SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAMME STRUCTURE

M.TECH. COMPUTER SCIENCE AND ENGINEERING

SPECIALIZATION: ARTIFICIAL INTELLIGENCE

Academic Batch 2026-2028 and Onwards

GAUTAM BUDDHA UNIVERSITY

GAUTAM BUDH NAGAR, GREATER NOIDA, UP, INDIA

SEMESTER I

S.No.	Course Code	Course Name	L	T	P	Credits
1	CS521	Advanced Data Base Management System	3	0	0	3
2	CS523	Advanced Data Structure and Algorithm	3	1	0	4
3	CS525	Research Techniques in ICT	3	0	0	3
4	CS527	Python Programming	3	0	0	3
5	ES415	Energy and Environment	3	0	0	3
6	CD581	Advanced Data Base Management System Lab	0	0	4	2
7	CD583	Python Programming Lab	0	0	4	2
		Total Hours and Credits	**15**	**1**	**8**	**20**

SEMESTER II

S.No.	Course Code	Course Name	L	T	P	Credits
1	CA522	Computer Vision Applications	3	0	0	3
2	CS524	Deep Learning	3	0	0	3
3	CA526	Cognitive Computing	3	0	0	3
4	CS528	Blockchain Technology	3	1	0	4
5	CSXXX	Programme Elective-1	3	0	0	3
6	CA582	Computer Vision Applications Lab	0	0	4	2
7	CS584	Deep Learning Lab	0	0	4	2
8	GP	General Proficiency	Non Credit			
		Total Hours and Credits	**15**	**1**	**8**	**20**

ELECTIVES FROM DCSE

S.No.	Course Code	Course Name	L	T	P	Credits
1	CA530	Natural Language Processing	3	0	0	3
2	CA532	Speech Processing and Systems	3	0	0	3
3	CA534	Pattern Matching	3	0	0	3
4	CA536	Recommendation Systems	3	0	0	3
5	CS538	Parallel and High Performance Computing	3	0	0	3

SEMESTER III

S.No.	Course Code	Course Name	L	T	P	Credits
1	CS621	Cloud and Edge Computing	3	0	0	3
2	CA623	Generative AI	3	0	0	3
3	CAXXX	Programme Elective-II	3	0	0	3
4	CAXXX	Programme Elective-III	3	0	0	3
5	CA681	Generative AI Lab	0	0	4	2
6	CA691	Dissertation Part-I	0	0	12	6
		Total Hours and Credits	**12**	**0**	**16**	**20**

SEMESTER IV

S.No.	Course Code	Course Name	L	T	P	Credits
1	CA692	Dissertation Part-II	0	0	40	20
		Total Hours and Credits	**0**	**0**	**40**	**20**

ELECTIVE BASKET-II/III (SELECT ANY TWO)

S.No.	Course Code	Course Name	L	T	P	Credits
1	CA631	AI for Drone Systems	3	0	0	3
2	CA633	Autonomous Mobile Robots	3	0	0	3
3	CA635	Agentic AI	3	0	0	3
4	CA637	Intelligent Information Retrieval	3	0	0	3
5	CA639	Introduction to Large Language Models	3	0	0	3
6	CA641	AI Planning and Decision Making	3	0	0	3
7	CA643	Prompt Engineering	3	0	0	3
8	CA645	Evolutionary Computation	3	0	0	3
9	CA647	Trustworthy and Responsible AI	3	0	0	3
10	CA649	Predictive Analytics	3	0	0	3

SEMESTER I

Advanced Database Management System			
Course Code:	**CS521**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Knowledge of database design
2. A general understanding of database design and dependency
3. Understanding of different types of databases
4. Knowledge of databases on the internet
5. Application of enhanced databases

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Basic knowledge and understanding of ER diagram and UML class diagram.
2. Ability to apply functionality and Normalization in relational databases.
3. Recognise and fetch data from object-oriented, parallel and distributed databases.
4. Use XML and understand unstructured data.
5. Implement concepts and deductions of enhanced databases in different applications.

UNIT I: INTRODUCTION TO DATABASE DESIGN

Entities, Attributes, Entity Sets, Relationships, Key Constraints, Participation Constraints, Weak Entities, UML Class Diagrams, Subclasses, Superclasses, Inheritance, Specialization, Generalisation, Constraints and Characteristics of Specialization and Generalization Hierarchies, Modelling of UNION Types Using Categories, Representing Specialization and Generalization In UML Class Diagrams, Data Abstraction, Knowledge Representation and Ontology Concepts.

UNIT II: DATABASE DESIGN THEORY

Problems Caused by Redundancy, Decompositions, Problems Related to Decomposition, Reasoning About FD's, FIRST, SECOND, THIRD Normal Form, BCNF, Fourth Normal Form, Lossless Join Decomposition, Dependency Preserving Decomposition, Schema Refinement in Database Design, Multi-Valued Dependencies.

UNIT III: OBJECT- ORIENTED, PARALLEL AND DISTRIBUTED DATABASES

Overview of Object-Oriented Concepts, Object Identity, Object Structure, Type Constructor, Encapsulation of Operations, Methods and Persistence; Architectures for Parallel Databases, Parallel Query Evaluation, Parallelizing Individual Operations, Sorting Joins, Distributed Database Concepts, Data Fragmentation, Replication and Allocation Techniques for Distributed Database Design, Query Processing in Distributed Databases, Concurrency Control and Recovery in Distributed Databases.

UNIT IV: DATABASES ON THE WEB AND SEMI–STRUCTURED DATA

Web interface, XML, structure of XML data, querying XML data, storage of XML data, XML applications, semi- structured data model, indexes for text data.

UNIT V: ENHANCED DATA MODELS FOR ADVANCED APPLICATIONS

Active database concepts, temporal database concepts, spatial databases: concept and architecture, deductive databases and query processing, mobile databases, Geographic Information Systems (GIS).

Textbooks/ References Books:

1. Elmasri and Navathe, Fundamentals of Database Systems
2. Ramakrishnan and Gehrke, Database Management Systems
3. Korth, Silberschatz, Sudarshan, Database System Concepts
4. Rob and Coronel, Database Systems: Design, Implementation and Management
5. Date and Longman, Introduction to Database Systems

Advanced Data Structure and Algorithm			
Course Code:	**CS523**	**Course Credits:**	**4**
Course Category:	**PCC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 01**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 15**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Recognise the appropriate data structures, ADT libraries, and using it to design algorithms for a specific problem.
2. Be capable of solving problems using abstraction techniques.
3. Be able to choose appropriate algorithms for a specific problem.
4. Be able to analyse algorithms in terms of their efficiency and correctness.
5. To accept the recent developments in the area of algorithm design.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Design and analyse programming problem statements.
2. Choose appropriate data structures and algorithms for a specific problem.
3. Gain the necessary mathematical abstraction to solve problems.
4. Come up with an analysis of efficiency and proof of correctness.
5. Comprehend and select algorithm design approaches in a problem-specific manner.

UNIT I: INTRODUCTION

Review of Basic Concepts: Abstract data types, Data structures, Algorithms, Big-Oh, Small-Oh, Omega, Small-Omega and Theta Notations, finding time complexity of programs, Recurrence Relations: Solving Recurrence Relations, Substitution Method, Master Theorem.

UNIT II: HASHING

Hashing: Review of Hashing, Hash Function, Collision Resolution Techniques in Hashing, Separate Chaining, Open Addressing, Linear Probing, Quadratic Probing, Double Hashing, Rehashing, Extendible Hashing, Recent Trends in Hashing.

UNIT III: TREES & GRAPH

Trees: Binary Search Trees, AVL Trees, Red-Black Trees, 2-3 Trees, B-Trees, Splay Trees, Minimum Spanning Tree (MST), Kruskal's Algorithm and Prim's Algorithm, Applications to MST.

Graph: Graph, Breadth First Search, Depth First Search, Shortest path in edge-weighted case (Dijkstra's), Bellman-Ford Algorithm, Topological Sorting.

UNIT IV: SELECTED TOPICS

Strassen's Matrix Multiplication, Greedy method VS Dynamic Programming, Job sequencing with deadlines, Fractional Knapsack Problem, 0/1 Knapsack Problem, Travelling Salesman Problem, Huffman coding, Pre-order, Post-order, Inorder traversal, Postfix to infix notation, Infix to Postfix notation.

UNIT V: LINEAR PROGRAMMING & RECENT TRENDS

Linear Programming: Geometry of the feasibility region and Simplex algorithm.
NP-completeness: Examples, proof of NP-hardness and NP-completeness.
Recent Trends: Recent Trends in problem-solving paradigms using recent searching and sorting techniques by applying recently proposed data structures.

Text Books/ Reference Books:

1. Introduction to Algorithms, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein.
2. Algorithms Unlocked: Thomas H. Cormen.
3. The Algorithm Design Manual, Steven S. Skiena.
4. Algorithms: Robert Sedgewick and Kevin Wayne.
5. Advanced Data Structures: Peter Brass.

Research Techniques in ICT			
Course Code:	**CS525**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To understand the fundamentals, purpose, types, and processes of research in Information and Communication Technology.
2. To develop the ability to identify, formulate, and define research problems and objectives.
3. To understand and apply statistical, mathematical, and analytical techniques used in ICT research.
4. To understand algorithmic research, experimental design, modelling, optimization, and validation techniques.
5. To understand the role of simulation and computational techniques in conducting and evaluating ICT research.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Explain the fundamentals of research methodology, research processes, and research problem formulation.
2. Apply appropriate statistical and analytical techniques to analyse research data.
3. Develop and validate mathematical and computational models for ICT research problems.
4. Design and evaluate algorithmic and experimental approaches for solving research problems.
5. Apply simulation and computational techniques to conduct, analyse, and report ICT research studies.

UNIT I: INTRODUCTION TO RESEARCH TECHNIQUES

Meaning of research, objectives of research, motivation in research, types of research, characteristics and prerequisites of research, significance of research, research process, sources of research problem, criteria of identifying the problem, necessity of defining the problem, errors in selecting research problem, technique involved in defining the problem, report, and paper writing.

UNIT II: DATA ANALYSIS AND STATISTICAL TECHNIQUES

Data and their analyses, quantitative methods and techniques, Measure of central tendency, measures of variation, frequency distribution, analysis of variance, methods, Correlation analysis, regression analysis, time series and forecasting, introduction to discriminant analysis, factor analysis, cluster analysis, conjoint analysis, probability distribution, binomial distribution, poisson distribution, uniform distribution, exponential distribution, and normal distribution, sampling methods, test of hypothesis.

UNIT III: MATHEMATICAL MODELING

Steps of modelling, operations research models like queuing theory, stochastic processes, application of models, conceptual framework development and validation techniques, optimization techniques.

UNIT IV: ALGORITHMIC RESEARCH

Algorithmic research problems, types of algorithmic research, types of solution procedure, steps of algorithm development, steps of algorithmic research, design of experiments.

UNIT V: SIMULATION AND SOFT COMPUTING TECHNIQUES

Introduction to soft computing, artificial neural network, genetic algorithm, fuzzy logic and their applications, tools of soft computing, need for simulation, types of simulation, simulation language, fitting the problem to simulation study, simulation models, output analysis, data simulation packages like MATLAB, NS2, ANSYS, Cadence.

Text books/ Reference Books:

1. Research Methodology: Methods and Techniques, C.R. Kothari
2. Research Methodologies, R. Panneerselvam, Prentice Hall, 2007.
3. Research in Education, Best, John V. and James V Kahn, Wiley Eastern, 2005.
4. Elements of Educational Research, Sukhia, S.P., P.V. Mehrotra, and R.N. Mehrotra, PHI publication, 2003.
5. Methodology of Research Education, K. Setia, IEEE publication, 2004.
6. Research methodology, Methods and Techniques, Kothari, C.R., 2000.

Python Programming			
Course Code:	**CS527**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To understand the fundamental concepts, syntax, and programming constructs of the Python programming language.
2. To develop Python programs using conditional statements, loops, expressions, and basic input/output operations.
3. To understand and apply functions, strings, and built-in data structures such as lists, tuples, sets, and dictionaries in Python programming.
4. To develop Python programs using file handling, exception handling, modules, abstract data types, and object-oriented programming concepts.
5. To apply recursion, searching, sorting, and merging techniques to solve computational problems using Python.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Develop and execute Python programs using basic syntax, expressions, operators, and input/output operations.
2. Apply conditional statements and looping constructs to solve computational problems using Python.
3. Develop modular Python programs using functions, strings, and built-in data structures such as lists, tuples, sets, and dictionaries.
4. Implement file handling, exception handling, modules, and object-oriented programming concepts in Python.
5. Apply recursion, searching, sorting, and merging techniques to solve computational problems efficiently.

UNIT I: INTRODUCTION

The Programming Cycle for Python, Python IDE, Interacting with Python Programs, Elements of Python, Type Conversion. Basics: Expressions, Assignment Statement, Arithmetic Operators, Operator Precedence, Boolean Expression.

