Program of Study

Science and Engineering

Machine Learning for Healthcare - HST.956

Introduces students to machine learning in healthcare, including the nature of clinical data and the use of machine learning for risk stratification, disease progression modeling, precision medicine, diagnosis, subtype discovery, and improving clinical workflows. Topics include causality, interpretability, algorithmic fairness, time-series analysis, graphical models, deep learning and transfer learning. Guest lectures by clinicians from the Boston area, and projects with real clinical data, emphasize subtleties of working with clinical data and translating machine learning into clinical practice. Limited to 55.

Mathematics and Statistics

Stochastic Processes - 18.615

Basics of stochastic processes. Markov chains, Poisson processes, random walks, birth and death processes, Brownian motion.

High-Performance Computing

Parallel Processing - 6.846

Introduction to parallel and multicore computer architecture and programming. Topics include the design and implementation of multicore processors; networking, video, continuum, particle and graph applications for multicores; communication and synchronization algorithms and mechanisms; locality in parallel computations; computational models, including shared memory, streams, message passing, and data parallel; multicore mechanisms for synchronization, cache coherence, and multithreading. Performance evaluation of multicores; compilation and runtime systems for parallel computing. Substantial project required.

Software Performance Engineering - 6.172 (Undergraduate)

Project-based introduction to building efficient, high-performance and scalable software systems. Topics include performance analysis, algorithmic techniques for high performance, instruction-level optimizations, vectorization, cache and memory hierarchy optimization, and parallel programming.

Computer Science

Approximation Algorithms - 6.854

First-year graduate subject in algorithms. Emphasizes fundamental algorithms and advanced methods of algorithmic design, analysis, and implementation. Surveys a variety of computational models and the algorithms for them. Data structures, network flows, linear programming, computational

geometry, approximation algorithms, online algorithms, parallel algorithms, external memory, streaming algorithms.

Algorithmic Game Theory - 6.853
Presents research topics at the interface of computer science and game theory, with an emphasis on algorithms and computational complexity. Explores the types of game-theoretic tools that are applicable to computer systems, the loss in system performance due to the conflicts of interest of users and administrators, and the design of systems whose performance is robust with respect to conflicts of interest inside the system. Algorithmic focus is on algorithms for equilibria, the complexity of equilibria and fixed points, algorithmic tools in mechanism design, learning in games, and the price of anarchy.

Parallel Computing and Scientific Machine Learning - 6.338
Introduction to scientific machine learning with an emphasis on developing scalable differentiable programs. Covers scientific computing topics (numerical differential equations, dense and sparse linear algebra, Fourier transformations, parallelization of large-scale scientific simulation) simultaneously with modern data science (machine learning, deep neural networks, automatic differentiation), focusing on the emerging techniques at the connection between these areas, such as neural differential equations and physics-informed deep learning. Provides direct experience with the modern realities of optimizing code performance for supercomputers, GPUs, and multicores in a high-level language.

Other Relevant Courses
Market Design - 14.125
Theory and practice of market design, building on ideas from microeconomics, game theory and mechanism design. Prominent case studies include auctions, labor markets, school choice, prediction markets, financial markets, and organ exchange clearinghouses.

Previously Completed Courses
CMSC828W - Deep Learning
In this course, we are going to explore empirically-relevant theoretical foundations of deep learning (DL). We will cover topics including DL optimization, DL generaliation, Neural Tangent Kernels, Deep Generative Models, DL Robustness, DL Interpretability, Domain Adaptation and Generalization, Self-Supervised Learning and Deep Reinforcement Learning.

CMSC754 - Computational Geometry

Introduction to algorithms and data structures for computational problems in discrete geometry (for points, lines, and polygons) primarily in two and three dimensions. Topics include triangulations and planar subdivisions, geometric search and intersection, convex hulls, Voronoi diagrams, Delaunay triangulations, line arrangements, visibility, and motion planning.

CMSC723 - Computational Linguistics 1
Introduce fundamental concepts, techniques, and algorithms for the computational handling of natural language. Statistical and machine learning techniques, models, and algorithms that enable computers to deal with the ambiguity and implicit structure of human language. Approaches that focus on uncovering linguistic structure, such as syntactic or semantic parsing, as well as those that focus on manipulating text in useful ways, such as question answering or machine translation.

