UNIVERSITY SCHOOL
OF
INFORMATION AND COMMUNICATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAMME STRUCTURE

M.TECH. COMPUTER SCIENCE AND ENGINEERING

SPECIALIZATION : ARTIFICIAL INTELLIGENCE AND ROBOTICS

2025-2027

GAUTAM BUDDHA UNIVERSITY

GAUTAM BUDH NAGAR, GREATER NOIDA, UP, INDIA

SEMESTER I

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	CS521	Advanced Data Base Management System	3	0	0	3	CC1
2	CS523	Advanced Data Structure and Algorithm	3	1	0	4	CC2
3	CS525	Research Techniques in ICT	3	0	0	3	CC3
4	CS527	Python Programming	3	0	0	3	CC4
5	ES415	Energy and Environment	3	0	0	3	OE1 /AECC
6	CD581	Advanced Data Base Management System Lab	0	0	4	2	CC-L1
7	CD583	Python Programming Lab	0	0	4	2	CC-L2 / SEC
8	GP	General Proficiency	Non Credit				
		Total Hours and Credits	**15**	**1**	**8**	**20**	

SEMESTER II

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	CA522	Computer Vision Applications	3	0	0	3	CC5
2	CS524	Deep Learning	3	0	0	3	CC6
3	CA526	Cognitive Computing	3	0	0	3	CC7 / SEC
5	CS528	Blockchain Technology	3	1	0	4	CC8
		Elective-1	3	0	0	3	E1 / DSE
8	CA582	Computer Vision Applications Lab	0	0	4	2	CC-L3
9	CS584	Deep Learning Lab	0	0	4	2	CC-L4
10	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	Types
1	CA530	Natural Language Processing	3	0	0	3	E1
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 1 is approved in 33rd BoS and Semester 2 is approved in 35th BoS.

SEMESTER 1

ADVANCED DATABASE MANAGEMENT SYSTEM			
Course Code:	**CS521**	**Course Credits:**	**3**
Course Category:	**CC1**	**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**
Total No. of Lectures (L + T):	**45 + 00**	**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 on enhanced database

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 on relational database.
3 Understand and fetch data from object oriented, parallel and distributed databases.
4 Use XML and understand unstructured data
5 Implement concept and deduction of enhanced database on different applications

UNIT I INTRODUCTION TO DATABASE DESIGN

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

UNIT II DATABASES DESIGN THEORY

Problems Caused by Redundancy, Decompositions, Problems Related to Decomposition, Reasoning About FD's, FIRST, SECOND, THIRD Normal Form, BCNF, Forth 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).

Text Books:

1. Elmasri and Navathe, Fundamentals of Database Systems,
2. Ramakrishnan and Gehrke, Database Management Systems,

References Books:

1. Korth, Silberschatz, Sudarshan, Database System Concepts,
2. Rob and Coronel, Database Systems: Design, Implementation and Management,
3. Date and Longman, Introduction to Database Systems,

ADVANCED DATA STRUCTURE AND ALGORITHM			
Course Code:	**CS523**	**Course Credits:**	**4**
Course Category:	**CC2**	**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**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 Understand the appropriate data structures, ADT libraries, and use 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 analyze algorithms in terms of their efficiency and correctness.
5 Understanding 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 analyze programming problem statements.
2 Choose appropriate data structures and algorithms for a specific problem.
3 Understand the necessary mathematical abstraction to solve problems.
4 Come up with analysis of efficiency and proofs 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 Algorithms, 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:

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.

References Books:

1. Algorithms: Robert Sedgewick and Kevin Wayne.
2. Advanced Data Structures: Peter Brass.

RESEARCH TECHNIQUES IN ICT			
Course Code:	**CS 525**	**Course Credits:**	**3**
Course Category:	**CC3**	**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**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**
COURSE OBJECTIVES			
1 To get the understanding about what is research.			
2 To have the knowledge about research methodology.			
3 Awareness of methodology to be followed for report and paper writing.			
4 Familiarization to different models and algorithms during research.			
5 Conversant with various simulation and soft computing techniques.			
COURSE OUTCOMES			
At the end of the course the students should be able to:			
1 Acquainted with procedure for carrying out the research and its importance.			
2 Can identify and apply the appropriate research techniques according to domain of research.			
3 Recognize models and algorithms required for their study.			
4 Know about research paper writing and publication procedure.			
5 Will get the direction about different simulation packages.			