UNIT II: CONDITIONALS

Conditional statement in Python (if-else statement, its working and execution), Nested-if statement and Elif statement in Python, Expression Evaluation & Float Representation. Loops: Purpose and working of loops, while loop including its working, For Loop, Nested Loops, Break and continue.

UNIT III: FUNCTION

Parts of A Function, Execution of A Function, Keyword and Default Arguments, Scope Rules. Strings: Length of the string and perform Concatenation and repeat operations in it. Indexing and Slicing of Strings.

Python Data Structures: Tuples, Unpacking Sequences, Lists, Mutable Sequences, List Comprehension, Sets, Dictionaries. Higher Order Functions: Treat functions as first-class Objects, Lambda Expressions.

UNIT IV: FILE I/O

File input and output operations in Python Programming. Exceptions and Assertions Modules: Introduction, Importing Modules, Abstract Data Types: Abstract data types and ADT interface in Python Programming. Classes: Class definition and other operations in classes, Special Methods (such as _init_, _str_, comparison methods and Arithmetic methods, etc.), Class Example, Inheritance, Inheritance and OOP.

UNIT V: ITERATORS & RECURSION

Recursive Fibonacci, Tower Of Hanoi, Search: Simple Search and Estimating Search Time, Binary Search and Estimating Binary Search Time, Sorting & Merging: Selection Sort, Merge List, Merge Sort, Higher Order Sort.

Text Books/Reference Books:

1. Allen B. Downey, ``Think Python: How to Think Like a Computer Scientist‘‘, 2nd edition, Updated for Python 3, Shroff/O‘Reilly Publishers, 2016
2. Guido van Rossum and Fred L. Drake Jr, ―An Introduction to Python – Revised and updated for Python 3.2, Network Theory Ltd., 2011.

Energy and Environment			
Course Code:	**ES415**	**Course Credits:**	**3**
Course Category:	**OEC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**
COURSE OBJECTIVES			
1. To provide in-depth knowledge of renewable and non-renewable energy resources			
2. To provide knowledge about energy harnessing techniques			
3. To study the energy-environment issues			
COURSE OUTCOMES			
At the end of the course the students can use the knowledge so gathered and utilized to:			
1. Meet the challenges of energy vis-a-vis environmental security			

UNIT I: SUN AS SOURCE OF ENERGY

Nature of its radiation, solar radiation and its spectral characteristics; Conventional energy sources (coal, oil, biomass and natural gas), non-conventional energy sources (hydro-electric power, tidal, wind, geothermal, solar, nuclear magneto-hydrodynamic power MHD); Energy use pattern in India and parts of world, Energy security.

UNIT II: FOSSIL FUELS

Classification, composition, physico-chemical characteristics; Calorific value – gross and net; Energy content of coal, petroleum and natural gas, shale oil, coal bed methane, gas hydrates Concept of Green Energy; Principles of generation of hydropower, tidal energy, ocean thermal energy conversion, wind power, geothermal energy, solar energy (solar collectors, photo-voltaic modules, solar ponds).

UNIT III: NUCLEAR ENERGY

Fission and fusion, nuclear fuels, nuclear reactor – principles and types; Mechanism of radiation action on living organisms - Stochastic and Non-stochastic effects, delayed effects; Radioactivity from nuclear reactors, fuel processing and radioactive waste, hazards related to power plants.

UNIT IV: BIOENERGY

Types, importance, methods of energy production from biomass

UNIT V: ENVIRONMENTAL IMPLICATIONS OF ENERGY USE

CO2 emission and atmosphere –scenario in developed and developing world (and India), Global warming, Radiative forcing, Impacts of large-scale exploitation of solar, wind, hydro, nuclear and bioenergy sources, National Solar Mission, National Mission for Enhanced Energy Efficiency, case studies.

TextBooks/ Reference Books:

1. Fay, J.A. and Golomb, D.S. 2011. Energy and the Environment, Oxford University Press, New Delhi.
2. Iqbal, M. 1983. An Introduction to Solar Radiation. Academic Press, New York.
3. Kaushika, N.D. and Kaushik, K. 2004. Energy, Ecology and Environment: A Technological Approach, Capital Publications, New Delhi. Website - https://nptel.ac.in/course.html

Advanced Database Management System Lab			
Course Code:	**CD581**	**Course Credits:**	**2**
Course Category:	**PCC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Labs (Hrs/Week):	**1 (4 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To develop practical skills in database creation, manipulation, and querying using relational database management systems.
2. To implement database design concepts including data types, constraints, normalization, and relational operations.
3. To develop and execute SQL queries using joins, aggregation, functions, and subqueries for retrieving and analysing data.
4. To understand and implement database transaction management and query-processing concepts through practical exercises.
5. To apply database concepts to solve practical data-management problems using SQL and relational database systems.
COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Develop and execute SQL statements for creating, modifying, and managing relational databases.
2. Apply constraints, normalization, joins, and relational operations to design and manage database structures.
3. Construct SQL queries using aggregation, functions, subqueries, and other query-processing techniques.
4. Implement and analyse transaction-management concepts in relational database systems.
5. Apply database concepts to solve practical data-management and analysis problems.

List of Experiments:

1. Introduction to MySQL, Postgre Sql, Microsoft Sql softwares.
2. An exercise on data types in PostgreSQL & Data Definition Language Commands
3. Exercise on Data Manipulation Language and Transaction Control Commands using PostgreSQL.
4. Exercise on Types of Data Constraints using PostgreSQL.
5. Exercise on JOINS (Single-Table) Using Normalization
6. Exercise on JOINS (Multiple-Table) Using Normalization
7. Exercise on GROUP BY/ORDER BY Clause and Date Arithmetic using PostgreSQL.
8. Exercise on different Functions (Aggregate, Math and String)
9. Exercise on different types of subqueries
10. Procedures, Views and Triggers

Python Programming Lab			
Course Code:	**CD583**	**Course Credits:**	**2**
Course Category:	**PCC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Labs (Hrs/Week):	**1 (4 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Interpret the use of procedural statements like assignments, conditional statements, loops and function calls.
2. Infer the supported data structures like lists, dictionaries and tuples in Python
3. Illustrate the application of matrices and regular expressions in building the Python programs.
4. Discover the use of external modules in creating excel files and navigating the file systems.
5. Describe the need for Object-oriented programming concepts in Python.

COURSE OUTCOME
At the end of the course the students should be able to:
1. To write, test, and debug simple Python programs.
2. To implement Python programs with conditionals and loops.
3. Use functions for structuring Python programs.
4. Represent compound data using Python lists, tuples, dictionaries.
5. Read and write data from/to files in Python.

List of Experiments:

1. Write a python program find the maximum of a list of numbers.
2. Write a python program to perform Matrix Multiplication.
3. Write a python program first n prime number
4. Write a python program selection sort.
5. Write a python program to compute the GCD of two numbers.
6. Write a python program to find the most frequent words in a text file.
7. Write a Python program to create a scientific calculator
8. Write a Python program to print all the Disarium numbers between 1 and 100.
9. Write a Python program to encrypt the text using Caesar Cipher technique. Display the encrypted text. Prompt the user for input and the shift pattern.
10. Write a Python program to construct a linked list. Prompt the user for input. Remove any duplicate numbers from the linked list.

SEMESTER II

Computer Vision Applications			
Course Code:	**CA522**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Explain the fundamental concepts of Computer Vision, including image formation, image acquisition, camera models, color spaces, digital image types, and image conversion techniques.
2. Analyse object shape representation and motion estimation techniques using Principal Component Analysis (PCA), eigenvalues and eigenvectors, matrix transformations, and motion models.
3. Apply image feature extraction and camera projection techniques using edge and corner detectors, SIFT, SURF, HOG, scale-space analysis, stereo vision, and epipolar geometry.
4. Apply mathematical and machine learning techniques for computer vision applications, including linear algebra, PCA, supervised learning algorithms, and model evaluation using appropriate performance metrics.
5. Design and evaluate deep learning-based computer vision solutions using neural networks and CNN architectures, including transfer learning, data augmentation, regularization, and modern optimization techniques.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Explain the fundamental concepts, history, and real-world applications of Computer Vision.
2. Describe the working principles of image acquisition systems and different types of cameras used in various domains.
3. Differentiate between color spaces (RGB, CMYK, HSV) and various types of digital images (Binary, Grayscale, Color).
4. Interpret camera specifications such as pinhole model, CMOS, CCD, and image specifications including pixel resolution, aspect ratio, HD, and interlacing.
5. Apply image conversion techniques to transform images between different formats and color spaces.

UNIT I: INTRODUCTION TO COMPUTER VISION

General introduction, History of CV, Required component, Useful application, Image acquisition using a camera, DiAerent types of cameras for diAerent domain- Stills, Video, DSLR, Bodycam, Drone, Color spaces: RGB, CMYK, HSV, Camera specifications: Pinhole, CMOS, CCD, Image specifications: Pixel (Picture element), Aspect ratio, HD, Interlacing, Type of digital images: Binary, Grayscale, Color, Conversion techniques.

UNIT-II: SHAPE OF OBJECTS AND MOTION

Medial axis, Boundary coding, Chain Coding, Shape Numbering, Bounding box, Principal Component Analysis, Eigen Values and Vectors, Finding Eigen sets, Simple motion, Image differentiation, Single constant threshold, Weighted aggregate, Hierarchical Motion Estimation, 3D motion of a point, Matrix operations for different motion in objects, 2D matrix motion, Translation Motion, Sine Motion, Spatial Pattern of where motion occurred.

UNIT-III: FEATURE EXTRACTION AND CAMERA PROJECTION

Edges - Canny, LOG, DOG; Line detectors (Hough Transform), Corners - Harris and Hessian AAine, Orientation Histogram, SIFT, SURF, HOG, GLOH, Scale-Space Analysis- Image Pyramids and Gaussian derivative filters, Gabor Filters and DWT, Ambiguity in single, Geometry for simple stereo system View, depth and Calibration, Epipolar Geometry: Baseline, Epipole, Epipolar Line, Epipolar Plane.

UNIT-IV: MATHEMATICAL FOUNDATIONS OF MACHINE LEARNING

Linear Algebra for Vision, Vectors, matrices, eigenvalues, eigenvectors, SVD (Singular Value Decomposition), PCA for dimensionality reduction, Matrix factorization, Supervised Learning, Linear Regression, Logistic Regression, k-Nearest Neighbors (kNN), Naïve Bayes, Support Vector Machine (SVM), Decision Trees & Random Forest, Model Evaluation, Confusion matrix, Precision, Recall, F1-score, ROC Curve, Cross-validation.

UNIT-V: DEEP LEARNING AND CONVOLUTIONAL NEURAL NETWORKS

Neural Networks, Perceptron, Multi-Layer Perceptron (MLP), Activation functions (ReLU, Sigmoid, Tanh, Softmax), Loss functions (Cross-Entropy, Convolutional Neural Networks (CNN), Convolution operation, Padding & Stride, Pooling layers, Fully connected layers, CNN architectures: LeNet-5, AlexNet, VGG16, ResNet, EfficientNet, Training Techniques, Batch Normalization, Dropout, Transfer Learning, Data Augmentation, Learning rate scheduling, Optimizers (Adam, AdamW, RMSProp).

Text Books/Reference Books:

1. Pattern Recognition and Machine Learning, Christopher M. Bishop, Springer, 2006.
2. Deep Learning, Ian Goodfellow, Yoshua Bengio, Aaron Courville, MIT Press, 2016.
3. Computer Vision: Algorithms and Applications, Richard Szeliski, Springer, 2010.
4. Deep Learning for Computer Vision, Rajalingappaa Shanmugamani, Packt Publishing, 2018.
5. The Elements of Statistical Learning, Trevor Hastie, Robert Tibshirani, Jerome Friedman, Springer, 2nd Edition, 2009.
6. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, Aurélien Géron, O'Reilly Media, 2nd Edition, 2019.
7. Multiple View Geometry in Computer Vision, Richard Hartley and Andrew Zisserman, Cambridge University Press, 2nd Edition, 2003.

Deep Learning			
Course Code:	**CS524**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To gain a historical perspective of Deep Learning and become familiar with the building blocks of neural networks — the perceptron, backpropagation, autoencoders and batch-normalisation.
2. To understand the fundamental building blocks of neural networks, including artificial neurons, activation functions, loss functions, and network architectures.
3. To apply gradient descent, backpropagation and regularisation techniques to train deep networks and control overfitting.
4. To design and implement convolutional and recurrent architectures (CNNs, RNNs, LSTMs) for vision and sequence tasks.
5. To explore advanced deep architectures such as GANs and transfer learning, and evaluate deep learning models across application domains.

