An NSF REU Site Based on Trust and Reproducibility of Intelligent Computation: Experience Report

Mary Hall
University of Utah
Salt Lake City, Utah, USA
mhall@cs.utah.edu

Ganesh Gopalakrishnan
University of Utah
Salt Lake City, Utah, USA
ganesh@cs.utah.edu

Eric Eide
University of Utah
Salt Lake City, Utah, USA
eeide@cs.utah.edu

Johanna Cohoon
University of Utah
Salt Lake City, Utah, USA
hannah.cohoon@utah.edu

Jeff M. Phillips
University of Utah
Salt Lake City, Utah, USA
jeffp@cs.utah.edu

Mu Zhang
University of Utah
Salt Lake City, Utah, USA
muzhang@cs.utah.edu

Shireen Y. Elhabian
University of Utah
Salt Lake City, Utah, USA
shireen@sci.utah.edu

Aditya Bhaskara
University of Utah
Salt Lake City, Utah, USA
bhaskara@cs.utah.edu

Harvey Dam
University of Utah
Salt Lake City, Utah, USA
harvey.dam@utah.edu

Artem Yadrov
University of Utah
Salt Lake City, Utah, USA
artemyadrov87@gmail.com

Tushar Kataria
University of Utah
Salt Lake City, Utah, USA
tushar9818@gmail.com

Amir Mohammad Tavakkoli
University of Utah
Salt Lake City, Utah, USA
tavak@cs.utah.edu

Sameeran Joshi
University of Utah
Salt Lake City, Utah, USA
joshisameeran17@gmail.com

Mokshagna Sai Teja Karanam
University of Utah
Salt Lake City, Utah, USA
kmokshagna999@gmail.com

ABSTRACT

This paper presents an overview of an NSF Research Experience for Undergraduate (REU) Site on Trust and Reproducibility of Intelligent Computation, delivered by faculty and graduate students in the Kahlert School of Computing at University of Utah. The chosen themes bring together several concerns for the future in producing computational results that can be trusted: secure, reproducible, based on sound algorithmic foundations, and developed in the context of ethical considerations. The research areas represented by student projects include machine learning, high-performance computing, algorithms and applications, computer security, data science, and human-centered computing. In the first four weeks of the program, the entire student cohort spent their mornings in lessons from experts in these crosscutting topics, and used one-of-a-kind research platforms operated by the University of Utah, namely NSF-funded CloudLab and POWDER facilities; reading assignments, quizzes, and hands-on exercises reinforced the lessons. In the subsequent five weeks, lectures were less frequent, as students branched into small groups to develop their research projects. The final week focused on a poster presentation and final report. Through describing our experiences, this program can serve as a model for preparing a future workforce to integrate machine learning into trustworthy and reproducible applications.

SC-W 2023, November 12–17, 2023, Denver, CO, USA
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0785-8/23/11.
https://doi.org/10.1145/3624062.3624100

CCS CONCEPTS

• Social and professional topics → Model curricula; **• Computing methodologies → Machine learning algorithms**; **Machine learning**; **• Human-centered computing → Ubiquitous and mobile computing**; **• Networks → Wireless access points, base stations and infrastructure**; **• Security and privacy**;

KEYWORDS

undergraduate education, artifact evaluation, HPC, ML, networking

ACM Reference Format:
Mary Hall, Ganesh Gopalakrishnan, Eric Eide, Johanna Cohoon, Jeff M. Phillips, Mu Zhang, Shireen Y. Elhabian, Aditya Bhaskara, Harvey Dam, Artem Yadrov, Tushar Kataria, Amir Mohammad Tavakkoli, Sameeran Joshi, and Mokshagna Sai Teja Karanam. 2023. An NSF REU Site Based on Trust and Reproducibility of Intelligent Computation: Experience Report. In *Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis (SC-W 2023), November 12–17, 2023, Denver, CO, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3624062.3624100

1 INTRODUCTION

Life in modern society is increasingly dependent on trustworthy components that work as expected, such as verified arithmetic libraries that form the bedrock of climate simulation codes typically executed on high-performance computing (HPC) machines. Scientists and engineers conduct their day-to-day work based on such components, and also on existing scientific results (e.g., prior studies on trade wind flows) to deliver *newer* trustworthy components and scientific results (e.g., prediction of where forest-fire-carried smoke will next land). Other examples of trust include face-recognition systems in airports that are based on artificial intelligence (AI) and machine learning (ML): people trust digital face recognition systems to classify images without bias (e.g., against skin tones), respect privacy (e.g., not to leak the images), and be intrusion-proof (have sufficient network security). Another aspect of enhancing trust is through community-level debugging, where issues using the Git tool are posted against software components found inadequate, and enhancements are made to benefit all users.

