DEPARTMENT OF COMPUTER SCIENCE

MSc in Computer Science

Programme Syllabus

Ramakrishna Mission Vivekananda Educational and Research Institute

Belur Campus

Programme Outcomes

1. Inculcate critical thinking to carry out scientific investigation objectively without being biased with preconceived notions.
2. Equip the student with skills to analyze problems, formulate an hypothesis, evaluate and validate results, and draw reasonable conclusions thereof.
3. Prepare students for pursuing research or careers in industry in mathematical sciences and allied fields.
4. Imbibe effective scientific and/or technical communication in both oral and writing.
5. Learn the necessary tools and programming skills for building apps.
6. Continue to acquire relevant knowledge and skills appropriate to professional activities and demonstrate highest standards of ethics in learning and work

Programme Specific Outcomes

1. Understanding the theoretical underpinnings in computing and computing systems.
2. Knowledge of the synergy between s/w and h/w through the study of computer architecture, compilers, and systems programming.
3. Knowledge and application of various algorithms, algorithmic methods, and data structures in solving computational problems drawn from various fields such computer graphics, computational geometry, distributed systems, data mining, mobile computing.
4. Training in contemporary technologies such as IoT, Blockchain etc.
5. Use software development tools, software systems on modern computing platforms.
6. Communicate computer science concepts, designs, and solutions effectively and professionally through written and oral forms.

MSc in Computer Science
Syllabus

Contents

Artificial Intelligence

Course Code: CS246

Course Description: Artificial Intelligence (AI) is a vast field. Nowadays people often think that AI is nothing but machine Learning and the reason is mainly the widespread use of machine learning. The fact is machine learning is just a part of the domain of AI. We will try to explore the full breadth of the course, which encompasses logic, probability, reasoning, learning, decision making, and action.
We will define AI as the study of intelligent agents where each such agent receives precept from the environment and act upon the precept. We will discuss goal-based models, knowledge-based models,
After a basic introduction, we will jump into different modules like problem-solving (for example, Sudoku solver), planning, and acting under uncertainty. Application of artificial intelligence in various fields like natural language processing, perception (computer vision), robotics will be covered.

Prerequisite(s): Basic knowledge of computer science such as algorithms, data structures, probability, linear algebra.
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Artificial Intelligence: A Modern Approach ;
Stuart Russell and Peter Norvig

Course Objective: Students will get to know

(1) Understand the fundamental concepts of Artificial Intelligence (AI), including intelligent agents, environments, and the broad scope of AI beyond machine learning

(2) Learn various problem-solving strategies such as searching, informed exploration, and constraint satisfaction, and apply them to solve complex problems

(3) Explore planning techniques and decision-making processes in the presence of uncertainty, including probabilistic reasoning and making both simple and complex decisions

(4) Gain an overview of learning methodologies, including supervised, unsupervised, and reinforcement learning, and understand their applications in AI problem-solving

(5) Understand the principles of Bayesian Networks and Hidden Markov Models (HMMs), including their representations, inference methods, and applications in real-world scenarios
(6) Explore the diverse applications of AI in fields such as robotics, computer vision, natural language processing, and computer games, including deep reinforcement learning techniques

Course Outcome: The students will be able to
(1) Analyze and identify problems suitable for the application of Artificial Intelligence techniques across various domains, including but not limited to problem-solving, planning, and decision-making
(2) Implement selected AI techniques to effectively address and solve complex problems, demonstrating proficiency in foundational AI concepts such as problem-solving by searching, constraint satisfaction, and adversarial search
(3) Evaluate and apply state-of-the-art AI techniques, including probabilistic reasoning, reinforcement learning, and Bayesian Networks, to address real-world challenges in diverse fields such as robotics, computer vision, and natural language processing
(4) Demonstrate a comprehensive understanding of key AI principles and methodologies, as outlined in the course syllabus, including the principles of intelligent agents, uncertain knowledge and reasoning, and learning algorithms.
(5) Critically assess the applicability of basic and advanced AI techniques to different problem domains, making informed decisions about selecting and implementing appropriate methods based on problem characteristics and constraints.
(6) Through a combination of theoretical understanding, practical application, and critical analysis, students will develop the necessary skills and knowledge to contribute effectively to the field of Artificial Intelligence and apply AI techniques to solve complex real-world problems.

Grade Distribution:

Assignments and Class Tests	30%
Midterm Exam	20%
Final Exam	50%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Week	Topic
1	Introduction, Intelligent Agents and Environments
2	Problem Solving: Solving problem by searching
3	Problem Solving: Informed search and exploration
4	Problem Solving: Constraint satisfaction problem
5	Problem Solving: Adversarial Search
6	Uncertain Knowledge and Reasoning: Uncertainity
7	Uncertain Knowledge and Reasoning: Probabilistinc reasoning
8	Uncertain Knowledge and Reasoning: Making simple decisions
9	Uncertain Knowledge and Reasoning: Making complex decisions
10	Learning: Brief overview of various learning methods
11	Learning: Reinforcement learning
12	Bayes Net (1)
13	Bayes Net (2)
14	Application: Computer vision, Natural Language Processing
15	Deep Reinforcement Learning
16	Course Wrap-up

Applied Machine Learning (reading course)

Course Code: DA342

Course Description: DA342 will explore some Machine Learning (ML) techniques to solve different practical problems that occur in one of the area of *Audio, Text, Image, and Video processing* task. We will start with the definition of the particular problem and some initial results. Then try to touch some state-of-the-art results from the papers published in the last *ten* years in the top venues like ICML, NeurIPS, CVPR, ICCV, ECCV and related journals.

Prerequisite(s): Student should have some knowledge in

- Mathematics: Linear Algebra, Multivariate Calculus, Basic Optimization, Basic Probability and Statistics
- Computer programming: Any one from C/C++/Python(recommended for the class project)/MATLAB/Octave
- Basis ML concepts (classification, regression, etc.)

Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Probabilistic Machine Learning: An Introduction;
Kevin Patrick Murphy

Pattern Recognition and Machine Learning;
Christopher M. Bishop

Statistical Learning with Sparsity: The Lasso and Generalizations;
Trevor Hastie, Robert Tibshirani and Martin Wainwright

Pattern Classification and Scene Analysis;
R. O. Duda, P. E. Hart and D. G. Stork

Foundations of Machine Learning;
Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar

Understanding Machine Learning: From Theory to Algorithms;
Shai Shalev-Shwartz and Shai Ben-David

Speech and Language Processing;
Dan Jurafsky and James H. Martin.

Deep Learning;
Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar

Course Objective: Students will get to know

(1) critical reading and analysis skills for different practical applications of machine learning algorithms

(2) identifying main concepts, key arguments, evaluating evidence, and understanding the author's perspective and biases

(3) recognizing the limitations of machine learning algorithms in addressing real-world problems

Course outcomes: After successful completion of this course, students will be able to:

(1) decompose a data related problem (in the area of *Audio, Text, Image, and Video processing*) into subproblems in ML and identify potential solutions

(2) apply state-of-the-art ML tools and techniques to solve data related problems that occur in one of the area of *Audio, Text, Image, and Video processing* task

(3) start critically read, assess, and synthesize scientific literature in applied ML and related fields

(4) discuss the potential and limitation of ML models in a particular area

Grade Distribution:

Paper reading and presentation	40%
Project and report writing	50%
Paper/Project review	10%

Course Outline (tentative) and Syllabus:

There is no predefined(fixed) syllabus for this course! The instructor will give some topics based on the current hot research topic and industry problems with a brief introduction. Students have to choose one topic based on their interests. The instructor will share some materials related to the selected topics. Each student has

to present those at least once a week and have to do a project on that or a related topic.
In 2022, initially, four students opted for this course, and two have left! They have selected the following topics:

- Student-1: **Fake news/misinformation detection.**
- Student-2: **Tackling climate change with machine learning.**
- Student-3 (left): **Player/team performance prediction in IPL game.**
- Student-4 (left): **Sparse coding/learning and its application in images and video data.**

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Reading assignments are from reference textbook will be mentioned in the suggested reading material in the *course url*.

Week	Content
Week 1	• Introduction to the course • Course logistics • A brief discussion of some practical problems (research and industry)
Week 2	• A brief discussion of more practical problems (research and industry) and asked the student to explore some focused workshops happening in the top venues like ICML, NeurIPS, CVPR, ICCV, ECCV, ICLR etc.. • Assignment of 2-3 topics to each student.
Week 3	• Particular topic selection for each student – Student has to give a brief presentation on the topics allocated in the last week
Week 4	• Student presentation
Week 5	• Student presentation
Week 6	• Student presentation • Finalise the class project (through a public presentation)
Week 7	• Student presentation and project update
Week 8	• Student presentation and project update
Week 9	• Student presentation and project update
Week 10	• Student presentation and project update
Week 11	• Student presentation and project update
Week 12	• Student presentation and project update

Week 13	• Student presentation and project update – Discussion on project report
Week 14	• Student presentation and project update – Review of the project report
Week 15	• Final presentations (public)

Approximation and Online algorithms

Course Code: CS312

Course Description: CS312 is going to cover many topics of approximation and online algorithms in computer science.

Prerequisite(s): Design and Analysis of Algorithms
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Computers and Intractibility: A Guide to the Theory of NP-completeness;
M. Garey and D. Johnson

Approximation Algorithms;
PV. Vazirani

The Design of Approximation Algorithms;
D. P. Williamson and D. B. Shmoys

Course Objective: Students will get to know
(1) approximation algorithms for a wide range of NP-hard optimization problems
(2) online algorithms for the real world problems
(3) various algorithmic and proof techniques
Course Outcome: The students will be able to
(1) design approximation and online algorithms
(2) implement them in the computer

Grade Distribution:

Assignments and Class Tests	20%
Midterm Exam	20%
Final Exam	60%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Week	Content
Week 1	• Introduction • Unweighted Vertex Cover Problem
Week 2	• Minumum-Degree Spanning Tree • The Euclidean Traveling Salesman Problem
Week 3	• The k-Center Problem • Multiway Cut and K-Cut Problems
Week 4	• Scheduling Jobs with Deadlines on a Single Machine • Scheduling Jobs on Identical Parallel Machines
Week 5	• Class Test 1 • The Set Cover Problem • An Application of Set Cover to the String Matching problem
Week 6	• An Application of Layering via Set Cover and Vertex Cover • An Application of Set Cover to Art Gallery problems
Week 7	• The Knapsack Problem • The Bin-Packing Problem
Week 8	• Problem Session • Review for Midterm examination
Week 9	• The Primal-Dual Method for Weighted Vertex Cover Problem • Introduction to Online Algorithms
Week 10	• The Paging Problem • Amortized Analysis of Online Algorithms
Week 11	• List Update Problem • Scheduling Jobs on Identical Parallel Machines
Week 12	• Online Algorithms for Machine Learning • Online Algorithms for Graph Colouring
Week 13	• Class Test 2 • K-Server Problem
Week 14	• Searching for a Target in an Unbounded Region • Searching for a Target in a Street
Week 15	• Problem Session • Review for Final Examination

Basic Statistics

Course Code: DA102

Course Description: DA102 is going to provide an introduction to some basic statistical methods for analysis of categorical and continuous data. Students will also learn to make practical use of the statistical computer package R.

Prerequisite(s): NA
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.
Credit Hours: 4

Text(s):

Statistics;
David Freedman, Pobert Pisani and Roger Purves

Fundamentals of Statistics (V-1 & 2),
A.M. Goon, M. K.Gupta, B. Dasgupta

The visual display of Quantitative Information;
Edward Tufte

Mathematical Statistics with Applications;
Kandethody M. Ramachandran and Chris P. Tsokos

Fundamental of Mathematical Statistics
S. C. Gupta and V. K. Kapoor

Course Objectives: Students will get to know

(1) fundamental statistical concepts and some of their basic applications in real world.
(2) organizing, managing, and presenting data,
(3) how to use a wide variety of specific statistical methods, and,
(4) computer programming in R.

