Undergraduate Major Handbook

B.S. Computer Science

MAJOR CODE

Version 1.0 (Mar 17, 2025)

1. The University Mission, Vision, and Graduate Attributes

Mission

The mission of Fulbright University Vietnam is to inspire new generations of leaders and ambitious thinkers of diverse origin to serve Vietnamese society and contribute to a better world. As Vietnam's first liberal arts university, founded on the promise of Vietnamese-American partnership, we practice and teach continuous innovation in pursuit of bold, ethical answers to local and global challenges. Our design for academic programs of exceptional quality embraces scientific, social, and humanistic modes of inquiry and action, optimized for a dynamic and impatient future.

Vision

In 25 years, Fulbright University Vietnam will be the top choice for students in Vietnam and across Asia. Our students will represent the full regional, disciplinary, socio-economic and gender diversity we deeply value.

Supported by world-renowned faculty, our students will engage in research, internships and student organizations in Vietnam and across the globe. In turn, they will pursue opportunities in the world's best companies and graduate programs.

As our students thrive, Vietnamese higher education will adapt alongside us, increasingly integrating student-centered pedagogy and liberal arts approaches. Vietnamese youth will become more effective change-makers and leaders to tackle 21st-century challenges.

Graduate Attributes

Knowledge - Students will acquire the knowledge of human perspectives and the natural world through:

GA1. *Interdisciplinary Innovation* - The liberal arts approach to integrating foundational knowledge and methods across diverse fields of inquiry.
GA2. *Disciplinary Depth* - In-depth understanding, application, and synthesis of specialized knowledge and skills from within a defined major course of study.
GA3. *Cultural Outlook* - The awareness of and sensitivity to local and global challenges, with an emphasis on understanding and developing Vietnam in the twenty-first century.

Skills - Students will develop the following four intellectual and practical competencies for future success:

GA4. *Creative Thinking* - The understanding and synthesis of ideas, approaches, and expertise to find novel and innovative solutions.
GA5. *Critical Reasoning* - The systematic exploration of issues, questions, or objects through comprehensive analyses of evidence, resulting in informed conclusions.
GA6. *Effective Communication* - The clear and effective presentation of meaning to increase knowledge, foster understanding, or promote change in the listeners' attitudes, values, beliefs, or behaviors.
GA7. *Information Literacy* - Strategically navigate, critically evaluate, and ethically use digital and non-digital information to understand and solve problems in a changing and complex information environment.

Mindsets - Students will form positive attitudes that foster lifelong learning and commitment to society as good citizens through the following associated competencies:

GA8. *Ethical Responsibility* - The recognition of the ethical ramifications of personal and public values and actions, and understanding of how they impact the civic life of our communities.
GA9. *Intellectual Curiosity* - The pursuit of purposeful and ongoing learning to enhance knowledge and improve oneself personally and professionally throughout their life.
GA10. *Collaborative Engagement* - The ability to work successfully as part of a team, characterized by respectful interaction, adaptive thinking, and meaningful contribution.

2. B.S. Computer Science: Program Description and Objectives

Program Description

The Computer Science (CS) major prepares students with an adaptable skill set to respond to the astonishing speed of technological change and develop solutions for the problems of today and tomorrow. Using a student-centered, interdisciplinary, and future-focused approach, the Computer Science major aims to educate the next generation of local leaders who will make a meaningful and lasting societal impact both in Vietnam – one of the most quickly emerging and innovative technology economies in the world – and beyond. As part of the major, students will be equipped with the foundational knowledge in Computer Science and relevant disciplines. They will be exposed to essential areas of the CS discipline including theory, systems, and applications. They will learn about the underlying mathematical ideas that are critical for computation, establish proficiency in the process of designing systems and applications, gain experience in collecting and analyzing data using modern technologies, and begin to develop an understanding for the role of users in the design of systems and applications. Courses in Computer Science go beyond content to help students learn through direct experiences in projects and problems. Students will also have the opportunity to further focus their studies by selecting a concentration, such as software engineering, artificial intelligence and machine learning, data science, business analytics. The Computer Science major at Fulbright is designed to prepare students for work in industry or continue their lifelong learning as well as potential graduate-level studies.