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 modeling, 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:
1. Research Methodology: Methods and Techniques, C.R. Kothari

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

PYTHON PROGRAMMING			
Course Code:	**CS527**	**Course Credits:**	**3**
Course Category:	**CC4**	**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**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

1. To learn and understand Python programming basics and paradigm
2. To learn and understand python looping, control statements and string manipulations.
3. Students should be made familiar with the concepts of GUI controls and designing GUI applications.
4. To learn and know the concepts of file handling, exception handling and database connectivity.
5. To learn and understand database connectivity in the python programming language.

COURSE OUTCOMES

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

1. To read and write simple Python programs.
2. To develop Python programs with conditionals and loops.
3. To define Python functions and to use Python data structures –– lists, tuples, dictionaries
4. To do input/output with files in Python
5. To do searching ,sorting and merging in Python

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 Structure : 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 the 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:

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:	**OE1**	**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
Total No. of Lectures (L + T):	**40+00**	**End Sem. Exam Hours:**	3
COURSE OBJECTIVES			
To provide in-depth knowledge of renewable and non-renewable energy resources, their harnessing techniques and energy-environment issues.			
COURSE OUTCOMES			
The knowledge so gathered could be utilized to meet the challenges of energy vis-a-vis environmental security.			

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

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 hydro-power, tidal energy, ocean thermal energy conversion, wind power, geothermal energy, solar energy (solar collectors, photo-voltaic modules, solar ponds)
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

Bioenergy: Types, importance, methods of energy production from biomass

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 bio-energy sources; National Solar Mission, National Mission for Enhanced Energy Efficiency, case studies.

Text 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

ADVANCED DATABASE MANAGEMENT SYSTEM LAB			
Course Code:	**CS581**	**Course Credits:**	2
Course Category:	**CC-L1**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lab (Hrs/Week) / Total No. of Lab	**03 / 10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
. To explore the features of a Database Management Systems
. To interface a database with front end tools
. To understand the internals of a database system
. To provide a strong foundation in advanced database concepts from an industry perspective.
. To learn query processing and transaction management concepts for object-relational database and distributed database

COURSE OUTCOME
At the end of the course the students should be able to:
1. Develop and apply critical thinking skills.
2. Design and present Lab as well as project reports
3. Apply appropriate methods for the analysis of raw data
4. Perform logical troubleshooting as and when required.
5. Verify and implement the concepts and theory learnt in class.

List of Experiments:

1. Introduction to MySQL, PostgreSql, Microsoft Sqlsoftwares.
2. An exercise of 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 sub queries
10. Procedures, View and Triggers

PYTHON PROGRAMMING LAB			
Course Code:	CS583	**Course Credits:**	2
Course Category:	CC-L2	**Course (U / P)**	U / P
Course Year (U / P):	1P	**Course Semester (U / P):**	2P
No. of Lab (Hrs/Week) / Total No. of Lab	03 / 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 2

Computer Vision Applications

Course Code:	**CA522**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**IP / P**
Course Year (U / P):	**4IP / 1P**	**Course Semester (U / P):**	**8IP / 2P**
No. of Lectures + Tutorials (Hrs/Week):	**3**	**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 Computer Vision, including image formation, camera models, color spaces, digital image types, and image acquisition techniques.
2. To analyze object shape representation and motion estimation techniques using mathematical tools such as PCA, eigenvalues/eigenvectors, matrix transformations, and motion modeling.
3. To extract and interpret image features using edge detection, corner detection, scale-space analysis, feature descriptors (SIFT, SURF, HOG), and understand geometric concepts like stereo vision and epipolar geometry.
4. To develop a strong mathematical foundation for Machine Learning in vision applications, including linear algebra concepts, supervised learning algorithms, and model evaluation metrics.
5. To design and implement deep learning models, particularly Convolutional Neural Networks (CNNs), for image classification and recognition tasks using modern architectures and training 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:

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.

Reference Book:

1. The Elements of Statistical Learning Trevor Hastie, Robert Tibshirani, Jerome Friedman, Springer, 2nd Edition, 2009.

2. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, Aurélien Géron, O'Reilly Media, 2nd Edition, 2019.

3. 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:	**CC**	**Course (U / P)**	**IP / P**
Course Year (U / P):	**4IP / 1P**	**Course Semester (U / P):**	**8IP / 2P**
No. of Lectures + Tutorials (Hrs/Week):	**3**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

1. Gain a historical perspective of Deep Learning and its foundations. Become familiar with basic principles of AI toward problem-solving, inference, perception, knowledge representation and learning.
2. Experience Deep Learning Algorithms and development various AI tools.
3. Explore the current scope, potential, limitations, and implications of Deep learning.
4. Analysis on the effect of deep learning practices in software systems.
5. Evaluation of deep learning algorithms

COURSE OUTCOME

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

1. Demonstrate knowledge of the building blocks of Deep Learning system.
2. Analyze and formalize the problem as a state space, graph, design heuristics, and select different search or game-based techniques to solve them.
3. Develop intelligent algorithms for constraint satisfaction problems and also design intelligent systems for Game Playing
4. Attain the capability to represent various real-life problem domains using logic-based techniques and use this to perform inference or planning.
5. To evaluate and compare deep learning models across different application domain.