COURSE OUTCOME
At the end of the course the students should be able to:
1. Demonstrate knowledge of the building blocks of a Deep Learning system.
2. Build and train neural networks using TensorFlow/Keras with appropriate activation and loss functions.
3. Apply gradient descent, backpropagation and regularisation to optimise deep networks and mitigate overfitting.
4. Design and apply convolutional and recurrent models (CNNs, RNNs, LSTMs) to image and sequence tasks.
5. Evaluate and compare deep learning models — including GANs and transfer learning — across different application domains.

UNIT I INTRODUCTION TO DEEP LEARNING

Introduction to machine learning - Linear models (SVMs and perceptrons, logistic regression)- Introduction to Neural Nets: What a shallow network computes- Training a network: loss functions, back propagation and stochastic gradient descent- Neural networks as universal function approximators.

UNIT II: INTRODUCTION TO TENSORFLOW AND ACTIVATION FUNCTIONS

Computational Graph, Key highlights, creating a Graph, Regression example, Gradient Descent, TensorBoard, Modularity, Sharing Variables, Keras Perceptrons: What is a Perceptron, XOR Gate, Sigmoid, ReLU, Hyperbolic Fns, Softmax Artificial Neural Networks: Introduction, Perceptron Training Rule, Gradient Descent Rule.

UNIT III DIMENTIONALITY REDUCTION

Linear (PCA, LDA) and manifolds, metric learning - Autoencoders and dimensionality reduction in networks - Introduction to Convnet - Architectures – AlexNet, VGG, Inception, ResNet - Training a Convnet: weight initialisation, batch normalisation, hyperparameter optimisation.

UNIT IV OPTIMIZATION AND GENERALIZATION IN DEEP LEARNING

Optimization in deep learning– Non-convex optimization for deep networks- Stochastic Optimization Generalization in neural networks- Spatial Transformer Networks- Recurrent networks, LSTM Recurrent Neural Network Language Models- Word-Level RNNs & Deep Reinforcement Learning - Computational & Artificial Neuroscience.

UNIT V APPLICATIONS OF DEEP LEARNING

Image-net- Detection-Audio WaveNet-Natural Language Processing Word2Vec - Joint Detection Bio-Informatics- Face Recognition- Scene Understanding- Gathering Image Captions.

Textbooks/Reference Books:

1. Ian Goodfellow, Yoshua Bengio, Aaron Courville, Deep Learning, MIT Press, 2016.
2. Deng & Yu, Deep Learning: Methods and Applications, Now Publishers, 2013.
3. Michael Nielsen, Neural Networks and Deep Learning, Determination Press, 2015.

Cognitive Computing			
Course Code:	**CA526**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P):**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

1. To understand the fundamentals of cognitive computing and how it differs from traditional computing and conventional artificial intelligence systems.
2. To study human cognitive processes such as perception, learning, memory, reasoning, and decision-making, and their computational representation.
3. To understand machine learning, neural networks, natural language processing, and knowledge representation techniques used in cognitive computing systems.
4. To apply cognitive computing concepts to practical domains such as healthcare, cybersecurity, education, and decision-support systems.
5. To evaluate cognitive computing systems with respect to performance, explainability, reliability, ethics, and responsible AI.

COURSE OUTCOMES

At the end of the course, the students should be able to:

1. Explain the fundamental concepts, characteristics, and architecture of cognitive computing systems.
2. Analyse computational models of perception, learning, memory, reasoning, and knowledge representation.
3. Apply machine learning, neural networks, NLP, and knowledge-based techniques to develop cognitive computing solutions.
4. Design cognitive computing solutions for real-world applications involving intelligent interaction and decision-making.
5. Evaluate cognitive computing systems using appropriate performance, reliability, explainability, and usability measures.

UNIT I: FOUNDATIONS OF COGNITIVE COMPUTING

Introduction to Cognitive Computing; Evolution from Traditional Computing to Cognitive Computing; Cognitive Computing vs Conventional AI; Characteristics of Cognitive Systems; Human Cognition and Computational Intelligence; Perception, Learning, Memory, Reasoning and Decision-Making; Cognitive Computing Architecture; Cognitive Systems and Intelligent Agents; Knowledge Representation; Structured and Unstructured Data; Context Awareness; Human–Computer Interaction; Cognitive Computing Applications and Use Cases.

UNIT II: PERCEPTION, LEARNING AND KNOWLEDGE REPRESENTATION

Computational Models of Perception; Machine Learning for Cognitive Systems; Supervised, Unsupervised and Reinforcement Learning; Feature Representation and Learning; Neural Networks for Cognitive Computing; Deep Learning Fundamentals; Representation Learning; Knowledge Representation Techniques; Ontologies and Knowledge Graphs; Semantic Networks; Reasoning over Knowledge; Context and Memory Management; Uncertainty and Probabilistic Reasoning; Cognitive Learning Systems.

UNIT III: REASONING, DECISION-MAKING AND COGNITIVE SYSTEM DESIGN

Reasoning in Cognitive Systems; Deductive, Inductive and Abductive Reasoning; Probabilistic Reasoning; Decision-Making under Uncertainty; Cognitive Search and Information Retrieval; Recommendation and Decision-Support Systems; Planning and Problem Solving; Learning from Feedback; Context-Aware Decision-Making; Cognitive System Architecture and Workflow; Integration of Machine Learning, Knowledge Representation and NLP; Evaluation of Cognitive System Performance; Case Studies and Design Considerations.

UNIT IV: APPLICATIONS, EVALUATION AND RESPONSIBLE COGNITIVE COMPUTING

Cognitive Computing Applications in Healthcare; Cybersecurity; Education; Finance; Smart Cities; Customer Service; Industrial Systems; Autonomous and Intelligent Systems; Explainable Cognitive Computing; Reliability and Robustness; Performance Evaluation; Human Oversight; Privacy and Data Protection; Fairness and Bias; Security; Ethical and Societal Implications; Responsible AI; Governance and Regulatory Considerations; Emerging Trends in Cognitive Computing; Industry Case Studies.

UNIT V: APPLICATIONS IN AI AND ROBOTICS

Pattern matching in computer vision: edge/contour matching, Hough transform. Sensor fusion matching: LiDAR, IMU data alignment. Real-time implementations: optimizations, hardware acceleration. Case studies: robotic grasping, autonomous navigation.

Textbooks/Reference Books:

1. Judith Hurwitz, Marcia Kaufman, Adrian Bowles, Cognitive Computing and Big Data Analytics, Wiley.
2. Bernard Marr, Data Strategy: How to Profit from a World of Big Data, Analytics and the Internet of Things, Kogan Page.
3. Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, Pearson.
4. Reference Books
5. Ian Goodfellow, Yoshua Bengio and Aaron Courville, Deep Learning, MIT Press.
6. Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press.
7. Daniel Jurafsky and James H. Martin, Speech and Language Processing, Pearson/Prentice Hall.
8. Russell and Norvig, Artificial Intelligence: A Modern Approach, Pearson.

Blockchain Technology			
Course Code:	**CS528**	**Course Credits:**	**4**
Course Category:	**PCC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 01**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 15**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To get acquainted with the concept of Distributed ledger system and Blockchain.
2. To learn the concepts of consensus and mining in Blockchain through the Bitcoin network.
3. To understand Ethereum and develop-deploy smart contracts using different tools and frameworks.
4. To understand permissioned Blockchain and explore Hyperledger Fabric.
5. To understand different types of crypto assets.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Describe the basic and fundamental concept of Blockchain and Distributed Ledger Technology.
2. Interpret the knowledge of the Bitcoin network, nodes, keys, wallets and transactions
3. Implement smart contracts in Ethereum using different development frameworks.
4. Develop simple applications using Solidity language on Ethereum platform.
5. Interpret different Crypto assets and Crypto currencies.

UNIT I: FUNDAMENTALS OF BLOCKCHAIN TECHNOLOGY

Blockchain – Definition, architecture, elements of blockchain, benefits and limitations, types of blockchain. Consensus – definition, types, consensus in blockchain.
Decentralization – Decentralization using blockchain, Methods of decentralization, Routes to decentralization, Blockchain and full ecosystem decentralization.

UNIT II: FUNDAMENTALS OF CRYPTOGRAPHY

Introduction to Cryptography, Symmetric cryptography – AES. Asymmetric cryptography – RSA. Elliptic curve cryptography, Digital signatures – RSA digital signature algorithms. Secure Hash Algorithms – SHA-256. Applications of cryptographic hash functions – Merkle trees, Distributed hash tables.

UNIT III: CONSENSUS ALGORITHMS AND BITCOIN

Consensus Algorithms, Crash fault-tolerance (CFT) algorithms – Paxos, Raft. Byzantine fault-tolerance (BFT) algorithms – Practical Byzantine Fault Tolerance (PBFT), Proof of work (PoW), Proof of stake (PoS), Types of PoS. Bitcoin – Definition, Cryptographic keys – Private keys, public keys, addresses. Transactions – Lifecycle, coinbase transactions, transaction validation. Blockchain – The genesis block. Mining – Tasks of miners, mining algorithm, hash rate. Wallets – Types of wallets.

UNIT IV: SMART CONTRACTS AND USE CASES

Smart Contracts – Definition, Smart contract templates, Oracles, Types of oracles, Deploying smart contracts. Decentralization terminology – Decentralized applications, Decentralized Autonomous Organizations. Use cases of Blockchain technology – Government, Health care, Finance, Supply chain

management. Blockchain and allied technologies – Blockchain and Cloud Computing, Blockchain and Artificial Intelligence.

UNIT V: ETHEREUM AND SOLIDITY

Ethereum – The Ethereum network. Components of the Ethereum ecosystem – Keys and addresses, Accounts, Transactions and messages. The Ethereum Virtual Machine, Blocks and blockchain. The Solidity language – The layout of a Solidity source code, Structure of a smart contract, variables, data types, control structures, events, inheritance, libraries, functions, error handling. Smart contracts Case study: Voting, Auction.

Text Books/Reference Books:

1. Imran Bashir, Mastering Blockchain: A deep dive into distributed ledgers, consensus protocols, smart contracts, DApps, cryptocurrencies, Ethereum, and more, Packt Publishing, Third edition, 2020. Mastering Bitcoin: Unlocking Digital Cryptocurrencies, Andreas M. Antonopoulos, O'Reilly Media
2. Ritesh Modi, Solidity Programming Essentials: A beginner's guide to build smart contracts for Ethereum and blockchain, Packt Publishing, First edition, 2018.
3. Kumar Saurabh, Ashutosh Saxena, Blockchain Technology: Concepts and Applications, First Edition, Wiley Publications, First edition, 2020.
4. Chandramouli Subramanian, Asha A George, et al, Blockchain Technology, Universities Press (India) Pvt. Ltd, First edition, August 2020.
5. Blockchain Technology: Concepts and Applications, Kumar Saurabh and Ashutosh Saxena, Wiley.

Computer Vision Applications Lab			
Course Code:	**CA582**	**Course Credits:**	**2**
Course Category:	**PCC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Labs (Hrs/Week):	**1 (4 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

1. Understand and implement fundamental image processing techniques Apply OpenCV-based operations such as image transformation, filtering, edge detection, feature extraction, and histogram analysis for image preprocessing.
2. Develop classical machine learning models for image analysis Extract handcrafted features (HOG, SIFT, SURF), apply dimensionality reduction (PCA), and train SVM classifiers using linear and RBF kernels for image classification.
3. Design and train deep learning models for computer vision tasks Build and evaluate CNN architectures for image classification, analyze training performance, and compare with traditional ML approaches.
4. Apply advanced deep learning techniques for real-world applications Implement transfer learning (e.g., VGG16, ResNet), real-time object detection (YOLO), image captioning (CNN + LSTM), and generative models (GANs).
5. Evaluate model robustness and performance in vision systems Analyze classification metrics, study adversarial attacks (e.g., FGSM), and assess the impact of dimensionality reduction and model design choices on performance.

COURSE OUTCOMES

At the end of the course the students should be able to:

1. Apply image preprocessing and enhancement techniques using OpenCV, including image transformation, filtering, feature extraction, and edge detection methods.
2. Analyze and implement classical feature-based image processing algorithms such as HOG, SIFT/SURF, keypoint matching, and evaluate their performance in image classification tasks using SVM.
3. Develop and evaluate machine learning models (Linear SVM, RBF SVM, PCA-based dimensionality reduction) for image classification and analyze their impact on accuracy and computational efficiency.
4. Design, train, and evaluate deep learning models including CNNs for image classification, and implement transfer learning using pre-trained architectures such as VGG16 and ResNet.
5. Implement advanced computer vision applications including object detection (YOLO), image captioning (CNN-LSTM), generative models (GANs), and adversarial attack techniques (FGSM) to assess model robustness.

LIST OF EXPERIMENTS:

1. Write a program to perform image loading, resizing, rotation, grayscale conversion, and histogram plotting using OpenCV.
2. Implement and compare Sobel, Laplacian, and Canny edge detection techniques on a given image.

