COMPUTER SCIENCE

A.K. Watson Hall, 203.432.1246
http://cpsc.yale.edu
M.S., M.Phil., Ph.D.

Chair
Zhong Shao

Director of Graduate Studies
Lin Zhong (lin.zhong@yale.edu)

Professors Dana Angluin (Emerita), James Aspnes, Dirk Bergemann,* Ronald Coifman,* Aaron Dollar,* Julie Dorsey, Joan Feigenbaum, Michael Fischer, Robert Frank,* David Gelernter, Mark Gerstein,* John Lafferty,* Rajit Manohar,* Vladimir Rokhlin,† Holly Rushmeier, Brian Scassellati, Martin Schultz (Emeritus), Zhong Shao, Avi Silberschatz, Daniel Spielman, Phillipp Strack,* Leandros Tassiulas,* Nisheeth Vishnoi, Y. Richard Yang, Lin Zhong, Steven Zucker†

Associate Professors Abhishek Bhattacharjee, Yang Cai, Amin Karbasi,* Theodore Kim, Smita Krishnaswamy,* Sahand Negahban,* Charalampos Papamanthou, Ruzica Piskac, Robert Soule, Jakub Szefer*

Assistant Professors Kim Blenman,* Arman Cohan, Yongshan Ding, Benjamin Fisch, Tesca Fitzgerald, Wenjun Hu,* Julian Jara-Ettinger,* Anurag Khandelwal, Daniel Rakita, Katerina Sotiraki, David van Dijk,* Marynel Vázquez, Andre Wibisono, Alex Wong, Zhitao Ying, Manolis Zampetakis, Fan Zhang

Senior Lecturers James Glenn, Andrew Sherman, Stephen Slade

Lecturers Timos Antonopoulos, Timothy Barron, Ozan Erat, Kyle Jensen,* Janet Kayfetz, Jay Lim, Dylan McKay, Cody Murphey, Sohee Park, Scott Petersen, Brad Rosen, Andrew Sherman,* Inyoung Shin, Alan Weide, Cecillia Xie

* A secondary appointment with primary affiliation in another department or school.
† A joint appointment with another department.

FIELDS OF STUDY
Algorithms and computational complexity, artificial intelligence, data networking, databases, graphics, machine learning, programming languages, robotics, scientific computing, security and privacy, and systems.

RESEARCH FACILITIES
The department operates a high-bandwidth, local-area computer network-based mainly on distributed workstations and servers, with connections to worldwide networks. Workstations include Dell dual-processor PCs (running Linux or Windows/XP). Laboratory contains specialized equipment for graphics, vision, and robotics research. Various printers, including color printers, as well as image scanners, are also available. The primary educational facility consists of thirty-seven PC workstations supported by a large Intel PC server. This facility is used for courses and unsponsored research
by Computer Science majors and first-year graduate students. Access to computing, through both the workstations and remote login facilities, is available to everyone in the department.

**SPECIAL REQUIREMENTS FOR THE PH.D. DEGREE**

There is no foreign language requirement. To be admitted to candidacy, a student must (1) pass ten courses (including CPSC 690 and CPSC 691) with at least two grades of Honors, the remainder at least High Pass, including three advanced courses in an area of specialization; (2) take six advanced courses in areas of general computer science; (3) successfully complete a research project in CPSC 690, CPSC 691, and submit a written report on it to the faculty; (4) pass a qualifying examination in an area of specialization; (5) be accepted as a thesis student by a regular department faculty member; (6) serve as a teaching assistant for two terms; and (7) submit a written dissertation prospectus, with a tentative title for the dissertation. Grades of Pass will not count toward the Ph.D.

To satisfy the distribution requirement (requirement 2 above), the student must take one course in programming languages or systems, one programming-intensive course, two theory courses, and two in application areas. In order to gain teaching experience, all graduate students are required to serve as teaching assistants for two terms during their first three years of study. All requirements for admission to candidacy must be completed prior to the end of the third year. In addition to all other requirements, students must successfully complete CPSC 991, Ethical Conduct of Research, prior to the end of their first year of study. This requirement must be met prior to registering for the second year of study.

**MASTER’S DEGREES**

**M.Phil.** See Degree Requirements under Policies and Regulations.

**M.S. (en route to the Ph.D.)** To qualify for the M.S., the student must pass eight courses at the 500 level or above from an approved list. An average grade of at least High Pass is required, with at least one grade of Honors.

**Terminal Master’s Degree Program** Students may also be admitted to a terminal master’s degree program directly. There are two options for the terminal master’s degree:

- **Terminal Master’s Degree Program (coursework-only option)** The requirements are the same as for the M.S. en route to the Ph.D. This program is normally completed in one year, but a part-time program may be spread over as many as four years.