Course Outcome: The students will be able to

(1) apply technologies in organizing different types of data,
(2) present results effectively by making appropriate displays, summaries, and tables of data,
(3) perform simple statistical analyses using R

(4) analyze the data and come up with correct interpretations and relevant conclusions.

Grade Distribution:

Assignments	20%
Quizzes	10%
Midterm Exam	20%
Final Exam	50%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Week	Content
Week 1	• Introduction: What is Statistics • Types of Data: – Quantity/Qualitative – Identity – Freq/Non-freq • Data Collection: Direct observation, Direct question, Questionnaire
Week 2	• Data Representation: Textual, Tabular • Series Data and Grouped Data • Graphical Representation: – Bar diagram – Pie-chart – Frequency bar chart – Histogram – Ogive
Week 3	• Descriptive Numerical Measures – Measures of Central Tendency: AM, GM, HM, Median, Quantile, Mode – Box Plot
Week 4	• Descriptive Numerical Measures – Measures of Dispersion: Range, IQR, Mean Deviation, Root mean square deviation (SD, Variance), Gini Coefficient, CV, CQD. – Measure of Skewness – Measure of Kurtosis
Week 5	• Problem Session • Quiz 1
Week 6	• Bivariate Data – Scatter plot • Correlation – Pearson's correlation, Spearman's correlation, Kendall's correlation – Correlation with ties
Week 7	• Method of Curve Fitting – Method of least square – RSS, R^2

Week	Content
Week 9	• Normal Distributions and its associates – χ^2 distribution – t distribution – F distribution
Week 10	• Statistical Inference • Estimation – Point Estimation – Interval Estimation
Week 11	• ForestGreenMethods of calculating point estimates – Method of moments – Method of maximum likelihood
Week 12	• ForestGreenProperties of Point Estimators: Unbiased – Bias – MSE – Minimum Variance Unbiased Estimator
Week 13	• ForestGreenProperties of Point Estimators: Sufficiency – Neyman–Fisher factorization theorem – Joint Sufficiency – RAO–BLACKWELL Theorem
Week 14	• ForestGreenProperties of Point Estimators: Sufficiency – Minimal sufficient statistic – Lehmann and Scheffe method • Properties of Point Estimators: Consistency – Test for Consistency
Week 15	• ForestGreenProperties of Point Estimators: Efficiency – Uniformly Minimum Variance Unbiased Estimator – Cramer–Rao Lower bound – Fisher Information

Computational Complexity

Course Code: CS301

Course Description: The course *Computational Complexity* aims to teach the theory of computational complexity classes to the postgraduate students. The subject focuses on classifying computational problems according to their inherent difficulty, and then relating these classes of problems to each other. A computational problem is a task solved and solvable by a Turing Machine as a universal model of computers. By a computation problem, we assume that the problem is solvable by a mechanical application of some definite and practical mathematical steps, such as a plausible algorithm. In this course we stress on classes P, NP, L, NL, PSPACE, EXP, NEXP, and the classes in the polynomial hierarchy. We also study circuit complexity classes.

Prerequisite(s): Algorithms and Data Structures, Theory of Computation, Automata Theory
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Computational Complexity: A Modern Approach;
Sanjeev Arora and Boaz Barak

Computational Complexity: A Conceptual Perspective;
Oded Goldreich

Computational Complexity;
Christos H. Papadimitriou

Introduction to the Theory of Computation;
Michael Sipser

The Nature of Computation;
Cristopher Moore and Stephan Mertens

Theory of Computational Complexity;
Ding-Zhu Du and Ker-I Ko

Course Objective: Students will get to know

(1) Computational complexity concepts
(2) Knowledge of complexity classes
(3) Limits of important computational models
(4) Theorems and proofs related to common complexity classes

Course Outcome: The students will be able to

(1) Analysis of complexity classes of problems
(2) Modeling algorithms to well known computational models
(3) Formulating, analyzing and solving complexity problems
(4) Present the solution to a defined computational complexity problem orally, and answer question about the solution
(5) Read and integrate academic material from various online sources
(6) Improved problem analysis skills
(7) Better understanding of time and complexity analysis
(8) Improvement in asking quality questions

Grade Distribution:

Assignments and Class Tests	20%
Midterm Exam	30%
Final Exam	50%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 lecture hours. Quizzes will be unannounced, so students should maintain close to 100% attendance.

Chapters in reading assignments are from reference textbook (AB is textbook by Arora and Barak) unless specified otherwise.

Week	Content
Week 1	• Introduction to computational complexity — notations and preliminary definitions, Big-oh notation • Reading assignment: Introductory chapters in all references
Week 2	• The computational model — Turing machine (TM), efficiency and running time of TMs, string representation of TMs. uncomputability, the class P, efficient universal machine • Reading assignment: Chapter 1, AB, relevant portion • Home assignment 1 • Quiz 1
Week 3	• NP and NP completeness — the class NP, reducibility and NP-completeness • Reading assignment: Chapter 2, AB, relevant portion
Week 4	• NP and NP completeness — The Cook-Levin theorem, the web of reduction, decision vs search, classes coNP, EXP and NEXP, summary remarks on classes P, NP, etc. • Reading assignment: Chapter 2, AB, relevant portion • Home assignment 2 • Quiz 2
Week 5	• Diagonalization — time hierarchy theorem, diagonalization theorems • Reading assignment: Chapter 3, AB, relevant portion
Week 6	• Diagonalization — Ladner's theorem: existence of NP-intermediate problems • Reading assignment: Chapter 3, AB, relevant portion
Week 7	• Diagonalization — Oracle machines and the limits of diagonalization • Reading assignment: Chapter 3, AB, relevant portion • Home assignment 3 • Quiz 3
Week 8	• Space complexity — definition of space bounded computation, PSPACE completeness • Reading Assignment: Chapter 4, AB, relevant portion • Review for Midterm Exam

Week	Content
Week 9	• Space complexity — classes L and NL, NL completeness • Reading assignment: Chapter 4, AB, relevant portion • Home assignment 4 • Quiz 4
Week 10	• Polynomial hierarchy and alternations — the class Σ_2^p, the polynomial hierarchy • Reading assignment: Chapter 5, AB, relevant portion
Week 11	• The polynomial hierarchy and alternations — Alternating Turing machines, time versus alternations: time-space trade-offs for SAT, defining the hierarchy via oracle machines • Reading assignment: Chapter 5, AB, relevant portion • Home assignment 5 • Quiz 5
Week 12	• Boolean Circuits — Boolean circuits and class $P_{/poly}$, uniformly generated circuits, Turing machines that take advice, classes $P_{/poly}$ vs NP • Reading assignment: Chapter 6, AB, relevant portion
Week 13	• Boolean circuits — Circuit lower bounds, nonuniform hierarchy theorem, finer gradation among circuit classes, circuits of exponential sizes • Reading assignment: Chapter 6, AB, relevant portion • Home assignment 6 • Quiz 6
Week 14	• Randomized computation — Probabilistic Turing machines (PTM), some examples of PTMs, one-sided and "zero-sided" error: RP, coRP and ZPP classes, the robustness of definitions, relationship between BPP and other classes, randomized reductions, randomized space-bounded computation • Reading assignment: Chapter 7, AB, relevant portion • Home assignment 7 • Quiz 7
Week 15	• Interactive proofs — Some variations of interactive proofs, public coins and AM, IP=PSPACE, the power of the prover, multiprover interactive proofs, program checking, interactive proof for the permanent • PCP theorem and hardness of approximation — An introduction • Reading assignment: Chapter 8,11 AB, relevant portion • Review for Final Exam

Computer Vision and Pattern Recognition

Course Code: CS342

Course Description: In an era where cameras and digital imagery permeate our daily lives, understanding computer vision has become essential. CS342 provides an introductory exploration into this fascinating field. From casual "selfies" to immersive augmented reality experiences, the applications of computer vision are ubiquitous.

In this course, we delve into critical questions that shape the landscape of computer vision today:How do computers efficiently search for specific images amidst vast datasets? What algorithms enable the automatic tagging of photos on social media platforms? Can machines accurately interpret natural gestures or sign languages for seamless human-computer interaction? How do self-driving cars navigate complex terrains using visual input?

Our journey spans a wide range of topics, from the basics of image processing to cutting-edge visual recognition techniques. We explore machine learning methods, including supervised algorithms and deep neural networks, to tackle intricate research problems.

Moreover, we engage in discussions about ethical considerations, privacy, and bias in the responsible development of AI-driven systems.

Prerequisite(s): Basic knowledge of probability and linear algebra; data structures, algorithms; programming experience. Previous experience with image processing will be useful but is not assumed.
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Computer Vision: Algorithms and Applications;
Richard Szeliski

Digital Image Processing;
Rafael Gonzalez and Richard Woods

Computer Vision: A Modern Approach;

David A. Forsyth and Jean Ponce

Computer Vision;
Linda G. Shapiro and George C. Stockman

Multiple View Geometry in Computer Vision;
Richard Hartley and Andrew Zisserman

Pattern classification;
Richard O. Duda, Peter E. Hart, and David G. Stork

Pattern Recognition and Machine Learning;
Christopher M. Bishop

Course Objective: Students will get to know

(1) a comprehensive understanding of the theoretical and practical aspects of computing with images, including image formation, measurement, and analysis.

(2) Students will implement common methods for robust image matching and alignment, demonstrating proficiency in image processing techniques.

(3) Students will grasp the geometric relationships between 2D images and the 3D world, enabling them to understand concepts such as image transformations, alignment, and epipolar geometry (4) Students will gain exposure to object and scene recognition and categorization from images, including techniques such as object detection, supervised classification algorithms, and probabilistic models for sequence data

(5) Students will comprehend the principles of state-of-the-art deep neural networks, starting with convolutional neural networks (CNNs), and their applications in computer vision

(6) Students will develop practical skills necessary to build computer vision applications, enabling them to apply learned concepts in real-world scenarios effectively

(7) Students will view Computer Vision as a research area, understanding its ongoing developments, challenges, and potential applications

Course Outcome: The students will be able to

(1) demonstrate proficiency in image processing techniques, including feature extraction, filtering and edge detection through practical implementation and application exercises.

(2) Students will acquire the ability to employ segmentation and clustering algorithms, Hough transform, and robust fitting techniques, such as RANSAC for grouping and fitting objects and contours within images.

(3) Students will develop a deep understanding of multiple view geometry, local invariant feature detection and description, image transformations, planar homography, and stereo vision techniques for analyzing images and objects from different perspectives. (4) Students will demonstrate competence in object and scene recognition, categorization, and classification using supervised learning algorithms, probabilistic models, support vector machines, and neural networks
(5) Students will gain practical experience in building computer vision applications, utilizing learned techniques to address real-world problems, such as object detection and image classification
(6) Through assignments, projects, and exams, students will develop critical thinking skills and problem-solving abilities necessary for tackling challenges in computer vision and pattern recognition domains
(7) Students will develop a research-oriented perspective towards computer vision, gaining exposure to advanced topics, current trends, and emerging techniques in the field through lectures, readings, and discussions
(8) Students will enhance their communication skills through project and paper presentations
(9) Students will explore ethical considerations and implications related to computer vision technologies, understanding issues such as privacy, bias, and societal impact, and developing a responsible approach towards the use and development of computer vision systems
(10) By the end of the course, students will recognize the importance of lifelong learning in the rapidly evolving field of computer vision, being prepared to adapt to new technologies, methods, and research findings throughout their careers

Grade Distribution:

Assignments and Class Tests	30%
Midterm Exam	20%
Final Exam	50%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Week	Topic
1	Introduction and Basics of Digital Image
2	Filtering, Edge Detection and Template Matching
3	Frequency Response and Hough Transform
4	Harris Corner, SIFT
5	Subsampling, Fitting Interpolation
6	Tracking, Binary Image
7	Camera Stereo, Multiview Geometry
8	Camera Parameters, Panorama, Disparity
9	Machine Learning in Computer Vision
10	Machine Learning to Deep Learning
11	CNN, RNN, LSTM
12	Transformers, GNN
13	Segmentation
14	Generative AI for Image Generation
15	Paper Presentations
16	Advanced topics (if time permits), Applications, Course Wrap-up

Design and Analysis of Algorithms

Course Code: CS241

Course Description: This course deals with topics in design and analysis of algorithms. In particular, the course will cover different techniques of algorithm design illustrating them with several examples and also highlight some of the lower bounding techniques in algorithm design such as NP-Completeness.