3. Extract HOG features from an image dataset and perform classification using SVM.
4. Detects and matches keypoints between two images using SIFT and/or SURF feature descriptors.
5. Train and evaluate an SVM classifier for image classification using linear and RBF kernels.
6. Apply PCA for dimensionality reduction on an image dataset and analyze its impact on classification performance.
7. Implement a basic CNN model for image classification using PyTorch or TensorFlow and plot training performance.
8. Perform transfer learning using a pre-trained CNN model (e.g., VGG16 or ResNet) for image classification.
9. Implement real-time object detection using a pre-trained YOLO model on images or webcam input.
10. Implement an image captioning system using CNN for feature extraction and LSTM for sequence generation.
11. Implement a basic GAN model to generate synthetic images from a training dataset.
12. Demonstrate an adversarial attack (e.g., FGSM) on a trained CNN model and analyze robustness.

Deep Learning Lab			
Course Code:	**CS584**	**Course Credits:**	**2**
Course Category:	**PCC**	**Course (U / P):**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Labs (Hrs/Week):	**1 (4 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Provide hands-on experience with deep learning frameworks and tools.
2. Implement and experiment with neural network models for real-world problems
3. Understand training, optimization, and evaluation of deep learning models.
4. Apply deep learning techniques in domains such as computer vision, NLP, and time-series analysis.
5. Encourage research-oriented thinking and reproducible experimentation.

COURSE OUTCOMES
At the end of the course the students should be able to understand:
1. Implement basic to advanced neural network architectures using Python-based frameworks.
2. Train, tune, and evaluate deep learning models on benchmark datasets.
3. Apply CNNs, RNNs, and Transformer-based models to domain-specific problems.
4. Analyze model performance using appropriate metrics and visualization tools
5. Develop mini-projects and research prototypes using deep learning techniques.

Software and Tools:

- Programming Language: Python
- Frameworks/Libraries: TensorFlow, Keras, PyTorch
- Supporting Libraries: NumPy, Pandas, Matplotlib, Seaborn, Scikit-learn, OpenCV, NLTK / SpaCy
- Platforms: Jupyter Notebook, Google Colab (optional)

List of Practical:

1. Python & DL Environment Setup: NumPy, Pandas, Matplotlib, TensorFlow/Keras installation, dataset preprocessing.
2. Single Layer Perceptron: Implementation from scratch and logical gate classification.
3. Multi-Layer Perceptron (MLP): Implementation from scratch and logical gate classification.
4. Backpropagation Algorithm: ANN model training and performance visualization.
5. Activation & Loss Functions: Comparison of Sigmoid, ReLU, Tanh, Softmax, MSE, Cross-Entropy.
6. Convolutional Neural Network (CNN): MNIST digit classification.
7. Image Classification using CNN: CIFAR-10 dataset with Dropout and Batch Normalization.
8. Sequence modeling or text classification using RNN.
9. Hyperparameter Tuning: Learning rate and optimizer comparison (SGD, Adam, RMSprop).
10. Recurrent Neural Network (RNN): Sequence prediction implementation.
11. LSTM Implementation: Time series forecasting and future value prediction.

PROGRAM ELECTIVES

Natural Language Processing			
Course Code:	**CA530**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P):**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**-**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To learn the fundamentals of Natural language Processing and its algorithm
2. To understand machine translation and applications of NLP
3. Basic understanding of deep learning models for NLP
4. To understand and implement deep learning architectures like RNN, LSTM, and GRU for solving NLP tasks
5. To explore speech processing, text-to-speech, multimodal language systems, and ethical aspects of speech and language technologies.

COURSE OUTCOMES
At the end of the course the students should be able to understand:
1. Learn the fundamentals of Natural language Processing and its algorithm
2. Understand machine translation and applications of NLP
3. Provide basic understanding of deep learning models for NLP
4. Apply the concept of NLP in the real domain.
5. Implement basic speech recognition and multimodal NLP techniques and address ethical concerns in voice-based AI systems

UNIT I: FUNDAMENTALS OF NATURAL LANGUAGE PROCESSING

Introduction to NLP: Characteristics of Natural Language, Language structure, Sentence Structure, Language analyzer, Lexicon, word formation, Morphology, syntax analysis (parsing), semantics, ambiguity, pragmatics and discourse.

UNIT II: TEXT PROCESSING AND REPRESENTATION TECHNIQUES

NLP Algorithms: Understanding Corpus and data attributes, Corpus Formats CSV, JSON, XML, LibSVM, Operations on Text Corpus, Tokenisation, stop words, Term Frequency-Inverse Document Frequency (TF-IDF), Text Analysis and word embedding using word2vec, doc2vec, GLoVe, Bag-of- words (BoW).

UNIT III: MACHINE TRANSLATION AND NLP APPLICATIONS

Machine Translation and Applications of NLP: Introduction to Machine Translation (MT), Approaches, Structure of Anusaraka: an Interlingua-based MT system, Example/Analogy based MT, Word/phrase based MT, Neural MT. Applications of NLP: Sentiment analysis, chatbots, conversational models (Question Answering system) for Digital Assistants

UNIT IV: DEEP LEARNING FOR NATURAL LANGUAGE PROCESSING

Deep learning models for NLP: Neural Net-based NLP models: Study of Convolutional Neural Network(CNN), Recurrent Neural Network(RNN), Long Short-Term Memory (LSTM) and Gated Recurrent Unit(GRU) using Natural Language Toolkit (NLTK)

UNIT V: MULTIMODAL AND EMERGING NLP APPLICATIONS

Introduction to speech and language interaction; Overview of Automatic Speech Recognition (ASR) and Text-to-Speech (TTS); Speech-based NLP applications; Multimodal NLP: integration of text, speech, and vision; NLP for low-resource languages; Conversational and voice-based AI

applications; Ethical considerations in language technologies; Privacy, bias, and responsible NLP systems.

Textbooks/Reference Books:

1. Daniel Jurafsky, James H. Martin,"Speech and Language Processing: An Introduction to Natural Language Processing", Computational Linguistics and Speech, Pearson Publication, 2014.
2. Thanaki, Jalaj, "Python natural language processing". Packet Publishing Ltd, 2017.

Speech Processing and Systems			
Course Code:	**CA532**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To introduce the speech signal as a stochastic, non-stationary time-series and relate it to human speech production and perception.
2. To develop mathematical and algorithmic understanding of short-time spectral/cepstral analysis used in speech processing front-ends.
3. To model speech and language statistically for recognition, including acoustic models, lexicons, and language models.
4. To design and evaluate modern speech systems (ASR, speaker recognition, TTS) using classical and deep-learning methods.
5. To analyse robustness, real-time constraints, and ethical considerations (privacy, bias) in deploying speech technologies.
COURSE OUTCOME
At the end of the course the students should be able to:
1. Explain the speech production mechanism and characterize speech signals in time, frequency, and time-frequency domains.
2. Implement and compare key feature extraction pipelines (e.g., LPC, MFCC, PLP) and interpret their parameters.
3. Build baseline speech recognition components using DTW/HMMs and evaluate them using standard metrics (e.g., WER).
4. Train and assess modern neural architectures for speech tasks (e.g., CNN/RNN/Transformer; CTC/attention) and articulate trade-offs.
5. Design an end-to-end speech application prototype considering noise robustness, latency, compute constraints, and responsible AI.

UNIT I: INTRODUCTION

Overview of speech processing systems; speech production and perception; speech signal characteristics (voiced/unvoiced, pitch, formants); sampling, quantization and pre-emphasis; framing and windowing; short-time stationarity; speech corpora and annotations; evaluation basics and datasets.

UNIT II: SPEECH FEATURE EXTRACTION AND ANALYSIS

Short-Time Fourier Transform (STFT) and spectrograms; filter banks; Linear Predictive Coding (LPC) and LP spectrum; cepstrum and liftering; Mel-Frequency Cepstral Coefficients (MFCC) and delta features; Perceptual Linear Prediction (PLP); feature normalization (CMN/CMVN); voice activity detection (VAD).

UNIT III: STATISTICAL MODELS FOR SPEECH AND LANGUAGE

Dynamic Time Warping (DTW); probabilistic modeling and Gaussian Mixture Models (GMM); Hidden Markov Models (HMM) for speech: forward-backward, Viterbi decoding, Baum-Welch training; pronunciation lexicon; n-gram language models and smoothing; decoding with acoustic + language model integration.

UNIT IV: DEEP LEARNING FOR SPEECH

Neural acoustic modeling: DNN, CNN, RNN/LSTM/GRU; sequence criteria (CTC) and attention-based encoder-decoder; Transformers for speech; self-supervised learning (e.g., wav2vec 2.0-style) and fine-tuning; speaker embeddings (i-vectors, x-vectors); keyword spotting and diarization overview.

UNIT V: SPEECH SYSTEMS AND APPLICATIONS

Automatic Speech Recognition (ASR) pipelines and end-to-end systems; Text-to-Speech (TTS) fundamentals (vocoder concepts, neural TTS overview); speech enhancement and noise-robust processing; speaker verification/identification; deployment aspects: streaming, latency, memory/compute constraints, edge vs cloud; evaluation metrics (WER, CER, EER, MOS); privacy, security, bias and responsible deployment.

Text Books/Reference Books:

1. Rabiner, L. R., & Schafer, R. W. (1978). Digital Processing of Speech Signals. Prentice-Hall, Englewood Cliffs, NJ.
2. Jurafsky, D., & Martin, J. H. (2026). Speech and Language Processing (3rd ed. draft, Jan 6, 2026 release). Stanford University. (Online) https://web.stanford.edu/~jurafsky/slp3/
3. O'Shaughnessy, D. (1999). Speech Communications: Human and Machine (2nd ed.). Wiley-IEEE Press.
4. Huang, X., Acero, A., & Hon, H.-W. (2001). Spoken Language Processing: A Guide to Theory, Algorithm, and System Development. Prentice Hall PTR.
5. Quatieri, T. F. (2002). Discrete-Time Speech Signal Processing: Principles and Practice. Pearson Education.
6. Gold, B., Morgan, N., & Ellis, D. (2011). Speech and Audio Signal Processing: Processing and Perception of Speech and Music (2nd ed.). Wiley-Interscience.
7. Rabiner, L. R., & Juang, B.-H. (1993). Fundamentals of Speech Recognition. Prentice Hall.
8. Benesty, J., Sondhi, M. M., & Huang, Y. (Eds.). (2008). Springer Handbook of Speech Processing. Springer.
9. Furui, S. (2001). Digital Speech Processing: Synthesis, and Recognition (2nd ed.). CRC Press.
10. Holmes, J., & Holmes, W. (2001). Speech Synthesis and Recognition (2nd ed.). CRC Press.
11. Jelinek, F. (1997). Statistical Methods for Speech Recognition. The MIT Press.
12. Gales, M., & Young, S. (2008). The Application of Hidden Markov Models in Speech Recognition. Foundations and Trends in Signal Processing, 1(3), 195–304.
13. Watanabe, S., et al. (2018). ESPnet: End-to-End Speech Processing Toolkit. In INTERSPEECH 2018.

Pattern Matching			
Course Code:	**CA534**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Introduce foundational concepts of pattern matching for AI and robotics applications
2. Develop skills in exact, approximate, and structural matching algorithms
3. Enable implementation of pattern matching for vision, sensor data, and robotic perception
4. Explore applications in robotics navigation, object detection, and control systems.
5. Foster understanding of efficiency analysis for real-time robotic systems.
COURSE OUTCOMES
At the end of the course the students should be able to understand:
1. Explain core principles of pattern matching and distinguish from pattern recognition.
2. Design and analyze string, sequence, and template matching algorithms.
3. Implement image and geometric pattern matching for robotic vision tasks.
4. Apply approximate matching techniques to noisy sensor data in robotics.
5. Evaluate pattern matching methods for efficiency in real-time AI systems.

UNIT I: FUNDAMENTALS

Introduction to pattern matching: definitions, types (exact, approximate, syntactic). String matching: Naive, Knuth-Morris-Pratt (KMP), Boyer-Moore algorithms. Time/space complexity analysis. Applications in robotics: command parsing, sensor signal matching

UNIT II: ADVANCED STRING AND SEQUENCE MATCHING

Rabin-Karp hashing for strings. Finite automata for multiple pattern matching (Aho-Corasick). Sequence alignment: edit distance (Levenshtein), longest common subsequence. Robotics use: trajectory matching, motion planning sequences.

UNIT III: TEMPLATE AND IMAGE MATCHING

Template matching: normalized cross-correlation, sum of squared differences. Feature-based matching: SIFT, SURF descriptors. Geometric transformations: affine, projective matching. Applications: robotic object detection, visual servoing.

