Towards Authentic Undergraduate Research Experiences in Software Engineering and Machine Learning

Suranjan Chakraborty  
Department of Computer and Information Sciences  
Towson University  
Towson, Maryland, USA  
schakraborty@towson.edu

Lin Deng  
Department of Computer and Information Sciences  
Towson University  
Towson, Maryland, USA  
lind@towson.edu

Josh Dehlinger  
Department of Computer and Information Sciences  
Towson University  
Towson, Maryland, USA  
jdehlinger@towson.edu

ABSTRACT

Authentic undergraduate research experiences have been shown to be very effective at sustaining students’ learning motivation and enhancing students’ theoretical knowledge and practical skills. However, there still exist some common challenges in undergraduate research. In this paper, we describe an approach that offers undergraduate students authentic and immersive research experience focusing on applied machine learning for software engineering and discuss our experiences with example undergraduate research projects and outcomes. A survey was designed to assess students’ overall experience of participating in authentic undergraduate research projects in machine learning for software engineering. Preliminary results from this survey are provided.

CCS CONCEPTS
• Computing methodologies ~ Machine learning  • Software and its engineering  • Applied computing ~ Education

KEYWORDS
Machine learning, software engineering, computing education, undergraduate research

ACM Reference format:

1 Introduction

For decades, software engineering (SE) and machine learning (ML) educators have kept reinventing and improving their courses to keep up with the pace of rapid advancement of information technology. As with most of Computer Science, topics in SE and ML have significantly changed in the last ten years. Such change is constant, so educators need to aim to not only sustain students’ learning motivation, ensure students attain necessary technical and problem-solving skills and experience, but also provide them with skills to keep up with the constant changes in the disciplines. This is essential for continued success in their future professional career.

However, traditional computer science education still relies on textbooks and focuses on lectures, which sometimes becomes less motivating to students compared to project-based learning approaches. This often leads to students feeling that they can learn more usefully from online platforms than from the traditional classroom. This issue has often been dealt with creation of courses that leverage project-driven pedagogical approaches to motivate students and engage them with in-demand techniques and skills.

Yet, there are still two critical issues. Project-based learning courses usually run only one semester, which is too short for students to complete a moderate size project, given the time and effort that students must spend on learning new techniques and skills. Second, blurring of boundaries within the various sub-areas of Computer Science has become the norm, especially in SE and ML fields. For example, huge software requirements documentation written in natural language are analyzed and classified through ML models [1,3], countless test inputs are automatically generated by autonomous agents [5], hidden threats and security vulnerabilities buried in program source code are automatically detected by neural networks [9], and new powerful ML techniques are employed to help solve challenging problems in the domain of SE. This requires rich experience and profound understanding and knowledge of both SE and ML. All of these are difficult to teach even with project-based learning pedagogical approaches within the timeframe of a single semester.

Research projects, thesis and dissertation work provide a mechanism to resolve the aforementioned issues, through contextualizing cross-area research problems, practicing students’ problem-solving skills, and enrich their technical experience on latest technology. However, unlike graduate programs, most undergraduate computing programs often do not prioritize authentic undergraduate research experiences.

In this paper, we report an approach that offers undergraduate students authentic and immersive research experience in SE and
ML and describe our experiences, including example undergraduate research projects. We employ different cross-area research projects in SE and ML, to demonstrate and contextualize knowledge, concepts, and techniques. We utilize an agile engineering approach to complement students’ learning styles and engage students in attaining latest technology, algorithms, and models. Specifically, students look for challenging research questions, propose hypotheses, solutions, design empirical studies, apply research methodologies and statistical analysis that are typically taught in graduate schools.

The contributions of this paper include:

- We develop a strongly mentored, undergraduate research approach that provides an authentic and immersive undergraduate research experience, aiming at developing highly qualified, computing professionals.
- This approach increases the opportunities of undergraduate computing students participating in out-of-classroom research activities involving applied ML for SE problems to bolster their professional development.
- This out-of-classroom undergraduate research approach is highly customized to individual students’ interests and catered to their own future career goals.
- This work enhances students’ ability to plan, organize, and carry out authentic scientific research, and provides a basis for educational programs to offer undergraduate students research opportunities that bridge theory and knowledge to practice by focusing on real-world challenging problems.
- This work develops additional out-of-classroom opportunities to attract, retain and guide students in undergraduate programs.

2 Background and Related Work

Incorporating research in undergraduate education has frequently been recommended [2, 8, 11]. Existing scholarly work suggests that systematic exposure to research is necessary for students to develop greater competence in science, engineering, mathematics, and technology [2]. In addition, participation in programmatic research projects provide an excellent setting for undergraduate students to develop and practice teamwork, complex problem solving, and effective communication - skills considered essential for the real-world work place [2]. Russell et al. [8] report the results of an extensive survey of STEM students conducted between 2003-2005, which suggests that such authentic undergraduate research opportunities lead to: a) increased confidence, awareness and understanding related to the students’ STEM discipline; b) increased clarity about the conduct of research and development of research skills; c) greater understanding of the expectations of graduate school; d) an inclination to pursue graduate education; and, d) increased interest in pursuing and sustaining a career in STEM. Thus, authentic undergraduate research opportunities not only enhance skills but also have the potential to serve as critical pipeline for the development of STEM researchers from the undergraduate pool of students. However, traditional CS undergraduate research approaches have some common challenges: a) providing students breadth of knowledge and domain expertise needed to do research; and, b) lack of time needed to complete meaningful research [2, 10, 12]. Thus, there is a need to develop systematic undergraduate research opportunities, such as the one described in the paper, that provide authentic and immersive undergraduate research experience.