Program Educational Objectives (PEOs)

PEO1. To equip students with the necessary skills and knowledge to gain employment in industry or to study in graduate schools.
PEO2. To ingrain in students a strong code of professional and personal ethics, and the ability to effectively and creatively collaborate and communicate in the workplace and beyond.
PEO3. To cultivate students to become innovators and thinkers to solve computer science related challenges.

Program Learning Outcomes (PLOs)

PLO no.	Description of PLO	FUV Graduate Attribute
1	Demonstrate an understanding of fundamental computer science by illustrating core concepts, comparing methods, analyzing and solving problems in both theory and systems.	GA2, GA4, GA5, GA6, GA7
2	Explain emerging aspects of the computer science discipline (e.g., artificial intelligence, machine learning, data science, business analytics, digital media, etc.).	GA2, GA4, GA5, GA6, GA7, GA9
3	Develop real-world applications (e.g., software systems and frameworks, mobile apps, data pipeline, etc.) by using knowledge of computer science and other disciplines.	GA1, GA3, GA8, GA9, GA10

3. Graduate Studies and Career Paths for the B.S. Computer Science

Graduates with the B.S. Computer Science have successfully gained employment as:

- Software engineer
- Testing engineer
- AI engineer
- Data engineer
- Business analyst
- Product owner
- Product development manager

Computer Science students have successfully applied to graduate programs in the United States and Europe. These programs include:

- MSc/PhD Computer Science
- MSc Data Science
- MSc Bioinformatics

4. Program Structure and Curriculum

A Bachelor of Science in Computer Science is awarded following the successful completion of at least 128 credits, including:

- Core and exploratory requirements
- Elective and experiential learning
- Major requirements

Core curriculum

Students need to complete the following four core courses:

- Global Humanities (4 credits)
- Modern Vietnamese Culture and Society (4 credits)
- Design and Systems Thinking (4 credits)
- One course in QUEST (Quantitative Understanding and Empirical Scientific Thinking), a core group consisting of the following four courses:
    - Quantitative Reasoning for a Digital Age (4 credits)
    - Scientific Inquiry (4 credits)
    - Introduction to Statistics (4 credits)
    - Programming for Data Science and Visualization (4 credits).

These courses are pre-assigned to students in their first year or the first semester of the second year.

Exploratory requirements

Exploratory requirements encourage students to step out of their comfort zone by exploring broad areas of study and discover more fully where their interests and passions lie. Students need to complete 8 credits (2 courses at the 100-level or 200-level) chosen from each of the three categories: E1- Arts and Humanities, E2-Social Sciences, and STEM.

No course can be taken to fulfill more than one category. At most, 8 exploratory credits can be taken as Pass/No Pass. Up to two 8 exploratory credits (2 courses) can be counted towards the student's major requirements. If a student has more than one major, the 8-credit limit is independently applied to each major.

Students should aim to complete the Exploratory requirements by the end of their second year.

Notes:

- *Class of 2023 students are exempted from the Exploratory requirements.*
- *There are four exploratory categories for the students of classes of 2024, 2025, 2026, and 2027: E1-Arts and Humanities, E2-Social Sciences, E3- Sciences and Engineering, and E4-Mathematics and Computing.*
- *Class of 2024 students need to complete 16 credits (4 courses) to fulfill their Exploratory requirements, one course from each category, with at most 4 out of 16 credits counted as Pass/No Pass.*
- *Students of classes of 2025, 2026, and 2027 need to complete 32 credits (8 courses), eight credits from each category, to fulfill their Exploratory requirements.*

Elective and Experiential Learning Requirements

Electives

To complete the degree, students are free to take elective courses from the Course Catalogue. While the number of elective credits may vary, the total number of earned credits must be at least 128. Electives enable students to explore different subjects across the curriculum to build a broad and diverse background or to pursue more specialized studies within their major. Alternatively, students may use electives to fulfill requirements for another minor or major.

Experiential Learning

This requirement is subject to the school policy.

Major Requirements (56 credits)

- CS 101: Introduction to Programming (4 credits)
- CS 201: Data Structures & Algorithms (4 credits)
- CS 203: Computer Organization (4 credits)
- MATH 202: Discrete Math (4 credits)
- Three intermediate courses (12 credits)
- CS 302: Algorithms & Theory of Computing (4 credits)
- Four advanced courses (16 credits)
- Capstone I OR an additional intermediate course OR an advanced course (4 credits)
- Capstone II OR an additional intermediate course OR an advanced course (4 credits)

To formally declare Computer Science as their major, the student must complete at least two Computer Science courses (MATH 202 - Discrete Math is also counted).