UNIT IV: APPROXIMATE AND STRUCTURAL MATCHING

Approximate matching: fuzzy string matching, dynamic programming variants. Tree and graph matching: subtree isomorphism, edit distance for graphs. Syntactic pattern matching: grammars, parsing techniques. Robotics: map matching, SLAM feature alignment.

UNIT V: APPLICATIONS IN AI AND ROBOTICS

Pattern matching in computer vision: edge/contour matching, Hough transform. Sensor fusion matching: LiDAR, IMU data alignment. Real-time implementations: optimizations, hardware acceleration. Case studies: robotic grasping, autonomous navigation.

Textbooks/Reference Books:

1. Gusfield, D. Algorithms on Strings, Trees, and Sequences. Cambridge University Press, 1997.
2. Mount, D. Bioinformatics: Sequence and Genome Analysis. CSHL Press, 2004.
3. Aho, A.V., et al. Compilers: Principles, Techniques, and Tools. Addison-Wesley, 2006 (pattern matching chapters).
4. Sedgewick, R., Wayne, K. Algorithms. 4th ed., Addison-Wesley, 2011 (string matching sections).

5. Jain, A.K. Pattern Recognition and Machine Learning (select chapters on matching). Springer, 2006.
6. Tim Roughgarden, Beyond the Worst-Case Analysis of Algorithms, Cambridge University Press, 2021.

Recommendation Systems			
Course Code:	**CA536**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Understand the fundamental concepts of recommender systems including user preferences and prediction mechanisms.
2. Learn various recommendation approaches such as collaborative, content-based, knowledge-based, and hybrid systems.
3. Understand evaluation designs and performance metrics for recommender systems.
4. Gain insights into human-centered evaluation and practical deployment of recommender systems.
COURSE OUTCOMES
At the end of the course the students should be able to:
1. Describe the basic concepts, architecture, and framework of recommender systems.
2. Explain and compare various approaches used for building recommender systems.
3. Apply appropriate evaluation metrics and methodologies to assess recommender systems.
4. Analyze real-world applications of recommender systems and design simple recommendation models for different domains.

UNIT I: INTRODUCTION TO RECOMMENDER SYSTEMS

Definition and functions of recommender systems, need and importance of recommender systems, framework and architecture of recommendation engines, domains and applications, personalization concepts, user profiling, personalized and non-personalized recommendation, privacy issues, user data collection and trust, human-computer interaction in recommender systems, visualization, conversational recommender systems, cold start problem and data sparsity.

UNIT II: COLLABORATIVE FILTERING TECHNIQUES

Understanding ratings and rating data, rating matrix representation, user-based collaborative filtering, similarity measures, nearest neighbour methods, item-based collaborative filtering, similarity computation, model-based approaches, preprocessing techniques, comparison of user-based and item-based methods, scalability issues, data drift and concept drift.

UNIT III: CONTENT-BASED AND KNOWLEDGE-BASED RECOMMENDER SYSTEMS

Architecture of content-based recommender systems, advantages and limitations, content representation, item profiles, feature extraction techniques, learning user profiles, similarity-based retrieval, classification algorithms, knowledge-based recommender systems, knowledge representation, reasoning, constraint-based recommenders, case-based recommenders.

UNIT IV: HYBRID AND CONTEXT-AWARE RECOMMENDER SYSTEMS

Hybrid recommender systems concepts, motivation for hybridization, feature combination, feature augmentation, weighted, switching and mixed hybrid approaches, cascade and meta-level designs, limitations of hybrid systems, context-aware recommender systems, trust and context modelling, contextual pre-filtering, contextual post-filtering, contextual modelling.

UNIT V: EVALUATION AND APPLICATIONS OF RECOMMENDER SYSTEMS

Evaluation objectives and paradigms, offline and online evaluation methods, dataset preparation, training and testing strategies, accuracy metrics such as MAE and RMSE, ranking evaluation metrics, user-centric evaluation, case study of Netflix prize dataset, applications of recommender systems in e-commerce, social networks, media streaming, education and healthcare.

Text Books /Reference Books:

1. Aggarwal, C. C., “Recommender Systems: The Textbook,” Springer, 2016.
2. Jannach, D., Zanker, M., Felfernig, A., “Recommender Systems: An Introduction”, Cambridge University Press, 2011.
3. Manouselis N., Drachsler H., Verbert K., Duval E., “Recommender Systems for Learning”, Springer, 2013.
4. Kim Falk, “Practical Recommender Systems", Manning, 1st Edition, 2019
5. Rounak Banik, “Hands-On Recommendation Systems with Python: Start building powerful and personalized, recommendation engines with Python", 2018.
6. Ekstrand, M. D., Riedl, J. T., Konstan, J. A., Collaborative Filtering Recommender Systems, Now Publishers, 2011.
7. Ricci, F., Rokach, L., Shapira, B., Recommender Systems Handbook, Springer, 2011.
8. P. Pavan Kumar, S. Vairachilai, Sirisha Potluri, \Recommender Systems: Algorithms and Applications", CRC Press, 1st Edition, 2021.

Parallel and High Performance Computing			
Course Code:	**CS538**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U/ P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L +T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Introduce the fundamental concepts of parallel and high-performance computing, including concurrency, parallelism, scalability, and performance limitations.
2. To develop the ability to analyse performance bottlenecks in parallel programs using profiling tools, memory hierarchy concepts, and efficiency metrics.
3. To enable students to identify parallelism in applications and design efficient parallel algorithms using CPU-based optimization and vectorization techniques.
4. To provide hands-on experience in developing and optimizing shared-memory parallel programs using the OpenMP programming model with appropriate scheduling and synchronization strategies.
5. To familiarize students with distributed-memory and accelerator-based computing by implementing parallel programs using MPI and GPU programming models and evaluating their scalability and performance.
COURSE OUTCOMES
At the end of the course the students should be able to:
1. Explain the fundamental concepts of parallel and high-performance computing, including concurrency, scalability, and performance limitations.
2. Analyse performance bottlenecks in parallel programs using profiling techniques, memory hierarchy concepts, and efficiency metrics.
3. Design parallel solutions by identifying parallelism in applications and applying suitable parallel algorithms and CPU-based optimization techniques.
4. Develop and optimize shared-memory parallel programs using OpenMP, applying scheduling, synchronization, and vectorization strategies for improved performance.
5. Implement distributed and accelerator-based parallel programs using MPI and GPU programming models, and evaluate their scalability and performance.

UNIT I: INTRODUCTION TO PARALLEL AND HIGH PERFORMANCE COMPUTING

Need for parallel computing, real-world applications of parallel and high-performance computing, concurrency vs parallelism, scalability basics, Amdahl's Law and motivation for high-performance computing

UNIT II: PERFORMANCE ANALYSIS

Performance bottlenecks in parallel programs, profiling techniques for performance analysis, performance limits and efficiency metrics, data layout and memory hierarchy, cache effects on program performance

UNIT III: PARALLEL PROGRAM DESIGN AND CPU PARALLELISM

Planning parallel computing projects, identifying parallel opportunities in programs, parallel algorithms and common computation patterns, vectorization concepts, SIMD architecture and speedup

UNIT IV: SHARED MEMORY PROGRAMMING

Loop transformations for performance optimization, compiler support for vectorization, OpenMP programming model, OpenMP performance optimization techniques, scheduling, reduction, synchronization overhead

UNIT V: DISTRIBUTED MEMORY PARALLELISM AND GPU COMPUTING

Practical multicore performance tuning, need for MPI and distributed high-performance computing, MPI programming basics, point-to-point communication, collective communication operations, performance considerations in MPI, communication overhead, scalability, GPU architecture and computational advantages, GPU programming model, GPU execution concepts

Text Books/Reference Books:

1. Parallel and High Performance Computing, Robert Robey and Yuliana Zamora, 2021.
2. Introduction to High Performance Computing for Scientists and Engineers, Georg Hager, Gerhard Wellein, 2011.
3. Introduction to parallel computing (2nd edition). Pearson Addison Wesley, 2003.

SEMESTER III

Cloud and Edge Computing			
Course Code:	**CS621**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P):**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Explain the principles and terminology of the foundations of Cloud and Edge Computing.
2. Analyse problems and trade-offs concerning cloud architecture and virtualization.
3. Apply suitable methods drawn from distributed data and compute.
4. Design solutions that integrate edge–cloud continuum.
5. Evaluate approaches and responsible practice relating to security, reliability and operations.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Explain the concepts and terminology of the foundations of Cloud and Edge Computing
2. Analyse problems using the methods covered under cloud architecture and virtualization.
3. Apply appropriate techniques from distributed data and compute to realistic datasets or systems.
4. Design a solution that integrates concepts from the edge–cloud continuum.
5. Evaluate alternative approaches using technical, performance and reliability criteria.
6. Develop and communicate a responsible, reproducible solution drawing on security, reliability and operations.

UNIT I: FOUNDATIONS OF DISTRIBUTED COMPUTING

Evolution from clusters and grids to cloud and edge computing; Definition and essential characteristics of cloud computing — on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service; Overview of service models (IaaS, PaaS, SaaS) and deployment models (public, private, hybrid, multi-cloud); Drivers for edge computing — latency, bandwidth, privacy and data locality; The cloud-to-edge continuum at a glance; Representative cloud and edge use cases; Benefits, limitations and cost/performance trade-offs.

UNIT II: CLOUD ARCHITECTURE AND VIRTUALIZATION

IaaS, PaaS and SaaS; Virtual machines and containers; Orchestration; Service discovery; Autoscaling; Multi-tenancy; Cloud storage; Hypervisor types; Full and para-virtualization; Hardware support; Namespaces and control groups; Container images and registries; VM–container comparison; Orchestration control loops; Resource isolation and scheduling; Public, private, hybrid and multi-cloud models; Regions and availability zones; Reference architectures; Elasticity policies; Resource provisioning; Load balancing; Cloud service selection; Performance, availability and cost comparison; Worked examples and case analysis.

UNIT III: DISTRIBUTED DATA AND COMPUTE

Distributed file systems; Key-value and NoSQL stores; Partitioning and replication; MapReduce, stream processing and serverless workflows; Data locality; Shuffling and partition strategies; Distributed joins; Lineage and recomputation; Batch-stream unification; Data skew; Checkpointing; Back-pressure; Execution-plan analysis; Scalability experiments; Single-writer and multi-writer replication; Leader-follower and leaderless designs; Quorum reads and writes; Conflict-free replicated data types; Consistency anomalies; Isolation levels; Compensating actions; Failure recovery; Worked examples and case analysis; Comparative evaluation of techniques.

UNIT IV: EDGE–CLOUD CONTINUUM

Fog and edge architectures; Placement and offloading; Latency, bandwidth and energy trade-offs; IoT integration; Federated edge learning; Public, private, hybrid and multi-cloud models; Regions and availability zones; Reference architectures; Elasticity policies; Resource provisioning; Load balancing; Cloud service selection; Performance, availability and cost comparison; Edge node taxonomy; Cloudlets and gateways; Task-placement formulations; Computation offloading; Mobility support; Intermittent connectivity; Edge caching; Privacy-preserving analytics; Latency-energy modelling; Smart-city and industrial-IoT examples; Worked examples and case analysis.

UNIT V: SECURITY, RELIABILITY AND OPERATIONS

Zero-trust basics; Identity and access; Observability; SRE indicators; Resilience patterns; Cost governance; Sustainable computing; Case studies; Availability and reliability measures; Redundancy; Health checks; Graceful degradation; Bulkheads; Recovery-point and recovery-time objectives; Backup and disaster recovery; Fault injection; Chaos experiments; Reliability-cost trade-offs; Logs, metrics and traces; Service-level indicators, objectives and error budgets; Capacity planning; Deployment strategies; Runbooks; Incident response; Root-cause analysis; Post-incident reviews; Operational dashboards; Worked examples and case analysis.

Text Books/Reference Books:

1. M. van Steen and A. S. Tanenbaum, Distributed Systems, 4th ed., distributed-systems.net, 2023.
2. Erl, R. Puttini and Z. Mahmood, Cloud Computing: Concepts, Technology & Architecture, Pearson.
3. T. Hwang, G. Fox and J. Dongarra, Distributed and Cloud Computing, Morgan Kaufmann.
4. R. Buyya and A. V. Dastjerdi (eds.), Internet of Things: Principles and Paradigms, Morgan Kaufmann. C. Pahl et al., Cloud Container Technologies: a State-of-the-Art Review, IEEE Transactions on Cloud Computing.