CMSC828M - Mechanism Design for Social Good
How do we allocate food to food banks, children to schools, organs to patients, residents to hospitals, security forces to patrol routes, and credit to multiple authors of an open source project—all in the face of competing incentives affecting selfish participants? Can we fairly divide a house, furniture, cars, and the family dog between a divorcing couple? Is Oprah going to be our next president, and—if so—why? Can we design mechanisms for these problems that perform well in practice, are computationally tractable, and whose workings and results are understandable by humans?

CMSC657 - Quantum Computing
Quantum computers have the potential to efficiently solve problems that are intractable for classical computers. This course will explore the foundation of quantum computing. As this is a multidisciplinary subject, the course will cover basic concepts in theoretical computer science and physics in addition to introducing core quantum computing topics. No previous background in quantum mechanics is required. Strong background in linear algebra is recommended!

Tentative topics include: quantum mechanics of qubits; quantum entanglement; quantum protocols; quantum circuits and universality; simple quantum algorithms; quantum Fourier transform; Shor factoring algorithm; Grover search algorithm; quantum complexity theory; selected additional topics as time permits.

CMSC426 - Computer Vision
An introduction to basic concepts and techniques in computer vision. This includes low-level operations such as image filtering and edge detection, 3D reconstruction of scenes using stereo and structure from motion, and object detection, recognition, and classification.

Field Of Interest (2250 characters)

I spent my first couple years in college jumping between various artificial intelligence (AI) projects, realizing my research direction through a project on fairness in rideshare with Prof. John Dickerson. For the project, I worked to improve the fairness of rideshare matching algorithms, which suffer from driver-side income inequality and rider-side discrimination, through new fairness-based objective functions [1]. Performing this research taught me about fairness-related issues in AI, ways to mitigate them, and the potential of AI for social good.

I'm interested in using AI to improve the utility and fairness of healthcare algorithms. I was inspired to pursue these applications after attending talks that detailed the use of algorithms to predict patient health outcomes [2] and detect ailments using chest cardiographs [3]. Algorithms for healthcare need to be fair across demographic lines so all groups can be serviced fairly, so I plan to focus on the fairness of healthcare algorithms and develop methods that ensure fairness.

Algorithms for healthcare use big data and high-performance computing to infer medical patterns. For example, models can learn from time-series clinical data and biological data to predict future clinical outcomes [2], which are done using long-short term memory (LSTM) and recurrent neural network (RNN) architectures. High-performance computing makes it easier to build bigger or deeper models by expanding the feasible search space of models, and also makes it possible for models to take in vaster swaths of data. Additionally, more computing resources allow for larger feature vectors that can capture patient and biological data more precisely.

Improving models for healthcare can improve the utility and fairness of real-world medical care. Better models lead to better predictions, leading to earlier ailment detection and better preventive care. These models can be used across other applications of time-series data as well, such as for price prediction in economics [4]. Finally, by incorporating fairness into our model, predictions are fair across demographics, and so are applicable to diverse populations. increasing trust in medical AI [5].

Citation

[1] Raman, Naveen, Sanket Shah, and John Dickerson. "Data-Driven Methods for Balancing Fairness and Efficiency in Ride-Pooling." arXiv preprint arXiv:2110.03524 (2021).

[2] Chen, Irene, Krishnan, Irene, and Sontag, David "Clustering Interval-Censored Time-Series for Disease Phenotyping" arXiv preprint arXiv:2102.07005 (2021)

[3] Seyyed-Kalantari, Laleh, et al. "Underdiagnosis bias of artificial intelligence algorithms applied to chest radiographs in under-served patient populations." Nature medicine (2021): 1-7.

[4] Tsay, Ruey S. Analysis of financial time series. Vol. 543. John wiley & sons, 2005.

[5] Pham, Quynh, et al. "The Need for Ethnoracial Equity in Artificial Intelligence for Diabetes Management: Review and Recommendations." Journal of medical Internet research 23.2 (2021): e22320.

Computational Power (2250 characters)

Artificial intelligence (AI) algorithms aim to generalize from known training data to unseen testing data. To understand this better, I worked with a team to study the generalization of AI-based deep learning algorithms from "easy" training to "hard" testing examples [1]. We extend prior work, which showed that recurrent networks help with generalization for simple examples, such as mazes [2], to more difficult problems in vision, specifically denoising. Recurrent networks work well for this task, as they can run for more iterations for harder problems, and so can reuse the same architecture for easy and hard problems. We found that these networks assist with generalization, and can be further improved through the use of image-specific features such as dynamic filters [3] and perceptual loss [4], which lead to improvements over state-of-the-art denoisers.