Trust *fundamentally depends on reproducibility.* A person must be able to take an existing scientific result or a pre-existing software component, test it, and see if they can reproduce the published specifications or claims. Since results are reproducible only when the exact setup conditions are obeyed, practices and habits that promote reproducibility—such as the use of Jupyter Notebook tool—must become ingrained into common practice.

The TREU REU site at the Kahlert School of Computing, University of Utah was designed with the goals set forth thus far. By the time our REU funds arrived (end of February), we advertised our site, vigorously promoting the opportunity in multiple ways, with a closing deadline of April 15.

We were surprised at the encouraging response rate (total 85 applicants received for 10 positions) and started making offers soon after the closing date. The offers were imparted a personal touch through a Zoom call followed by a formal offer letter. A few aspiring local undergraduates from Utah were also added (using other REU supplements) to make the experience wholesome for the group. The offers were slanted toward institutions without an established research program, and emphasized gender and ethnic diversity. The external students selected were spread more or less evenly between sophomores and juniors, with many having taken basic computer science classes, and with most of them having their first such REU site experience.

What now follows is a description of the student projects, intended to convey what the students could accomplish after about five weeks of lessons. This paper does not focus on any particular topic (e.g., parallelism); instead, it is oriented at how a blend of areas were taught in the context of supporting intelligent computation (i.e., AI and ML). It is to be emphasized that *the success would have been far less had we not engaged the help of our very capable PhD students and a postdoctoral researcher.* We conclude the paper with an assessment of the outcomes using our surveys.

2 STUDENT PROJECTS

We now describe the student projects, listing their goals, concepts involved, and experiments performed, including whether the computational platforms—GPUs in particular—were a bottleneck (especially closer to project finish). We provide qualitative summary assessments in each section.

2.1 Artifact Evaluation Work and Challenges

Goal: Students were to help prepare an IRB-approved study of conferences' artifact evaluation processes and gain practice in human-centered computing research methods. The study design was completed before the REU began. Involved students were to pilot study materials (i.e., diary study questions and interview protocols) and refine them prior to data collection beginning.

Concepts involved: This project related to human-centered computing, reproducibility and sociotechnical factors affecting its achievement, diary studies, semi-structured interviews, trace data collection, pilot studies, and data triangulation.

Experiments conducted, platforms used: Students participated in four pilot sessions and collected feedback on the study materials' clarity and comprehensiveness. Materials were designed to capture data on how reviewers evaluate research artifacts' reproducibility and challenges reviewers face. Diary studies were piloted using Qualtrics. Interviews were conducted over Zoom. Students substantially revised the materials, improving their validity and utility.

Overall assessment: Through assigned readings and engagement in the research process, students gained an appreciation for the sociotechnical factors that affect reproducibility (e.g., time to create an artifact, reward for such work, available instructions and infrastructure). Students gained practice in conducting and scheduling interviews. Attempts to use third-party packages to collect trace data from artifact repositories were unsuccessful. However, students did gain practice in communicating with package developers and troubleshooting. Piloting revealed that authors conceive of research artifacts as distinct from the documentation that explains them; to computational researchers, artifacts are code.

2.2 Particle Filters for Event Location

Goal: Particle filters are often used to estimate the position of an object in an environment given a map of its features and (imperfect) sensor readings. Usual implementations of particle filters require environment features to be repeatedly observable, and we sought ways around this limitation. The case study involved locating events in a musical concert.

Concepts involved: Our particle filter used ideas from reinforcement learning, positional encoding layers, and attention layers. Its implementation involved PyTorch Tensor operations on the GPU.

Experiments conducted, platforms used: We performed accuracy and time experiments comparing our particle filter with the typical particle filter. We used a variety of machines from laptops to desktops, with and without GPUs.