Generative AI			
Course Code:	**CA623**	**Course Credits:**	**3**
Course Category:	**PCC**	**Course (U / P):**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Explain the principles and terminology of generative models for artificial intelligence
2. Analyse problems and trade-offs concerning image generation and synthesis.
3. Apply suitable methods drawn from diffusion models and vision transformers.
4. Design solutions that integrate multimodal generative vision systems.
5. Evaluate approaches and responsible practice relating to generative AI applications

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Explain the fundamental concepts and terminology of generative models for artificial intelligence
2. Analyse problems using the methods covered under image generation and synthesis.
3. Apply appropriate techniques from diffusion models and vision transformers to realistic vision applications.
4. Design a solution that integrates concepts from multimodal generative vision systems.
5. Evaluate alternative generative AI approaches using technical, performance and reliability criteria.
6. Develop and communicate a responsible, reproducible solution drawing on generative AI

UNIT I: FOUNDATIONS OF GENERATIVE AI

Generative AI fundamentals; Probabilistic generative models; Latent representations; Image distributions; GANs; VAEs; Diffusion models; Transformer-based vision models; Datasets; Image generation pipeline; Evaluation metrics; Applications; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises.

UNIT II: IMAGE GENERATION AND SYNTHESIS

Image synthesis; GAN architectures; Conditional GANs; StyleGAN; Image-to-image translation; Image super-resolution; Image inpainting; Image colorization; Style transfer; Controllable image editing; Quality assessment; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises; Hardware deployment issues.

UNIT III: DIFFUSION MODELS AND TRANSFORMERS

Diffusion probabilistic models; Denoising diffusion; Latent diffusion; Stable Diffusion; Text-to-image generation; Prompt conditioning; Vision Transformers; Attention mechanisms; Controllable generation; Model fine-tuning; Performance optimization; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises; Reproducibility practices.

UNIT IV: MULTIMODAL GENERATIVE VISION

Vision-language models; CLIP; Image captioning; Visual question answering; Text-guided image editing; Image retrieval; Multimodal embeddings; Cross-modal learning; Synthetic data generation; Evaluation and benchmarking; Worked examples and case analysis; Comparative evaluation of

techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises.

UNIT V: APPLICATIONS AND RESPONSIBLE GENERATIVE AI

Medical image generation; Autonomous driving; Industrial inspection; Robotics; Digital content creation; Synthetic datasets; Deployment considerations; Model compression; Explainability; Fairness; Bias; Copyright and intellectual property; Privacy; Responsible AI; Emerging trends; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Ethical and societal implications.

Textbooks / Reference books:

1. David Foster, Generative Deep Learning, 2nd ed., O'Reilly Media.
2. Ian Goodfellow, Yoshua Bengio and Aaron Courville, Deep Learning, MIT Press.
3. Richard Szeliski, Computer Vision: Algorithms and Applications, 2nd ed., Springer.
4. Rafael C. Gonzalez and Richard E. Woods, Digital Image Processing, 4th ed., Pearson.
5. Ashish Vaswani et al., Attention Is All You Need, NeurIPS Proceedings.
6. C. Blum and D. Merkle, Swarm Intelligence: Introduction and Applications, Springer.

Generative AI Lab			
Course Code:	**CA 681**	**Course Credits:**	**2**
Course Category:	**PCC**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Labs (Hrs/Week):	**1 (4 HRS)**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Understand the fundamental concepts and techniques of Generative AI.
2. Implement generative models such as VAE and GAN.
3. Apply diffusion models for image generation.
4. Explore transformer-based and multimodal generative AI applications.
5. Evaluate Generative AI systems considering performance, safety and responsible AI.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Explain the fundamental concepts of Generative AI.
2. Implement basic generative models for image synthesis.
3. Apply diffusion models for text-to-image generation.
4. Use transformer and multimodal models for Generative AI applications.
5. Evaluate and analyse Generative AI models and their limitations.

List of Experiments:

1. Implement a basic Generative AI image generation pipeline using Python.
2. Implement a Variational Autoencoder (VAE) for image generation.
3. Implement a Generative Adversarial Network (GAN) for image synthesis.
4. Implement a Conditional GAN for controlled image generation.
5. Perform image-to-image translation, inpainting and colorization using Generative AI.
6. Implement a basic Diffusion Model for image generation.
7. Perform text-to-image generation using Stable Diffusion with different prompts.
8. Implement a Vision Transformer (ViT) and demonstrate the attention mechanism.
9. Implement image captioning and Visual Question Answering using a multimodal model.
10. Develop a Generative AI application and evaluate it for quality, bias, privacy and responsible AI.

PROGRAM ELECTIVES

AI for Drone Systems			
Course Code:	**CA631**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P):**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Explain the fundamentals of Artificial Intelligence in autonomous drone systems.
2. Analyse AI techniques for drone perception, navigation, and decision-making.
3. Apply computer vision, machine learning, and deep learning for intelligent aerial applications.
4. Design autonomous drone solutions integrating perception, planning, and edge AI.
5. Evaluate AI-enabled drone systems considering performance, safety, reliability, and regulations.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Explain AI concepts and architectures used in autonomous drone systems.
2.Analyze perception, localization, and navigation techniques for drones.
3. Apply machine learning and deep learning algorithms to drone applications.
4. Design intelligent drone solutions for real-world autonomous missions.
5. Evaluate AI-based drone systems using technical, operational, and safety metrics.
6. Develop AI-enabled drone applications considering ethics, privacy, security, and regulatory requirements.

UNIT I: INTRODUCTION TO AI-ENABLED DRONE SYSTEMS

Drone platforms and architectures; UAV classifications; Flight dynamics fundamentals; Sensors and actuators; GPS, IMU and LiDAR; Drone communication systems; Embedded AI; AI-enabled autonomy; Mission planning; Drone operating modes; System architecture; Drone applications; Drone software stack; AI frameworks for UAVs; Performance measures; Safety considerations; Contemporary drone platforms; Industry case studies.

UNIT II: COMPUTER VISION FOR DRONES

Image acquisition; Camera calibration; Image preprocessing; Feature extraction; Object detection; Image segmentation; Object tracking; Optical flow; Stereo vision; Depth estimation; Pose estimation; Visual odometry; Visual SLAM; Semantic segmentation; Scene understanding; Obstacle detection; Landing zone detection; Aerial image analysis; Remote sensing; Deep learning for aerial vision; YOLO; Vision Transformers; Comparative evaluation of techniques; Performance measures.

UNIT III: AI FOR AUTONOMOUS NAVIGATION

Localization; Mapping; Path planning; Motion planning; Trajectory optimization; Obstacle avoidance; Simultaneous Localization and Mapping (SLAM); Reinforcement Learning; Decision making; Autonomous waypoint navigation; Mission optimization; Multi-objective optimization; Adaptive navigation; Sensor fusion; Kalman Filter; Particle Filter; Deep Reinforcement Learning; Dynamic path replanning; Swarm intelligence; Multi-drone coordination; Comparative evaluation of techniques; Industry applications.

UNIT IV: EDGE AI AND INTELLIGENT DRONE APPLICATIONS

Edge AI; TinyML; Onboard inference; Model optimization; Model compression; Quantization; TensorRT; NVIDIA Jetson; Qualcomm RB5; Embedded GPU computing; Real-time analytics; Aerial surveillance; Precision agriculture; Infrastructure inspection; Disaster management; Environmental

monitoring; Smart city applications; Healthcare delivery; Logistics and delivery drones; Energy-efficient AI; Comparative evaluation of techniques; Performance optimization; Contemporary tools and practices.

UNIT V: SAFETY, SECURITY AND FUTURE DIRECTIONS

AI safety; Fault detection; Predictive maintenance; Explainable AI; Responsible AI; Privacy-preserving AI; Secure drone communication; Cybersecurity threats; Adversarial attacks; Federated learning; Digital twins; Cloud robotics; Regulatory frameworks; DGCA regulations; Airspace management; UTM systems; Ethical AI; Sustainable drone systems; AI governance; Emerging research trends; Industry case studies; Scalability considerations.

Textbooks / Reference books:

1. Autonomous Flying Robots – Kenzo Nonami, Springer.
2. Programming Autonomous Drones with ROS 2 – Packt Publishing.
3. Artificial Intelligence for Robotics – Robin Murphy, MIT Press.
4. Computer Vision: Algorithms and Applications – Richard Szeliski.
5. Deep Learning – Ian Goodfellow, Yoshua Bengio, Aaron Courville.
6. Small Unmanned Aircraft: Theory and Practice – Randy Beard and Timothy McLain.

Autonomous Mobile Robots			
Course Code:	**CA633**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P):**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Explain the principles and terminology of mobile robot models.
2. Analyse problems and trade-offs concerning localization.
3. Apply suitable methods drawn from mapping and SLAM.
4. Design solutions that integrate planning and navigation.
5. Evaluate approaches and responsible practice relating to system integration.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Explain the fundamental concepts and terminology of mobile robot models.
2. Analyse problems using the methods covered under localization.
3. Apply appropriate techniques from mapping and SLAM to realistic datasets or systems.
4. Design a solution that integrates concepts from planning and navigation.
5. Evaluate alternative approaches using technical, performance and reliability criteria.
6. Develop and communicate a responsible, reproducible solution drawing on system integration.

UNIT I: MOBILE ROBOT MODELS

Locomotion; Wheeled configurations; Kinematic and dynamic models; Motion constraints; Differential-drive, tricycle, car-like and omnidirectional robots; Wheel constraints; Instantaneous centre of rotation; Forward and inverse kinematics; Non-holonomic constraints; Odometry; Dynamic effects; Controllability; Calibration; Trajectory feasibility; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises; Hardware deployment issues.

UNIT II: LOCALIZATION

Odometry; Probabilistic motion and sensor models; Kalman, histogram and particle-filter localization; Motion and observation models; Dead reckoning; Landmark association; Covariance propagation; EKF, UKF and particle filtering; Global localization; Kidnapped-robot problem; Accuracy and consistency measures; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises; Hardware deployment issues.

UNIT III: MAPPING AND SLAM

Occupancy grids; Feature maps; Graph-based SLAM; Loop closure; Map quality; Inverse sensor models; Scan matching; ICP; Visual and LiDAR odometry; EKF-SLAM; Particle-filter SLAM; Pose graphs; Loop-closure detection; Graph optimization; Map consistency and quality evaluation; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises; Hardware deployment issues.

UNIT IV: PLANNING AND NAVIGATION

Global and local planning; Costmaps; Reactive methods; Dynamic obstacles; Exploration; Problem representation; Search trees and graphs; Dijkstra and A*; Admissible and consistent heuristics; Potential fields; Dynamic window; RRT and RRT*; Trajectory smoothing; Replanning; Completeness and optimality; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises.

UNIT V: SYSTEM INTEGRATION

Navigation stacks; Multi-robot systems; Field robotics; Benchmarking; Safety and deployment case studies; Sensor, compute and actuator interfaces; Middleware timing; State-estimation pipeline; Navigation-stack configuration; Diagnostics; Logging and replay; Simulation-to-hardware transfer; Safety interlocks; Acceptance testing; Field benchmarking; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises.

Textbooks / Reference books:

1. R. Siegwart, I. Nourbakhsh and D. Scaramuzza, Introduction to Autonomous Mobile Robots, 2nd ed., MIT Press.
2. S. Thrun, W. Burgard and D. Fox, Probabilistic Robotics, MIT Press.
3. S. M. LaValle, Planning Algorithms, Cambridge.
4. P. Corke, Robotics, Vision and Control, 3rd ed., Springer.
5. ROS 2 and Nav2 official documentation.

Agentic AI			
Course Code:	**CA635**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P):**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Explain the principles and terminology of agentic artificial intelligence and autonomous intelligent agents.
2. Analyse problems and trade-offs concerning planning, reasoning and decision-making in agentic systems.
3. Apply suitable methods drawn from large language models, tool integration and multi-agent collaboration.
4. Design agentic AI systems for solving real-world autonomous tasks.
5. Evaluate approaches and responsible practice relating to autonomous AI agents.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Explain the fundamental concepts and terminology of agentic AI and intelligent agents.
2. Analyse problems using the methods covered under planning, reasoning and autonomous decision-making.
3. Apply appropriate techniques for tool usage, workflow automation and multi-agent collaboration.
4. Design an agentic AI solution for a specified real-world application.
5. Evaluate alternative agentic AI approaches using technical, performance and reliability criteria.
6. Develop and communicate a responsible, reproducible agentic AI solution.

UNIT I: FOUNDATIONS OF AGENTIC AI

Intelligent agents; Agentic AI fundamentals; Characteristics of autonomous agents; Agent architectures; Perception, reasoning and action; Reactive and deliberative agents; Goal-oriented behaviour; Planning concepts; Memory and context management; Agent life cycle; Applications; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations.

UNIT II: AGENT REASONING AND TOOL INTEGRATION

Reasoning strategies; Task decomposition; Planning and execution; Tool calling; API integration; External knowledge retrieval; Workflow automation; Memory management; Agent orchestration; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises.

UNIT III: MULTI-AGENT SYSTEMS

Multi-agent systems; Collaborative agents; Communication protocols; Coordination strategies; Task allocation; Negotiation; Conflict resolution; Distributed problem solving; Human-agent collaboration; Agent evaluation; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Reproducibility practices.