3 Approach Towards Authentic Undergraduate Research Experience in SE and ML

Since 2018, the authors established the Towson University (TU) Intelligent Software Engineering Research Group, aiming to engage more undergraduate computing students in SE and ML research activities and offer them authentic undergraduate research experience. The general approach is phased as follows:

1. Student Recruitment Phase
   - **Marketing and outreach stage.** We created a website, documents, posters, and handouts that clearly describe the research opportunities. After that, we hosted a webinar that introduces the Intelligent Software Engineering Research Group initiatives and specific details and advertised in campus email lists. We established a presence to distribute marketing materials at TU events and visit classes.
   - **Interview and selection stage.** Prospective students submit a resume and a brief statement of purpose describing goals and motivations to participate in research activities in SE and ML. Then, students are invited for an interview. During the interview, professors assess each applicant’s ability and research interests.

2. Training and Research Awareness Phase
   - **Knowledge and training stage.** In this stage, we provide students with necessary knowledge, concepts, and background information. Students are mentored by the professors in our group and learn conceptual knowledge of SE and selected ML approaches (e.g., topic modeling, artificial neural networks, etc.). Then, students practice the implementation of such ML models and approaches.
   - **Research awareness stage.** This stage helps students understand what research is. We help students increase their awareness and acquaintance of different research methodologies (e.g., empirical evaluation approaches, statistical analysis, hypothetical analysis, etc.) related to applying techniques to research problem domains in SE and ML. Students learn how to search for related literature in different research publication databases, read and criticize papers, and use appropriate formats of citations. Much of this stage is accomplished by directing to students through directed reading of related research papers and getting them to discuss their learning from such research.

3. Research Apprentice Phase
   - **Research topic matching stage.** Professors provide students with research topics related to applied ML in key areas of SE. After that, students choose their preferred topics based on their individual interests and experience.
• **Proposal stage.** Students search for related literature and conduct literature review. Through a 40-minute seminar presentation, students present literature review, propose their own research ideas and design necessary experimental studies. After the presentation, students are required to answer questions from fellow students and professors. All students in the group are required to provide insights, comments, and constructive feedback.

• **Investigation stage.** Following their proposal, students empirically investigate their research projects. We employ an agile engineering style approach to manage each student’s weekly progress by requiring them to report: a) what has been done since last meeting; b) what is the plan before next meeting; and, c) what are the difficulties.

4. Research Dissemination Phase

• **Project dissemination stage.** After collecting sufficient experimental results, students identify publication outlets and develop research manuscripts ready for submission. Also, students are required to make a final presentation to report their findings and future possible research directions, so that other students may take over.

4 Example Projects

TU is the second largest public university in Maryland, with a strong emphasis on undergraduate education. The student body at TU is diverse; for example, the incoming freshman class for 2019 is made up of 48% identifying as a minority. Since the establishment in 2018, hundreds of students expressed their interests in working with our Intelligent Software Engineering Research Group. Due to the limited resources, we have mentored 20 undergraduate students. Eight of them have co-authored at least one research publication. Projects are chosen based on learning outcomes, technical contributions, and latest research trends. This section introduces three representative projects.

4.1 Project 1: Automatic Non-Functional Software Requirements Classification Using Neural Networks

In SE, requirements engineering is a critical step that significantly impacts the success of a software system. Software requirements can be classified into functional requirements (FRs) and non-functional requirements (NFRs). FRs define services, while NFRs define all kinds of constraints. Usually, the elicitation, specification, classification, and documentation of requirements are very time-consuming, labor intensive and error prone, which involve a large amount of manual work from analysts and engineers.

This project applies ML techniques, such as neural networks, to achieve automated classification for NFRs. We constructed different neural network models, such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), and Deep Neural Network (DNN), used for Natural Language Processing (NLP). Then, we trained the neural network models with the training data that include NFRs in different classes (e.g., usability, security, performance, and operational) so that the models can “understand” different types of NFRs. After that, we evaluate the performance of each model in terms of classifying NFRs.

**Outcomes:** Five undergraduate students have participated this project. Two research papers [1, 4], with undergraduate students as primary authors, have been published in international conferences.

4.2 Project 2: Automatic Equivalent Mutants Classification Using Neural Networks

Mutation testing has been found to be one of the most effective software testing techniques that can help software developers and testers identify software bugs, design powerful test cases, and deliver high quality software systems. However, the cost of mutation testing is expensive, in terms of a large amount of testing requirements (mutants). For example, testers need to manually identify and eliminate equivalent mutants (i.e., the mutants that can never be distinguished by any test cases), which requires domain knowledge and is very time consuming and labor intensive.