UNIT I INTRODUCTION TO DEEP LEARNING

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

UNIT II INTRODUCTION TO DEEP LEARNING

History of Deep Learning- A Probabilistic Theory of Deep Learning- Back propagation and regularization, batch normalization- VC Dimension and Neural Nets-Deep Vs Shallow Networks Convolutional Networks- Generative Adversarial Networks (GAN), Semi-supervised Learning.

UNIT III DIMENTIONALITY REDUCTION

Linear (PCA, LDA) and manifolds, metric learning - Auto encoders and dimensionality reduction in networks - Introduction to Convnet - Architectures – AlexNet, VGG, Inception, ResNet - Training a Convnet: weights initialization, batch normalization, hyper parameter optimization.

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:

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:	CC	Course (U / P)	IP/P
Course Year (U / P):	4IP / 1P	Course Semester (U / P):	8IP / 2P
No. of Lectures + Tutorials Hrs/Week):	03	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 AI systems.
2. To study human cognitive processes such as learning, memory, perception, and reasoning, and apply these concepts to the design of intelligent systems.
3. To learn machine learning, neural networks, and NLP techniques that enable systems to learn from data and interact naturally with humans.
4. To apply cognitive computing concepts to practical domains like healthcare, cybersecurity, education, and decision-support systems while considering ethical implications
5. Foster understanding of efficiency analysis for cognitive computing 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 1: 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 2: 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 3: 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 4: 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 5: 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

1. Gusfield, D. Algorithms on Strings, Trees, and Sequences. Cambridge University Press, 1997.
2. Mount, D. Bioinformatics: Sequence and Genome Analysis. CSHL Press, 2004.

Reference Books

1. Aho, A.V., et al. Compilers: Principles, Techniques, and Tools. Addison-Wesley, 2006 (pattern matching chapters).
2. Sedgewick, R., Wayne, K. Algorithms. 4th ed., Addison-Wesley, 2011 (string matching sections).
3. Jain, A.K. Pattern Recognition and Machine Learning (select chapters on matching). Springer, 2006.
4. Tim Roughgarden, Beyond the Worst-Case Analysis of Algorithms, Cambridge University Press, 2021.

BLOCKCHAIN TECHNOLOGY			
Course Code:	CS528	Course Credits:	4
Course Category:	CC	Course (U / P)	IP / P
Course Year (U / P):	4IP / 1P	Course Semester (U / P):	8IP / 2P
No. of Lectures + Tutorials (Hrs/Week):	3 +1	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. understand permissioned Blockchain and explore Hyperledger Fabric.			
5. To understand different types of crypto assets.			
COURSE OUTCOME			
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 Book

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

Reference Books:

1. Ritesh Modi, Solidity Programming Essentials: A beginner's guide to build smart contracts for Ethereum and blockchain, Packt Publishing, First edition, 2018.

2. Kumar Saurabh, Ashutosh Saxena, Blockchain Technology: Concepts and Applications, First Edition, Wiley Publications, First edition, 2020.

3. Chandramouli Subramanian, Asha A George, et al, Blockchain Technology, Universities Press (India) Pvt. Ltd, First edition, August 2020.

4. Blockchain Technology: Concepts and Applications, Kumar Saurabh and Ashutosh Saxena, Wiley.

Computer Vision Applications Lab			
Course Code:	**CA582**	**Course Credits:**	**3**
Course Category:	**CC-L**	**Course (U / P)**	**IP/P**
Course Year (U / P):	**4IP/1P**	**Course Semester (U / P):**	**8IP/2P**
No. of Lectures + Tutorials (Hrs/Week):	**04+ 00**	**Mid Sem. Exam Hours:**	**NIL**
Total No. of Lectures (L + T):	**10**	**End Sem. Exam Hours:**	**2**