Minor Requirements (24 credits)

The minor requires students to take a total of six courses:

- One programming course (4 credits) which is either
    - CS 101: Introduction to Programming
    - or CS 103: Programming for Data Science and Visualization
    - or CS 104: Introduction to Computer Science
- One 100-level or 200-level CS course (4 credits)
- CS 201 – Data Structures & Algorithms (4 credits)
- One intermediate course (4 credits)
- One 200-level or 300-level CS course (4 credits)
- One advanced course (4 credits)

Major Outline: Curricular Structure and Progression

All students are encouraged to take the course Introduction to Computer Science first to explore all different areas of computer science. Then the students are required to take five courses that will lay out the foundation of programming, mathematics, and hardware. After having the knowledge from these foundation courses, the students will continue their journey with three intermediate courses, which are designed to cover the important and basic knowledge in the major aspects in computer science. Following the completion of their foundation and intermediate courses, students will have flexibility in their choice of four elective advanced courses to explore their areas of interest deeper. Finally, in the fourth year, students can choose to do the capstone to graduate with honors or to take two more intermediate or advanced courses to graduate without honors.

Foundational Courses:

Sample Foundational Courses:

- Introduction to Programming
- Data Structures & Algorithms
- Computer Organization
- Discrete Mathematics
- Algorithms & Theory of Computing

Intermediate Courses:

Sample Intermediate Courses:

- Software construction
- Programming Language Paradigms
- Operating Systems
- Computer Networks
- Introduction to Artificial Intelligence

Advanced Courses:

Sample Advanced Courses

- Foundations of Software Engineering
- Database Systems
- Computer Security
- Cloud Computing
- Human-Computer Interaction
- Computer Graphics
- Game Design & Development
- Machine Learning
- Deep Learning
- Advanced Deep Learning
- Computer Vision
- Natural Language Processing
- Optimization
- Computational Social Media
- Bioinformatics

CS 401/402: Capstone Project

Eligibility criteria for enrollment in CS 401: Capstone I include a minimum major GPA of 3.5 and approval of a successful application. More detailed guidelines for the Capstone in Computer Science can be found in the Computer Science Capstone Handbook.

COMPUTER SCIENCE FLOWCHART

Sample Student Journeys for the Major and Minor

Sample Major Plan (15 equivalent courses)

Year 1	Year 2	Year 3	Year 4
Core Curriculum Introduction to CS Computer Science I 2 ELECTIVES	Discrete Math Computer Science II Computer Organization Intermediate CS course 4 ELECTIVES	Algorithms & Theory of Computing Intermediate CS course Intermediate CS course Advanced CS course Advanced CS course 3 ELECTIVES	Advanced CS course Advanced CS course Capstone I & II (or two elective intermediate/advanced courses) 4 ELECTIVES

Sample Minor Plan

Year 1	Year 2	Year 3	Year 4
Core Curriculum Introduction to CS 1 Major course 2 ELECTIVES	Computer Science I Computer Science II 3 Major courses 3 ELECTIVES	Intermediate CS course Intermediate CS course 5 Major courses 1 ELECTIVES	Advanced CS course 5 Major courses 2 ELECTIVES

Course Alignment with the Program Learning Outcomes (PLOs)

PLO1. Demonstrate an understanding of fundamental computer science by illustrating core concepts, comparing methods, analyzing and solving problems in both theory and systems.

PLO2. Explain emerging aspects of the computer science discipline (e.g., artificial intelligence, machine learning, data science, business analytics, digital media, etc.).

PLO3. Develop real-world applications (e.g., software systems and frameworks, mobile apps, data pipeline, etc.) by using knowledge of computer science and other disciplines.

