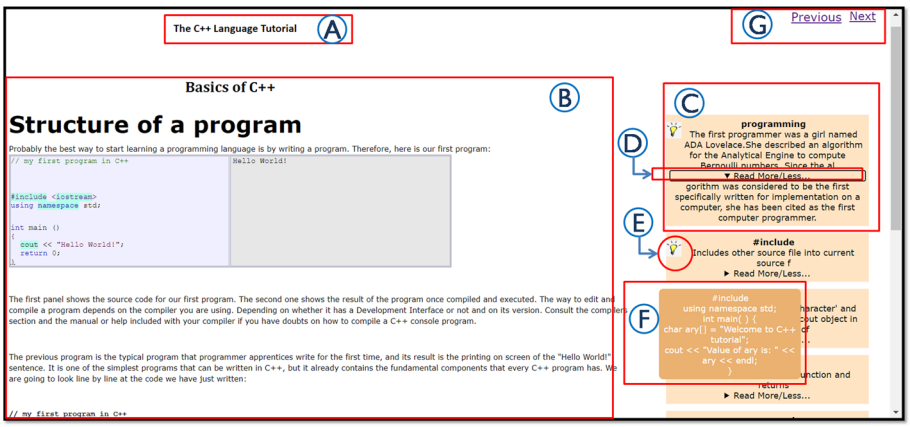

Fig. 3 A Snapshot of *TAnnotator* representing **A** name of the e-textbook, **B** content of the textbook, **C** facts being displayed, **D** option to expand or collapse the fact, **E** option to view example code snippet, **F** the example code snippet being displayed and **G** option to navigate across pages in the e-textbook

in e-textbook is also permitted to be extracted, then the e-textbook could be annotated with additional information.

User scenario

Suppose *Risha* is a computer science student, and is interested in reading the textbook-The C++ Language tutorial. She visits the e-textbook and reads through the book, and wishes to know more about the topics being discussed in the e-book. She learns about *TAnnotator* and considers that it would aid her learning. Hence, she visits *TAnnotator* portal and decides to read the e-textbook integrated with *tooltips* that include facts and examples.

She is displayed with text of *The C++ Language tutorial* e-textbook, page by page as shown in Fig. 3. The title of the text book is displayed as shown in Fig. 3A, and the text in each page of the textbook is displayed as shown in Fig. 3B. The text of the page displayed is processed using topic modelling technique Latent Dirichlet Allocation (LDA), and the topics such as *programming, #include, cout* and so on are extracted, as shown in the right panel of Fig. 3. *Risha* is displayed with facts as *tooltips*, that are dynamically extracted from preidentified sources based on the keywords of extracted topics. These *tooltips* are displayed to the right of the web page, as shown in Fig. 3C. A '*Read More/Less*' icon is presented at every tooltip to enable further reading (as highlighted in Fig. 3D). When *Risha* clicks on the Read More/Less icon, she is displayed with more elaborated text on the tooltip. When *Risha* hovers on the image highlighted by in Fig. 3E, she is displayed with example code snippet corresponding to the *#include* keyword, as shown in Fig. 3F. Further, *Risha* is facilitated to navigate back and forth through pages using the *Previous* and *Next* tabs represented in Fig. 3G.

Evaluation

TAnnotator has been developed with main aim of supporting readers and enhancing user experience with e-textbooks, specifically in the area of programming, keeping in mind, the ever evolving nature of technologies in the domain of computer science

and engineering. Hence, our evaluation was based on identifying the extent to which *TAnnotator* is useful, the complexity associated with using *TAnnotator* and broadly, the user experience with *TAnnotator*. Research tools and approaches such as *DynamiQue* (Venigalla & Chimalakonda, 2020), proposed with similar goals of improving user experience in the existing literature have been evaluated through user surveys, based on multiple models such as TAM (Technology Acceptance Model), UTAUT (Unified Theory of Acceptance & Use of Technology) (Venkatesh et al., 2003, 2012), IDT (Innovation Diffusion Theory), integration of TAM and IDT, and other integrated models (Venigalla & Chimalakonda, 2020).