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:	CC-L	Course (U / P)	IP / P
Course Year (U / P):	4 IP / 1P	Course Semester (U / P):	8IP / 2P
No. of Labs(Hrs/Week):	04	Mid Sem. Exam Hours:	NIL
Total No. of Labs	10	End Sem. Exam Hours:	2
COURSE OBJECTIVES			
4. Provide hands-on experience with deep learning frameworks and tools.			
5. Implement and experiment with neural network models for real-world problems			
6. Understand training, optimization, and evaluation of deep learning models.			
7. Apply deep learning techniques in domains such as computer vision, NLP, and time-series analysis.			
8. Encourage research-oriented thinking and reproducible experimentation.			
COURSE OUTCOMES			
At the end of the course the students should be able to understand:			
6. Implement basic to advanced neural network architectures using Python-based frameworks.			
7. Train, tune, and evaluate deep learning models on benchmark datasets.			
8. Apply CNNs, RNNs, and Transformer-based models to domain-specific problems.			
9. Analyze model performance using appropriate metrics and visualization tools			
10. 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.
12. Autoencoder: Dimensionality reduction and reconstruction analysis.
13. Transfer Learning: Fine-tuning pre-trained models (ResNet) on a custom dataset.

ELECTIVES

ELECTIVES

Natural Language Processing			
Course Code:	CA530	**Course Credits:**	3
Course Category:	E1	**Course (U / P)**	IP/P
Course Year (U / P):	4IP/2P	**Course Semester (U / P):**	8IP/2P
No. of Lectures (Hrs/Week):	03	**Mid Sem. Exam Hours:**	1.5
Total No. of Lectures (L):	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:

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: Speech and Multimodal Processing in NLP

Fundamentals of Speech Recognition, Speech Synthesis and Text-to-Speech (TTS), Audio signal processing for NLP, Speaker identification and verification, Multimodal NLP: Text, Speech, and Vision Integration, NLP for low-resource languages, Ethical considerations in speech and language technologies

Text 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:	**E1**	**Course (U / P)**	**IP/P**
Course Year (U / P):	**4 IP / 1P**	**Course Semester (U / P):**	**8IP / 2P**
No. of Lectures + Tutorials (Hrs/Week):	**3 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45+ 00**	**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:

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.

Reference Books:

1. Rabiner, L. R., & Juang, B.-H. (1993). Fundamentals of Speech Recognition. Prentice Hall.
2. Benesty, J., Sondhi, M. M., & Huang, Y. (Eds.). (2008). Springer Handbook of Speech Processing. Springer.
3. Furui, S. (2001). Digital Speech Processing: Synthesis, and Recognition (2nd ed.). CRC Press.
4. Holmes, J., & Holmes, W. (2001). Speech Synthesis and Recognition (2nd ed.). CRC Press.
5. Jelinek, F. (1997). Statistical Methods for Speech Recognition. The MIT Press.
6. Gales, M., & Young, S. (2008). The Application of Hidden Markov Models in Speech Recognition. Foundations and Trends in Signal Processing, 1(3), 195–304.
7. 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:	E1	Course (U / P)	IP/P
Course Year (U / P):	4 IP / 1P	Course Semester (U / P):	8IP / 2P
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. 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:

11. Explain core principles of pattern matching and distinguish from pattern recognition.
12. Design and analyze string, sequence, and template matching algorithms.
13. Implement image and geometric pattern matching for robotic vision tasks.
14. Apply approximate matching techniques to noisy sensor data in robotics.
15. Evaluate pattern matching methods for efficiency in real-time AI systems.

Unit 1: 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 2: 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 3: 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 4: 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 5: 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

1. Gusfield, D. Algorithms on Strings, Trees, and Sequences. Cambridge University Press, 1997.
2. Mount, D. Bioinformatics: Sequence and Genome Analysis. CSHL Press, 2004.

Reference Books

Aho, A.V., et al. Compilers: Principles, Techniques, and Tools. Addison-Wesley, 2006 (pattern matching chapters).

Sedgewick, R., Wayne, K. Algorithms. 4th ed., Addison-Wesley, 2011 (string matching sections).

Jain, A.K. Pattern Recognition and Machine Learning (select chapters on matching). Springer, 2006.

Tim Roughgarden, Beyond the Worst-Case Analysis of Algorithms, Cambridge University Press, 2021.

RECOMMENDATION SYSTEMS

Course Code:	**CA536**	**Course Credits:**	**3**
Course Category:	**E1**	**Course (U / P)**	**IP/P**
Course Year (U / P):	**4 IP / 1P**	**Course Semester (U / P):**	**8IP / 2P**
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. 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:

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.

Reference Books

1. Ekstrand, M. D., Riedl, J. T., Konstan, J. A., Collaborative Filtering Recommender Systems, Now Publishers, 2011.
2. Ricci, F., Rokach, L., Shapira, B., Recommender Systems Handbook, Springer, 2011.
3. 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:	E1	Course (U/ P)	IP /P
Course Year (U / P):	4 IP / 1P	Course Semester (U / P):	8IP / 2P
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. 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 1 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:

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.