COURSE CODE	COURSE TITLE	PLO1	PLO2	PLO3
CS101	Introduction to Programming	R	I	
CS103	Programming for Data Science and Visualization	R	I	
CS104	Introduction to Computer Science	I	I	
CS201	Data Structures & Algorithms	R	I	
CS203	Computer Organization	R	I	
CS207	Software Construction	R	I	
CS208	Machine Learning for Data Science	R	I	
CS211	Operating Systems	R	I	
CS212	Programming Language Paradigms	R	I	
CS301	Foundations of Software Engineering	M	R	M/A
CS302	Algorithms & Theory of Computing	M	I	
CS303	Human-Computer Interaction	M	I	
CS304	Database Systems	M	I	
CS306	Computer Security	M	R	M/A
CS307	Machine Learning	M	I	
CS310	Web Applications & Human-Computer Interaction	M	R	M/A
CS311	Advanced Deep Learning	M	R	M/A
CS312	Natural Language Processing	M	R	M/A
CS313	Deep Learning for AI	M	R	
CS3xx	Bioinformatics	M	I	M/A
MATH308	Statistical Learning	M	I	
MATH301	Optimization	M	I	
ENG301	Computer Vision	M	I	M/A
CS 401/402	Capstone I and II	M/A	M/A	M/A

I: Introduced R: Reinforced/Practiced M: Mastery A: Assessment opportunity

5. Teaching, Learning, and Assessment

Teaching and assessment in the Computer Science major are designed on the principle of active and collaborative learning. In addition to summative methods, courses rely on continuous formative assessment of student learning through discussion (instructor-led and student-led), in-class activities, and applied learning.

Program Learning Outcomes	**Teaching and Learning Methods**	**Assessment Examples**
1. Demonstrate an understanding of fundamental computer science by illustrating core concepts, comparing methods, analyzing and solving problems in both theory and systems.	Lecture Lab Worksheet Group discussion	Programming test Quiz Exam Group presentation Capstone
2. Explain emerging aspects of the computer science discipline (e.g., artificial intelligence, machine learning, data science, business analytics, digital media, etc.).	Reading Lecture Lab Group discussion Class project	Quiz Exam Group presentation Capstone
3. Develop real-world applications (e.g., software systems and frameworks, mobile apps, data pipeline, etc.) by using knowledge of computer science and other disciplines.	Lecture Lab Field trip Class project	Presentation Capstone

6. Course Catalogue

*Subject to staffing

** All courses are one semester (4 credits) unless indicated otherwise

COURSE CODE	**COURSE TITLE**	**CROSS-LISTING**	**AVAILABILITY (tentative)**	**PREREQUISITES**
CS101	Introduction to Programming		Once a year	None
CS103	Programming for Data Science and Visualization		Twice a year	None
CS104	Introduction to Computer Science		Once every two years	None
CS201	Data Structures & Algorithms		Once a year	CS101
CS203	Computer Organization		Once a year	CORE105
CS207	Software Construction		Once a year	CS201
CS208	Machine Learning for Data Science		Once a year	CS103
CS210	Computer Networks		**On holding**	CS201
CS211	Operating Systems		Once a year	CS201
CS212	Programming Language Paradigms		Once a year	CS201
CS301	Foundations of Software Engineering		Once a year	CS207
CS302	Algorithms & Theory of Computing		Once a year	CS201, MATH202

CS303	Human-Computer Interaction		**On holding**	One 200-level CS course
CS304	Database Systems		Once a year	CS201
CS306	Computer Security		**On holding**	CS211
CS307	Machine Learning		Once a year	CS201
CS310	Web Applications & Human-Computer Interaction		**On holding**	CS201
CS311	Advanced Deep Learning		Once a year	CS313
CS312	Natural Language Processing		Once a year	CS201, MATH103
CS313	Deep Learning for AI		Once a year	CS208
CS3xx	Bioinformatics	IS	**On holding**	One 200-level CS course
Courses in other majors cross-listed with Computer Science				
MATH308	Statistical Learning	CS	TBA	
MATH301	Optimization	CS	TBA	
ENG301	Computer Vision	CS	TBA	

Course Descriptions

CS101: Introduction to Programming

Prerequisites: None (CS104 is recommended)

This course aims to equip students with the skills necessary to solve computational problems using a high-level programming language. Students will develop their abilities to use (i) fundamental programming constructs such as branching statements, loops, functions, and recursion and (ii) abstract data types such as lists and maps. Students will also be exposed to basic concepts of object-oriented programming.