Considering attributes such as ease of use and usefulness, against which *TAnnotator* has to be evaluated, and considering the advantages and appropriateness of UTAUT2 model (Venkatesh et al., 2012) over other TAMs (Rondan-Catalun˜a et al., 2015), we evaluate *TAnnotator* through a user survey based on UTAUT2 model, adapted to the functionalities of *TAnnotator*. Certain factors of UTAUT2 such as *social influence* and *facilitating conditions* that could strongly influence book readers, have made UTAUT2 more appropriate to evaluate *TAnnotator*, than the other existing TAMs. UTAUT refers to Unified Theory of Acceptance and Use of Technology, that aims to evaluate a novel technological contribution based on multiple factors, with behavioral intention to use and user behavior as the prominent attributes for evaluation.

UTAUT2 (Venkatesh et al., 2012) model evaluates technology based on seven attributes—Performance Expectancy (PE), Effort Expectancy (EE), Social Influence (SI), Facilitating conditions (FC), Hedonic Motivations (HM), Price Value (PV) and Habit (H). These attributes were identified to influence the users' decision on willing to use the portal. PE describes the usefulness of technology, as perceived by the users, with respect to providing users with the necessary information. EE refers to the effort to be spent by users in order to use the technology, that includes clarity and ease of use. SI indicates the opinion of users' society towards using the technology, while FC indicates the availability of resources to use the technology and the information and skillset to be learned before using the technology. HM refers to the fun and pleasure obtained by using the technology. PV refers to the users' opinion on value of the price associated with the technology. H deals with interest of users in using the technology in the users' daily routine, and BI deals with the tendency of user to use and continue to use the technology. *TAnnotator* can currently be accessed without any pricing from the source, thus, not requiring to be evaluated against PV attribute. Hence, an adapted UTAUT2 model, with the following six attributes, which are considered for evaluation of *TAnnotator*, has been used to perform a user survey.

- Performance expectancy (PE)
- Effort expectancy (EE)
- Social influence (SI)
- Facilitating conditions (FC)
- Hedonic motivations (HM)
- Habit (H)
- Behavioral intention to use (BI)

Table 1 Adapted UTAUT2 Questionnaire and the corresponding mean and standard deviation values obtained after the user survey

Factors	Questions	Mean	SD
PE1	I find TAnnotator useful in my daily life	3.75	0.91
PE2	Using the TAnnotator increases my chances of achieving things that are important to me	3.65	0.74
PE3	Using the TAnnotator helps me accomplish things more quickly	3.9	0.71
PE4	Using TAnnotator would increase my productivity in learning	4.1	0.71
EE1	Learning how to use the TAnnotator is easy for me	4.25	0.78
EE2	My interaction with the TAnnotator is clear and understandable	4.15	0.67
EE3	I find the TAnnotator easy to use	4	0.79
EE4	It is easy for me to become skillful at using TAnnotator	4	0.56
SI1	People who are important to me would think that I should use TAnnotator	3.65	0.67
SI2	People who influence my behavior would think that I should use TAnnotator	3.7	0.65
SI3	People whose opinions that I value prefer that I use TAnnotator	3.85	0.67
FC1	I have the resources necessary to use TAnnotator	4.2	0.83
FC2	I have the knowledge necessary to use TAnnotator	4.3	0.65
FC3	TAnnotator is compatible with other technologies I use	4.05	0.76
FC4	I can get help from others when I have difficulties using TAnnotator	3.95	0.76
HM1	Using the TAnnotator is fun	3.85	0.48
HM2	Using the TAnnotator is enjoyable	3.85	0.58
HM3	Using the TAnnotator is very entertaining	3.75	0.63
HT1	The use of the TAnnotator has become a habit for me	3.1	0.96
HT2	I must use the TAnnotator	3.35	0.81
HT3	Using the TAnnotator has become natural to me	3.45	0.88
BI1	I intend to continue using the TAnnotator in the future	3.7	0.57
BI2	I will always try to use the TAnnotator in my daily life	3.7	0.65
BI3	I plan to continue to use TAnnotator frequently	3.65	0.67
CR1	TAnnotator has displayed facts and examples relevant to content in the e-textbook	4.1	0.71

Invites were sent out to 20 under graduate students and 10 industry developers requesting for participation in the evaluation survey. While the invites were sent randomly to 20 undergraduate students from a class of 40, we had a filter that the undergraduate students selected have computer science as their major, in our academic institution. The industry developers were randomly chosen based from different organizations, through their Linkedin[12] India profiles, where the users mention their work profile and details.