- **Terminal Master’s Degree Program (thesis option)** To qualify for the M.S. thesis option the student must (1) pass six courses at the 500 level or above from an approved list with an average grade of at least High Pass and with at least one grade of Honors; (2) complete a research thesis, generally in the second year; and (3) serve as a teaching assistant for four terms. This program is normally completed in two years.

Please use the links provided for additional information about the department, faculty, courses, and facilities online; You may also reach out to the departmental registrar at cs-admissions@cs.yale.edu.
COURSES

**CPSC 513a, Computer System Security**  Timothy Barron
Overview of the principles and practice behind analyzing, designing, and implementing secure computer systems. The course covers problems that have continued to plague computer systems for years as well as recent events and research in this rapidly evolving field. Students learn to think from the perspective of an adversary, to understand systems well enough to see how their flaws could be exploited, and to consequently defend against such exploitation. The course offers opportunities for hands-on exploration of attacks and defenses in the contexts of web applications, networks, and system-level software. It also addresses ethical considerations and responsibilities associated with security research and practice.

**CPSC 519a or b, Full Stack Web Programming**  Alan Weide
This course introduces students to a variety of advanced software engineering and programming techniques in the context of full-stack web programming. The focus of the course includes both client- and server-side programming (and database programming), client/server communication, user interface programming, and parallel programming.

**CPSC 520b / ENAS 820b, Computer Architecture**  Abhishek Bhattacharjee
This course offers a treatment of computer architectures for high-performance and power/energy-efficient computer systems. Topics include the foundations of general-purpose computing, including instruction set architectures, pipelines, superscalar and out-of-order execution, speculation, support for precise exceptions, and simultaneous multi-threading. We also cover domain-specific hardware (e.g., graphics processing units), and ongoing industry efforts to elevate them to the status of first-class computing units. In tandem, we cover topics relevant to both general-purpose and domain-specific computing, including memory hierarchies, address translation and virtual memory, on-chip networks, machine learning techniques for resource management, and coherence techniques. If time permits, we study the basics of emerging non-classical computing paradigms like neuromorphic computing. Overall, this course offers insights on how the computing industry is combating the waning of traditional technology scaling via acceleration and heterogeneity. Prerequisites: Courses similar to CPSC 323, 223, and 202. This is a programming-intensive course, so comfort with large programming projects is essential.

**CPSC 521a, Compilers and Interpreters**  Jay Lim
Compiler organization and implementation: lexical analysis, formal syntax specification, parsing techniques, execution environment, storage management, code generation and optimization, procedure linkage, and address binding. The effect of language-design decisions on compiler construction.

**CPSC 522b, Operating Systems**  Anurag Khandelwal
The design and implementation of operating systems. Topics include synchronization, deadlocks, process management, storage management, file systems, security, protection, and networking.

**CPSC 524a, Parallel Programming Techniques**  Andrew Sherman
Practical introduction to parallel programming, emphasizing techniques and algorithms suitable for scientific and engineering computations. Aspects of processor and machine architecture. Techniques such as multithreading, message passing, and data parallel
computing using graphics processing units. Performance measurement, tuning, and debugging of parallel programs. Parallel file systems and I/O.

**CPSC 528b, Language-Based Security**  Zhong Shao

Basic design and implementation of language-based approaches for increasing the security and reliability of systems software. Topics include proof-carrying code; certifying compilation; typed assembly languages; runtime checking and monitoring; high-confidence embedded systems and drivers; and language support for verification of safety and liveness properties.

**CPSC 529a, Principles of Computer System Design**  Lin Zhong

Humans are stupid; computers are limited. Yet a collaboration of humans and computers has led to ever more powerful and complex computer systems. This course examines the limitations of humans and computers in this endeavor and how they shape the design, implementation, and evaluation of computer systems. It surveys the empirical knowledge reported by scholars and practitioners who overcome such limitations. The lectures, reading assignments, and classroom discussions travel through psychology and philosophy and revisit important results from theoretical computer science, with a goal of elucidating the rationales behind the best practices in computer systems research and development. Prerequisite: CPSC 323 or equivalent. Students should have the ability to write significant system programs in at least one system programming language (e.g., C, C++ and Rust).

**CPSC 531a, Computer Music: Algorithmic and Heuristic Composition**  Scott Petersen

Study of the theoretical and practical fundamentals of computer-generated music. Music and sound representations, acoustics and sound synthesis, scales and tuning systems, algorithmic and heuristic composition, and programming languages for computer music. Theoretical concepts are supplemented with pragmatic issues expressed in a high-level programming language.

**CPSC 532b, Computer Music: Sound Representation and Synthesis**  Scott Petersen

Study of the theoretical and practical fundamentals of computer-generated music, with a focus on low-level sound representation, acoustics and sound synthesis, scales and tuning systems, and programming languages for computer music generation. Theoretical concepts are supplemented with pragmatic issues expressed in a high-level programming language. Prerequisite: ability to read music.