Prerequisite(s): High School Mathematics and basic coputer programming in C, C++ or Python.
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Mining of Massive Datasets;
Jure Leskovec, Anand Rajaraman, Jeffrey David Ullman

Introduction to Algorithms;
Thomas H. Cormen, Charles E. Leiserson, Ronald Rivest, Clifford Stein

Algorithm Design;
Eva Tardos, Jon Kleinberg

Algorithms;
Sanjoy Dasgupta, Christos Papadimitriou, Umesh Vazirani

Algorithm Design and Applications;
Michael T. Goodrich, Roberto Tamassia

Data Structures and Algorithm Analysis in C;
Mark Allen Weiss

Course Objective: Students will get to know

(1) To fully understand standard the data mining methods and techniques such as association rules, data clustering and classification

(2) Familiarize students with various algorithmic paradigms such as greedy algorithms, dynamic programming, divide and conquer, and backtracking, enabling them to recognize when each approach is applicable.

(3) Enable students to analyze the time and space complexity of algorithms using mathematical techniques and Big O notation
(4) To recognize trade-offs between different algorithmic approaches, such as time complexity vs. space complexity, and help them make informed decisions based on problem constraints

Course outcomes: After successful completion of this course, students will be able to:
(1) Students should be able to understand various algorithmic paradigms such as greedy algorithms, dynamic programming, divide and conquer, etc., and apply them to solve problems efficiently
(2) Ability to analyze the time and space complexity of algorithms. This includes understanding Big O notation and being able to analyze worst-case, average-case, and best-case scenarios
(3) Proficiency in designing algorithms to solve specific types of problems, such as sorting, searching, graph algorithms, etc.
(4) Ability to implement algorithms in programming languages, understanding data structures, and optimizing code for performance
(5) Recognizing trade-offs between different algorithmic approaches, such as time complexity vs. space complexity, and making informed decisions based on problem constraints
(6) Understanding how to classify problems based on their inherent characteristics and choosing appropriate algorithmic techniques to solve them

Grade Distribution:

Assignments and Class Tests	20%
Midterm Exam	30%
Final Exam	50%

Course Outline (tentative) and Syllabus: The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures.

Week	Content
Week 1	• Different order notations like $O, \Theta, \Omega, o, \omega$ and compare two different functions using order notation. Pseudo-Polynomial Time Complexity. • Amortized analysis - Aggregate, Accounting and Potential methods notations.
Week 2	• Data Strutures - Binary Search Trees, Augmenting Data Structures - Range Queries and Index-Based Searching • Balanced Binary Search Trees - AVL Trees, Red-Black Trees
Week 3	• Priority Queues and Heaps, Heapsort, Binomial Queues, Fibonacci Heap
Week 4	• Hashing and Hash Tables, Hash Functions - Division and Multiplication methods, Universal Hashing • Hash Functions, Handling collisions - Chainng, Open Addressing - Linear Probing, Quadratic Probing and Double Hashing
Week 5	• Introduction of the Divide and Conquer paradigm of algorithm design, Master Theorem • Devising algorithms using divide and conquer for merge sort, counting inversions, Interger Multiplication, Maxima-Set Problem, Large integer, Local Maximum, Counting Inversion, Maximum Subsequence Sum, Majority Element Finding multiplication
Week 6	• Median and order statistics, Selection in worst-case linear time • Lower Bound for Comparison based Sorting, Soritng in Linear time - Counting sort, Radix sort, Bucket sort
Week 7	• Greedy methods of algorithm design. • Devising greedy algorithm for various problems like Interval Scheduling, Scheduling All Intervals, Fractional Knapsack.
Week 8	• Introducing the concept of Dynamic Programming and use of memoization. • Devising algorithms using dynamic programming for the problems like Matrix Chain Multiplication, Weighted interval scheduling, 0-1 Knapsack.
Week 9	• Dynamic programming algorithm - Weighted interval scheduling, longest increasing subsequence, edit distance, Shortest paths in DAG, Largest independent sets in trees.
Week 10	• Breadth First Search (BFS), Depth First Search (DFS) in graphs. • Articulation points, bridges and biconnected components for undirected graphs

Week	Content
Week 11	• Directed Acyclic Graphs and Topological Ordering. • Strongly connected components for directed graphs.
Week 12	• Disjoint Sets Data Structures - Linked representation, Parent link representation-naive linking, link-by-size, link-by-rank, link-by-path cpmpressionwith naive linking etc. • Kruskal, Prim's, Reverse deletion, Boruvka's algorithms for minimum spanning trees.
Week 13	• Algorithms for single source shortest paths in directed graphs - Dijkstra's algorithm, Bellman-Ford algorithm.
Week 14	• Algorithms for all pair shortest paths - Floyd-Warshall algorithm, Johnson's algorithm for sparse graphs.
Week 15	• Introduction to the concept of P, NP, NP-Completeness, Polynomial Reduction • CNF-SAT and 3SAT, Hamiltonian-Cycle and TSP

Discrete Mathematics

Course Code: CS100

Course Description: CS100 deals with discrete mathematical structures which are encountered in Computer Science. The topic broadly includes methods of enumeration, graph theory and mathematical logic.

Prerequisite(s): None
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Introductory Combinatorics;
Richard. A. Brualdi

Applied Combinatorics;
Alan Tucker

Extremal Combinatorics;
Stasys Jukna

Combinatorial Mathematics;
D. B. West

Abstract Algebra;
David S. Dummit, Richard M. Foote

Contemporary Abstract Algebra;
Joseph Gallian

Course Objective: Students will get to know
(1)basic principles of counting
(2)basics of existential arguments in combinatorics
(3) finite sets, relations and their combinatorics
(4) group theoretic arguments

Course outcomes: After successful completion of this course, students will be able to:

(1)basic principles of counting,
(2)basics of existential arguments in combinatorics
(3) finite sets, relations and their combinatorics,
(4) group theoretic arguments,
(5)use counting arguments to calculate complexity of algorithms,
(6)explore paradigms of software verification,
(7)model real life problems in many cases using graph theory,
(8)explore the area of group algorithms and group based algorithms.

Grade Distribution:

Assignments and Class Tests	20%
Midterm Exam	30%
Final Exam	50%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Week	Content
Week 1	• Basic Counting Principles (Addition, Multiplication, Subtraction, Division), Linear Permutations, Circular Arrangements, Combinations • Reading assignment: Chapter 2, RB
Week 2	• Binomial Theorem: Proof of the Binomial Theorem, Unimodality of Binomial Coefficients, Multinomial Theorem, Newton's Binomial Theorem, Binomial Identities • Reading assignment: Chapter 5, RB
Week 3	• Pigeonhole Principle–Basic Examples • Reading assignment: Chapter 3, RB
Week 4	• Pigeonhole Principle–graph theory examples • Reading assignment: Chapter 10.1, CM, Chapter 4, Jukna
Week 5	• Principle of Inclusion and Exclusion (PIE) and Application, Proof for the formula for union of sets, Combination with Repititions, Derangement • Reading assignment: Chapter 6, RB

Week	Content
Week 6	• Permutations with Forbidden Positions • Derangement • Reading assignment: Chapter 6, RB
Week 7	• Evaluating Sums using PIE • Concept of Signed Involution • Reading assignment: Chapter 4.1, CM
Week 8	• Proving identities using combinatorial arguments using bijections • Counting spanning trees • Reading assignment: Chapter 1.1, 1.2, 1.3 CM
Week 9	• Catalan numbers and bijections • Designing Recurrence–Lattice Path • Reading sssignment: Chapter 1.2, 1.3 CM
Week 10	• Special Counting Sequences: Catalan numbers, Stirling numbers–their recurrences • Fibonacci numbers • Reading assignment: Chapter 8, RB
Week 11	• Generating functions • Solving Recurrences • Reading assignment: Chapter 1, West
Week 12	• Groups–Definitions, examples • Permutation–cycle decomposition • Even and odd permutation • Symmetric groups • Reading assignment: Chapter 1.1, 1.3, DF
Week 13	• Dihedral Groups • Group generators • Reading assignment: Chapter 1.2, DF
Week 14	• Concept of Homomorphism and Isomorphism • Group Action • Reading assignment: Chapter 1.6,1.7, DF
Week 15	• Subgroups • Cyclic Subgroups • Subgroup Generators • Reading assignment: Chapter 2, DF
Week 16	• Centralizer, Normalizer, Stabilizer, Kernels • Reading assignment: Chapter 2, DF • Review for Final Exam

Advanced Algorithms

Course Code: CS313

Course Description: CS313 is going to cover many advanced topics in computer science on
data structures, dynamic programming, graph algorithms and randomized algorithms.

Prerequisite(s): Design and Analysis of Algorithms
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.
Course url:
Credit Hours: 4

Text(s):

1. *The Design and Analysis of Computer Algorithms*, A. Aho, J. Hopcroft, and J. Ullman, Addison-Wesley, Reading, Mass., 1974.
2. *Introduction to Algorithms*, T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, MIT Press, 2009.
3. *Algorithm Design*, Jon Kleinberg and Eva Tardos, Pearson Education, Inc., 2006.
4. *The Design and Analysis of Algorithms*, Dexter C. Kozen, Springer, 1991.
5. *Design and Analysis of Computer Algorithms*, David Mount, Lecture Notes, University of
   Maryland, Fall 2003.

Course Objectives:

Knowledge acquired: Students will learn
(1) algorithm design for a wide range of graph and optimization problems, and
(2) various algoritmic techniques, and
(3) how to prove the correctness of algorithms.

Skills gained: The students will be able to design algorithms using sofisticated data structures and algorithmic techniques, and can implement them in the computer.

Grade Distribution:

Class Tests	20%
Midterm Exam	20%
Final Exam	60%

Grading Policy: Grades are based on the actual marks obtained in the examinations and there will be no relative grading.
Approximate grade assignments:

>= 90	A+
89 – 80	A
79 – 70	B
69 – 50	C
49 – 35	D
<= 34	F

Course Policies:

- **Grades**

  Grades in the **C** range represent **average performance** and is **below the expectation**; Grades in the **B** range represent performance that is **substantially better** and **meets expectations**; Grades in the **A** range represent performance that is **very good**. Grades in the **A+** range represent performance that is **excellent**.

- **Attendance and Absences**

  1. Attendance is expected and will be taken each class. Students are not supposed to miss class without prior notice/permission. Any absences may result in point and/or grade deductions.
  2. Students are responsible for all missed work, regardless of the reason for absence. It is also the absentee's responsibility to get all missing notes or materials.

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures.