We received acceptance responses from 11 students and 4 industry developers. Thus, we considered 15 volunteers, in the age group of 19–44 years. The volunteers included 6 female students, 5 male students and 2 female industry developers and 2 male industry developers. Of all the participants, 12 of them stated that they were very familiar or familiar to using e-textbooks, while three of the participants marked their familiarity as neutral. A 5-point Likert scale based questionnaire of 30 questions, based on the six attributes of UTAUT2, presented in Table 1 and other required demographic information such as age and familiarity, was then circulated among the volunteers. While the

[12] https://www.linkedin.com/.

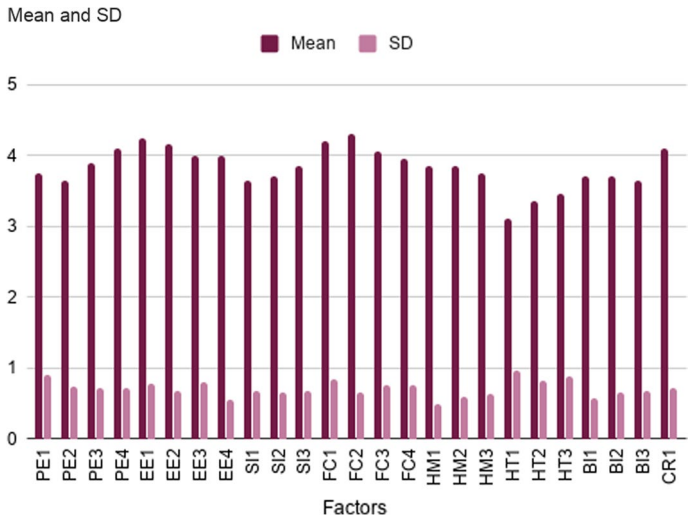

Fig. 4 Mean and standard deviation plots for factors of UTAUT2 in the questionnaire

first 6 questions of the questionnaire referred to demographic information, the rest 24 questions dealt with attributes of adapted UTAUT2 model, similar to the following questions.

- Using *TAnnotator* would increase my productivity in learning. *(Rate from Strongly Disagree to Strongly Agree)*
- *TAnnotator* has displayed facts and examples relevant to content in the e- textbook. *(Rate from Strongly Disagree to Strongly Agree)*
- My interaction with the *TAnnotator* is clear and understandable. *(Rate from Strongly Disagree to Strongly Agree)*

The volunteers were explained about *TAnnotator* and were requested to use the portal through the provided link. They were asked to read through the textbook provided and consequently view the tips presented on the portal. They were then asked to answer the circulated questionnaire.

The questionnaire and corresponding results are presented here.[13]

Results

All questions of the questionnaire sent for user survey were classified into corresponding UTAUT2 factors and labelled accordingly. These questions were answered based on a 5-point Likert scale from *Strongly Disagree* to *Strongly Agree*. During analysis of the results, the highest score, 5 has been assigned to *strongly agree* and the lowest score, 1 has been assigned to *strongly disagree*. Mean and standard deviation of the results have been calculated for each of the questions and the resultant scores are plotted on a graph, that is presented in Fig. 4. The horizontal axis of the graph comprises of question codes for all the questions based on the factors they correspond to, while the vertical

[13] http://bit.ly/35PT9lC.

Table 2 Correlation analysis results of UTAUT2 variables

	PE	EE	SI	FC	HM	HT	BI
PE	1						
EE	0.625	1					
SI	0.546	0.510	1				
FC	0.618	0.606	0.638	1			
HM	0.574	0.520	0.479	0.346	1		
HT	0.373	0.376	0.661	0.387	0.555	1	
BI	0.228	0.337	0.353	0.272	0.243	0.671	1

axis comprises of mean and standard deviation scores. The mean value of the questions indicates the extent of acceptance, while standard deviation indicates the mutual level of agreement among the participants.