**CPSC 533b, Computer Networks**  Y. Richard Yang

An introduction to the design, implementation, analysis, and evaluation of computer networks and their protocols. Topics include layered network architectures, applications, transport, congestion, routing, data link protocols, local area networks, performance analysis, multimedia networking, network security, and network management. Emphasis on protocols used in the Internet.

**CPSC 534a, Topics in Networked Systems**  Y. Richard Yang

Study of networked systems such as the Internet and mobile networks which provide the major infrastructure components of an information-based society. Topics include the design principles, implementation, and practical evaluation of such systems in new settings, including cloud computing, software-defined networking, 5G, Internet of things, and vehicular networking.
CPSC 535b, Building an Internet Router  Robert Soule
Over the course of the term, students build a fully functioning Internet router. Students design the control plane in Python on a Linux host and design the data plane in the new P4 language on the bmv2 software switch. To provide context and background for the design of their router, students read a selection of papers to get both a historical perspective and exposure to current research in networking. Prerequisite: CPSC 533.

CPSC 537a or b, Introduction to Database Systems  Avi Silberschatz

CPSC 538a, Big Data Systems: Trends and Challenges  Anurag Khandelwal
Today’s Internet-scale applications and cloud services generate massive amounts of data. At the same time, the availability of inexpensive storage has made it possible for these services and applications to collect and store every piece of data they generate, in the hopes of improving their services by analyzing the collected data. This introduces interesting new opportunities and challenges designing systems for collecting, analyzing, and serving the so-called big data. This course looks at technology trends that have paved the way for big data applications, surveys state-of-the-art systems for storage and processing of big data, and considers future research directions driven by open research problems. Our discussions span topics such as cluster architecture, big data analytics stacks, scheduling and resource management, batch and stream analytics, graph processing, ML/AI frameworks, and serverless platforms and disaggregated architectures.

CPSC 539a or b, Software Engineering  Timos Antonopoulos
Introduction to building a large software system in a team. Learning how to collect requirements and write a specification. Project planning and system design. Increasing software reliability: debugging, automatic test generation. Introduction to type systems, static analysis, and model checking.

CPSC 540a, Database Design and Implementation  Robert Soule
This course covers advanced topics in Database Systems, explaining on the material covered in CPSC 437/537. Topics covered include complex data types, application development, big data, data analytics, parallel and distributed storage, parallel and distributed query processing, advanced indexing techniques, advanced relational database design, and object-based databases.

CPSC 541a, Zero-Knowledge Proofs  Ben Fisch
This is a course in cryptographic proof systems. In the digital world today, we trust services to perform many kinds of computations on our data, from managing financial ledgers and databases to complex analytics. We trust these services not only to operate correctly but also to keep our information private. Proof systems allow us to remove this trust. A succinct proof system is a system that enables a service to attach a small certificate on the correctness of its computation, and the certificate can be verified by small devices, even if the original computation needs substantial computation to compute this result. Beyond correctness, a zero-knowledge proof system enables us to prove knowledge of secret information, including hidden inputs to a computation that
achieves a certain output. Both types of proof systems have incredible applications to privacy and verifiability in a decentralized web. CPSC 567 (Cryptography), MATH 225 (Linear Algebra) are recommended, but not required, prior to taking the course.

**CPSC 542a, Theory of Computation**  Dylan McKay
This course first introduces core, traditional ideas from the theory of computation with more modern ideas used in the process, including basic ideas of languages and automata. Building on the core ideas, the course then covers a breadth of topics in modular units, where each unit examines a new model and potentially a new perspective on computation. Topics may include: basic notions of Complexity Theory, provability and logic, circuits and non-uniform computation, randomized computation, quantum computation, query-based computation, notions of machine learning, compression, and algebraic models of computation. Additional topics might be introduced in lectures or student projects, according to student interests, including mechanism design, voting schemes, cryptography, biological computation, distributed computation, and pseudorandomness. Prerequisite: One of CPSC 365, 366, or 368 is required. This course is a proof-based theory course and mathematical maturity is expected.

**CPSC 543a / MATH 543a, Optimal Transport: Theory, Algorithms, and Applications to Data Science**  Smita Krishnaswamy
Optimal transport started with Gaspart Monge in the 1700s when he stated the problem of moving a large pile of sand (whose shape is a probability distribution) to a target pile with minimal effort. The optimal transport plan not only gives a coupling between distributions but also a metric between such probability measures, which has found use in everything from modern neural networks to economic resource allocation problems, to shape matching in computer vision. This course covers the theoretical foundations as well as computational aspects of optimal transport starting with the original formulations as maps between discrete measures and extending to general measures as well as the key Kantorovich relaxation as a coupling between measures and its metric properties. We also cover algorithmic foundations of optimal transport using linear programs that have recently been sped-up via entropic regularizations. In addition to the primal form, we cover the dual form and relaxations which lead to integral probability metrics. We vary the ground space of optimal transport from Euclidean, to arbitrary metrics, to graphs. We move from static to dynamic formulations of optimal transport, which can provide paths of flow for dynamics that are energy-constrained. Finally, we cover important extensions such as unbalanced optimal transport which allows for transport between generic measures (without the same volume) and for Gromov-Wasserstein distances between measures on different metric spaces. Prerequisites: MATH 241, CPSC 202, CPSC 223, and CPSC 365. Knowledge of Python programming is also required.