Overall assessment: We developed a particle filter technique that can be used to estimate the temporal location of a sequence of distinct events that approximately follows an expected schedule. In addition, we developed a fast weighting function that, according to our experiments, is much faster and almost as accurate as the typical Gaussian weighting function, which may be preferred in applications that demand low latency or frequent updates. The student made meaningful, non-trivial contributions to our research. We intend to submit this to a conference such as ICLR.

2.3 Machine Unlearning

Goal: While the primary goal of a machine learning system is making inferences from data, we are sometimes required (e.g. for legal reasons) to have a model that "forgets" certain ideas, such as certain classes. However, there are no techniques that we could find for making a model behave as if it had never been trained on certain data, besides completely retraining a model from scratch without that data. We sought such a technique.

Concepts involved: This project involved general deep learning concepts and experiments were written in PyTorch.

Experiments conducted, platforms used: The student who participated in this project participated in other projects as well. For this reason, we did not have time to prepare experiments using larger datasets, so GPU availability was not a bottleneck. Experiments were performed on a desktop machine with a single GPU.

Overall assessment: We developed a technique that avoids complete retraining, and our initial experiments demonstrate comparable performance to models that were not required to unlearn from training data. The student implemented and ran experiments for our non-trivial machine unlearning technique.

2.4 Semantic Classification: Spatial Trajectories

Goal: The goal of the project was to include semantic information within a recent framework for classifying spatial trajectories (e.g., a series of GPS way points).

Concepts involved: The project involves spatial data, machine learning, and reproducibility. The student needed to initially reproduce experiment from a research code base for classifying spatial trajectories, and then extend it.

Experiments conducted, platforms used: The students reproduced existing experiments, in the process, understood how to run various algorithmic variants, on different datasets, and reproduce experimental results figures. Then the student extended the method which only treated spatial trajectories as shapes to also include semantic information about various spatial points of interest.

Overall assessment: The project was success. The student got exposed to the reproduction process of most empirical research, and then was able to extend it and demonstrate clear improvement in a controlled experiment.

2.5 Compiler Optimization: ML Primitives

Goal: The project focused on performance of machine learning workloads. Students learned how to measure and model performance, and develop an understanding of the performance impact of programming language choice, memory hierarchy optimization, and architecture. Parallel code for GPUs was developed and measured against the state-of-the-art TVM compiler[5].

Concepts involved: Students used state-of-the-art performance analysis and code optimization tools. The lessons included how to optimize matrix-vector multiplication, convolution 1D, convolution 2D, transposed matrix-matrix multiplication, and matrix-matrix multiplication, which are used at the heart of scientific computing and deep learning benchmarks. Another lesson discussed the roofline model, a performance modeling tool for understanding performance bottlenecks. Students were introduced to the concept of scheduling languages, which provide an interface to compilers to describe transformations to be applied to code. Autotuners compare the performance of different schedules to find the schedule that achieves the best performance.

Experiments conducted, platforms used: Students used an autotuner called Ansor [23] to generate the best schedule for a set of kernels for the state-of-the-art TVM compiler. Ansor uses genetic algorithms to generate potential candidates. Students were interested in whether the schedules in Ansor could be replicated in another compiler framework, Multi-Level IR (MLIR) [12], and achieve the same performance. MLIR's transform dialect makes it possible to express schedules as code. To compare the results, students conducted experiments on Nvidia A100 GPU and AMD EPYC 7513 CPU for common deep learning benchmarks such as convolution 1D, convolution 2D, matrix-vector multiplication and matrix-matrix multiplication.

Overall assessment: The students were able to generate MLIR schedules and achieve high performance on matrix-vector multiplication, which exceeded the performance of TVM+Ansor. For other kernels, there were some performance gaps, for which they worked with the graduate students to find explanations.

2.6 Object Detection and Classification Studies

Goal: The primary goal of the project was to investigate the performance of object detection models trained on video frames containing images of lettuce and weeds. The original dataset, being from video, contained many frames with overlapping content. We created a second *deaugmented* dataset, where each frame is of unique content, and investigated its impact on training behavior and generalization performance.

Concepts involved: The project involved general deep learning concepts. Image detection and classification principles were gained in the course of the project.

Experiments conducted, platforms used: Experiments were done using YOLO v8 [9], an object detection and classification model, due to its ease of use through the corresponding web application. The student preprocessed the dataset using Roboflow [6] with different frame frequencies into two different datasets consisting of 24 frames. Afterwards, two YOLO v8 models were trained on the two datasets and the results were evaluated using a validation dataset.