UNIT IV: AGENTIC AI FRAMEWORKS AND APPLICATIONS

Agent development frameworks; Workflow design; Conversational agents; Coding assistants; Autonomous research assistants; Enterprise automation; Robotics applications; Software engineering applications; Decision support systems; Healthcare and finance use cases; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations.

UNIT V: DEPLOYMENT, EVALUATION AND RESPONSIBLE AGENTIC AI

Agent evaluation; Benchmarking; Reliability; Scalability; Monitoring; Explainability; Human oversight; Fairness; Bias; Privacy; Security; Ethical and legal considerations; Governance; Responsible AI; Emerging trends in Agentic AI; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance measures; Common limitations and failure scenarios; Future research directions.

Textbooks / Reference books:

1. Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, 4th Edition, Pearson.
2. Wooldridge, M., An Introduction to MultiAgent Systems, 2nd Edition, Wiley.
3. Bratman, M., Intentions, Plans, and Practical Reason, Harvard University Press.
4. Foster, D., Generative Deep Learning, 2nd Edition, O'Reilly Media.
5. Attention mechanisms. Luger, G. F., Artificial Intelligence: Structures and Strategies for Complex Problem Solving, 8th Edition,

Intelligent Information Retrieval			
Course Code:	**CA637**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P):**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Explain the principles and architecture of intelligent information retrieval systems.
2. Analyse indexing, ranking, and retrieval techniques for structured and unstructured data.
3. Apply machine learning and deep learning techniques for intelligent search.
4. Design retrieval systems using semantic search, knowledge graphs, and Retrieval-Augmented Generation (RAG).
5. Evaluate intelligent retrieval systems using effectiveness, efficiency, and user-centric metrics.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Explain the concepts and architecture of intelligent information retrieval systems.
2. Analyse indexing, ranking, and query processing techniques for information retrieval.
3. Apply machine learning and deep learning methods for intelligent search applications.
4. Design semantic retrieval systems integrating embeddings, knowledge graphs, and RAG.
5. Evaluate retrieval models using effectiveness, efficiency, scalability, and relevance metrics.
6. Develop intelligent retrieval solutions using contemporary AI frameworks and responsible AI practices.

UNIT I: FOUNDATIONS OF INFORMATION RETRIEVAL

Information retrieval fundamentals; Information retrieval models; Structured, semi-structured and unstructured data; Document representation; Tokenization; Text normalization; Stop-word removal; Stemming and lemmatization; Inverted index; Positional index; Boolean retrieval; Vector Space Model; TF-IDF; BM25 ranking; Query processing; Query expansion; Relevance feedback; Retrieval architecture; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures.

UNIT II: MACHINE LEARNING FOR INFORMATION RETRIEVAL

Learning-to-rank; Feature engineering; Ranking functions; Query classification; Document classification; Text clustering; Recommendation techniques; Personalised search; Collaborative filtering; Content-based filtering; Search log analysis; Click-through prediction; User behaviour modelling; Search analytics; Learning-to-rank algorithms; Supervised and unsupervised retrieval; Comparative evaluation of techniques; Performance measures; Contemporary tools and practices; Scalability considerations; Industry case studies.

UNIT III: DEEP LEARNING AND SEMANTIC SEARCH

Word embeddings; FastText; Word2Vec; GloVe; Contextual embeddings; BERT; Sentence Transformers; Dense retrieval; Dual encoders; Cross encoders; Semantic search; Neural ranking; Vector databases; Approximate Nearest Neighbor (ANN) search; FAISS; Pinecone; ChromaDB; Hybrid retrieval; Cross-lingual retrieval; Multimodal retrieval; Worked examples and case analysis; Comparative evaluation of techniques; Performance and quality measures; Contemporary tools and practices.

UNIT IV: KNOWLEDGE-AWARE RETRIEVAL AND GENERATIVE AI

Knowledge graphs; Entity linking; Entity retrieval; Ontologies; SPARQL; Graph; Graph neural networks for retrieval; Retrieval-Augmented Generation (RAG); Vector search; Hybrid search; agentic AI for retrieval; Query rewriting; Document chunking; Embedding generation; Context retrieval; Hallucination mitigation; Citation generation; Enterprise search; AI-powered question answering; Comparative evaluation of techniques; Contemporary AI tools and practices; Responsible AI considerations.

UNIT V: EVALUATION, DEPLOYMENT AND APPLICATIONS

Evaluation metrics; Precision; Recall; MAP; MRR; NDCG; User satisfaction; Search latency; Scalability; Distributed indexing; Cloud-based retrieval; Edge search; Privacy-preserving retrieval; Explainable retrieval; Bias and fairness in search; Secure retrieval; Retrieval monitoring; Model deployment; Search APIs; Enterprise search systems; Web search; Academic search; Healthcare retrieval; Legal information retrieval; Recommendation systems; Worked examples and case analysis; Contemporary tools and practices; Industry case studies.

Textbooks / Reference books:

1. Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press.
2. Ricardo Baeza-Yates and Berthier Ribeiro-Neto, Modern Information Retrieval, Addison-Wesley.
3. Stefan Büttcher, Charles L. A. Clarke, and Gordon V. Cormack, Information Retrieval: Implementing and Evaluating Search Engines, MIT Press.
4. Tommaso Teofili, AI-Powered Search, Manning Publications.
5. Emmanuel Ameisen, Building Machine Learning-Powered Applications, O'Reilly Media.

Introduction to Large Language Models			
Course Code:	**CA639**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P):**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Explain the principles and terminology of large language models and transformer-based architectures.
2. Analyse problems and trade-offs concerning language modelling, prompting and model adaptation.
3. Apply suitable methods drawn from prompt engineering, retrieval augmentation and fine-tuning.
4. Design LLM-based solutions for real-world natural language processing applications.
5. Evaluate approaches and responsible practice relating to large language models.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Explain the fundamental concepts and terminology of large language models and transformer architectures.
2. Analyse problems using the methods covered under language modelling and prompt engineering.
3. Apply appropriate techniques for prompting, retrieval augmentation and model adaptation to realistic applications.
4. Design an LLM-based solution for a specified natural language processing task.
5. Evaluate alternative LLM approaches using technical, performance and reliability criteria.
6. Develop and communicate a responsible, reproducible LLM-based solution.

UNIT I: FOUNDATIONS OF LARGE LANGUAGE MODELS

Natural language processing fundamentals; Evolution of language models; Statistical language models; Neural language models; Transformer architecture; Tokenization; Attention mechanism; Positional encoding; Pre-training concepts; Foundation models; Applications of LLMs; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations.

UNIT II: LLM ADAPTATION AND CONTEXT MANAGEMENT

Prompting fundamentals for LLM interaction; Zero-shot, one-shot, and few-shot learning; Basic instruction and structured prompting; Context-window management; Retrieval-augmented generation fundamentals; Fine-tuning concepts; Parameter-efficient adaptation; Model alignment and instruction tuning; Model limitations and adaptation trade-offs; Evaluation of adapted LLMs.

UNIT III: LLM APPLICATIONS

Text generation; Summarization; Question answering; Conversational AI; Information extraction; Text classification; Code generation; Translation; Document analysis; Enterprise applications; Domain-specific LLMs; Multimodal LLM overview; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Reproducibility practices.

UNIT IV: LLM DEPLOYMENT AND INTEGRATION

LLM inference; APIs and cloud-based LLM services; Open-source LLMs; Model serving; Workflow automation; Tool calling; Function calling; AI assistants; Integration with external knowledge sources; Performance optimization; Monitoring and evaluation; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations.

UNIT V: RESPONSIBLE LLMs AND EMERGING TRENDS

Hallucinations; Explainability; Bias and fairness; Privacy; Security; Copyright and intellectual property; Ethical considerations; Responsible AI; LLM evaluation benchmarks; Governance; Model alignment; Emerging trends in LLMs; Agentic AI overview; Future directions; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance measures; Common limitations and failure scenarios; Contemporary tools and practices; Ethical and societal implications.

Textbooks / Reference books:

1. Lewis Tunstall, Leandro von Werra and Thomas Wolf, Natural Language Processing with Transformers, O'Reilly Media.
2. Denis Rothman, Transformers for Natural Language Processing, 3rd Edition, Packt Publishing.
3. Jacob Eisenstein, Natural Language Processing, MIT Press.
4. Yoav Goldberg, Neural Network Methods for Natural Language Processing, Morgan & Claypool Publishers.
5. Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, 4th Edition, Pearson.

AI Planning and Decision Making			
Course Code:	**CA641**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P):**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Explain the principles and terminology of state-space planning.
2. Analyse problems and trade-offs concerning classical planning.
3. Apply suitable methods drawn from planning under uncertainty.
4. Design solutions that integrate temporal and multi-agent planning.
5. Evaluate approaches and responsible practice relating to integrated decision systems.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Explain the fundamental concepts and terminology of state-space planning.
2. Analyse problems using the methods covered under classical planning.
3. Apply appropriate techniques from planning under uncertainty to realistic datasets or systems.
4. Design a solution that integrates concepts from temporal and multi-agent planning.
5. Evaluate alternative approaches using technical, performance and reliability criteria.
6. Develop and communicate a responsible, reproducible solution drawing on integrated decision systems.

UNIT I: STATE-SPACE PLANNING

Problem representation; Uninformed and heuristic search; Admissibility; Planning graphs; Problem states, actions, goals and costs; Uninformed search; Uniform-cost search; Greedy best-first and A-star; Admissible and consistent heuristics; Graph-search implementation; Duplicate detection; Memory-bounded search; Complexity and optimality; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises.

UNIT II: CLASSICAL PLANNING

STRIPS/PDDL; Forward and backward planning; Partial-order planning; Heuristics; STRIPS and PDDL representation; Preconditions and effects; Planning graphs; Delete relaxation; Heuristic construction; Landmark methods; Plan validation; Benchmark domains; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises; Hardware deployment issues; Reproducibility practices; Ethical and societal implications.

UNIT III: PLANNING UNDER UNCERTAINTY

Bayesian decision theory; MDPs; Value and policy iteration; POMDPs; Belief states; Markov decision processes; Bellman equations; Partially observable MDPs; Belief update; Approximate planning; Risk and uncertainty measures; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises; Hardware deployment issues; Reproducibility practices; Ethical and societal implications.

UNIT IV: TEMPORAL AND MULTI-AGENT PLANNING

Time and resources; Scheduling; Decentralized planning; Game-theoretic reasoning; Durative actions; Temporal constraints; Resources; Planning with deadlines; Coordination protocols; Negotiation; Conflict resolution; Multi-agent plan evaluation; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises; Hardware deployment issues; Reproducibility practices; Ethical and societal implications; Recent research directions.

UNIT V: INTEGRATED DECISION SYSTEMS

Task and motion planning; Online/receding-horizon planning; Learning-guided planning; Safety and explainability; Task-and-motion planning; Receding-horizon and model-predictive decisions; Planning-learning interfaces; Execution monitoring; Contingency handling; Recovery; Resource constraints; Explainable decisions; Safety constraints; Scenario-based evaluation; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Common limitations and failure scenarios; Contemporary tools and practices; Safety and robustness considerations; Simulation-based exercises; Hardware deployment issues.

Textbooks / Reference books:

1. S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, 4th ed., Pearson.
2. M. Ghallab, D. Nau and P. Traverso, Automated Planning and Acting, Cambridge.
3. S. M. LaValle, Planning Algorithms, Cambridge.
4. M. Puterman, Markov Decision Processes, Wiley.
5. D. P. Bertsekas, Dynamic Programming and Optimal Control, Athena Scientific.

Prompt Engineering			
Course Code:	**CA643**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P):**	
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Explain the fundamentals of Prompt Engineering and Large Language Models.
2. Analyse prompt design strategies for diverse AI tasks and applications.
3. Apply prompt engineering techniques for text, code, image, and multimodal AI systems.
4. Design intelligent AI applications using Retrieval-Augmented Generation (RAG), function calling, and AI agents.
5. Evaluate prompt quality considering accuracy, robustness, safety, ethics, and performance.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Explain the concepts and architecture of prompt engineering for foundation models.
2. Analyse prompt optimization techniques for different AI applications.
3. Apply prompt engineering methodologies for solving real-world problems.
4. Design AI-powered applications integrating prompts, RAG, APIs, and AI agents.
5. Evaluate prompt effectiveness using technical, qualitative, and quantitative metrics.
6. Develop responsible, secure, and scalable prompt-driven AI solutions.

UNIT I: FOUNDATIONS OF PROMPT ENGINEERING

Artificial Intelligence foundations; Generative AI; Foundation Models; Transformer architecture; Tokens and tokenization; Context windows; Embeddings; Prompt lifecycle; Prompt anatomy; Zero-shot learning; One-shot learning; Few-shot learning; Instruction prompting; Role prompting; Chain-of-thought prompting; Self-consistency; Prompt templates; Prompt libraries; Prompt management; Prompt evaluation; AI model capabilities and limitations; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures.