**CPSC 546a, Data and Information Visualization**  Holly Rushmeier
Visualization is a powerful tool for understanding data and concepts. This course provides an introduction to the concepts needed to build new visualization systems, rather than to use existing visualization software. Major topics are abstracting visualization tasks, using visual channels, spatial arrangements of data, navigation in visualization systems, using multiple views, and filtering and aggregating data. Case studies to be considered include a wide range of visualization types and applications in humanities, engineering, science, and social science. Prerequisite: CPSC 223.
CPSC 547a, Introduction to Quantum Computing  Yongshan Ding
This course introduces the fundamental concepts in the theory and practice of quantum computing. Topics covered include information processing, quantum programming, quantum compilation, quantum algorithms, and error correction. The objective of the course is to engage students in applying fresh thinking to what computers can do. We establish an understanding of how quantum computers store and process data, and we discover how they differ from conventional digital computers. We anticipate this course will be of interest to students working in computer science, electrical engineering, physics, or mathematics. Students must be comfortable with programming, discrete probability, and linear algebra. Prior experience in quantum computing is useful but not required.

CPSC 552b / AMTH 552b / CB&B 663b, Deep Learning Theory and Applications  Smita Krishnaswamy
Deep neural networks have gained immense popularity within the past decade due to their success in many important machine-learning tasks such as image recognition, speech recognition, and natural language processing. This course provides a principled and hands-on approach to deep learning with neural networks. Students master the principles and practices underlying neural networks, including modern methods of deep learning, and apply deep learning methods to real-world problems including image recognition, natural language processing, and biomedical applications. Course work includes homework, a final exam, and a final project—either group or individual, depending on enrollment—with both a written and oral (i.e., presentation) component. The course assumes basic prior knowledge in linear algebra and probability. Prerequisites: CPSC 202 and knowledge of Python programming.

CPSC 553a / AMTH 553a / CB&B 555a / GENE 555a, Unsupervised Learning for Big Data  Staff
This course focuses on machine-learning methods well-suited to tackling problems associated with analyzing high-dimensional, high-throughput noisy data including: manifold learning, graph signal processing, nonlinear dimensionality reduction, clustering, and information theory. Though the class goes over some biomedical applications, such methods can be applied in any field. Prerequisites: knowledge of linear algebra and Python programming.

CPSC 554a, Software Analysis and Verification  Ruzica Piskac
Introduction to concepts, tools, and techniques used in the formal verification of software. State-of-the-art tools used for program verification; detailed insights into algorithms and paradigms on which those tools are based, including model checking, abstract interpretation, decision procedures, and SMT solvers.

CPSC 555a, Economics and Computation  Emmanouil Zampetakis
A mathematically rigorous investigation of the interplay of economic theory and computer science, with an emphasis on the relationship of incentive-compatibility and algorithmic efficiency. Particular attention to the formulation and solution of mechanism-design problems that are relevant to data networking and Internet-based commerce.

CPSC 557a, Sensitive Information in a Connected World  Michael Fischer
Issues of ownership, control, privacy, and accuracy of the huge amount of sensitive information about people and organizations that is collected, stored, and used by
today’s ubiquitous information systems. Readings consist of research papers that explore both the power and the limitations of existing privacy-enhancing technologies such as encryption and “trusted platforms.”

**CPSC 558b, Automated Decision Systems  Stephen Slade**
People make dozens of decisions every day in their personal and professional lives. What would it mean for you to trust a computer to make those decisions for you? It is likely that many of those decisions are already informed, mediated, or even made by computer systems. Explicit examples include dating sites like match.com or recommendation systems such as Amazon or Netflix. Most Internet ads on sites like Google or Facebook are run by real-time-bidding (RTB) systems that conduct split-second auctions in the hopes of getting your attention. Driverless cars offer the promise of safer highways. Corporations and other enterprises invest in decision support systems to improve the quality of their products and services. This course considers the spectrum of automated decision models and tools, examining their costs and effectiveness. Examples come from a variety of fields including finance, risk management, credit-card fraud, robotics, medicine, and politics.