Overall assessment: As a result, the investigation showed that the model trained on deaugmented set produced better generalization performance than the model trained on the original dataset.

Unfortunately, we did not notice that the deaugmented set was not a subset of the original dataset until the poster was printed. Because the deaugmented set covered 24 times the video length compared to the original dataset, we find the result unsurprising. Nonetheless, the student gained proficiency in using object detection and classification models.

2.7 ML-based Computational Histopathology

Goal: Deep learning models for cell detection/counting [8, 22] in digital histopathology are trained independently from tissue/tumor segmentation[11] models as two separate tasks. But a pathologist zooms out of a patient sample to identify tissues of interest and zooms in to detect cells necessary for a diagnosis. This workflow indicates a dependence between these tasks. The aim of this project was to train a deep learning model that closely matches a pathologist's workflow. OCELOT [19] dataset was used where tissue annotations and cell annotations are available for overlapping patches and multi-task learning could be used to share features among different tasks [16, 19].

Concepts involved: During the course of the project, students learned about semantic segmentation [18], data augmentation [3], fine-tuning pre-trained models [10], image post-processing (for cell counting), and cross-validation. Writing their own data loader and training configuration in PyTorch exposed students to implementation details that assisted them in debugging inferior models performance.

Experiments conducted, platforms used: Students trained models to examine: (a) Training on a CPU versus a GPU, (b) hyper-parameter search for semantic segmentation, (c) the impact of utilizing various data augmentation techniques, and (d) fine-tuning pre-trained backbone for improved convergence. As the majority of the experiments required GPUs with more RAM, students utilized CHPC resources at the University of Utah (specifically GPU nodes owned by the Kalhert School of Computing), which also exposed them to slurm scripting/scheduling.

Overall assessment: The project was a success because it exposed students to multi-scale digital histopathology, medical imaging, and issues that all deep learning researchers face while working in this domain, including low training sample sizes, hyper-parameter search, use of data augmentation, and utilization of pre-trained models. Students were motivated to continue working on the project to attain all specified objectives.

2.8 Reinforcement Learning Studies

Goal: Reinforcement learning (RL) agents can exhibit superhuman performance in certain tasks such as Atari games, but often do so unreliably, i.e. they may not exhibit acceptable performance with high probability. The goal of the project was to compare the reliability of using CNNs vs. vision transformers for estimating Q values in deep Q networks [15].

Concepts involved: The project involved usage of two vision transformers (SwinNet [13] T and S) and two CNNs (EfficientNetv2 [20] S and M), as well as Q learning and general RL concepts. The experiments were done using Python and Atari environments were obtained from Gymnasium [21].

Experiments conducted, platforms used: We trained deep Q learning agents that used the CNNs or the vision transformers in different Atari environments.

Overall assessment: We observed a slightly better sum of average rewards in the Froggerv5 environment than in other environments. Because existing work using these environments used hyperparameters that were impractical given our resources, we were unable to rule out that GPU availability was not a bottleneck. The students implemented the deep Q learning agents as well as the experiments swapping out environments and Q value estimator models. Because compute resources were limited, they were not able to fully investigate the original research question.

2.9 Malware Classification using ML

Goal: The default ML model for text classification is some transformer-based classifier such as BERT [7]. However, one disadvantage of transformers is that their number of parameters scales quadratically with the input sequence length. Previous work [14] has used convolutional neural networks (CNNs) to classify Android opcode sequences, which can be hundreds of thousands of opcodes long, into benign software or malware. We noticed that this work did not mention transformers or recurrent neural networks. We wanted a comparison between their CNN-based technique and using transformers on truncated opcode sequences.

Concepts involved: Reproducing the CNN classifier experiments required understanding of convolution layers and CNN classifiers in general. Implementing the BERT-like classifier involved the encoder-decoder structure of transformers and relevant layers such as embedding, positional encoding, and attention.

Experiments conducted, platforms used: We first attempted to run the CNN classifier authors' experiments implemented in Lua, but later opted to redo them in Python. These lightweight experiments completed within minutes. The transformer experiments, using a single GPU with 24 GB of memory, could not take in sequences longer than about 2^{13} opcodes. Because this was not close to the entire sequence length, we did not seek other GPUs.