As reported in the Fig. 4 the mean of all the questions is greater than 3 and close to 4 in majority of the cases, indicating better acceptance of *TAnnotator* among the participants of user survey. Questions corresponding to Effort Expectancy (EE) and Facilitating Conditions (FC) aspects have highest means, close to 4, and reasonably low standard deviations. This indicates that majority of the participants have collectively found learning to use and being skillful at *TAnnotator* to involve minimal effort from the participants. The mean value 4.1 for question based on correctness indicates that *TAnnotator* has rendered correct and relevant information on the respective pages of the e-textbook. However, the mean values of questions related to Habit (H), though close to 3, are observed to be lesser than other factors, indicating the need for improvements in *TAnnotator* towards being regularly used by users, and consequently be put into practice.

This analysis of usefulness through the user survey based evaluation answers RQ2, by indicating that majority of the users consider annotating an e-textbook with additional information to be useful. However, the analysis also indicates that *TAnnotator* could be further improved to support users in making it a habit to use *TAnnotator* frequently.

Participants have also suggested improving *TAnnotator* to support multiple e- textbooks. Some suggestions from users include:

- "A code editor could be integrated for better usage of code examples"
- "More facts along with source from which they are extracted could be useful"

The correlation among the variables from UTAUT2 model considered for evaluation of *TAnnotator* are presented in Table 2. We calculated the correlation of UTAUT2 variables using the following formula, where r: correlation coefficient, x_i: values of the x-variable in a sample, $\bar{x}$: mean of the values of the x-variable, y_i: values of the y-variable in a sample and $\bar{y}$: mean of the values of the y-variable, and x and y correspond to UTAUT2 variables.

$$r = \frac{\sum (x_i - \bar{x})(yi - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

The values of correlation among the variables displayed in Table 2 indicate positive correlation among all the factors. The strongest correlation (0.638) is observed among Social Influence (SI) and Facilitating Conditions (FC), indicating that the users' facilitating conditions to use *TAnnotator* are strongly affected by Social Influence on the user. Also, users' behavioral intention (BI) to use *TAnnotator* is observed to be strongly influenced by the extent to which *TAnnotator* contributes to users' habit (H), than any other factor.

Discussion and limitations

TAnnotator currently demonstrates the idea of annotating e-textbooks with facts and examples through only one e-textbook- *The C++ Language Tutorial*. However, it could be extended with minimal technical effort to any other e-textbook in the programming domain, based on the e-textbooks' policies. Extension of *TAnnotator* to other e-textbooks of any other domains requires identification of active discussion forums that could be used as sources for fetching facts about the concepts being discussed in the textbook.

Currently, facts and examples are only being extracted from three sources. Increasing the number of sources could further improve the number of facts being presented to the readers and could consequently contribute to increased knowledge base for the readers. Moreover, the topics for which facts and examples are being fetched is based on the keywords obtained through LDA and frequent word identification, when specific paragraphs are processed. Hence, the LDA model used influences the accuracy of topics obtained as keywords to represent topic of discussion in the textbook. Moreover, the accuracy of topics depends on efficiency of the LDA model, and these topics might sometimes be irrelevant. The current version being a prototype version, we only included 6 pages of the textbook. We can add more pages in the current portal before making it available to be used by a wider range of audience. The facts and examples augmented to the e-textbook using *TAnnotator* could be aligned more attractively to further improve readers' experience.

TAnnotator displays additional information only in extra margin adjacent to the text, and does not disturb the inherent flow of content in the original e-textbook. However, there is need for qualitative assessment towards determining the position of annotations. An expert in the field corresponding to topics of discussion in the e-textbook could be consulted to validate the annotated information and the information that could disturb the didactic concepts could be omitted. This exercise could be conducted for multiple annotations and the actions could be drafted, which could help in training machine learning model that could help in validating the annotations. This could reduce the need for human interference, while facilitating validation of the annotations.

We demonstrated the annotation of one programming-based e-textbook using *TAnnotator* in this work. This could be applicable and adapted to other similar e-textbooks, but is limited to the associated permissions. The e-textbook under consideration should allow permissions to extract the data present in it, and the relevant sources corresponding to this e-textbook should allow permissions to extract and reuse the data. However, the sources and the e-textbook considered for demonstration in this work permit data to be extracted and reused.