UNIT II: ADVANCED PROMPTING TECHNIQUES

Prompt optimization; Prompt decomposition; Step-by-step reasoning; Tree-of-Thoughts; Graph-of-Thoughts; ReAct prompting; Self-reflection; Prompt chaining; Structured prompting; XML and JSON prompting; Function calling; Tool calling; Context engineering; Long-context prompting; Prompt compression; Prompt versioning; Prompt debugging; Prompt testing; Prompt automation; Comparative evaluation of techniques; Performance measures; Contemporary prompting frameworks; Design trade-offs; Tool-supported exercises.

UNIT III: CONTEXT-AWARE PROMPTING AND RETRIEVAL

Context engineering; Retrieval-Augmented Generation (RAG) fundamentals; Vector representations and semantic search; Document chunking and context selection; Grounding and citation generation; Prompting with retrieved context; Function and tool calling from a prompt-engineering perspective; Context-window management; Prompt-based knowledge retrieval; Evaluation of retrieval-augmented prompts; Practical applications and case studies.

UNIT IV: MULTIMODAL PROMPT ENGINEERING

Multimodal AI; Vision-language models; Image prompting; Image generation; Image editing prompts; Audio prompting; Video prompting; Code generation; SQL generation; Report generation; Prompt engineering for software development; Prompt engineering for data analytics; Prompt engineering for education; Healthcare AI; Business intelligence; Domain-specific prompt design; Workflow automation; Human-AI collaboration; Comparative evaluation of techniques; Contemporary AI platforms; Industry case studies.

UNIT V: EVALUATION, SAFETY AND ENTERPRISE APPLICATIONS

Prompt evaluation; Benchmarking; Hallucination detection; Robustness testing; Prompt injection attacks; Jailbreak attacks; AI security; Responsible AI; Explainability; Fairness; Bias mitigation; Privacy; Ethical AI; AI governance; Cost optimization; Latency optimization; Model monitoring; PromptOps; Enterprise deployment; AI compliance; Emerging trends in Prompt Engineering; Worked examples and case analysis; Comparative evaluation of techniques; Implementation considerations; Performance and quality measures; Contemporary tools and practices.

Text Books / Reference books:

1. Lee Boonstra, Prompt Engineering, O'Reilly Media.
2. Noah Gift, Practical Generative AI with ChatGPT, O'Reilly Media.
3. Chip Huyen, AI Engineering, O'Reilly Media.
4. John Berryman and Albert Ziegler, Prompt Engineering for LLMs, O'Reilly Media.
5. LangChain Documentation (latest edition).
6. OpenAI API Documentation.
7. Anthropic Prompt Engineering Guide.
8. Google Gemini API Documentation.

Evolutionary Computation			
Course Code:	**CA645**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P):**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To introduce the main concepts, techniques, and applications in the field of Evolutionary Computation.
2. To give students practical experience in evolutionary computation techniques and how to implement them using different programming languages.
3. To develop an understanding of issues in computing using evolutionary algorithms, particularly Genetic Algorithms.
4. To learn how Evolutionary Computation can be applied to various application areas.
5. To learn multi-objective optimization.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Understand the implementation issues of evolutionary algorithms.
2. Determine appropriate parameter settings to make different evolutionary algorithms work effectively.
3. Design new evolutionary operators, representations, and fitness functions for specific practical and scientific applications.
4. Effectively present an evolutionary computation article to an audience.
5. Review and critique evolutionary computation articles.

UNIT I: FUNDAMENTAL CONCEPTS AND DIALECTS

Metaheuristics; Requirements of metaheuristics in optimization problems; Characteristics of problems suitable for application of metaheuristics; Evolutionary Computing metaphor and inspiration from biology; Applications, pros and cons of Evolutionary Computation; Components of Evolutionary Computation; Example applications – Eight Queen Problem, Knapsack Problem; Various dialects of Evolutionary Computation.

UNIT II: GENETIC ALGORITHMS

Introduction; Canonical Genetic Algorithm; Binary, Integer, Real-valued and Permutation representations and variation operators for them; Population models; Parent selection – Fitness proportionate selection, Rank-based selection, Tournament selection; Survivor strategy; Implementation issues; Parameter control and effect on GA dynamics.

UNIT III: THEORETICAL FOUNDATION OF GENETIC ALGORITHMS

Schemas and Hyperplane Sampling; Schemata Theorem; Limitations and Building Block Hypothesis; Two-Armed Bandit Problem; Deceiving a Genetic Algorithm; Minimal Deceptive Problem; Royal Road Functions; SAHC; NAHC; RMHC; Hitchhiking; Exact mathematical models of SGA; Statistical-mechanics approaches.

UNIT IV: PROBLEM SOLVING USING EVOLUTIONARY COMPUTATION

Evolving computer programs; Evolving LISP programs; Evolving Cellular Automata; Data analysis and prediction; Predicting dynamical systems; Predicting protein structure; Evolving Neural Networks – Evolving weights, Architecture (Direct Encoding and Grammatical Encoding), and Learning Rules;

Baldwin Effect and Evolutionary Reinforced Learning.

UNIT V: PARALLEL IMPLEMENTATION AND OTHER DIALECTS OF EVOLUTIONARY COMPUTATION

Parallel implementation of Genetic Algorithms; Genetic Programming; Evolutionary Programming; Evolutionary Strategies; Learning Classifier Systems; Memetic Algorithms; Introduction to Multi-Objective Genetic Algorithm (MOGA); Introduction to Simulated Annealing; Artificial Bee Colony Algorithms; Ant Colony Optimization Algorithms; Swarm Intelligence; Evolutionary Computation.

Textbooks / Reference books:

1. Melanie Mitchell, An Introduction to Genetic Algorithms, MIT Press.
2. David E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Pearson Education.
3. Lawrence Davis, Handbook of Genetic Algorithms, Van Nostrand Reinhold.
4. Kalyanmoy Deb, Multi-Objective Optimization Using Evolutionary Algorithms, Wiley.
5. Marco Dorigo and Thomas Stützle, Ant Colony Optimization, MIT Press.

Trustworthy and Responsible AI			
Course Code:	**CA647**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Understand the fundamental principles and characteristics of Trustworthy and Responsible AI.
2. Analyse issues related to fairness, bias, transparency, explainability and accountability in AI systems.
3. Apply techniques for developing reliable, robust, secure and privacy-preserving AI systems.
4. Evaluate AI models using appropriate ethical, technical and performance criteria.
5. Develop responsible AI solutions considering legal, societal and ethical implications.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1. Explain the fundamental concepts and principles of Trustworthy and Responsible AI.
2. Identify and analyse bias, fairness and discrimination in AI systems.
3. Apply explainability, transparency, privacy and security techniques to AI applications.
4. Evaluate AI systems for reliability, robustness, safety and accountability.
5. Analyse legal, ethical and societal implications of AI technologies.
6. Develop and communicate responsible AI solutions following appropriate governance and ethical practices.

UNIT I: FOUNDATIONS OF TRUSTWORTHY AND RESPONSIBLE AI

Trustworthy AI: Concepts and Principles; Responsible AI; AI Ethics; Human-Centred AI; Reliability; Safety; Robustness; Accountability; Transparency; AI Lifecycle; Trust in AI Systems; Challenges and Risks; Case Studies.

UNIT II: FAIRNESS, BIAS AND INCLUSION

AI Bias; Sources of Bias; Data Bias; Algorithmic Bias; Fairness in AI; Types of Fairness; Discrimination; Fairness Metrics; Bias Detection and Mitigation; Inclusive AI; Demographic and Representation Issues; Case Studies.

UNIT III: EXPLAINABLE AND TRANSPARENT AI

Explainable AI (XAI); Interpretability; Transparency; Model-Agnostic and Model-Specific Explainability; Feature Importance; LIME; SHAP; Counterfactual Explanations; Explainability for Deep Learning; Human-AI Interaction; Limitations of Explainability.

UNIT IV: PRIVACY, SECURITY AND ROBUST AI

AI Privacy; Data Protection; Privacy-Preserving Machine Learning; Anonymization; Differential Privacy; Federated Learning; AI Security; Adversarial Attacks; Robustness; Model Reliability; AI Safety; Secure AI Deployment; Risk Assessment.

UNIT V: AI GOVERNANCE, LAW AND SOCIETAL IMPACT

AI Governance; Accountability and Responsibility; AI Regulations and Standards; Intellectual Property; Copyright; Data Protection Laws; Ethical Decision-Making; Societal Impact of AI; Human Oversight; AI Risk Management; Responsible Generative AI; Emerging Trends and Case Studies.

Text Books/Reference Books:

1. Virginia Dignum, Responsible Artificial Intelligence, Springer.
2. Mark Coeckelbergh, AI Ethics, MIT Press.
3. Christoph Molnar, Interpretable Machine Learning.
4. Patrick Hall, Navdeep Gill and others, Responsible Machine Learning.
5. NIST, Artificial Intelligence Risk Management Framework (AI RMF).

Predictive Analytics			
Course Code:	**CA649**	**Course Credits:**	**3**
Course Category:	**PEC**	**Course (U / P):**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Understand the concepts and techniques of predictive analytics.
2. Apply statistical and machine learning methods for predictive modeling.
3. Analyse structured and unstructured data to extract predictive insights.
4. Develop predictive models for real-world decision-making.
5. Evaluate predictive models using appropriate performance metrics and ethical considerations.

COURSE OUTCOMES
At the end of the course, the students should be able to:
1**.** Explain the principles and applications of predictive analytics.
2**.** Apply statistical and machine learning techniques for prediction.
3. Analyse datasets to identify patterns and predictive insights.
4. Develop predictive models for real-world applications.
5. Evaluate predictive models using appropriate statistical and machine learning performance metrics.
6. Assess predictive models in deployment with respect to reliability, data drift, fairness, privacy, security, and responsible AI considerations.

UNIT I: INTRODUCTION TO PREDICTIVE ANALYTICS

Introduction to Predictive Analytics; Predictive, Descriptive and Prescriptive Analytics; Data Analytics Life Cycle; Data Collection and Integration; Data Preprocessing; Data Cleaning; Feature Engineering; Exploratory Data Analysis (EDA); Data Visualization; Statistical Foundations; Probability Distributions; Sampling Techniques; Business Applications of Predictive Analytics.

UNIT II: STATISTICAL AND MACHINE LEARNING TECHNIQUES

Regression Analysis; Linear and Logistic Regression; Decision Trees; Random Forest; Naïve Bayes; K-Nearest Neighbours (KNN); Support Vector Machines (SVM); Ensemble Learning; Model Selection; Cross Validation; Bias-Variance Tradeoff; Hyperparameter Tuning; Feature Selection.

UNIT III: TIME SERIES AND FORECASTING

Introduction to Time Series Data; Trend and Seasonality; Time Series Decomposition; Moving Average Models; Exponential Smoothing; ARIMA Models; Forecast Accuracy Measures; Demand Forecasting; Financial Forecasting; Sales Prediction; Weather Prediction; Predictive Maintenance; Anomaly Detection.

UNIT IV: DEEP LEARNING AND ADVANCED PREDICTIVE ANALYTICS

Neural Networks; Deep Learning Fundamentals; Artificial Neural Networks (ANN); Recurrent Neural Networks (RNN); Long Short-Term Memory (LSTM); Predictive Analytics using Deep Learning; Natural Language Processing for Prediction; Sentiment Analysis; Recommendation Systems; Predictive Analytics using Big Data; AutoML; Explainable AI (XAI).

UNIT V: DEPLOYMENT, ETHICS AND INDUSTRY APPLICATIONS

Model Evaluation Metrics; Accuracy, Precision, Recall, F1-Score, ROC-AUC; Model Deployment; MLOps Fundamentals; Model Monitoring; Data Drift and Concept Drift; Responsible AI; Fairness and Bias in Predictive Models; Privacy and Security; Cloud-Based Predictive Analytics; Industry Case Studies in Healthcare, Finance, Retail, Manufacturing & Smart Cities; Future Trends.

Textbooks / Reference books:

1. Eric Siegel, Predictive Analytics: The Power to Predict Who Will Click, Buy, Lie, or Die, Wiley.
2. Dean Abbott, Applied Predictive Analytics, Wiley.
3. Trevor Hastie, Robert Tibshirani and Jerome Friedman, The Elements of Statistical Learning, Springer.
4. Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani, An Introduction to Statistical Learning, Springer.
5. Jiawei Han, Micheline Kamber and Jian Pei, Data Mining: Concepts and Techniques, Morgan Kaufmann.
6. Aurélien Géron, Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow, O'Reilly Media.
7. Ian Goodfellow, Yoshua Bengio and Aaron Courville, Deep Learning, MIT Press.

SEMESTER IV

CA692: Dissertation Part-II