Overall assessment: We found the CNN to have better classification accuracy. The student disassembled Android software into opcode sequences, implemented both classifiers and their vocabularies in PyTorch, and ran the experiments.

2.10 Robust High-Dimensional Statistics

Goal: The goal was to reproduce, extend, and make practical recent algorithmic improvements for high-dimensional robust statistics. The recent developments have been mostly theoretical with only simple proof-of-concept code. This project seeks to understand and simplify the mathematical and empirical properties of these algorithms on very high-dimensional data.

Concepts involved: This project involves mathematics of data, statistical analysis, and reproducibility in the context of an active machine learning topic. The student first needed to reproduce the existing work, both the mathematical proofs and the experiments; these efforts were needed so the new work could be properly evaluated against the state-of-the-art.

Experiments conducted, platforms used: Empirical experiments were performed, and included running existing code in MATLAB and reproducing them in python. The main computational bottlenecks were in linear algebra (SVD), and repetition of randomized algorithms, so GPUs were not needed at this stage. The research for this first-year undergraduate also involved novel mathematical proofs on statistical properties of high-dimensional data. The faculty mentor and student worked out sketches of the proofs on the board, and the student provided formal write-ups in a latex document.

Overall assessment: This was a highly successful project that managed to pair a motivated undergraduate with a like-minded faculty mentor and an accessible topic. The faculty's initial goal was just for the student to reproduce the experiments in python, but the student showed engagement and proclivity in the mathematics allowing for a much deeper exploration. We are currently working to extend the work, and prepare for a paper aimed at a top venue in machine learning.

2.11 Computing Statistical Shape Atlases

Goal: Use Shapeworks [4] to compute a statistical shape model for different anatomies (left atrium and prostate [1]) available in the public domain.

Concepts involved: Statistical shape modeling [4], data grooming and preprocessing, hyper-parameter tuning, optimization, models visualization and evaluation, and 3D medical image processing were among the concepts used during project development. The student was also required to analyze the modes of variation in the data and report population-level changes in anatomy using principal component analysis (PCA).

Experiments conducted, platforms used: All experiments were done using either the ShapeWorksStudio [4] or the command line interface of ShapeWorks on the student's own desktop. The student was instructed to compute a shape atlas and principal modes of variations for synthetic 3D spherical data (one mode of variation) to familiarize themselves with the entire computational pipeline. Later, the student was able to compute the statistical shape model for a dataset on the left atrium and analyze and report the modes of variation present in the dataset. The student also conducted an ablation study by analyzing the modes of variation using varying quantities of particles for the same anatomy.

Overall assessment: The project introduced the student to 3D medical image analysis in the context of morphology quantification and motivated them to work further on computing models for different anatomies available in the public domain.

3 ASSESSMENT

Resource issues: Some students launched a job requiring a huge allocation and that was fine but others who were even slightly late to launch were stuck (GPU availability was a bottleneck). This is a

Table 1: Number (out of nine) of post hoc survey respondents who accomplished the goals set at the beginning of the REU. Note that we did not target any specific area other than broadly AI/ML trust and reproducibility.

Student-set Goals	# Students
• Collaborate with peers	9
• Create a research poster	8
• Create or work with ML models	9
• Develop professional relationships	9
• Work on paper-yielding research projects	5
• Identify engrossing research areas	7
• Improve (social) networking skills	6
• Improve ability to grasp research papers	8
• Improve time management skills	4
• Improve writing skills	4
• Increase awareness of CS research areas	9
• Increase knowledge of career options	7
• Increase knowledge of cybersecurity	6
• Increase knowledge of HPC	8
• Increase knowledge of ML and AI	9
• Learn a new programming language	2
• Make a decision about pursuing a PhD	4
• Meet researchers at different career stages	8
• Produce demonstrable research artifacts	8

Table 2: Students' confidence in various research skills. Students were asked to rate their confidence on a scale of 1 (very unconfident) to 5 (very confident). Survey items were derived from Borrego et al. [2]. The attained confidence boost is also noted.