Week	Content
Week 1	• Introduction • Mathematical Background • Review of Sorting and Selection
Week 2	• Dynamic Programming: (a) Longest Common Subsequence • Dynamic Programming: (b) Chain Matrix Multiplication • Dynamic Programming: (c) Minimum Weight Triangulation
Week 3	• Dynamic Programming: (d) Segmented Least Square • Dynamic Programming: (e) A Solved Exercise • Dynamic Programming: (f) Sequence Alignment
Week 4	• Huffman Coding • Introduction to Graphs • Planar and Plane Graphs
Week 5	• Class Test 1 • Minimum Spanning Tree • Union-Find and Ackermann's Function
Week 6	• Bionomial Heaps • Fibonacci Heaps • Shortest Paths and Fibonacci Heaps
Week 7	• Splay Tree • Graph Traversal • Topological Sorting
Week 8	• Biconneced Component • Strongly Connected Component • Review for Midterm examination
Week 9	• All-Pairs Shortest Paths and Transitive Closure • Planar Separator Theorem
Week 10	• Network Flows: (a) The Max Flow-Min Cut Theorem • Network Flows: (b) Ford and Fulkerson's Algorithm • Network Flows: (c) Edmonds and Karp's Algorithms
Week 11	• Network Flows: (d) Dinic's Algorithm • Network Flows: (e) MPM Algorithm • Network Flows: (f) Applications of Max Flow
Week 12	• Unweighted Matching: (a) Berge's Theorem • Unweighted Matching: (b) Hopcroft and Karp's Algorithm
Week 13	• Class Test 2 • Randomized Algorithms: (a) Verifying Matrix Multiplication • Randomized Algorithms: (b) Polynomial Identity
Week 14	• Randomized Algorithms: (c) Randomized Quicksort • Randomized Algorithms: (d) Selection Problem • Randomized Algorithms: (e) Smallest Enclosing Disk
Week 15	• Randomized Algorithms: (f) Randomized Min-Cut • Randomized Algorithms: (g) Random Graphs • Review for Final Examination

Deep Learning (DL) and its Application in Natural Language Processing (NLP)

Course Code: **DA345**

Course description:

This course presents an introduction to the **Deep Learning (DL)** and its application in **Natural Language Processing (NLP)**. We will explore theoretical foundation of DL and their practical implementation to solve different NLP related tasks such as *text classification*, *sentiment analysis*, *machine translation*, *text summarization*, *text generation*, etc..

Prerequisite (s):

Student should have some knowledge in

- Mathematics: *Linear Algebra, Multivariate Calculus, Basis Optimisation and Basic probability*
- Computer programming: *Any one from C/C++/Python (recommended for the class project and assignments)/MATLAB/Octave*
- Basic concept in Algorithms and Data Structure
- Introduction to Machine Learning

Credit Hours: 4, approximately 60 credit hours

Text(s):

- Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.*, MIT Press, 1st edition, 2016. online
- Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning*, Cambridge University Press, 2023. online
- Simon J.D. Prince. *Understanding Deep Learning*, MIT Press, 2023. online
- Eugene Charniak. *Introduction to deep learning*, MIT Press, 2018.
- Michael Nielsen. *Neural Networks and Deep Learning*, online

- Dan Jurafsky and James H. Martin. *Speech and Language Processing*. draft, 3rd edition, 2024. [online]
- Delip Rao, Brian McMahan. *Natural Language Processing with PyTorch*, O'Reilly Media, Inc, 2019
- Lewis Tunstall, Leandro von Werra, and Thomas Wolf. *Natural Language Processing with Transformers Book*, O'Reilly Media, Inc, 2022. online code only
- Yoav Goldberg. *A Primer on Neural Network Models for Natural Language Processing*. online

Course Objective:

Students will get to know:

- the core concepts of DL, including network design, training and evaluation.
- common DL architechtures for tasks like object recognition, image generation, sentiment analysis, text summarization, and machine translation, etc..
- techniques for text preprocessing to clean and prepare textual data for NLP tasks.
- advanced deep learning models for different text processing tasks in NLP
- use of different DL and NLP toolkits (like PyTorch, NLTK, PyTorch) to solve practical NLP tasks.
- the performance of DL models and identify potential biases or errors.

Course outcomes:

After successful completion of this course, students will be able to:

- understand the mathematical basis of various deep-learning models.
- explain key deep learning concepts such as activation functions, loss functions, backpropagation, regularization, etc..
- design, implement, and evaluate feedforward, convolutional, recurrent neural networks and modern transformer based architechtures.
- explain textual data and extract representative features.
- solve practical problems in NLP using deep learning methods.

- gain hands-on experience with deep learning models in PyTorch [1].

Grade Distribution:

Quizzes and assignments	25%
Midterm Exam	10%
Class project	25%
Final Exam	40%

Course outline (tentative) and syllabus:

The tentative syllabus is as follows:

- Artificial neural network (ANN): Modelling Single neuron activity, different types of activity functions (sigmoid, tanh, ReLU, ELU etc.), how to connect multiple neurons to form a network, Multi-layer perceptron
- Optimization: Back propagation, different loss functions, gradient decent, stochastic gradient decent and different update rules (AdaGrad, RPMSprops, Adam etc.) for network parameters, regularization, dropout, batch normalisation etc.
- Deep learning toolbox: Explore a deep learning toolbox like PyTorch (my personal choice)/ TensorFlow and their autograd functionalities
- Convolutional neural network (CNN): Concept of kernel and convolution, some pooling operation (max, average etc.), some standard CNN architectures like LeNet, AlexNet, VggNet, ResNet etc. and concept of transfer learning
- Recurrent neural network (RNN): Sequential data and how to handle those using neural network, general RNN architecture, some popular RNN architectures like *Long short-term memory (LSTM)*, *Gated recurrent unit (GRU)* and their different variants
- Deep generative models: *Variational Autoencoders (VAE)*, *Generative Adversarial Networks (GAN)*, *Normalizing flows*, *Diffusion models*, etc.
- Neural language model:

[1] https://pytorch.org/

- Introduction to NLP
- Text preprocessing: tokenisation, stop words, stemming, lemmatisation, etc.
- Vector representations of text: *Bag of Words*, *TF-IDF*, *word embeddings*, *Word2Vec*, *GloVE*, etc.
- Sequence modelling: *Recurrent neural network (RNN)*, *Self-Attention network*, etc.
- Transformers: *Attention, BERT* and its different variants, Encoder-Decoder models
- Large language model (LLM): *GPT* different variants, *pre-trained language model*, *transfer learning*
- Application: *text classification*, *sentiment analysis*, *Named Entity Recognition (NER)*, *machine translation*, *text summarization*, *text generation*, etc.

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Reading assignments are from reference textbook will be mentioned in the suggested reading material in the *course url*.

Week	Content
Week 1	• Motivation: Artificial neural network (ANN): Modelling Single neuron activity, different types of activity functions (sigmoid, tanh, ReLU, ELU etc.), how to connect multiple neurons to form a network, Multi-layer perceptron
Week 2	• Optimization: Back propagation, different loss functions, gradient decent, stochastic gradient decent and different update rules (AdaGrad, RPMSprops, Adam etc.) for network parameters, regularization, dropout, batch normalisation etc. • Assignment-1
Week 3	• Deep learning toolbox: Explore a deep learning toolbox like PyTorch (my personal choice)/ TensorFlow and their autograd functionalities • Assignment-2
Week 4	• Convolutional neural network (CNN): Concept of kernel and convolution, some pooling operation (max, average etc.), some standard CNN architectures like LeNet, AlexNet, VggNet, ResNet etc. and concept of transfer learning • Assignment-3 • Class project discussion
Week 5	• Recurrent neural network (RNN): Sequential data and how to handle those using neural network, general RNN architecture, some popular RNN architectures like *Long short-term memory (LSTM)*, *Gated recurrent unit (GRU)* and their different variants • Assignment-4
Week 6	• Class project proposal • Deep generative models: *Variational Autoencoders (VAE)* • Assignment-5
Week 7	• Deep generative models: *Generative Adversarial Networks (GAN)*, *Normalizing flows*, *Diffusion models*, etc. • Course review for the mid-term examination
Week 8 - 15	• Neural language model: – Introduction to NLP – Text preprocessing: tokenisation, stop words, stemming, lemmatisation, etc. – Vector representations of text: *Bag of Words*, *TF-IDF*, *word embeddings*, *Word2Vec*, *GloVE*, etc. – Sequence modelling: *Recurrent neural network (RNN)*, *Self-Attention network*, etc. – Transformers: *Attention, BERT* and its different variants, Encoder-Decoder models – Large language model (LLM): *GPT* different variants, *pre-trained language model*, *transfer learning*

Graph Theory

Course Code: CS304

Course Description: The course CS304—*Graph Theory* aims to teach graph theory to students. We mainly study structural aspects of graphs in this course using combinatorial tools. We mainly focus on study of graph invariants which also have algorithmic implications.

Prerequisite(s): Discrete Math
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Combinatorial Mathematics;
D.B. West

Introduction to Graph Theory;
D.B. West

Graph Theory;
R. Diestal

Course Objective: Students will get to know

(1) Graph Theory concepts
(2) Basic theorems about common graph invariants like chromatic number, matching number etc.
(3) Advanced results in structural graph theory

Course outcomes: After successful completion of this course, students will be able to:

(1) Modelling problems using graph theory
(2) Using graph theoretic reasoning to compute structural invariants
(3) Designing graph networks that optimize different graph parameters as per application requirement
(4) Read and integrate academic material from various online sources
(5) Better combinatorial reasoning
(6) Improvement in asking quality questions

Grade Distribution:

Assignments and Class Tests	20%
Midterm Exam	30%
Final Exam	50%

Course Outline (tentative) and Syllabus: The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 lecture hours. Quizzes will be unannounced, so students should maintain close to 100% attendance.
Chapters in reading assignments are from reference textbook unless specified otherwise. Topics marked * will be optionally covered.

Week	Content
Week 1	• Definitions of basic Graph parameters for Undirected graphs like subgraphs, induced subgraphs, clique, chromatic number, matching number, independent set [CM 5.1, 5.2] • Directed graphs– basic structures like tournaments, strongly connected components • Degree sequences[CM 5.2]
Week 2	• Walk, paths, trees [CM 5.3] • connectivity [CM 5.3]
Week 3	• Petersen Graph—properties [CM 5.1] • Generalization of Petersen graph*
Week 4	• Definition of vertex and edge connectivity [CM 7.1] • Whitney's Inequality for vertex and edge connectivity [CM 7.1] • Construction of highly connected graphs–Harary Graphs [CM 7.1]
Week 5	• Menger's theorem [CM 7.2] • Application of Menger's theorem [CM 7.2]
Week 6	• 2-(edge) connected graph– characterization, Ear decomposition [CM 7.2] • 3- connected graph–characterization [CM 7.2]

Week	Content
Week 7	• Highly Connected Orientation– Lovasz shortcut Lemma [CM 7.2] • Concept of Submodularity [CM 7.2] • Nash-William's Orientation Theorem [CM 7.2]
Week 8	• Bipartite Matching [CM 6.1, GTD Chap 2] • Augmenting path, alternating path [CM 6.1, 6.3] • Hall's Theorem [CM 6.1]
Week 9	• Properties Hamiltonian Graphs [CM 7.3] • Sufficiency Conditions: Ore's Theorem [CM 7.3, GTD 10.1, 10.2]
Week 10	• Dirac's sufficiency condition for Hamiltonian cycle [CM 7.3, GTD 10.1, 10.2] • Chvatal's sufficiency condition for Hamiltonian cycle [CM 7.3, GTD 10.1, 10.2]
Week 11	• Hamiltonian cycle in square of a graph [GTD 10.3]
Week 12	• Vertex coloring [CM 8.1] • k-critical graphs [CM 8.2] • Mycielski construction [CM 8.1]
Week 13	• Brook's Theorem • Hadwiger's Conjecture and Forced Subgraphs [CM 8.2]
Week 14	• Edge coloring and Vizing's Theorem [CM 8.3]
Week 15	• Chordal graphs [CM 8.3] • Perfect graphs–Lovasz Theorem [CM 8.3]

Applied Computer Graphics

Course Code: CS214

Course Description: The course CS214—*Applied Computer Graphics* aims to teach modern OpenGL programming fundamentals to students. We mainly stress on 3D programming including modeling, lighting, shadow, animationa techniques. This course also teaches how to make use of modern hardware to efficiently render 3D scenes.