High-performance computing is essential for the development of recurrent networks. More powerful computing resources speed up training and expand the types of models that can be created. High-performance computing resources allow for larger hyperparameter searches, larger models, and new features that can improve model accuracy. For example, we attempted to incorporate median filters but found that model training was too slow [5]; high-performance computing can speed this up and allow us to efficiently incorporate these filters.

By improving generalization across difficulties, we can develop models that perform well even in the presence of limited training data, and models which perform well on difficult unseen points. This idea is not only applicable to image denoising, but to other fields, such as image classification and segmentation, and is especially useful when training data is limited, as these models can generalize even in the presence of train-test differences. During our experiments, we also found that recurrent models are smaller than traditional feedforward ones, so our models consume significantly less power, which is increasingly important as state-of-the-art models become larger. By further researching recurrent networks for generalization, we can shrink models while potentially outperforming state-of-the-art across a variety of tasks.

Citation

[1] Raman, Naveen et al. "Using Recurrent Architectures to Generalize From Easy to Hard Problems in Image Denoising"
[2] Schwarzschild, Avi, et al. "Can You Learn an Algorithm? Generalizing from Easy to Hard Problems with Recurrent Networks." arXiv preprint arXiv:2106.04537 (2021).
[3] Li, Duo, et al. "Involution: Inverting the inherence of convolution for visual recognition." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021.
[4] Johnson, Justin, Alexandre Alahi, and Li Fei-Fei. "Perceptual losses for real-time style transfer and super-resolution." European conference on computer vision. Springer, Cham, 2016.

[5] Liang, Luming, et al. "Convolutional neural network with median layers for denoising salt-and-pepper contaminations." Neurocomputing 442 (2021): 26-35.

Program of Study (2250 characters)

My coursework strikes a balance between theory and practice so I can develop and deploy provably-correct artificial intelligence (AI) algorithms to solve problems in healthcare. Theoretical classes give me the tools to develop models of health-related mechanisms and prove algorithmic bounds, while practical programming classes turn my models and algorithms into code that runs efficiently, even in the presence of big data.

My theory classes include approximation algorithms, stochastic processes, and algorithmic game theory. These courses primarily assist with two things: model development and bound proving. Stochastic processes and algorithmic game theory allow me to incorporate randomness into my models and develop agent-based simulations. Classes on stochastic processes and approximation algorithms help provide bounds on random and approximate algorithms respectively, both of which are becoming increasingly prevalent when dealing with large amounts of data. Theory classes additionally improve my mathematical maturity, as they expose me to ideas from a variety of fields, such as probability theory, linear algebra, and game theory, increasing the range of research papers I can understand.

I plan to take applied programming classes so I can deploy efficient algorithms that leverage high-performance computing and big data resources. Classes at the intersection of high-performance computing and machine learning, such as parallel programming, high-performance computing, and parallel machine learning, allow me to develop faster machine learning algorithms. To apply these ideas to healthcare, I plan on taking machine learning for healthcare. The class covers a broad range of applications, such as clinical NLP, cooperative decision making, and interpretability. Additionally, the social aspect of the class will allow me to collaborate with others interested in machine learning for healthcare.

By taking both theoretical and applied classes, I'll be best prepared to develop algorithms for healthcare that utilize high-performance computing resources.

Programming Language/Models

List four programming language/models + 200 character descriptions

Python - I used Python for various research projects, taking advantage of the matplotlib and numpy libraries to analyze and visualize data, and developing machine learning algorithms through sklearn.
PyTorch - I used PyTorch to build deep learning models, which I used during my time at Lincoln Labs and my current project on image denoising. I optimize my PyTorch models so they train quickly using GPUs.
Docplex - Docplex is an optimization library, which I used to develop solutions to linear programs. I used this primarily for my work on rideshare, where I used it to determine optimal matches between riders and drivers.
R - I use R to develop statistical imputation algorithms, which predict missing data points from survey data. To do this, I used libraries such as editrules and MICE, which use linear programs to impute data.
What are the programming languages that you intend to use in your research? - I plan on primarily using Python for data analysis and model development because of the richness of libraries available, such as the PyTorch model for neural network development.