Research Skill	A priori mean confidence	Conf. boost
Designing own research	2.5	1
Writing a scientific report	2.5	1.2
Using tools in the lab	2.7	1.2
Preparing a scientific poster	2.9	1.6
Presenting results of my data	3.1	1.3
Using statistics to analyze data	3.2	0.5
Analyzing data	3.3	0.7
Collecting data	3.3	0.7
Managing my time	3.5	0.6
Problem solving in the lab	3.6	0.4
Understanding scientific articles	3.7	0.3
Observing research in the lab	3.7	0.4
Reading scholarly research	3.7	0.6
Understanding guest lectures	3.8	0.2
Research team experience	3.8	0.6
Speaking to/with professors	3.9	0.4
Research relevance recognition	3.9	0.7
Grasping summer research basics	3.9	0.7

reality likely to be faced by educational endeavors involving the use of GPUs that must be dedicated for long training runs. Suitable planning is needed to alleviate some of these resource contentions.

NSF acknowledges the uneven power in AI research. Universities that can allocate large numbers of DGX machines publish more papers. Unfortunately, one cannot prove a point on a smaller machine running for longer which is the unstated sad part of ML research. Things like ablation studies "ablate the planet."

Assessment of curricular impact: To evaluate REU outcomes, we surveyed students before and after the REU about their intent to pursue a PhD, the number of people they could ask for a letter of recommendation, their confidence with respect to various research skills, their goals for the REU, and their self-reported knowledge of several topic areas. We drew on Borrego et al. [2] to create these surveys, re-using their survey items to capture students' confidence. We received 15 responses to our a priori survey and 10 responses to the post hoc survey; one of the post hoc survey participants did not respond to all items. Survey responses were anonymous.

We saw a slight increase in intention to complete a PhD (a priori mean 3.2 and mode 3, post hoc mean 3.6 and mode 4).

Students networked effectively during the REU. In our post hoc survey, students said they would be comfortable asking a mode of 2 people from the REU program for a letter of recommendation (range 2–4). This is a meaningful increase over the number of potential recommenders students had before the program; students reported a mode of 2 potential recommenders from their home institution (range 1–5) and only 1 from outside their home institution and the REU (range 0–5).

All of the goals students set were accomplished by at least one person during the REU. Students were asked to list two goals that they had for the summer in the a priori survey using free text entry. Using this list, an REU instructor recognized 19 unique goals set by the students. In the post hoc survey, students were asked which of those 19 goals they accomplished (see Table 1). In this manner we tracked progress toward student-generated goals rather than instructor-identified goals. Nine students responded to these questions. Five of theseThe attained confidence boost is also noted. goals were accomplished by all nine respondents: collaborate with peers, develop professional relationships, create or work with machine learning models, increase awareness of research topics in computer science, and increase knowledge of machine learning and artificial intelligence.

Comparison of our a priori survey results to our post hoc survey results shows that students tended to gain the most confidence in areas where they were previously unsure of themselves (see Table 2). The five skills where students gained the most confidence were preparing a scientific poster (post hoc mean 4.4), presenting the results of data (post hoc mean 4.4), using tools in the lab (post hoc mean 3.9), writing a scientific report (post hoc mean 3.8), and designing research (post hoc mean 3.4).

Comparison of survey results also shows that students gained knowledge in the two core areas of our REU—trust and reproducibility in computational research (average increase of 1.6; post hoc means 3.6 and 3.9 respectively). Students also increased their knowledge of research careers, ethics in research, and engineering careers (see Table 3).

Overall, these results suggest that our REU was successful in several respects. We educated students about the REU's areas of focus (trust and reproducibility). We provided professional networking opportunities and introduced students to research topics in computer science. And, we guided students through the process of conducting and presenting research.

4 DISCUSSIONS, CONCLUSIONS

In summary, the REU Site that we launched touched on a collection of topics that are central to today's advanced computer science research, teaching, and practices. Anecdotal feedback (including comments conveyed by the REUs to our local PhD students) suggest that one might aim for a slightly reduced variety of topics next time this site is offered. This is especially true considering the fact that for cohort building and broad exposure, we had all the students participate in all our initial lectures. Not only did it increase stress on the instructors (to get ready with this variety of material quickly), it also tended to be received by the students with varying degrees of enthusiasm.

Two lesson modules for wider adoption have emerged from this REU site: one on how to conduct performance measurement of parallel computations[1], and another on all the Machine Learning modules that served as the backbone for this REU site.[2] These lesson modules are valuable beyond our REU site and may please be considered for adoption in future endeavors of EduHPC.