**CPSC 559a, Building Interactive Machines  Marynel Vazquez**
This advanced course brings together methods from machine learning, computer vision, robotics, and human-computer interaction to enable interactive machines to perceive and act in a variety of environments. Part of the course examines approaches for perception with different sensing devices and algorithms; the other part focuses on methods for decision-making and applied machine learning for control. The course is a combination of lectures, state-of-the-art reading, presentations and discussions, programming assignments, and a final team project. Prerequisites: CPSC 570 and understanding of probability, differential calculus, linear algebra, and planning (in Artificial Intelligence). Programming assignments require proficiency in Python and high-level familiarity with C++. Students who do not fit this profile may be allowed to enroll with the permission of the instructor.

**CPSC 563b / ECON 565b, Algorithms for Convex Optimization  Nisheeth Vishnoi**
Convex optimization has played a major role in the recent development of fast algorithms for problems arising in areas such as theoretical computer science, discrete optimization, and machine learning. The approach is to first formulate the problem as a continuous (convex) optimization problem, even if the problem may be over a discrete domain, adapt or develop deterministic or randomized continuous-time dynamical systems to solve it, and then design algorithms for the problem via appropriate discretizations. The goal of this course is to design state-of-the-art algorithms for various classical discrete problems through the use of continuous optimization/sampling. The algorithmic applications include maximum flow in graphs, maximum matching in bipartite graphs, linear programming, submodular function minimization, and counting problems involving discrete objects such as matroids. We present approaches gradient descent, mirror descent, interior-point methods, and cutting plane methods. Prerequisite: CPSC 365 or permission of the instructor. S&DS 630 and a solid background in calculus, linear algebra, probability, and algorithms are recommended.

**CPSC 564a, Algorithms and their Societal Implications  Nisheeth Vishnoi**
Today’s society comprises humans living in an interconnected world that is intertwined with a variety of sensing, communicating, and computing devices. Human-generated data is being recorded at unprecedented rates and scales, and powerful AI and ML
algorithms, which are capable of learning from such data, are increasingly controlling various aspects of modern society: from social interactions. These data-driven decision-making algorithms have a tremendous potential to change our lives for the better, but, via the ability to mimic and nudge human behavior, they also have the potential to be discriminatory, reinforce societal prejudices, violate privacy, polarize opinions, and influence democratic processes. Thus, designing effective tools to govern modern society which reinforce its cherished values such as equity, justice, democracy, health, privacy, etc. has become paramount and requires a foundational understanding of how humans, data, and algorithms interact. This course is for students who would like to understand and address some of the key challenges and emerging topics at the aforementioned interplay between computation and society. On the one hand, we study human decision-making processes and view them through the lens of computation, and on the other hand we study and address the limitations of artificial decision-making algorithms when deployed in various societal contexts. The focus is on developing solutions through a combination of foundational work such as coming up with the right definitions, modeling, algorithms, and empirical evaluation. The current focus is on bias and privacy, with additional topics including robustness, polarization, and democratic representation. The grade will be based on class participation and a project. The project grade will be determined by a midterm and endterm report/presentation. The course has four primary modules: (1) Data: human-generated data; data collection and aggregation; (2) Decision-Making Algorithms: human decision-making algorithms; traditional algorithmic decision-making models and methods; machine learning-based decision-making models and methods; (3) Bias: socio-technical contexts and underlying computational problems; definitions of fairness; interventions for ensuring fairness; human biases through the lens of computation; privacy; need for definitions of privacy; differential privacy; beyond differential privacy; (4) Other topics: robustness; polarization; elections and social choice. Solid mathematical and programming background is necessary to enroll in this course. CPSC 365 and S&DS 251 are recommended.

CPSC 565a, Theory of Distributed Systems  James Aspnes
Models of asynchronous distributed computing systems. Fundamental concepts of concurrency and synchronization, communication, reliability, topological and geometric constraints, time and space complexity, and distributed algorithms.

CPSC 566b, Blockchain and Cryptocurrency  Ben Fisch
This course is an introduction to blockchain systems, such as Bitcoin and Ethereum. We begin with a brief history of blockchains and an overview of how they are being used today before launching into foundational topics, including distributed consensus, smart contracts, cryptographic building blocks from signatures to authenticated datastructures, and the economics of blockchains. We then cover advanced topics including the scalability and interoperability of blockchain systems and applications such as “decentralized finance” (DeFi). The lectures and assignments engage students in both theoretical and applied aspects of blockchain systems. The course assumes background in various fundamental areas of CS, including discrete math, probability, algorithms, data structures, cryptography, and networks.

CPSC 567b, Introduction to Cryptography  Staff
This course introduces modern symmetric and public-key cryptography as well as their broad applications, both from a theoretical and practical perspective. There is an initial
emphasis on fundamental cryptographic primitives (e.g., block ciphers, pseudorandom functions, pseudorandom generators, one-way functions), their concrete efficiency and implementation, as well as their security definitions and proofs. Ways of combining such primitives that lead to more complex objects used to secure today’s internet (e.g., via TLS), such as key exchange, randomized encryption, message authentication codes, and digital signatures are also studied. The last part of the course is devoted to modern and more advanced applications of cryptography (some of which are deployed at scale today), such as authenticated data structures, zero-knowledge proofs, oblivious RAM, private information retrieval, secret sharing, distributed consensus, and cryptocurrencies. (e.g., Bitcoin).