CS103: Programming for Data Science and Visualization

Prerequisites: None

The Programming for Data Science and Visualization course aims at providing students with a set of skills to analyze the nature of data across different domains. The course demonstrates data visualization by understanding, questioning, and problematizing how data are generated, analyzed, and used. The students will be able to apply concepts and skills to visualize data, interpret the findings, and examine the impacts of data-driven decision. The course will also get the students familiar with Python programming language and its libraries to analyze, visualize and discover data insight leading to a proper data-driven decision-making process.

CS104: Introduction to Computer Science

Prerequisites: None

The course Introduction to Computer Science aims at teaching students the basics of computer science, programming, and data science, along with potential career paths in the fields. Computer science is the study of how computers work and how to use them for different purposes. Programming is the skill of writing instructions for computers to follow. The students will be able to create their own programs that can perform various tasks, understand how computers solve problems, and apply their knowledge and skills to any field that involves computing. The course will also help learners

develop their logical thinking, problem-solving, and creativity skills that are essential for any discipline or career. The course covers diverse topics, including data science, database, web development, computer networks and systems.

CS201: Data Structures & Algorithms

Prerequisites: CS101

How do we write large programs to solve real-world problems correctly and efficiently? How do we design programs that can be tested and extended easily? This course aims to achieve these goals by expanding the content from CS101 (Introduction to Programming), with a greater focus on theoretical concepts, abstraction, and larger programs. Topics include object-oriented programming (classes, objects, subclasses, and inheritances), unit testing and refactoring, fundamental data structures (arrays, lists, stacks, queues, trees, hash tables, and graphs), basic algorithms (sorting and searching) and their analysis. Java is the main programming language used in the course but some Python/C++ implementations are also introduced.

CS203: Computer Organization

Prerequisites: CORE105– Design & Systems Thinking

Computing systems, such as mobile phones, laptops or personal computers, have been an important parts of human beings' everyday life. How are computing systems are designed and implemented? In principle, computing systems often involve many layers of abstraction, from gates and circuits through machine and assembly code to software libraries and applications. This course introduces students to the abstract design and implementation of computer systems from the digital level in the hardware upwards to the interface between the hardware and the software. In particular, the course starts by revisiting the concept of bits and introducing arithmetic and logical operations on bits. Next, it takes the students from the building of logic gates based on the transistor as a switch, gated latches to more complex logic structures. The knowledge is then applied to implement memory and a finite state machine. From there, students study the instruction cycle that the central processing unit (CPU) of a computer follows. As an example, students study a particular computer that is able to capture the important structures of a modern computer, while simple enough to facilitate complete understanding and hands-on programming experiences. Students also explore decisions and tradeoffs involved in the design and implementation. Applied projects and/or lab assignments might include the design and simulation of a CPU, and the tools used to program low-level systems.

CS204: Introduction to Artificial Intelligence

Prerequisites: Take at least one 100-level CS course

This course introduces students to the basic knowledge on Artificial intelligence.Artificial intelligence (AI) is a research field that studies how to realize the intelligent human behaviors on a computer. The ultimate goal of AI is to make a computer that can learn, plan, and solve problems autonomously. In this course, student will learn the foundational principles and practice implementing some of these applications including representation, problem solving, and learning methods of artificial intelligence. Accordingly, students should be able to develop intelligent systems by assembling solutions to concrete computational problems; understand the role of knowledge representation, problem solving, and learning in intelligent-system engineering; and appreciate the role of problem solving, vision, and language in understanding human intelligence from a computational perspective.

CS207: Software Construction

Prerequisites: CS201

This course provides students the opportunities to strengthen their understanding of object-oriented concepts and to build their skills and experience in developing software systems by applying OO concepts and design principles. Topics covered in this course include class, object, information hiding, encapsulation, inheritance, interface, polymorphism, design patterns, multi-threading, database connection, unit testing, and Unified Modeling Language (UML). Students will work in teams to practice designing software using UML models such as use-case, sequence, and class diagrams. They will also implement the design via building Android applications using Java.