Prerequisite(s): Introduction to Computer Graphics, Linear Algebra, C++/C Programming
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Computer Graphics with OpenGL;
Donald Hearn and M Pauline Baker

Computer Graphics: Principles and Practice;
John F Hughes, Andries Van Dam, Morgan Mcguire, David F Sklar, James D Foley, Steven K Feiner, and Kurt Akeley

Learn OpenGL: An offline transcript of learnopengl.com;
Joey de Vries

Learning Modern 3D Graphics Programming;
Jason L. McKesson

OpenGL Programming Guide — the red book;
John Kessenich Graham Sellers and Dave Shreiner

The OpenGL Graphics System: A specification (v4.5) — the blue book;
Mark Segal and Kurt Akeley

Course Objective: Students will get to know
(1) OpenGL graphics pipeline for different versions
(2) Concepts of 3D programming
(3) Mathematical foundations of 3D computer graphics

(4) Lighting and shadow models
(5) Ray tracing method
(6) Theory of animation

Course Outcome: The students will be able to
(1) 3D models with loading, saving and display in OpenGL
(2) 3D Lights, creation and manupulation
(3) 3D Shadows modeling
(4) Rendering of 3D environments
(5) Basic 3D animation
(6) Basic 3D effects
(7) Present the solution to a defined computer graphics problem orally, and answer question about the solution
(8) Read and integrate academic material from various online sources
(9) Improved software development skills
(10) Learning version control and static code analysis
(11) Improvement in asking quality questions
(12) Understanding basic GPU hardware

Grade Distribution:

Assignments and Class Tests	20%
Midterm Exam	30%
Final Exam	50%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 lecture hours. Quizzes will be unannounced, so students should maintain close to 100% attendance.

Week	Content
Week 1	• Introduction to OpenGL • Reading assignment: Introductions in all references
Week 2	• OpenGL pipeline 1.x, 2.x, 3.x and 4.x, specifications and differences • Reading assignment: Chapter 1, redbook • Home assignment 1 • Quiz 1
Week 3	• Basic color theory, physical, physiological and psychological explanations of color perception • Reading Assignment: Relevant Chapters of CG w/OpenGL • Review for Midterm Exam
Week 4	• Vertex and Fragment Shaders, fundamentals and basic programming Analysis of a simple beginner OpenGL 4.x program • Reading assignment: Chapter 2 & Chapter 3 of LO
Week 5	• OpenGL data representation Data structures for graphics • Reading assignment: Chapter 2, blue book
Week 6	• Mathematics for 3D computer graphics Transformations • Reading assignment: Relevant chapters in CG w/OpenGL • Home assignment 2 • Quiz 2
Week 7	• Model, view and projection matrices local space, world space, view space and clip space • Reading assignment: Chapter 8,9 LO
Week 8	• Textures, storing and using • Reading assignment: Chapter 7, LO • Home assignment 3 • Quiz 3

Week	Content
Week 9	• Basic lighting and materials point lights, spot lights, directional light • Reading assignment: Chapter 13,14 LO • Home assignment 4
Week 10	• Shadows mapping, techniques and approximations • Reading assignment: Chapter 35, LO • Home assignment 5 • Quiz 4
Week 11	• Cubemaps, usage and programming • Reading assignment: Chapter 27, LO • Home assignment 6
Week 12	• Basic animation, data structures and programming • Reading assignment: Chapter 4, AB • Home assignment 7 • Quiz 5
Week 13	• Geometry Shaders • Reading assignment: Chapter 30, LO • Home assignment 8
Week 14	• Tessalation Shaders • Reading assignment: Chapter 9, redbook
Week 15	• Compute Shaders • Reading assignment: Chapter 12, redbook • Review for Final Exam

Linear Algebra and Matrix Computation

Course Code: DA109

Course Description: This course introduces fundamental concepts of linear algebra and explores practical matrix computation techniques. We will emphasize both theoretical understanding and practical implementation using computational tools. Through lectures, discussions, and problem-solving activities, students will gain a solid understanding of vector spaces, matrices, linear transformations, eigenvalues, and eigenvectors, matrix factorizations, as well as numerical methods for solving linear systems and eigenvalue problems.

Prerequisite(s): Student should have some knowledge in

- Basic concept on algebra, vectors and co-ordinate geometry
- Computer programming: Any one from C/C++/ Python(recommended for the class assignments)/MATLAB/Octave

Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Linear algebra;
A. Ramachandra Rao and P. Bhimasankaram

Linear Algebra Done Right;
Sheldon Axler

Linear Algebra;
Kenneth Hoffman and Ray Kunze

Introduction to linear algebra;
Gilbert Strang

Course Objective: Students will get to know

(1) concepts of vector spaces, including concepts of linear independence, span, and basis
(2) basic operations on vectors and matrices
(3) linear transformations to represent real-world problems

(4) different methods to solve a systems of linear equations
(5) eigenvalues and eigenvectors and their applications
(6) matrix decomposition methods for efficient computations

Course outcomes: After successful completion of this course, students will be able to:
(1) demonstrate the fundamental concepts in linear algebra, including vector spaces, matrices, linear transformations, eigenvalues, and eigenvectors
(2) implement algorithms related to linear algebra and matrix computation using Python programming language
(3) apply matrix computation techniques to solve practical problems, including matrix operations, Gaussian elimination, LU decomposition, matrix inverses, determinants, eigenvalue calculations, and matrix factorizations
(4) analyze and solve linear systems of equations using appropriate methods
(5) apply linear algebra concepts and techniques to solve practical problem in machine learning, data analysis and engineering fields

Grade Distribution:

Assignments and Class Tests	30%
Midterm Exam	20%
Final Exam	50%

Course Outline (tentative) and Syllabus:

The tentative syllabus is as follows:

- Introduction to Vectors: Vectors and their geometry, operations on vectors (addition, multiplication by a scalar, dot product, length).
- Vector Space: Vector space, Subspace, Basis and dimension, Change of basis.
- Linear Transformations: Introduction to linear transformations, Rank-Nullity theorem, Matrix of a linear transformation, Linear operators and isomorphism, Linear functionals.
- Matrix Algebra: Matrix addition and multiplication, transpose, inversion, Special matrices, Row rank and column rank of a matrix, Determinant of a matrix and its geometric interpretation, Cramer's rule to solve system of linear equations, Various matrix decompositions.

- Eigenvalues and Eigenvectors: Introduction to eigenvalues and eigenvectors of matrices, Characteristic polynomial, Cayley-Hamilton theorem, Algebraic and geometric multiplicities of eigenvalues, Matrix diagonalization, Positive (semi-) definite matrices, Solving linear recurrences.

- Normed Linear Spaces: Normed spaces, Cauchy-Schwarz inequality and triangle inequality, Projection, Gram-Schmidt orthogonalization, Hermitian operators, The Spectral theorem.

- Matrix Computations: Floating point numbers and operations, Error Analysis, Solving systems of linear equations - Direct (Gaussian elimination, LU factorization) and Iterative methods (Jacobi method, Gauss-Seidel method), Solving least square problems - QR decomposition, Gram-Schmidt orthogonalization, Singular value-decomposition (SVD), Solving Eigenvalue problems - Tridiagonal QR iteration, Jacobi method.

- Some practical applications (if time permits).

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures.

Week	**Content**
Week 1	• Motivation:Why do we need to study LAMC? • A brief discussion of some practical application • Vector space, Subspace, Span, Linearly dependent & independent, Basis and Dimension
Week 2	• Linear maps/transformation, null space, range of linear maps and rank-nullity theorem, Injective, Surjective and Bijective Linear map, System of linear equations and relation with linear map, Matrix, Linear map represent by a matrix
Week 3	• Direct sum, Linear map represent by a matrix and introduction to the Matrix algebra, Invertibility and Isomorphic Vector Spaces
Week 4	• Products and Quotients of Vector Spaces, Linear functional, Dual Space and Dual Map
Week 5	• Matrix algebra, special matrices, row space and column space, Matrix rank and inverse • Assignment-1

Week	Content
Week 6	• Rank factorization and change of basis • Elementary operations and Echelon form • Assignment-2
Week 7	• Elementary operations and Reduced Echelon form • Normal form and Hermite canonical form
Week 8	• Linear equation • Assignment-3 • Course review for the mid-term examination
Week 9	• Sweep-out method for solving $Ax = b$
Week 10	• Determinants • Assignment-4
Week 11	• Inner product, norm and orthogonality, Cauchy–Schwarz inequality • Orthogonal complement, projection, orthogonal and unitary matrices
Week 12	• Eigenvalues and Eigenvectors • Matrix diagonalization
Week 13	• Singular value decomposition • Gaussian elimination, LU factorization and Cholesky factorization • Assignment-5
Week 14	• QR factorization, solving least square problem and Power method to find the largest eigenvalue and corresponding eigenvector
Week 15	• IEEE 754 floating point number and Error Analysis • Course review for the final examination

Mathematical Logic

Course Code: CS222

Course Description: Mathematical logic is the mathematical study of formal logic. This include model theory, proof theory, set theory, and recursion theory (also known as computability theory). Research in mathematical logic commonly addresses the mathematical properties of formal systems of logic. This include their expressive or deductive power. We also study the use of logic to analyse correct mathematical reasoning and also to establish foundations of mathematics.
The course CS222—*Mathematical Logic* will teach precise and formal definitions of the language of logic theories and the notion of proofs for both propositional and predicate logic. We will study completeness, incompleteness and consistency theorems in logic. The completeness theorem says that we have all the rules of proof we could ever have. The Gödel incompleteness theorem says that they are not enough to decide all statements even about arithmetic. The compactness theorem exploits the finiteness of proofs to show that theories have unintended (nonstandard) models. We may have additional topics of computability and set theory, if time permits.

Prerequisite(s): Discrete Mathematics
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Introduction to Mathematical Logic;
Elliott Mendelson

Beginning Mathematical Logic, A Study Guide;
Peter Smith

A Friendly Introduction to Mathematical Logic;
Christopher C Leary and Lars Kristiansen

Fundamentals of Mathematical Logic;
Peter G Hinman

Logic and Structure;
Dirk van Dalen

Mathematical Logic;
Ian Chiswell and Wilfrid Hodges

Mathematical Logic;
Joseph R Shoenfield

Mathematical Logic;
Stephen Cole Kleene

Proofs and Types;
Jean-Yves Girard, tr. and apps. by Paul Taylor and Yves Lafont

Course Objective: Students will get to know
(1) Mathematical logic concepts
(2) Propositional logic
(3) Predicate logic
(4) Completeness, consistency and compactness theorems

Course Outcome: The students will be able to
(1) Logical analysis of a proof
(2) Understanding and writing correct proofs
(3) Formulating, analyzing and solving logical problems
(4) Presenting the problems, solutions, definitions in a formal manner orally or in writing, and ask questions and answer them in logical way
(5) Reading and integrating academic material from various online sources
(6) Improved problem analysis skills
(7) Better understanding of logical analysis
(8) Improvement in asking quality questions
(9) Better understanding of tractable algorithms and their design and analysis

Grade Distribution:

Assignments and Class Tests	20%
Midterm Exam	30%
Final Exam	50%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 lecture

hours. Chapters in reading assignments are from reference textbook (EM is textbook by Elliott Mendelson) unless specified otherwise.