Also, the keywords in a specific page are being identified only once to ensure faster response of the portal. Approaches to dynamically identify keywords every time a page loads, without compromising on the response time of the portal could be explored. This could further improve the knowledge base of users by displaying newer facts frequently. The evaluation performed largely focuses on the user experience and usability of *TAnnotator*, but does not focus on the learning outcome using *TAnnotator*. However, as programming courses differ with respect to their goals across various institutions, programmer age groups, educational and social backgrounds, it is difficult to arrive at common learning goals, and hence difficult to assess the learning outcome.

Conclusion and future work

We presented the idea of annotating online textbooks, specifically in the area of computer programming, with facts and examples in this paper. We demonstrated this idea by presenting a prototype version of *TAnnotator*, a web portal that displays *tooltips* and corresponding example code snippets adjacent to the text in the e-textbook. These *tooltips* are displayed based on the context of the topics being discussed in the textbook. The *tooltips* and examples are extracted from developer forums and blogs. We have evaluated the user experience and usefulness of *TAnnotator*, through a user survey based on UTAUT2 technology acceptance model. The results of the survey indicate that majority of the users appreciated the idea of annotating e-textbooks in the programming domain and that they were willing to use *TAnnotator*. It can be used to learn about the latest trends and specific *tooltips* for multiple topics being discussed in the e-textbooks. It could be useful to the software developers in the industry to be up-to-date about multiple facts in their domain. It could also help students in getting a better idea about the topics being discussed in the textbooks.

We plan to increase the number of sources being used for tooltip and example retrieval based on the content of the textbook. The responses of the survey indicate lesser inclination towards using *TAnnotator* on a day-to-day basis. The suggestions of the users also indicate need for improvement in the number of facts annotated and the addition of other features. We plan explore multiple sources by considering the participants' expertise and further explore ways to incorporate the participants' suggestions. We plan to extend *TAnnotator* to support more number of freely available e-textbooks in the programming domain. Also, we intend to identify appropriate rich sources of information for each of the textbooks that would be supported by *TAnnotator*. We also plan to integrate an automated question and answer mechanism, capable of answering questions of the users, to the current version. User activity on the portal could be tracked, with user permissions, and *tooltips* based on the activity could be provided in the future versions, thus moving towards a personalised textbook annotator.

Acknowledgements
We thank all the participants for their valuable time and honest feedback that helped us in evaluating TAnnotator. We would also like to thank Shruti Priya and Shubhankar for helping us with the development of TAnnotator.

Author contributions
AV has contributed more in terms of implementation of the idea, and SC has contributed more in terms of the idea. All authors read and approved the final manuscript.

Funding
Not applicable.

Availability of data and materials
The tool and the results of evaluation are available from the corresponding author on reasonable request.

Declaration

Competing interests
There are no competing interests.

Received: 1 August 2022 Accepted: 17 January 2023
Published online: 25 January 2023

References

Almansoori, M., Lam, J., Fang, E., Soosai Raj, A. G., & Chatterjee, R. (2021). Textbook underflow: Insufficient security discussions in textbooks used for computer systems courses. In *Proceedings of the 52nd ACM technical symposium on computer science education* (pp. 1212–1218).

Almiyad, M. A., Oakden-Rayner, L., Weerasinghe, A., & Billinghurst, M. (2017). Intelligent augmented reality tutoring for physical tasks with medical professionals. In *International conference on artificial intelligence in education* (pp. 450–454).

Bailey, J., & Zilles, C. (2019). uassign: Scalable interactive activities for teaching the unix terminal. In *Proceedings of the 50th ACM technical symposium on computer science education* (pp. 70–76).

Benotti, L., Aloi, F., Bulgarelli, F., & Gomez, M. J. (2018). The effect of a web-based coding tool with automatic feedback on students' performance and perceptions. In *Proceedings of the 49th ACM technical symposium on computer science education* (pp. 2–7).

Berns, C., Chin, G., Savitz, J., Kiesling, J., & Martin, F. (2019). Myr: A web-based platform for teaching coding using VR. In *Proceedings of the 50th ACM technical symposium on computer science education* (pp. 77–83).

Chang, C.-C., & Chen, T.-C. (2022). Emotion, cognitive load and learning achievement of students using e-textbooks with/without emotional design and paper textbooks. *Interactive Learning Environments*, 1–19.