This project applies ML techniques, such as neural networks, to achieve automated identification for equivalent mutants. We construct a neural network model can learn from program source code and other related programming features. Then, we train the model with existing equivalent mutants. After that, the trained neural network model is expected to help developers and testers to identify equivalent mutants from a pool of software mutants.

**Outcomes:** Two undergraduate students, as primary authors, have participated this project. One research paper [7] has been published in international conference.

4.3 Project 3: Automatic Identification of Vulnerable code using Neural Networks

Security vulnerabilities are prevalent in software systems, increasingly threatening every aspect of cyberspace. Identifying these vulnerabilities and defects requires domain expertise and is very time consuming and labor intensive. Many tools have been designed to help analyze software systems and to find vulnerabilities and defects. However, the vulnerabilities that are not caught by these tools are usually too complicated to find, do not fall inside of an existing ruleset for identification, or are hidden well.

This project investigates and develops a system, based on ML techniques, such as neural networks, which is trained with known security vulnerabilities. Then, the system is expected to automatically help software developers, testers, and cybersecurity analysts discover defects and vulnerabilities in software.

**Outcomes:** Three undergraduate students have participated this project. Two research papers [6, 9] have been published in international conferences.

5 Evaluation

To initially evaluate students’ experience, students were asked to anonymously and voluntarily complete surveys, based on a modified version of the evaluation surveys from [3], upon completion of their time with the project to assess their perceived experience in ML and SE research projects.
Table 1. Partial Student Experience Survey Results (n = 20)

<table>
<thead>
<tr>
<th>Survey Question</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>I can use a research process to pursue a software engineering research project.</td>
<td>4.55</td>
</tr>
<tr>
<td>I am sure that I can actively participate in a software engineering research project.</td>
<td>4.70</td>
</tr>
<tr>
<td>I have gained confidence in collaborating with research professionals.</td>
<td>4.60</td>
</tr>
<tr>
<td>Working on a machine learning for software engineering research project gives me a better appreciation for the usefulness of machine learning.</td>
<td>4.84</td>
</tr>
<tr>
<td>I have a greater awareness of the potential for machine learning to benefit society due to working on a machine learning software engineering research project.</td>
<td>4.84</td>
</tr>
<tr>
<td>Working with the machine learning community to develop this research project has increased my interest in research.</td>
<td>4.05</td>
</tr>
<tr>
<td>I plan to continue contributing to a machine learning project after this course/experience has ended.</td>
<td>3.50</td>
</tr>
<tr>
<td>Participation in a machine learning/software engineering research project has caused me to consider graduate/further education.</td>
<td>4.00</td>
</tr>
<tr>
<td>The subject matter of this machine learning/software engineering research project is highly relevant to my future career plans.</td>
<td>4.26</td>
</tr>
<tr>
<td>Overall, I am very satisfied with my learning in the machine learning/software engineering research project.</td>
<td>4.75</td>
</tr>
</tbody>
</table>

Table 1 provides representative average of the survey results presented on a scale of 1-5 with 1 equating to Strongly Disagree and 5 equating to Strongly Agree. Overall, students responded positively to the undergraduate research experience, as shown in the quantitative scores shown in Table 1. Qualitative feedback reinforces these results as students generally appreciated the research opportunity and the ability to apply new technologies to a research project. Representative qualitative comments include:

- "Having the research experience on my resume definitely helped me get my foot in the door for interviews. It’s a nice thing to have to distinguish yourself from other graduates."
- "I’ve done some other research projects, but none such that I immediately used the findings I found. Setting small goals of "Set out to learn more about x, and then use that information to test y each week really increased my productivity and drive to learn more about our topic."

These survey questions help us collect feedback on our proposed approach in providing undergraduate students authentic research experience in SE and ML and help us reflect on the design and experience of the approach, so that we can improve in the future.

6 Conclusion and Future Work

The exploration and investigation of a feasible, effective, and comprehensive approach for providing undergraduate students with authentic and immersive research experience are vital to modern higher education practitioners, because it might be many students’ first exposure to scientific research. Simply stated, a well-designed research experience approach is eagerly sought to offer undergraduate students valuable research experience by developing research methodologies, proposing research ideas, carrying out research plans, and presenting research findings.

In this paper, we introduce the approach that offers undergraduate students authentic and immersive research experience focusing on applied ML for SE and describe our experiences with example research projects and outcomes. The approach enables undergraduate students to become computing professionals with advanced technical skills and enhances students’ theoretical and practical understanding of SE and ML. Participating students benefit from focused mentoring, authentic undergraduate research experience at a critical stage in their undergraduate education that prepares them for pursuing graduate studies and/or entry into the workforce. For future work, we would like to continue improving our approach based on students’ feedback and seek additional resources and opportunities to extend and institutionalize our approach into a well-developed program.

ACKNOWLEDGMENTS

This project is supported by the General Endowment Funds from Fisher College of Science and Mathematics at TU. Lin Deng is supported as a Jess and Mildred Fisher Endowed Professor at TU.

REFERENCES