CS208: Machine Learning for Data Science

Prerequisites: CS103

The course Machine Learning for Data Science aims at providing students with a basic understanding of the principles of machine learning algorithms and deriving practical solutions using predictive analytics. The course demonstrates mathematical concepts and hands-on skills required for the algorithms that are typically used in practice. The students will be able to apply concepts and skills to analyze data across different domains, interpret the findings, then build learning systems and comprehend their performance. Topics include: supervised learning (Naïve Bayes classification, Decision Tree, Random Forest, Support Vector Machine, Deep Neural Network); unsupervised learning (K-means clustering, Hierarchical clustering). The course will also discuss recent applications of machine learning, such as to robotics, autonomous driving systems, face and speech recognition, text and web data processing, etc.

CS210: Computer Networks

Prerequisites: CS201

This course is an in-depth study of computer networks. Using the Internet as a vehicle this course explores issues, concepts and techniques that are essential in building modern computer networks with emphasis on architectures, protocols and implementation issues. The main objective of this course is to gain a solid knowledge on technologies that build the infrastructure for modern networked applications such as cloud computing, Web browsing, social networking, multimedia streaming etc…

CS211: Operating Systems

Prerequisites: CS201

The operating system serves as an interface between user programs and hardware devices. Understanding its workings is essential for both application developers, as it aids in enhancing application performance, and systems developers, who may need to extend or create new operating systems for emerging devices. This course addresses this necessity by introducing fundamental concepts and principles of modern operating systems. The course is divided into three main parts. The first part delves into concurrency, covering topics such as processes, threads, CPU scheduling, context switching, synchronization, and deadlock. The second part explores memory management issues, including linking, allocation, address translation, segmentation, paging, and swapping. The third part introduces file systems, discussing disk management, file organization, and crash recovery. After these three parts, the course will conclude with advanced topics such as scalability, virtualization, security, and support for distributed computing.

CS212: Programming Languages

Prerequisites: CS201

How many programming languages should a professional software developer know? The answer is often "the more, the better", as exposure to multiple programming languages provides a wider array of options for solving specific problems. However, it is impractical to learn all or even most programming languages, given the sheer number of languages and their continuous evolution. This course addresses this challenge by delving into the foundational concepts of modern programming languages and offering a comparative study of several significant language groups known as programming paradigms. Students will explore various programming paradigms, such as imperative, object-oriented, declarative, and functional programming, to gain insights into their strengths, weaknesses, and optimal use cases. In addition, the course also introduces the design of compilers to implement essential programming language concepts, helping students optimize their code and understand errors and bugs at a deeper level. Furthermore, advanced topics such as meta-programming and program synthesis are also discussed, enabling students to envision the evolving landscape of programming languages in the future.

CS301: Foundations of Software Engineering

Prerequisites: CS207

Large scale systems, including web-based applications, are fundamentally different and more complex than short programs. Software engineering is the study of how to develop large-scale software systems. In this course, students will learn about and practice a variety of interrelated activities and processes, including requirements analysis, system design, documentation, implementation, and testing. Students will also participate in and contribute to a major software project. Projects might vary from year to year depending on student and faculty interests, and could include mobile, web, game, or embedded systems.

CS302: Algorithms & Theory of Computing

Prerequisites: CS201 & MATH 202

Are there any efficient algorithms to find the shortest path between two cities on a map, schedule courses at FUV, or detect if a computer program is a virus or not? This Algorithms and Theory of Computing course aims to provide students with concepts and techniques to address these questions. Three common techniques (divide-and-conquer, greedy, and dynamic programming) and some algorithms on graphs are introduced to help students learn how to design efficient algorithms for problems in practice. Students will also learn basic computational models (finite automaton, push-down automaton, and Turing machine) to understand the concepts of decidability and tractability. By that, students will be able to prove if a problem is hard to solve efficiently or even can not be solved by any algorithm.

CS303: Human-Computer Interaction

Prerequisites: Take one 200-level CS course

What makes a smartphone app, a website, or any physical product, easy to use? And how can we design such user interfaces in hardware and software? The field of human-computer interaction (HCI) is concerned with these questions. In this course, students will learn about approaches to effectively design interfaces (for example through lo-fidelity prototyping and user-centered design methods) and will be exposed to different techniques to assess usability (including interface analysis and heuristic evaluations). Students will also have opportunities to design and assess their own software interfaces.

CS304: Database systems

Prerequisites: CS201

This course offers a comprehensive examination of the design and implementation of relational database management systems (DBMS). Teaches the logical organization of databases, E_R design, normalization and use of SQL for data description and retrieval, including triggers and stored procedures, concurrency and security issues and typical solutions. Includes a major project building web interfaces to databases using PHP and MySQL. Introduction to NoSQL solutions.