Week	**Content**
Week 1	• Introduction to mathematical logic— notations and preliminary definitions • Introduction to the propositional calculus: • Propositional connectives, truth tables, tautologies • Adequate sets of connectives Reading assignment: Introductory chapters in all references, Chapter 1.1–1.3, EM, relevant sections
Week 2	• An Axiom System for the Propositional Calculus Reading assignment: Chapter 1.4, EM, relevant sections Home assignment 1 Quiz 1
Week 3	• Independence: Many-Valued Logics • Other Axiomatizations Reading assignment: Chapter 1.5–1.6, EM, relevant sections
Week 4	• Introduction to first-order logic and model theory • Universal and existential quantifiers • First-order languages and their interpretations: satisfiability and truth: models • First-order theories: logical axioms, proper axioms, rules of inference • Properties of first-order theories • Additional metatheorems and derived rules, particularization rule A4, existential rule E4, rule C Reading assignment: Chapter 2.1–2.6, EM, relevant sections Home assignment 2 Quiz 2
Week 5	• Completeness theorems Reading assignment: Chapter 2.7, EM, relevant sections
Week 6	• First-order theories with equality • Definitions of new function letters and individual constants Reading assignment: Chapter 2.8–2.9, AB, relevant sections

Week	Content
Week 7	• Prenex normal forms • Isomorphism of interpretations: categoricity of theories • Generalized first-order theories: completeness, decidability and applications • Elementary equivalence: elementary extensions • Ultrapowers: reduced direct products and nonstandard analysis Reading assignment: Chapter 2.11–2.14, AB, relevant sections Home assignment 3 Quiz 3
Week 8	• Semantic trees • Basic principle of semantic trees • Quantification theory allowing empty domains Reading assignment: Chapter 4, AB, relevant sections Review for midterm exam
Week 9	• Introduction to formal number theory • An axiom system • Number-theoretic functions and relations Reading assignment: Chapter 3.1–3.2, EM, relevant sections Home assignment 4 Quiz 4
Week 10	• Primitive recursive and recursive functions Reading assignment: Chapter 3.3, EM, relevant sections
Week 11	• Arithmetization: Gödel numbers Reading assignment: Chapter 3.4, EM, relevant sections Home assignment 5 Quiz 5
Week 12	• The fixed-point theorem: Gödel's incompleteness theorem • Recursive undecidability: Church's theorem • Nonstandard models Reading assignment: Chapter 3.5–3.7, EM, relevant sections

Week	Content
Week 13	• Introduction to axiomatic set theory • An axiom system • Ordinal numbers • Equinumerosity: finite and denumerable sets • Finite sets • Hartogs' theorem: initial ordinals—ordinal arithmetic • The axiom of choice: the axiom of regularity Reading assignment: Chapter 4.1–4.5, EM, relevant sections Home assignment 6 Quiz 6
Week 14	• Other axiomatizations of set theory—Morse–Kelley (MK), Zermelo–Fraenkel (ZF), theory of types (ST), Quine's theories NF and ML, set theory with urelements Reading assignment: Chapter 4.6, EM, relevant sections Home assignment 7 Quiz 7
Week 15	• Introduction to computability • Algorithms: Turing machines • Partial recursive functions: unsolvable problems • The Kleene–Mostowski hierarchy: recursively enumerable sets • Other notions of computability: Herbrand-Gödel computability, Markov algorithms • Decision problems Reading assignment: Chapter 5 EM, relevant sections Review for final exam
Week 16	• Any other relevant subject matter within the scope of the course

Machine Learning

Course Code: DA222

Course Description: This This course introduces the fundamental concepts and techniques of machine learning. Students will gain a solid understanding of various machine learning algorithms, their applications, and the process of building and evaluating machine learning models. Through lectures, discussions, hands-on exercises, and projects, students will develop the skills to apply machine learning to real-world problems.

Prerequisite(s): Student should have some knowledge in

- Mathematics: (some topic might need to discuss in brief)
  - Linear Algebra: *Vector space, Basis, Dimension, Matrix algebra (Addition, Multiplication, Trace, Inverse etc.), Eigen value and Eigen vectors, Positive definite matrices, Singular value decomposition etc.*
  - Multivariate Calculus: *Derivative, Partial derivative, Taylor series expansion, Chain rules etc.*
  - Basic Optimisation: *Convex set, Convex hull, Convex function, Gradient of a function, Hessian, Constrained and Unconstrained optimisation problem, Optimality condition*
  - Probability: *Definition, Random variables, Distribution function and their different variants, Conditional probability, Independence, Expectation, Variance, Moments, Entropy, Low of large numbers, Central limit theorem*
- Computer programming: *Any one from C/C++/Python (recommended for the class project and assignments)/MATLAB/Octave*
- Basic concept in Algorithms and Data Structure

Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Machine Learning: a Probabilistic Perspective;
Kevin Patrick Murphy

Probabilistic Machine Learning: An Introduction;
Kevin Patrick Murphy

Probabilistic Machine Learning: An Introduction;
Kevin Patrick Murphy

Pattern Recognition and Machine Learning;
Christopher M. Bishop

Foundations of Machine Learning;
Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar

Understanding Machine Learning: From Theory to Algorithms;
Shai Shalev-Shwartz and Shai Ben-David

Statistical Learning with Sparsity: The Lasso and Generalizations;
Trevor Hastie, Robert Tibshirani and Martin Wainwright

airness and Machine Learning: Limitations and Opportunities;
Solon Barocas, Moritz Hardt and Arvind Narayanan

Deep Learning;
Ian Goodfellow, Yoshua Bengio, and Aaron Courville

Pattern Classification and Scene Analysis;
R. O. Duda, P. E. Hart and D. G. Stork

Pattern Recognition;
S. Theodoridis and K. Koutroumbas

Introduction to Statistical Pattern Recognition;
K. Fukunaga

A Probabilistic Theory of Pattern Recognition;
Luc Devroye, Laszlo Gyorfi, and Gabor Lugosi

Course Objective: Students will get to know

(1) the core concepts of machine learning, including supervised learning and unsupervised learning

(2) data preprocessing techniques to prepare data for machine learning models

(3) the appropriate machine learning algorithms for different types of problems (classification, regression, clustering, etc.)

(4) train and evaluate machine learning models through hands-on experience using Python and machine learning libraries
(5) eigenvalues and eigenvectors and their applications
(6) the performance of different machine learning models and identify potential biases or errors

Course outcomes: After successful completion of this course, students will be able to:
(1) explain the fundamental concepts of machine learning, including the different types of problems (supervised, unsupervised) and their applications
(2) apply machine learning algorithms such as linear regression, logistic regression, decision trees, and clustering algorithms to solve regression and classification problems
(3) identify the appropriate machine learning algorithms for various tasks, such as classification, regression, or clustering, based on the problem type and data characteristics
(4) implement machine learning algorithms using the Python programming language and popular libraries such as NumPy, pandas, and scikit-learn
(5) evaluate and interpret machine learning models using appropriate performance metrics such as accuracy, precision, recall, F1-score, and confusion matrices
(6) analyze and interpret machine learning results, identify patterns and insights in data, and make informed decisions based on model predictions and performance
(7) discuss the potential and limitations of machine learning models

Grade Distribution:

Assignments and Class Tests	35%
Midterm Exam	15%
Final Exam	50%

Course Outline (tentative) and Syllabus:

The tentative syllabus is as follows:

- Motivation: What is Machine Learning (ML) and why do we need to study ML?
- Data: Representation/Featurization, Normalization (after some discussion in classification/regression), Data partition (train, val, and test)

- Regression: Linear, Ridge, LASSO

- Classification: kNN, Bayes classifier, Linear discriminant analysis, Logistic regression, SVM, Decision tree, Random forest, Boosting, Ensemble methods

- Clustering: K-means, Hierarchical and agglomerative clustering/linkage clustering, Spectral graph clustering

- Dimensionality reduction and data visualization: PCA, Multidimensional scaling, Random projection, Isomap, t-SNE, UMAP, etc.

- Kernel methods: Definition, Reproducing Kernel Hilbert space, Kernel-SVM, Kernel-PCA, Kernel-Least square regression

- Low-rank matrix completion and compressive sensing

- ML and Society: Fairness, Explainability, and Environmental effects

- Learning theory: Approximation and estimation error, Empirical risk minimization, Convergence and consistency, Capacity measure of function classes, Shattering coefficient, VC dimension, Rademacher complexity, Occam's razor

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Reading assignments are from reference textbook will be mentioned in the suggested reading material in the *course url*.

Week	Content
Week 1	• Motivation: What is Machine Learning (ML) and why we need to study ML? • A brief discussion of some practical problems (research and industry) • Data: Representation/Featurization, Normalization (after some idea in classification/regression), Data partition (train, val and test)
Week 2	• Classification: kNN and Assignment-1 • Introduction Bayes classifier
Week 3	• Bayes classifier, Linear discriminant analysis • Assignment-2
Week 4	• Regression: Linear, Ridge, LASSO • Assignment-3 • Class project discussion
Week 5	• Linear classifier (perceptron) and Assignment-4 • Support Vector Machine (SVM)
Week 6	• Decision tree • Project proposal/Class test

Week	Content
Week 7	• Random forest, Boosting, Ensemble methods • Assignment-5
Week 8	• Clustering: K-means, Hierarchical and agglomerative clustering/linkage clustering, Spectral graph clustering • Assignment-6 • Project update • Course review for the mid-term examination
Week 9	• Dimensionality reduction and data visualization: PCA, Multidimensional scaling, Random projection, Issomap, t-SNE, UMAP etc. • Assignment-7
Week 10	• Kernel methods: Definition, Reproducing Kernel Hilbert space, kernel-SVM, kernel-PCA, kernel-Least square regression • Assignment-8 • Project update
Week 11	• Low rank matrix completion and compressive sensing
Week 12	• ML and Society: Fairness, Explainability and Environment effect • Project update
Week 13 - 15	• Learning theory: Approximation and estimation error, Empirical risk minimization, Convergence and consistency, Capacity measure of function classes, Shattering coefficient, VC dimension, Rademacher complexity, Occam's razor • Course review for the final examination

Machine Learning with Graphs

Course Code: CS410

Course Information

Credits: 2 (30 hrs)
Prerequisites: Calculus, Linear Algebra, Probability and Statistics, Machine Learning, Algorithms, Graph Theory, Python Programming.

Course Description

Complex data can be represented as a graph of relationships and interactions between objects. This course explores the computational, algorithmic, and modeling challenges specific to the analysis of large-scale graphs. By examining graph structures, students will learn machine learning and data mining techniques to improve predictions and gain insights into various networked systems.

Course Goals and Objectives

At the end of this course, students will be able to:

- Understand and apply **Graph Neural Networks (GNNs)**.
- Perform **representation learning** and generate **node embeddings**.
- Conduct **link analysis** and study the **graph structure of the web**.
- Explore **models of network evolution** and network cascades.
- Reason over **knowledge graphs**.
- Use **deep generative models** for graph data.
- Detect communities and clusters within networks.

Weekly Schedule (Tentative)

1. **Week 1:** Introduction; Structure of Graphs
2. **Week 2:** Traditional Machine Learning on Graphs
3. **Week 3:** Node embeddings
4. **Week 4:** Introduction to Graph Neural Networks (GNNs)

5. **Week 5:** A general perspective on GNNs
6. **Week 6:** GNN augmentation and training
7. **Week 7:** Theory of GNNs
8. **Week 8:** Heterogeneous graphs
9. **Week 9:** Knowledge graphs
10. **Week 10:** Reasoning over knowledge graphs
11. **Week 11:** Fast neural subgraph matching
12. **Week 12:** GNNs for recommenders
13. **Week 13:** Deep generative models for graphs
14. **Week 14:** Advanced topics in GNNs
15. **Week 15:** Scaling to large graphs

Reference Books

1. *Graph Machine Learning* by Claudio Stamile, Aldo Marzullo, and Enrico Deusebio
2. *Introduction to Graph Neural Networks* by Zhiyuan Liu, Jie Zhou
3. *Graph Representation Learning* by William L. Hamilton
4. *Deep Learning on Graphs* by Yao Ma, Jilian Tang
5. *Graph Neural Networks: Foundations, Frontiers, and Applications* by Lingfei Wu, Peng Cui, Jian Pei, Liang Zhao

Natural Language Processing

Course Code: DA243

Course Description: TThis course presents an introduction to the field of computational models of written natural language, known as **Natural Language Processing** (NLP). We will explore both classical statistical and modern neural network-based language models. The course will cover theories, algorithms, and implementation (hands-on) of NLP models, aiming to highlight areas of ongoing research.