Choi, H.-J., & Park, C. H. (2019). Emerging topic detection in twitter stream based on high utility pattern mining. *Expert Systems with Applications, 115*, 27–36.

Davis, R. C., & Song, X. (2020). Uncovering the mystery of how users find and use ebooks through guerilla usability testing. *Serials Review, 46,* 1–8.

Dutta, R., Mantri, A., & Singh, G. (2022). Evaluating system usability of mobile augmented reality application for teaching Karnaugh-maps. *Smart Learning Environments, 9*(1), 1–27.

Ellis, M., Shaffer, C. A., & Edwards, S. H. (2019). Approaches for coordinating etextbooks, online programming practice, automated grading, and more into one course. In *Proceedings of the 50th ACM technical symposium on computer science education* (pp. 126–132).

Ericson, B. J., Guzdial, M. J., & Morrison, B. B. (2015). Analysis of interactive features designed to enhance learning in an ebook. In *Proceedings of the eleventh annual international conference on international computing education research* (pp. 169–178).

Fischer, L., Hilton, J., Robinson, T. J., & Wiley, D. A. (2015). A multi-institutional study of the impact of open textbook adoption on the learning outcomes of post- secondary students. *Journal of Computing in Higher Education, 27*(3), 159–172.

Gerjets, P., Scheiter, K., & Catrambone, R. (2006). Can learning from molar and modular worked examples be enhanced by providing instructional explanations and prompting self-explanations? *Learning and Instruction, 16*(2), 104–121.

Ghosh, K. (2022). Remediating textbook deficiencies by leveraging community question answers. *Education and Information Technologies*, 1–41.

Harsley, R., Di Eugenio, B., Green, N., & Fossati, D. (2017). Enhancing an intelligent tutoring system to support student collaboration: Effects on learning and behavior. In *International conference on artificial intelligence in education* (pp. 519–522).

Holstein, K., McLaren, B. M., & Aleven, V. (2018). Student learning benefits of a mixed-reality teacher awareness tool in AI-enhanced classrooms. In *International conference on artificial intelligence in education* (pp. 154–168).

Hori, M., Ono, S., Kobayashi, S., Yamaji, K., Kita, T., & Yamada, T. (2015). Learner autonomy through the adoption of open educational resources using social network services and multi-media e-textbooks. *AAOU Journal, 10*(1), 23.

Kao, G.Y.-M., & Ruan, C.-A. (2022). Designing and evaluating a high interactive augmented reality system for programming learning. *Computers in Human Behavior, 132*, 107245.

Keuning, H., Heeren, B., & Jeuring, J. (2021). A tutoring system to learn code refactoring. In *Proceedings of the 52nd ACM technical symposium on computer science education* (pp. 562–568). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3408877.3432526.

Kim, H. J., Park, J. H., Yoo, S., & Kim, H. (2016). Fostering creativity in tablet-based interactive classrooms. *Journal of Educational Technology & Society, 19*(3), 207–220.

Krusche, S., & Seitz, A. (2018). Artemis: An automatic assessment management system for interactive learning. In *Proceedings of the 49th ACM technical symposium on computer science education* (pp. 284–289).

Lee, H. J., Messom, C., & Yau, K.-L.A. (2013). Can an electronic textbooks be part of k-12 education?: Challenges, technological solutions and open issues. *Turkish Online Journal of Educational Technology-TOJET, 12*(1), 32–44.

Ma, B., Lu, M., Taniguchi, Y., & Konomi, S. (2022). Exploring jump back behavior patterns and reasons in e-book system. *Smart Learning Environments, 9*(1), 1–23.