CS306: Computer Security

Prerequisites: CS211

This course is a comprehensive study of the security principles and practices for network and computer systems. Topics to be covered include basic security concepts, common attacking techniques, basic cryptographic tools and secure protocols. Defense techniques such as authentication, access control and network intrusion detection will also be discussed. An introduction to networking technologies and principles will also be covered. Hands-on laboratories will be used to reinforce student learning of concepts, principles and practices that are important to the security of networks and computer systems.

CS307: Machine Learning

Prerequisites: CS201

Machine learning is concerned with the question of how to construct programs that automatically improve their performance through experience. This course introduces the principles underlying the design of existing supervised and unsupervised machine learning algorithms. Topics to be covered include the following:

- Supervised learning models and methods: Decision trees, neural networks, nearest-neighbor algorithms, Bayesian learning, Hidden Markov Models, support vector machines
- Unsupervised learning: Clustering
- Reinforcement learning: Markov Decision Processes, online supervised learning, certainty equivalent learning, temporal difference learning, Q-learning
- General techniques: Feature selection, cross-validation, maximum likelihood estimation, expectation-maximization, gradient descent, ensemble learning

CS311: Advanced Deep Learning

Prerequisites: CS313

The course Advanced Deep Learning aims at providing students with a good understanding of advanced architectures of deep neural networks and algorithms, along with deriving practical AI solutions in various domains such as economics, fintech, computer vision, natural language processing. The course demonstrates mathematical concepts and hands-on skills required for the algorithms that are typically used in practice. The students will be able to apply concepts and skills to analyze complex data across different domains, then build learning systems and comprehend their performance. The course covers diverse topics including Transfer Learning, Generative Adversarial Network (GAN), Reinforcement Learning (RL), Attention and Transformer Networks, Graph Neural Network (GNN). On the other hand, advanced applications of deep learning will also be addressed, such as natural language processing, robotics, autonomous driving systems, time-series applications, etc

CS312: Natural Language Processing

Prerequisites: CS201 and MATH103

This course is designed to introduce students to the fundamental concepts and ideas in natural language processing (NLP) which is a subfield of linguistics, computer science, and artificial intelligence. Students will study the computational properties of natural languages and algorithms to interpret and manipulate these languages. The course starts with simple statistical models and word embeddings. Then the focus is shifted to the architecture of large language models. The final part of the course is about how pre-trained large language models can be utilized to solve specific tasks. In addition, applications such as translation, summarization, extracting information, and question answering are also introduced and discussed.

CS313: Deep Learning for AI

Prerequisites: CS208

The course Deep Learning for Artificial Intelligence aims at providing students with a good understanding of deep neural networks and algorithms, along with deriving practical AI solutions in various domains such as economics, fintech, computer vision, natural language processing. The course demonstrates mathematical concepts and hands-on skills required for the algorithms that are typically used in practice. The students will be able to apply concepts and skills to analyze data across different domains, then build learning systems and comprehend their performance. The course covers diverse topics including introductive deep learning, popular deep neural networks, data augmentation, transfer learning and recurrent neural networks. On the other hand, recent applications of deep learning will also be addressed, such as stock/cryptocurrency price prediction with time series data, natural language processing, robotics, autonomous driving systems, etc.

CS3xx: Bioinformatics

Prerequisites: One 200-level CS course

Recent advancements in biotechnology, particularly in high throughput techniques, have enabled us to collect an unprecedently volume of biological data. This course provides a broaden introduction to how we can analyze these data to answer research questions in biology and medicine. From the computational side, this course focuses on algorithms and machine learning methods to analyze biological data efficiently. From the biological side, this course focuses on how we can leverage the data to discover mechanisms of biological processes or diseases. Topics of this course include sequencing data analysis, gene expression data analysis, gene regulation data analysis, gene function analysis, biological data clustering, biological network analysis, and biological databases. Applications of these topics in studying cancer, autism, gene therapy, or drug discovery are also introduced.

APPENDIX I

The Computer Science Program Curriculum has been constructed with the reference to programs:

- BS Computer Science, Stanford University
- BS Computer Science, Carnegie Mellon University
- BS Computer Science, Cornell University