Prerequisite(s): Student should have some knowledge in

- Mathematics: *Linear Algebra, Multivariate Calculus, Basis Optimisation and Basic probability*
- Computer programming: *Any one from C/C++/Python (recommended for the class project and assignments)/MATLAB/Octave*
- Basic concept in Algorithms and Data Structure
- Introduction to Machine Learning

Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Speech and Language Processing;
Dan Jurafsky and James H. Martin

Introduction to Natural Language Processing;
Jacob Eisenstein

Foundations of Statistical Natural Language Processing;
Chris Manning and Hinrich Schütze

Deep Learning;
Ian Goodfellow, Yoshua Bengio, and Aaron Courville

Course Objective: Students will get to know
(1) the core concepts of NLP, including language representation, tokenization,

and stemming/lemmatization
(2) techniques for text preprocessing to clean and prepare textual data for NLP tasks
(3) common NLP algorithms for tasks like sentiment analysis, named entity recognition, and machine translation
(4) advanced topics in NLP, including deep learning models for sequence processing
(5) use of different NLP toolkits (like NLTK, PyTorch etc.) to solve practical NLP tasks
(6) the performance of NLP models and identify potential biases or errors

Course outcomes: After successful completion of this course, students will be able to:
(1) explain textual data and extract representative features
(2) understand the basic concepts and basic algorithms of NLP
(3) understand the mathematical basis of various deep-learning based language models
(4) implement NLP algorithms and techniques in PyTorch for some of the major tasks in NLP
(5) decompose a real-world problem into subproblems in NLP and identify potential solutions
(6) discuss the potential and limitation of NLP (large language) models

Grade Distribution:

Assignments and Class Tests	35%
Midterm Exam	15%
Final Exam	50%

Course Outline (tentative) and Syllabus:

The tentative syllabus is as follows:

- What is Natural Language Processing (NLP) and why do we need to study NLP?
- Language Models (classical): *n-grams, smoothing, class-based, brown clustering* etc.
- Distributional Semantics: *distributional hypothesis, vector space models*, etc.
- Sequence Labelling ((classical)): *HMM, CRFs*, and applications of these models in *Part of Speech (POS)* tagging, *Named-entity recognition (NER)* etc.

- Language theory: *Grammar, Regular expression, Finite state acceptors (FSA), Context-free grammar (CFG), Probabilistic Context-free grammar (PCFG), Constituency (CKY) Parsing, Dependency parsing* etc.
- Application: *text classification*, *Part of Speech (POS)* tagging, *Named-entity recognition (NER). Coreference Resolution*, some NLP toolkits, etc.
- Neural language model:
  - *Neural Networks (NN), Activation functions, Backpropogation, Softmax, Cross-entropy, Gradient descent, Stochastic Gradient descent, Layer Normalization, Dropout*, etc.
  - Word representation (Vector): *Feed-forward NN, Word2Vec, GloVE,* etc.
  - Sequence modelling: *Recurrent neural network (RNN), Long short-term memory (LSTM), Self-Attention network*, etc.
  - Transformers: *Attention, BERT* and its different variants, Encoder-Decoder models
  - Large language model (LLM): *GPT* different variants, *pre-trained language model, transfer learning* and application

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Reading assignments are from reference textbook will be mentioned in the suggested reading material in the *course url*.

Week	Content
Week 1	• Motivation: What is Natural Language Procession (NLP) and why we need to study NLP? • A brief discussion of some practical problems (research and industry) • Text document classification (Bag-of-Words model) and Assignment-1
Week 2	• Language model: N-gram, Smoothing • Assignment-2
Week 3	• Language model: class-based and brown clustering
Week 4	• Word embeddings: vector semantics, Word2Vec • Assignment-3 • Class project discussion
Week 5	• Sequence labelling: Part-of-speech tagging and named entities recognition • HMM, CRF
Week 6	• Formal language theory • Context-free parsing and Assignment-4
Week 7	• Class project proposal • Probabilistic Context-free grammar (PCFG), Constituency (CKY) Parsing
Week 8	• Dependency parsing • Assignment-5 • Course review for the mid-term examination
Week 9 - 15	• Neural language model: – *Neural Networks (NN), Activation functions, Backpropogation, Softmax, Cross-entropy, Gradient descent, Stochastic Gradient descent, Layer Normalization, Dropout*, etc. – Word representation (Vector): *Feed-forward NN, Word2Vec, GloVE,* etc. – Sequence modelling: *Recurrent neural network (RNN), Long short-term memory (LSTM), Self-Attention network*, etc. – Transformers: *Attention, BERT* and its different variants, Encoder-Decoder models – Large language model (LLM): *GPT* different variants, *pre-trained language model, transfer learning* and application • Assignment-6 & 7 • Course review for the final examination

Optimization for Machine Learning

Course Code: CS211

Course Description: This course focuses on select topics from convex optimization theory, algorithms of unconstrained and constrained optimization, batch gradient methods etc. that are needed to build a solid understanding of, as well as implement several machine learning algorithns. The purpose of this course is to summarize and analyze classical and modern optimization methods from a machine learning perspective.

Prerequisite(s): Linear Algebra, Multivariate calculus

Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Optimization for Machine Learning ;
Suvrit Sra, Sebastian Nowozin and Stephen Wright

Nonlinear Programming: Theory and Algorithms;
M. S. Bazaraa, Hanif D. Sherali, C. M. Shetty

Convex Optimization;
Stephen Boyd and Lieven Vandenberghe

Course Objective: Students will get to know

(1) Theory of convex functions, sets and convex optimization
(2) Geometric interpretation of feasibility and optimality conditions
(3) Conditions for existence and uniqueness of optimal solution
(4) Different unconstrained and constrained algorithms used in Machine Learning models

Course Outcome: The students will be able to

(1) How to formulate optimization problems
(2) How to write a computer program to solve a mathematical optimization problem
(3) How to perform sensitivity analysis on the optimal solution of a problem
(4) Ability to understand the interplay between optimization and machine

learning

Grade Distribution:

Assignments and Class Tests	20%
Midterm Exam	30%
Final Exam	50%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Table 1: **Topics**

Mathematical Preliminaries

- Norms of vectors, Sets and Functions in euclidean space
- Derivatives of multivariate functions, directional derivative, Gradient, Jacobian and Hessian

Theory of Convex Optimization

- Convex sets, convex functions and their characterizations
- Convexity preserving operations
- Composition of convex functions
- Generalized convexity
- Unconstrained Optimization: Necessary and Sufficient conditions
- Constrained Optimization: Necessary and Sufficient conditions for problems with equality and inequality constraints, KKT optimality conditions, Constraint Qualification, Lagrangian Duality and Saddle Point Optimality Criteria

Optimization in ML : – Analysis of ML model(s) from the point of Convex Optimization Theory

Optimization Algorithms used in Machine Learning

- First-order Algorithms: Steepest/Gradient Descent, Properties of GD
- (if time permits) Conjugate Gradient, Stochastic Gradient Descent

Emerging Technologies (Blockchain, LLM, Android, IoT)

Course Code: CS231 Credit:4

Course Description:

This course provides an in-depth exploration of emerging technologies that are shaping the future of digital transformation, including Blockchain, Large Language Models (LLMs), Mobile Application Development, and the Internet of Things (IoT). It covers foundational concepts, advanced applications, and hands-on implementation to help students gain practical skills in these cutting-edge fields. Blockchain topics include decentralized systems, smart contracts, and applications in industries such as finance and supply chain. LLMs focus on natural language processing, understanding, and generating human-like text using state-of-the-art models. Android development encompasses the creation of mobile applications, covering core aspects of the Android ecosystem, while IoT delves into sensor integration, data processing, and communication between devices. Throughout the course, students will work on real-world projects, equipping them with the expertise required to innovate and solve complex problems in these evolving domains.

Prerequisite(s): Basic Programming Skills in any language

Note: Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. Students, however, will be evaluated only on the basis of topics covered in the course.

Course Objectives:

- Introduce Large Language Models (Text)
- Introduce Blockchain, IoT
- Introduce Android Programming
- Introduce React JS

Course Outcomes:

- Gain hands on experience of using OpenAI, Llama and Mistral using Langchain
- Gain knowledge on theory behind unimodal large language model (e.g. RAG)
- Gain hands on experience of using specific chains and agents for structured and unstructured data processing with LLM (e.g. chains for SQL, Pandas, Graph DB)

- Gain basic understanding of improving RAG by Graphs (theory only)
- Gain Hands on experience of Finetuning a LLM (as a project)
- Gain knowledge of Deep Learning applications (CNN, Transformer, UNet, DeTR) as required by LLM topics
- Gain theoretical knowledge of Multimodal models (Diffusion)
- Gain hands on experience of Ethereum Blockchain
- Gain hands on experience of creating a blockchain from scratch
- Gain theory of Concensus algorithms in Blockchain
- Gain understanding of principles of cross platform development (multiple mobile , desktop environments)
- Gain hands on experience of React JS application development
- Gain hands on experience of Native Android application user interface and backend service development
- Gain hands on experience of developing a python server using flask and connecting that to mobile and web based application
- Gain Hands on experience of application development using QT5, Tkinter and Streamlit - comparison between these systems
- Gain hands on experience of Relational and Graph Database programming in python
- Gain hands on experience of Knowledge Graph creation using LLM (as a project)
- Gain theoretical understanding of IoT protocols (BLE in detail, ZigBee, WiFi, NFC in brief)
- Gain understanding of IoT application areas with one domain in detail
- Gain the hands on experience of self-learning a programming environment with multiple machines / nodes using standard LLMs

Approximate weights of different components in evaluation (this is modified based on interest and orientation of the students):

Assignments	20%
Midterm Exam	30%
Final Exam	50%

Course Policies:

General

1. Computing devices are not to be used during any exams unless instructed to do so.
2. Quizzes and exams are open books and open notes.
3. Practice sessions are planned for the second half of the course.
4. ***No makeup quizzes or exams will be given.***

Attendance and Absences

1. Attendance is expected and will be taken in each class. Students are not supposed to miss class without prior notice/permission.
2. Students are responsible for all missed work, regardless of the reason for absence. It is also the absentee's responsibility to get all missing notes or materials.

Textbooks(s):

1. Build a Large Language Model (From Scratch) 1^{st} Edition
   Sebastian Raschka
2. Understanding Large Language Models 1^{st} Edition
   Thimira Amaratunga
3. Transformers for Machine Learning: A Deep Dive, 1^{st} Edition
   Uday Kamath, Kenneth L. Graham, Wael Emara
4. The Blockchain Developer
   Elad Elrom
5. Blockchain Consensus: An Introduction to Classical, Blockchain, and Quantum Consensus Protocols *Imran Bashir*
6. React and React Native: A complete hands-on guide to modern web and mobile development with React.js 3^{rd} Edition
   Adam Boduch, Roy Derks

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. Each week assumes 4 hour lectures and 2 hour practicals (as and when required). Quizzes will be unannounced, so students should maintain close to 100% attendance.