**CPSC 569b, Randomized Algorithms**  James Aspnes  
Beginning with an introduction to tools from probability theory including some inequalities like Chernoff bounds, the course covers randomized algorithms from several areas: graph algorithms, algorithms in algebra, approximate counting, probabilistically checkable proofs, and matrix algorithms.

**CPSC 570a, Artificial Intelligence**  Tesca Fitzgerald  
Introduction to artificial intelligence research, focusing on reasoning and perception. Topics include knowledge representation, predicate calculus, temporal reasoning, vision, robotics, planning, and learning.

**CPSC 572a, Intelligent Robotics**  Brian Scassellati  
Introduction to the construction of intelligent, autonomous systems. Sensory-motor coordination and task-based perception. Implementation techniques for behavior selection and arbitration, including behavior-based design, evolutionary design, dynamical systems, and hybrid deliberative-reactive systems. Situated learning and adaptive behavior.

**CPSC 574a or b, Computational Intelligence for Games**  James Glenn  
A seminar on current topics in computational intelligence for games, including developing agents for playing games, procedural content generation, and player modeling. Students read, present, and discuss recent papers and competitions, and complete a term-long project that applies some of the techniques discussed during the term to a game of their choice.

**CPSC 575a / ENAS 575a / INP 575a, Computational Vision and Biological Perception**  Steven Zucker  
An overview of computational vision with a biological emphasis. Suitable as an introduction to biological perception for computer science and engineering students, as well as an introduction to computational vision for mathematics, psychology, and physiology students.

**CPSC 576b / AMTH 667b / ENAS 576b, Advanced Computational Vision**  Steven Zucker  
Advanced view of vision from a mathematical, computational, and neurophysiological perspective. Emphasis on differential geometry, machine learning, visual psychophysics, and advanced neurophysiology. Topics include perceptual organization, shading, color, and texture.

**CPSC 577b, Natural Language Processing**  Arman Cohan  
Linguistic, mathematical, and computational fundamentals of natural language processing (NLP). Topics include part of speech tagging, Hidden Markov models,
syntax and parsing, lexical semantics, compositional semantics, machine translation, text classification, discourse, and dialogue processing. Additional topics such as sentiment analysis, text generation, and deep learning for NLP.

**CPCS 578a, Computer Graphics**  Theodore Kim
Introduction to the basic concepts of two- and three-dimensional computer graphics. Topics include affine and projective transformations, clipping and windowing, visual perception, scene modeling and animation, algorithms for visible surface determination, reflection models, illumination algorithms, and color theory.

**CPCS 579b, Advanced Topics in Computer Graphics**  Julie Dorsey
An in-depth study of advanced algorithms and systems for rendering, modeling, and animation in computer graphics. Topics vary and may include reflectance modeling, global illumination, subdivision surfaces, NURBS, physically based fluids systems, and character animation.

**CPCS 580a, Introduction to Computer Vision**  Alex Wong
This course focuses on fundamental topics in computer vision. We begin with the image formation process and discuss the role of camera models and intrinsic calibration in perspective projection. Basic image processing techniques (i.e. filtering) is introduced. After which, we discuss techniques to describe an image, from edges to feature descriptors and methods to establish correspondences between different images of the same scene. The course additionally covers topics in recognition (i.e. image classification, segmentation, detection, etc.) and reconstruction (i.e. stereo, structure-from-motion, optical flow). Machine learning and deep learning based methods in a subset of the topics covered are also introduced. Students get hands-on experience in implementing the techniques covered in the class and applying them to real world datasets and applications. Students taking this course must have successfully passed courses in data structures and object-oriented programming (e.g. CPSC 223a or equivalent courses) and foundational mathematical tools such as discrete math and linear algebra (e.g. CPSC 202 or equivalent courses). It is recommended that students have taken or successfully passed calculus (e.g. MATH 112, MATH 115, MATH 120, or equivalent courses) and linear algebra (e.g. MATH 225, or equivalent courses). A background in statistics, machine learning and deep learning is useful but not required. Experience in programming with Python is preferable, as we use it for assignments and projects. Familiarity with Google Colab and numerical and image processing packages (i.e. NumPy, SciPy, and Sci-kit Image) is helpful throughout the course.