List of Publications

Papers

1) A Muffin-Theorem Generator at Fun with Algorithms (FUN) 2018, PI: Bill Gasarch, May 2016-August 2016
2) "Stress and Burnout in Open Source: Toward Finding, Understanding, and Mitigating Unhealthy Interactions" at ICSE 2020, PI: Bogdan Vasilescu and Christian Kaestner, May 2019-August 2019
3) "Data Driven Methods for Balancing Fairness and Efficiency in Ride-Pooling" at ML For Economics Policy NeurIPS 2020 Workshop, PI: John Dickerson, September 2019 - January 2021
4) "What more can entity linking do for Question Answering at HAMLETS" NeurIPS 2020 Workshop , PI: Jordan-Boyd Graber, September 2019-Current
5) "Data-Driven Methods for Balancing Fairness and Efficiency in Ride-Pooling", presented/published at IJCAI 2021, PI: John Dickerson, September 2019 - January 2021
6) "Eliciting Bias in Question Answering Models through Ambiguity" at MRQA EMNLP 2021 Workshop, PI: Jordan-Boyd Graber, January 2021-August 2021
7) "Improving human-AI deference algorithms through fine-tuning", PI: Michael Yee, done while at Lincoln Labs, May 2021-August 2021, Workshop on Human-Machine Decisions at NeurIPS

Talks

1) Discovering mutational signatures through non-negative matrix factorization, PI: Aravind Srinivasan and Max Leiserson, May 2018-August 2019
2) "Stress and Burnout in Open Source: Toward Finding, Understanding, and Mitigating Unhealthy Interactions" at ICSE 2020, PI: Bogdan Vasilescu and Christian Kaestner, May 2019-August 2019
3) "Data-Driven Methods for Balancing Fairness and Efficiency in Ride-Pooling", presented/published at IJCAI 2021, PI: John Dickerson, September 2019 - January 2021

Posters

1) "What more can entity linking do for Question Answering at HAMLETS" NeurIPS 2020 Workshop , PI: Jordan-Boyd Graber, September 2019-Current
2) "Investigating methods of balancing inequality and efficiency in Ride Pooling" at AAAI Undergraduate Consortium 2021, PI: John Dickerson, September 2019 - January 2021
3) "Data Driven Methods for Balancing Fairness and Efficiency in Ride-Pooling" at ML For Economics Policy NeurIPS 2020 Workshop, PI: John Dickerson, September 2019 - January 2021
4) "What more can entity linking do for Question Answering at HAMLETS" NeurIPS 2020 Workshop , PI: Jordan-Boyd Graber, September 2019-Current
5) "Eliciting Bias in Question Answering Models through Ambiguity" at MRQA EMNLP 2021 Workshop, PI: Jordan-Boyd Graber, January 2021-August 2021
6) "Improving human-AI deference algorithms through fine-tuning", PI: Michael Yee, done while at Lincoln Labs, May 2021-August 2021, Workshop on Human-Machine Decisions at NeurIPS

Laboratory and Research Experience/Other Employment

2018 - Worked in computational biology lab with Professors Aravind Srinivasan and Max Leiserson on cancer signatures

2019-Present - Working in CLIP NLP lab with Professor Jordan Boyd-Graber on entity linknig

2019-Present - Working with Professor John Dickerson on two projects, the first involving fairness in ride-pooling, and the second involving rideshare pricing algorithms

2019 - Worked at CMU as part of REU with Professors Bogdan Vasilescu and Christian Kaestner on detecting toxicity in online communities

2020 - Worked at Facebook as a software engineer, doing web development to debug their feedranking models

2021 - Worked at MIT Lincoln Labs on human-AI collaboration

2021-present - As part of CMSC723 class, worked on project on bias in NLP, published project at MRQA

2021-present - Working with Professor Partha Lahiri on improving imputation algorithms for survey data

2021-present - As part of CMSC828W class, working on project to improve Vision Transformers

----2018----

----September---

Started school at UMD, ACES Honors College for cybersecurity

Worked in Aravind Srinivasan and Max Leiserson's lab to discover mutational processes through non-negative matrix factorization

---October---

Attended StringBio conference in Orlando on string algorithms for computational biology

----2019----

---January---

Started as TA for class on web programming, CMSC122

---May to August----

Worked at Carnegie Mellon University for the REUSE REU program with Bogdan Vasilesuc and Christian Kaestner on detecting online toxicity in software engineering communities.