Mazumder, S. F., Latulipe, C., & P´erez-Quin˜ones, M. A. (2020). Are variable, array and object diagrams in java textbooks explanative? In *Proceedings of the 2020 ACM conference on innovation and technology in computer science education* (pp. 425–431).
McGowan, M. K., Stephens, P. R., & West, C. (2009). Student perceptions of electronic textbooks. *Issues in Information Systems, 10*(2), 459–465.
Miller, B. N., & Ranum, D. L. (2012). Beyond pdf and epub: Toward an interactive textbook. In *Proceedings of the 17th ACM annual conference on innovation and technology in computer science education* (pp. 150–155).
Mystakidis, S., Christopoulos, A., & Pellas, N. (2021). A systematic mapping review of augmented reality applications to support stem learning in higher education. *Education and Information Technologies, 27*, 1–45.
Oates, T. (2014). *Why textbooks count*. Cambridge: Cambridge Assessment.
Onan, A., Korukoglu, S., & Bulut, H. (2016). LDA-based topic modelling in text sentiment classification: An empirical analysis. *International Journal of Linguistic Applications, 7*(1), 101–119.
Pletea, D., Vasilescu, B., & Serebrenik, A. (2014). Security and emotion: Sentiment analysis of security discussions on github. In *Proceedings of the 11th working conference on mining software repositories* (pp. 348–351).
Plisson, J., Lavrac, N., Mladenic, D., et al. (2004). A rule based approach to word lemmatization. In *Proceedings of is* (Vol. 3, pp. 83–86).
Porter, M. F., et al. (1980). An algorithm for suffix stripping. *Program, 14*(3), 130–137.
Reinhardt, A., Zhang, T., Mathur, M., & Kim, M. (2018). Augmenting stack overflow with API usage patterns mined from github. In *Proceedings of the 2018 26th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering* (pp. 880–883).
Rockinson-Szapkiw, A. J., Courduff, J., Carter, K., & Bennett, D. (2013). Electronic versus traditional print textbooks: A comparison study on the influence of university students' learning. *Computers & Education, 63*, 259–266.
Rondan-Cataluña, F. J., Arenas-Gaitán, J., & Ramírez-Correa, P. E. (2015). A comparison of the different versions of popular technology acceptance models. *Kybernetes, 44*(5), 788–805.
Seeling, P. (2016). Switching to blended: Effects of replacing the textbook with the browser in an introductory computer programming course. In *2016 IEEE frontiers in education conference (fie)* (pp. 1–5).
Solcova, L. (2016). Interactive textbook–a new tool in off-line and on-line education. *Turkish Online Journal of Educational Technology-TOJET, 15*(3), 111–125.
Sun, J., Flores, J., & Tanguma, J. (2012). E-textbooks and students' learning experiences. *Decision Sciences Journal of Innovative Education, 10*(1), 63–77.
Tang, K.-Y. (2021). Paradigm shifts in e-book-supported learning: Evidence from the web of science using a co-citation network analysis with an education focus (2010–2019). *Computers & Education, 175*, 104323.
Venigalla, A. S. M., & Chimalakonda, S. (2020). Dynamique–a technical intervention to augment static textbook with dynamic q&a. *Interactive Learning Environments, 30*, 1–15.
Venigalla, A. S. M., Lakkundi, C. S., Agrahari, V., & Chimalakonda, S. (2019). Stackdoc-a stack overflow plug-in for novice programmers that integrates q&a with API examples. In *2019 IEEE 19th international conference on advanced learning technologies (ICALT)* (Vol. 2161, pp. 247–251).
Venkatesh, V., Morris, M. G., Davis, G. B., & Davis, F. D. (2003). User acceptance of information technology: Toward a unified view. *MIS Quarterly, 27*, 425–478.
Venkatesh, V., Thong, J. Y., & Xu, X. (2012). Consumer acceptance and use of information technology: Extending the unified theory of acceptance and use of technology. *MIS Quarterly, 36*, 157–178.
Vinaja, R. (2014). The use of lecture videos, ebooks, and clickers in computer courses. *Journal of Computing Sciences in Colleges, 30*(2), 23–32.
Weber, G., & Brusilovsky, P. (2016). Elm-art–an interactive and intelligent web-based electronic textbook. *International Journal of Artificial Intelligence in Education, 26*(1), 72–81.
Weng, C., Otanga, S., Weng, A., & Cox, J. (2018). Effects of interactivity in e- textbooks on 7th graders science learning and cognitive load. *Computers & Education, 120*, 172–184.
Zavala, L., & Mendoza, B. (2018). On the use of semantic-based AIG to automatically generate programming exercises. In *Proceedings of the 49th ACM technical symposium on computer science education* (pp. 14–19).

Publisher's Note