**CPCS 581b, Introduction to Machine Learning**  Alex Wong
This course provides an introduction to machine learning and the problem of learning from data. It introduces several frameworks for formulating the learning task as statistical and computational problems, and explores algorithms for solving them. Topics include supervised learning (classification, regression, kernel methods, neural networks), unsupervised learning (clustering, PCA, dimensionality reduction), reinforcement learning (Markov decision process, online learning), and examples of machine-learning applications in various domains. The course provides a foundation for students interested in pursuing further research or applications of machine learning. Students complete a final project, which can be a synthesis review of recent development and state-of-the-art results in some machine-learning applications. It should also have a research component, for example exploring different algorithms or
generalizing the results to different applications, ideally related to each student’s own research.

**CPSC 582b, Current Topics in Applied Machine Learning**  David van Dijk  
We cover recent advances in machine learning that focus on real-world data. We discuss a wide range of methods and their applications to diverse domains, such as finance, health care, genomics, protein folding, drug discovery, neuroscience, and natural language processing. The seminar is based on a series of lectures by the instructor and guest lecturers, as well as student presentations. The latter are expected to be on recent publications from leading journals and conferences in the field and are followed by discussions. A final project involves the application of a machine-learning method to real-world data. Graduate students are required to work on projects, which are optional for undergraduates. Prerequisites: mathematical tools for computer science (CPSC 202 or equivalent course), linear algebra (MATH 222/MATH 225 or equivalent course), calculus (MATH 120 or equivalent course), or permission of the instructor; and basic coding knowledge (e.g., Python).

**CPSC 583a, Deep Learning on Graph-Structured Data**  Rex Ying  
Graph structure emerges in many important domain applications, including but not limited to computer vision, natural sciences, social networks, languages, and knowledge graphs. This course offers an introduction to deep learning algorithms applied to such graph-structured data. The first part of the course is an introduction to representation learning for graphs and covers common techniques in the field, including distributed node embeddings, graph neural networks, deep graph generative models, and non-Euclidean embeddings. The first part also touches upon topics of real-world significance, including auto-ML and explainability for graph learning. The second part of the course covers important applications of graph machine learning. We learn ways to model data as graphs and apply graph learning techniques to problems in domains including online recommender systems, knowledge graphs, biological networks, physical simulations and graph mining. The course covers many deep techniques (graph neural networks, graph deep generative models) catered to graph structures. We cover basic deep learning tutorials in this course. Knowledge of graphs as a data structure, and understanding of basic graph algorithms are essential for applying machine learning to graph-structured data. Familiarity with Python and important libraries such as Numpy and Pandas are helpful. A foundation of deep neural networks is highly recommended. Experience in machine Learning and Graph Theory are welcomed as well.

**CPSC 584b, Introduction to Human-Computer Interaction**  Marynel Vazquez  
This course introduces students to the interdisciplinary field of human-computer interaction (HCI), with particular focus on human-robot interaction (HRI). The first part of the course covers principles and techniques in the design, development, and evaluation of interactive systems. It provides students with an introduction to UX design and user-centered research. The second part focuses on the emergent field of HRI and several other nontraditional interfaces, e.g., AR/VR, tangibles, crowdsourcing. The course is organized as a series of lectures, presentations, a midterm exam, and a term-long group project on designing a new interactive system. Prerequisites: CPSC 201 and CPSC 202 or equivalents. Students who do not fit this profile may be allowed to enroll with permission of the instructor.
CPSC 585a, Applied Planning and Optimization  Daniel Rakita
This course introduces students to concepts, algorithms, and programming techniques pertaining to planning and optimization. At a high level, the course teaches students how to break down a particular problem into a state-space or a state-action space, how to select an effective planning or optimization algorithm given the problem at hand, and how to ultimately apply the selected algorithm to achieve desired outputs. Concepts are solidified through grounded, real-world examples (particularly in robotics, but also including machine learning, graphics, biology, etc.). These examples come in the form of programming assignments, problem sets, and a final project. General topics include discrete planning, sampling-based path planning, optimization via matrix methods, linear programming, computational differentiation, non-linear optimization, and mixed integer programming. After the course, students are able to generalize their knowledge of planning and optimization to any problem domain. Knowledge of linear algebra and calculus is expected. Students should be familiar with matrix multiplication, derivatives, and gradients.

CPSC 586b, Probabilistic Machine Learning  Andre Wibisono
This course provides an overview of the probabilistic frameworks for machine learning applications. The course covers probabilistic generative models, learning and inference, algorithms for sampling, and a survey of generative diffusion models. This course studies the theoretical analysis of the problems and how to design algorithms to solve them. This course familiarizes students with techniques and results in literature and prepares them for research in machine learning. Prerequisites: Knowledge of machine learning, linear algebra, probability, and calculus.