On the positive side, we did interject the REU site activities not only with social activities on campus, but also other enriching activities such as a presentation on "how to build research posters" (by our Office of Undergraduate Research) and another on applying for NSF Graduate Fellowships (by our Graduate School staff). Another highlight was a talk given by Prof. Daniel Reed on the future of HPC being cloudy and uncertain [17]—a lecture that received rapt attention and lengthy follow-up questions.

Perhaps the biggest payoff of REU sites is the camaraderie established in the formative years amongst young students who would otherwise not have met each other. Many opined that this experience where we made the gender and racial diversity a high priority gave them a fresh appreciation of the power of inclusive education.

This REU site was an adventure, especially given that we really had no prior working relationship with the chosen group, and fearing their inability to sink roots in a campus far away from home (luckily, the REUs took to Salt Lake City with gusto—also helped by the social activities we organized). Given our Year-1 experience with the topics of this REU site, our future year goals will be to narrow-down the set of topics (the volume was troubling for some REUs) and perhaps target the topics to the student tastes/needs (we had all the students partake in all our activities, but it was clear that a different subset cared about a particular topic, with the others ignoring it). The other lesson is to incentivize the completion of exit surveys. We had difficulty collecting responses to our post hoc surveys after students left campus; collecting responses prior to their departure and offering incentive would likely address this issue. Last but not least, an array of ML/AI projects finishing at the same time resulted in GPU availability issues—something that

[1]https://github.com/CtopCsUtahEdu/TREUhpc
[2]https://github.com/damtharvey/reu2023/tree/main

Table 3: Students' self-reported knowledge of five topic areas. Students were asked to rate their knowledge on a scale of 1 (not at all knowledgeable) to 5 (extremely knowledgable).

Knowledge Area	A priori knowledge mean	Increase in knowledge
Trust in the context of computational research	2	1.6
Reproducibility of computational research	2.3	1.6
Research careers	2.4	0.8
Ethics in research	2.7	0.9
Engineering careers	2.9	0.5

needs to be addressed by staging GPU result collection across non-overlapping batches (requiring proactive planning).

ACKNOWLEDGMENTS

Thanks to the REUs who participated in our REU Site during 2023: Cullen Anderson, Joel Berg, Leia Donaway, Maria Hernandez-Rivero, Asma Khan, Aakash Kulkarni, Josef Leim, Lexus Lindeman, Emily Miller, Jedidiah Schmidt, Austin Strobel, Jordan Tan, Christa Tsao, Jayden Vankirk, and Maxwell Yaw.

This material is based upon work supported by the National Science Foundation under Grant Number 2244492.

REFERENCES

[1] Michela Antonelli, Annika Reinke, Spyridon Bakas, Keyvan Farahani, Annette Kopp-Schneider, Bennett A Landman, Geert Litjens, Bjoern Menze, Olaf Ronneberger, Ronald M Summers, et al. 2022. The medical segmentation decathlon. *Nature communications* 13, 1 (2022), 4128.

[2] Marialice Mastronardi Maura Borrego, Nathan Choe, and Risa Hartman. 2021. The Impact Of Undergraduate Research Experiences On Participants' Career Decisions. *Journal of STEM Education* 22, 2 (2021).

[3] A. Buslaev, A. Parinov, E. Khvedchenya, V. I. Iglovikov, and A. A. Kalinin. 2018. Albumentations: fast and flexible image augmentations. *ArXiv e-prints* (2018). arXiv:1809.06839

[4] Joshua Cates, Shireen Elhabian, and Ross Whitaker. 2017. Shapeworks: particle-based shape correspondence and visualization software. In *Statistical shape and deformation analysis*. Elsevier, 257–298.

[5] Tianqi Chen, Thierry Moreau, Ziheng Jiang, Lianmin Zheng, Eddie Yan, Meghan Cowan, Haichen Shen, Leyuan Wang, Yuwei Hu, Luis Ceze, Carlos Guestrin, and Arvind Krishnamurthy. 2018. TVM: An Automated End-to-End Optimizing Compiler for Deep Learning. In *Proceedings of the 13th USENIX Conference on Operating Systems Design and Implementation* (Carlsbad, CA, USA) *(OSDI'18)*. USENIX Association, USA, 579–594.