Week	Content
Week 1	• Objective of the course, Introduction to Large Language Models • Hands on Chatbot Development using Llama and Mistral
Week 2	• Concepts of Deep Learning: RNN to Transformer • RAG and Langchain
Week 3	• Graph Database, Knowledge Graph and LLM • Structured Data Processing
Week 4	• Concepts of Finetuning a LLM • OpenAI models (GPT)
Week 5	• Deep Learning: CNN and Vision Transformer • UNet and Detection Transformer • Concept of Stable Diffusion in Multimodal LLM
Week 6	• User Interface Development for LLMs, Conventional UI: Tkinter, PyQT • Streamlit, Flask
Week 7	• Introduction to Android Programming, Activity LifeCycle • Android: Using Sound, Image, Video
Week 8	• Android Content Provider and Observer • LLM: Conversation History Development • Android Interface for LLM
Week 9	• Blockchain Overview, Basics of Cryptography with focus on ECC • Custom Blockchain development from scratch using python
Week 10	• Consensus Mechanisms: Proof of Work, Proof of Stake • Ethereum: Private Deployment using a single node
Week 11	• Consensus Mechanisms: Paxos • Ethereum: Transactions using HardHat and Metamask
Week 12	• Consensus Mechanisms: PBFT, PoH • Ethereum: Multi-node operations, Concept of Smart Contracts
Week 13	• Scalabality and Interoperbility • Integration of IoT and Blockchain: Use Cases
Week 14	• React JS Overview: ReactDOM , Sample application • Class and Functional Component, State variables
Week 15	• Connecting to Server using React JS • Animation using React JS

Probability and Stochastic Process

Course Code: DA104

Course Description: This course aims at providing a comprehensive review of probabilistic tools, techniques and basics of stochastic process. We shall focus more on applications of the techniques in problem solving.

Prerequisite(s): NA
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

A First Course in Probability ;
Shelden M. Ross

Introduction to Stochastic Process;
Paul G. Hoel, Sydney C. Port and Charles J. Stone

Mathematical Statistics and Data Analysis;
John A. Rice

Probability - Random Variables and Stochastic Processes;
Athanasios Papoulis, S. Pillai

Course Objective: Students will get to know

(1) different discrete and continuous probability distributions

(2) various standard statistics from mass, distribution and density functions from numerous univariate as well as bivariate distributions,

(3) different random processes to formulate and solve practical engineering problems..

Course Outcome: The students will be able to

(1) identify and interpret the key parameters which lies behind the random nature of the problems from various fields,

(2) convert a real-world problem into a precise mathematical probabilistic problem,

(3) use different statistical principles and the properties of random variables to solve various probabilistic problems.

Grade Distribution:

Assignments and Class Tests	20%
Midterm Exam	30%
Final Exam	50%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Week	Content
Week 1	• Basic Probability • Introduction to sample spaces • Classical definition of probability
Week 2	• Countable probability space • Counting methods using combinatorial tools • Conditional probability • Independence of events
Week 3	• Discrete Random Variables • Probability mass function • Expectation
Week 4	• Discrete Distribution • Binomial • Poisson • Geometric
Week 5	• Continuous random variables • Probability density function • Cumulative distribution function • Expectation
Week 6	• Continuous distribution • Uniform • Exponential • Normal
Week 7	• Joint distribution • Joint PMF for discrete random variables • Joint density and CDF
Week 8	• Independence of random variables • Conditional distribution and expectation • Covariance and correlation • Review for Midterm Exam

Week	Content
Week 9	• Central limit theorem • Characteristic function • Central limit theorem for finite variance
Week 10	• Weak law of large numbers for finite variance • Random walks
Week 11	• Markov chains • Definition and examples
Week 12	• Decomposition of states into communicating classes • Recurrence and transience as class properties
Week 13	• Invariant distributions for finite irreducible chains • Expected return time
Week 14	• Poisson processes • Definition and basic properties
Week 15	• Conditional distribution of arrival times given the number of arrivals • Non-homogeneous Poisson processes • Review for Final Exam

Reinforcement Learning

Course Code: DA344

Course Description: Reinforcement Learning (RL) is a branch of machine learning that deals with sequential decision-making. RL, however is different from other machine learning paradigms, since there is no supervisor for training. Instead, there is a trial-and-error learning process, that involves reward, feedback from environment, action, and time of action. The agent interacts with a dynamic, stochastic, and incompletely known environment, with the goal of finding an action-selection strategy, or policy, to optimize some measure of its long-term performance. RL has a wide range of applications in many problem situations, such as robotics, healthcare, smart grids, finance, self-driving cars etc., where explicit instructive signals are not available.The goal of the course is to introduce the mathematical foundations of reinforcement learning drawing from Operations Research, and do hands-on project on some small scale recent problems.

Prerequisite(s): Optimization Techniques, Matrix Algebra, Python

Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Reinforcement Learning: An Introduction;
Richard S. Sutton and Andrew G. Barto, 2nd Edition, MIT Press, Cambridge, MA, 2018

Reinforcement Learning: State-of-the-Art;
Marco Wiering and Martijn van Otterlo, Eds. Springer-Verlag Berlin Heidelberg 2012

Artificial Intelligence: A Modern Approach;
Stuart J. Russell and Peter Norvig. Fourth edition, 2020

Deep Reinforcement Learning: Fundamentals, Research and Applications;
Hao Dong, Zihan Ding, Shanghang Zhang Ed.s © Springer Nature Singapore Pte Ltd. 2020

Deep Reinforcement Learning: Frontiers of Artificial Intelligence;

Course Objective: Students will get to know

(1) Reinforcement Learning (RL) is a general purpose formalism for automated decision-making where an agent explicitly takes actions and interacts with the environment. Understanding the importance and challenges of learning agents that make decisions is of vital importance in designing interactive AI agents. This course focuses on various important aspects of RL, viz., formulating problems as Markov Decision Processes, understanding the basic exploration methods and the exploration/exploitation tradeoff, understanding value functions and how to implement dynamic programming as an efficient solution approach, identifying the impact of choices on performance and validating the expected behaviour of algorithms

(2) Understanding of how RL relates to and fits under the broader umbrella of machine learning, deep learning, supervised and unsupervised learning.

(3) Understanding of the space of RL algorithms (Temporal- Difference learning, Monte Carlo, Q-learning, Policy Gradient etc.).

(4) Understanding of how to implement reinforcement learning in RL platforms such as, **TensorFlow/Open AI Gym, PyTorch** etc.

Grade Distribution:

Assignments and Class Tests	30%
Midterm Exam	20%
Final Exam	30%
Final Project	20%

Course Outline (tentative) and Syllabus:

The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures. Quizzes will be unannounced.

Course Project :

There will be a course project, aimed at implementing reinforcement learning theory and algorithms to practical applications. It will also give students opportunity to learn how to deploy and train reinforcement learning on AI frameworks (e.g., PyTorch, OpenAI Gym, Tensorflow). The topic would be finalised based upon detailed discussion with the Instructor. An intermediate presentation of the project topic has to be done after midterm exam. Students can possibly extend and expand on the course project later on and give it a more concrete shape leading to term project and/or masters thesis.

Table 2: **Topics (for 16 weeks)**

A. Preliminaries (1 week)

- The Reinforcement Learning (RL) framework, RL vis-a-vis supervised/unsupervised ML
- Interesting applications of RL

B. Mathematical and Algorithmic Foundations of RL (4 weeks) – *The following topics will be covered in detail*

- *Sequential decision making*: Dynamic Programming Problem, Bellman Optimality Equations
- *Markov Decision Process* (MDP): State, stage, action, total expected reward criteria, average reward criteria. Optimal policies and optimal value functions. Value iteration and Policy Iteration. Optimal solution vs. approximate solution of Bellman optimality equation.
- *Partially Observable Markov Decision Process* (POMDP): The observation model, Monahan's enumeration algorithm, pruning via Linear Programming, the Witness algorithm, Parsimonious representation of value functions, near-optimal solutions for POMDPs

C. Methods of RL (8-10 weeks) – *The following topics will be covered in detail*

- *Model based methods*: Value iteration, Policy Iteration, Conditions on convergence Exploration/Exploitation Tradeoff (the k-armed bandit problem)
- *Model free methods*: Watkins' Q-learning,Sutton's Dyna architecture, Monte Carlo and Gaussian Process Temporal Difference Learning
- *Approximate solution methods*: Policy gradient, Monte Carlo policy gradient, Actor-critic algorithm, Performance evaluation of policy gradient methods

D. Frontiers in RL (1 week)- *If time permits, an overview will be given on the following topics:*

- Hierarchical Reinforcement Learning
- Game Theory and Multi-agent Reinforcement Learning
- Decentralised POMDP

Theory of Computations

Course Code: CS200

Course Description: This course introduces some fundamental concepts in automata theory and formal languages including grammar, finite automaton, regular expression, formal language, pushdown automaton and Turing machine. This subject not only forms the basic models of computation, it also includes the foundation of many branches of computer science, e.g. compilers, software engineering, concurrent systems, etc. The properties of these models will be studied and various rigorous techniques for analysing and comparing them will be discussed, by using both formalism and examples.

Prerequisite(s): Discrete Mathematics, Computer Programming.
Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Credit Hours: 4

Text(s):

Automata Theory, Languages and Computation (3rd Edition);
John E. Hopcroft, Rajeev D. Motwani and Jeffrey D. Ullman

Introduction to Theory of Computation;
Michael Sipser

Automata and Computability;
Dexter C. Kozen

Course Objective: (1) Introduce concepts in automata theory and theory of computation
(2) Identify different formal language classes and their relationships
(3) Design grammars and recognizers for different formal languages
(4) Prove or disprove theorems in automata theory using its properties
(5) Determine the decidability and intractability of computational problems

Course outcomes: (1) Acquire a fundamental understanding of the core concepts in automata theory and formal languages.
(2) An ability to design grammars and automata (recognizers) for different language classes.

(3) An ability to identify formal language classes and prove language membership properties.
(4) An ability to prove and disprove theorems establishing key properties of formal languages and automata.
(5) Acquire a fundamental understanding of core concepts relating to the theory of computation and computational models including decidability and intractability.

Grade Distribution:

Assignments and Class Tests	20%
Midterm Exam	30%
Final Exam	50%

Course Outline (tentative) and Syllabus: The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments. Each week assumes 4 hour lectures.

Week	Content
Week 1	• Introduction to automata, Alphabet, Strings, Language
Week 2	• Finite Automata — Deterministic Finite Automata, Nondeterministic Finite Automata, Finite Automata With Epsilon-Transitions
Week 3	• Regular Expressions and Languages — Regular Expressions, Finite Automata and Regular Expressions, Algebraic Laws for Regular Expressions
Week 4	• Decision Properties of Regular Languages — Proving Languages not to be Regular, Closure Properties of Regular Languages, Decision Properties of Regular Languages
Week 5	• Regular Grammar, Equivalence and Minimization of Automata — The Myhill-Nerode Theorem
Week 6	• Context-Free Grammars and Languages — Context-Free Grammars, Parse Trees, Ambiguity in Grammars and Languages
Week 7	• Pushdown Automata — Acceptance by Final State, Acceptance by Empty Stack, From Empty Stack to Final State, From Final State to Empty Stack
Week 8	• Pushdown Automata — Equivalence of PDA's and CFG's, Deterministic Pushdown Automata
Week 9	• Properties of Context-Free Languages — Normal Forms for Context-Free Grammars, The Pumping Lemma for Context-Free Languages
Week 10	• Properties of Context-Free Languages – Closure Properties of Context-Free Languages
Week 11	• Properties of Context-Free Languages — Decision Properties of CFL's
Week 12	• Introduction to Turing Machines — Programming Techniques for Turing Machines
Week 13	• Introduction to Turing Machines — Extensions to the Basic Turing Machine, Restricted Turing Machines
Week 14	• Undecidability – Recursively Enumerable Languages, Recursive Language
Week 15	• Undecidability – Undecidable Problems About Turing Machines, Undecidable Problems About CFLs