CPSC 588a, AI Foundation Models  Arman Cohan
Foundation models are a recent class of AI models that are large-scale in terms of number of parameters and are trained on broad data (generally using self-supervision at scale). These models have demonstrated exceptional capabilities in natural language processing, computer vision, and other tasks. Examples of foundation models are GPT-4, ChatGPT, GPT-3, Dall-E, Stable Diffusion, etc. In this course, we discuss building blocks of foundation models, including transformers, self-supervised learning, transfer learning, learning from human feedback, power of scale, large language models, in-context learning, chain-of-thought prompting, parameter-efficient fine-tuning, vision transformers, diffusion models, generative modeling, safety, ethical and societal considerations, their impact, etc. While the course primarily focuses on advances on large language models, we also cover foundation models in computer vision, as well as multi-modal foundation models. Prerequisite: either CPSC 477/577 or CPSC 480/580, or permission of the instructor.

CPSC 611a, Topics in Computer Science and Global Affairs  Joan Feigenbaum and Ted Wittenstein
This course focuses on “socio-technical” problems in computing and international relations. These are problems that cannot be solved through technological progress alone but rather require legal, political, or cultural progress as well. Examples include but are not limited to cyber espionage, disinformation, ransomware attacks, and intellectual-property theft. This course is offered jointly by the SEAS Computer Science Department and the Jackson School of Global Affairs. It is addressed to graduate students who are interested in socio-technical issues but whose undergraduate course work may not have addressed them; it is designed to bring these students rapidly to the
Computer Science

point at which they can do research on socio-technical problems. Prerequisites: Basics of cryptography and computer security (as covered in Yale's CPSC 467), networks (as covered in Yale's CPSC 433), and databases (as covered in Yale's CPSC 437) helpful but not required.

CPSC 612b, Topics in Algorithmic Game Theory  Yang Cai
The course focuses on algorithms and the complexity of equilibrium computation as well as its connection with learning theory and optimization. As many recent machine learning approaches have moved from an optimization perspective to an “equilibration” perspective, where a good model is framed as the equilibrium of a game. The intersection of game theory, learning theory, and optimization is becoming increasingly relevant. The goal of the course is to cover the fundamentals and bring students to the frontier of this active research area. Prerequisite: A course in algorithms (CPSC 365 or 366) and a course in probability theory (MATH/S&DS 241). A course in algorithmic game theory (CPSC 455/555) is helpful but not required.

CPSC 640a or b / AMTH 640a or b / MATH 640a, Topics in Numerical Computation  Vladimir Rokhlin
This course discusses several areas of numerical computing that often cause difficulties to non-numericists, from the ever-present issue of condition numbers and ill-posedness to the algorithms of numerical linear algebra to the reliability of numerical software. The course also provides a brief introduction to “fast” algorithms and their interactions with modern hardware environments. The course is addressed to Computer Science graduate students who do not necessarily specialize in numerical computation; it assumes the understanding of calculus and linear algebra and familiarity with (or willingness to learn) either C or FORTRAN. Its purpose is to prepare students for using elementary numerical techniques when and if the need arises.

CPSC 659a, Advanced Topics in Cryptography: Lattices and Post-Quantum  Cryptography  Aikaterini Sotiraki
This course explores the role of lattices in modern cryptography. In the last decades, novel computational problems, whose hardness is related to lattices, have been instrumental in cryptography by offering: (a) a basis for “post-quantum” cryptography, (b) cryptographic constructions based on worst-case hard problems, and (c) numerous celebrated cryptographic protocols unattainable from other cryptographic assumptions. This course covers the foundations of lattice-based cryptography from fundamental definitions to advanced cryptographic constructions. More precisely, we introduce the Learning with Error (LWE) and the Short Integer Solutions (SIS) problems and study their unique properties, such as the fact that their average-case hardness is based on the worst-case hardness of lattice problems. Next, we cover lattice constructions of advanced cryptographic primitives, such as fully homomorphic encryption and signature schemes. Overall, this course offers insights on the foundations and recent advancements in lattice-based cryptography. There is no required textbook, but certain lectures are based on the book Complexity of Lattice Problems: A Cryptographic Perspective by Daniele Micciancio and Shafi Goldwasser. We supplement the textbook with lecture notes from similar courses taught by Vinod Vaikuntanathan, Oded Regev, Chris Peikert and Daniele Micciancio. Beyond the lecture notes, we also read recent research papers. The course grade is based on multiple assignments and a final project. Prerequisites: This is an advanced course, which requires mathematical maturity and comfort with
linear algebra. The course also assumes prior knowledge of fundamental notions in cryptography (CPSC 467 or equivalent).

**CPSC 690a or b, Independent Project I**  Vladimir Rokhlin
By arrangement with faculty.

**CPSC 691a or b, Independent Project II**  Staff
By arrangement with faculty.

**CPSC 692a or b, Independent Project**  Staff
Individual research for students in the M.S. program. Requires a faculty supervisor and the permission of the director of graduate studies.

**CPSC 990a, Ethical Conduct of Research for Master’s Students**  Inyoung Shin
This course is 0 credits for YC students.

**CPSC 991a / MATH 991a, Ethical Conduct of Research**  Inyoung Shin
0 Course cr