[6] Floriana Ciaglia, Francesco Saverio Zuppichini, Paul Guerrie, Mark McQuade, and Jacob Solawetz. 2022. Roboflow 100: A Rich, Multi-Domain Object Detection Benchmark. arXiv:2211.13523 [cs.CV]

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL]

[8] Simon Graham, Quoc Dang Vu, Shan E Ahmed Raza, Ayesha Azam, Yee Wah Tsang, Jin Tae Kwak, and Nasir Rajpoot. 2019. Hover-net: Simultaneous segmentation and classification of nuclei in multi-tissue histology images. *Medical image analysis* 58 (2019), 101563.

[9] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. 2023. *YOLO by Ultralytics*. https://github.com/ultralytics/ultralytics

[10] Tushar Kataria, Beatrice Knudsen, and Shireen Elhabian. 2023. To pretrain or not to pretrain? A case study of domain-specific pretraining for semantic segmentation in histopathology. *arXiv preprint arXiv:2307.03275* (2023).

[11] Hyeongsub Kim, Hongjoon Yoon, Nishant Thakur, Gyoyeon Hwang, Eun Jung Lee, Chulhong Kim, and Yosep Chong. 2021. Deep learning-based histopathological segmentation for whole slide images of colorectal cancer in a compressed domain. *Scientific reports* 11, 1 (2021), 22520.

[12] Chris Lattner, Mehdi Amini, Uday Bondhugula, Albert Cohen, Andy Davis, Jacques Pienaar, River Riddle, Tatiana Shpeisman, Nicolas Vasilache, and Oleksandr Zinenko. 2021. MLIR: Scaling compiler infrastructure for domain specific computation. In *2021 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*. IEEE, 2–14.

[13] Zhengyi Liu, Yacheng Tan, Qian He, and Yun Xiao. 2022. SwinNet: Swin Transformer Drives Edge-Aware RGB-D and RGB-T Salient Object Detection. *IEEE Transactions on Circuits and Systems for Video Technology* 32, 7 (jul 2022), 4486–4497. https://doi.org/10.1109/tcsvt.2021.3127149

[14] Niall McLaughlin, Adam Doupé, Gail Ahn, Jesus Martinez-del Rincon, BooJoong Kang, Suleiman Yerima, Paul Miller, Sakir Sezer, Yeganeh Safaei, Erik Trickel, and Ziming Zhao. 2017. Deep Android Malware Detection. 301–308. https://doi.org/10.1145/3029806.3029823

[15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.

[16] Nisha Ramesh and Tolga Tasdizen. 2018. Cell segmentation using a similarity interface with a multi-task convolutional neural network. *IEEE journal of biomedical and health informatics* 23, 4 (2018), 1457–1468.

[17] Daniel Reed, Dennis Gannon, and Jack Dongarra. 2023. HPC Forecast: Cloudy and Uncertain. *Commun. ACM* 66, 2 (jan 2023), 82–90. https://doi.org/10.1145/3552309

[18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 234–241.

[19] Jeongun Ryu, Aaron Valero Puche, JaeWoong Shin, Seonwook Park, Biagio Brattoli, Jinhee Lee, Wonkyung Jung, Soo Ick Cho, Kyunghyun Paeng, Chan-Young Ock, et al. 2023. OCELOT: Overlapped Cell on Tissue Dataset for Histopathology. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 23902–23912.

[20] Mingxing Tan and Quoc V. Le. 2021. EfficientNetV2: Smaller Models and Faster Training. arXiv:2104.00298 [cs.CV]

[21] Mark Towers, Jordan K. Terry, Ariel Kwiatkowski, John U. Balis, Gianluca de Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Arjun KG, Markus Krimmel, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Andrew Tan Jin Shen, and Omar G. Younis. 2023. Gymnasium. https://doi.org/10.5281/zenodo.8127026

[22] Royden Wagner and Karl Rohr. 2022. Cellcentroidformer: Combining self-attention and convolution for cell detection. In *Annual Conference on Medical Image Understanding and Analysis*. Springer, 212–222.

[23] Lianmin Zheng, Chengfan Jia, Minmin Sun, Zhao Wu, Cody Hao Yu, Ameer Haj-Ali, Yida Wang, Jun Yang, Danyang Zhuo, Koushik Sen, et al. 2020. Ansor: Generating High-Performance tensor programs for deep learning. In *14th USENIX symposium on operating systems design and implementation (OSDI 20)*. 863–879.