----September---

Started working with John Dickerson on rideshare fairness

Started working with Jordan Boyd-Graber on improving entity linking through data collection
Started as TA for class on programming languages, CMSC330

---2020---
---January---
Published work at Carnegie Mellon, Raman on toxicity, Naveen, et al. "Stress and burnout in open source: Toward finding, understanding, and mitigating unhealthy interactions." Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: New Ideas and Emerging Results. 2020.
Started as TA for class on coding interviews, CMSC389O

----March---
Presented work on entity linking at MASC-SLL conference on speech and language

----May-August----
Worked at Facebook as a software engineer, developing web UIs for their feed ranking tools

---November----
Published work on entity linking with Jordan Boyd-Graber at HAMLETS workshop at NeurIPS
Published work on rideshare fairness with John Dickerson at Machine Learning for Econ Policy at NeurIPS
Published work on rideshare fairness with John Dickerson at AAAI Undergraduate Consortium

---2021---
---January---
Became head facilitator for student-taught class on coding interviews (CMSC389O)
Proposed new project on improving pricing systems for rideshare systems by including network externalities

---January-May---
Took a class on computational linguistics, working on a class project on bias in question answering, published in September at Machine Reading for Question Answering at EMNLP 2021

---April---
Published work on rideshare fairness with John Dickerson at IJCAI 2021

---May-August---
Worked at Lincoln Labs with Michael Yee on improving human-AI collaboration through fine-tuning and semi-supervised learning

---September---
Started working with Partha Lahiri on improving edit imputation algorithms

--October--
Published work done at Lincoln Labs with Michael Yee at Workshop on Human-Machine Decisions, NeurIPS 2021

---Course Highlights Major----
1 semester in following computer science graduate classes: Computational Linguistics, Deep Learning, Mechanism Design, Quantum Computing, Computational Geometry
1 semester in following science undergraduate classes: Computer Vision, Cryptography, Databases, Algorithms 2
1 semester in following math classes: partial differential equations, real analysis, probability theory, statistical theory, numerical methods, advanced linear algebra
1 semester in following lower-level classes: programming languages, algorithms, computer systems, calculus 3, tested out of linear algebra

---Course Highlights, Non-Major---
1 semester in agent-based modeling, natural resource economics, educational policy, physical chemistry, naval ethics, reverse engineering
3 semesters in cybersecurity

Academic Awards and Honors

Goldwater Scholarship - Prestigious scholarship given to top undergraduate researchers nationally
Churchill Scholarship - Fellowship given to 15 seniors nationally for one year study at Cambridge
Philip Merill Presidential Scholarship - Given to top 15 seniors for academic and service record
Iribe Scholarship - Computer Science scholarship worth 11K for academic and research record, given for 2 years (11K each year)
Presidential Scholarship - 4 year scholarship worth 20K given to rising freshmen
Capital One Scholarship - Computer Science scholarship worth 1K for academic and research record

Corporate Partners Scholarship - Computer Science scholarship worth 2K for academic and research record
CS Honors College - Research-based honors college, allowing students to take classes on research, create senior thesis
Global Fellows Program - Honors program that explores international and policy issues, culminating in part-time internship
ACES Honors College - Cybersecurity-based honors college that teaches basics of cybersecurity, Linux, bash scripting. Developed cybersecurity-based research project
CMU REUSE REU Fellowship - Participated in research program at CMU for 10 weeks, researching toxicity in online communities.
Math Modelling - Received outstanding award for modeling of refugee populations for SCUDEM math modeling competition
Bloomberg Codecon - Received top 3 awards at University of Maryland programming competition, and qualified for national competition at Bloomberg headquarters in New York City

Extracurricular Activities

CMSC330 TA (Programming Languages) - Fall 2019-Present
CMSC389O TA (Coding Interviews) - Spring 2020 - Present, head TA Spring 2021-Present
CMSC122 TA - Spring 2019
Maryland Mentors Program, volunteer at local elementary schools - Fall 2020 - Present
College Park Academy Volunteer (volunteering at local charter school) - Fall 2018 - Spring 2021, assisted with cybersecurity club, math tutoring
College Mentors (helping middle schoolers with career pathways) - Spring 2021