University of Gour Banga

[Established under W.B. Act XXVI of 2007
Recognized by UGC U/S 2(f) & 12(B)
NAAC accredited (2016)]

Department of Computer Science

N.H.-34 (Near Rabindra Bhawan), P.O.: Mokdumpur
Dist.: Malda, West Bengal, Pin: 732103

M.Sc. in Computer Science

Two Years (Four Semesters) Syllabus

(In NEP 2020 Format w.e.f 2026)

Contents

1st Year							
Sem	**Paper Code**	**Paper Name**	**Type**	**Credit**	**Total Credits**	**Equivalent Marks /Paper**	**Total Marks**
I	COMP-DSC-101	Advanced Design and Analysis of Algorithm	TH	4	24	50	300
	COMP-DSC-102	Advanced Design and Analysis of Algorithm Lab	PR	4		50	
	COMP-DSC-103	Artificial Intelligence	TH	4		50	
	COMP-DSC-104	Artificial Intelligence Lab	PR	4		50	
	COMP-DSE-105	Elective Pool-I	TH	4		50	
	COMP- SEC-106	Professional Skill Development through Indian Knowledge Systems (IKS)	PR	4		50	
II	COMP-DSC-201	Deep Learning	TH	4	24	50	300
	COMP-DSC-202	Deep Learning Lab	PR	4		50	
	COMP-DSC-203	Secure Web Engineering Lab	PR	4		50	
	COMP-DSC-204	Advanced Compiler Design	TH	4		50	
	COMP-DSE-205	Elective Pool-II	TH	4		50	
	COMP-SEC-206	AI Tools and their Use Cases	TH	4		50	
2nd Year							
III	COMP-DSC-301	Data Analytics Lab	PR	4	24	50	300
	COMP-DSE-302	Elective Pool-III	TH	4		50	
	COMP-DSE-303	Elective Pool-IV	TH	4		50	
	COMP-DRP-304	Advance Research Methodology	TH	4		50	
	COMP-DRP-305	Dissertation/ Departmental Research Project / Internship	PR	8		100	
IV	COMP-DSC-401	Generative AI	TH	4	24	50	300
	COMP-DSE-402	Elective Pool-V	TH	4		50	
	COMP-DSE-403	Elective Pool-VI	TH	4		50	
	COMP-DRP-404	Dissertation/ Departmental Research Project / Internship	PR	12		150	
Total					**96**		**1200**

DSC: Discipline Specific Course

DSE: Discipline Specific Elective

SEC: Skill Based Course

TH: Theory

PR: Practical

List of Discipline Specific Elective (DSE) Papers:

Sem	Paper	Elective Pool	Name of the Paper
I	COMP-DSE-105	Elective Pool-I	A. Software Engineering B. Human Computer Interaction (HCI) C. Agile and DevOps Engineering D. Software Engineering for Big Data Systems E. SWAYAM MOOCs Course
II	COMP-DSE-205	Elective Pool-II	A. Interpretable and Responsible AI B. Computational Intelligence C. Computational Sustainability D. AI and Society E. SWAYAM MOOCs Course
III	COMP-DSE-302	Elective Pool-III	A. Pattern Recognition through Feature Engineering B. Cloud Computing C. Big Data Analytics D. Bioinformatics E. Quantum Computing F. SWAYAM MOOCs Course
	COMP-DSE-303	Elective Pool-IV	A. Data Mining & Data Warehousing B. Parallel Computing C. Cryptography D. Computational Geometry E. Block Chain Technology F. SWAYAM MOOCs Course
IV	COMP-DSE-402	Elective Pool-V	A. Image Processing B. Computer Vision C. Biometric System D. Steganography and Digital Watermarking E. Nano Computing F. SWAYAM MOOCs Course
	COMP-DSE-403	Elective Pool-VI	A. Natural Language Processing B. Reinforcement Learning C. Speech Processing D. Edge AI and Intelligent Systems E. VLSI Design F. SWAYAM MOOCs Course

Marks Distribution:

- **Each 4-credit course shall carry a total of 50 marks,** comprising:
  - **40 marks** for the **Semester-End Examination**; and
  - **10 marks** for **Continuous Evaluation**.

- **Question Pattern for Theory (TH) Paper for Semester End Written Examination will be as follows**:
  - Full Marks = **40** [**Group-A:** Compulsory (2 Marks x 5 Question) + **Group B:** (10 Marks x 3 Questions)]
  - **Note:** Question(s) containing 10 marks may be divided into smaller sub-parts. At-least two extra questions will be available for each group to answer.

- **Practical (Semester End Laboratory Based Test) - PR**: Full Marks = 40
- Full Marks of DRP-305: 100
- Full Marks of DRP-404: 150

Duration of Examination:

- Theory paper of 40 marks: 2 hours
- Practical paper of 40 marks: 3 hours

Semester-I

COMP-DSC-101: Advanced Design and Analysis of Algorithm

UNIT-I: Introduction: Fundamentals of Algorithms and Analysis of Algorithms - Orders of Magnitude (Asymptotic notations); Growth rates, some common bounds (constant, logarithmic, linear, polynomial, exponential); Average and worst-case analysis; Analysing control statements; Recurrence Relations-substitution, change of variables, master's method.

UNIT-II: Divide and conquer algorithms: Introduction - Quick sort, worst and average case complexity, Merge sort, Strassen's Matrix multiplication, Binary search, Finding the maximum and minimum, etc.

UNIT-III: Greedy algorithms: General Characteristics of greedy algorithms, Problem solving using Greedy Algorithm-Minimum Spanning trees (Kruskal's algorithm, Prim's algorithm), Shortest path (Dijkstra algorithm), Huffman coding.

UNIT-IV: Dynamic programming: Introduction, The Principle of Optimality, Problem Solving using Dynamic Programming- Fibonacci numbers, Warshall and Floyd algorithm; Matrix chain multiplication, Longest Common Subsequence (LCS), Optimal Binary Search Tree.

UNIT-V: Graph Algorithms: An introduction using graphs - Traversing Trees: Depth First Search, Breath First Search.

UNIT-VI: Backtracking and Branch and Bound: 0/1 Knapsack Problem, The Eight queens' problem, Travelling Salesman problem.

UNIT-VII: String matching: Introduction, The naive string-matching algorithm, The Rabin-Karp algorithm, KMP Algorithm.

UNIT-VIII: Introduction to Complexity Theory: The class P and NP, Polynomial reduction, NP-Complete Problems, NP-Hard Problems.

Text/ Reference Books:

1. Introduction to Algorithms: Cormen, Leiserson, Rivest and Stein: Prentice Hall of India
2. Fundamentals of Computer Algorithms: Sahni , Horowitz: Universites Press
3. Introduction to the Design and Analysis of Algorithms: AnanyLevitin
4. Data Structures and Algorithms: Aho, Hopcroft and Ullmann: Addison Wesley.
5. Data Structures and Algorithms in Java b: Michael T. Goodrich , Roberto Tamassia
6. Data Structures and Algorithms in C++ : Adam Drozdek
7. Teofilo F. Gonzalez, Handbook of NP-Completeness: Theory and Applications

COMP-DSC-102: Advanced Design and Analysis of Algorithm Lab

Programming must be done using Python preferably using Jupyter Notebook platform. Students will complete assignments involving the implementation and analysis of algorithms and conduct empirical studies corresponding to the topics covered in the theory component of the course.

The assignments listed below are illustrative examples and not an exhaustive list. They serve as a starting point to cover various aspects of the course.

1. Implementation and Time analysis of sorting algorithms. Bubble sort, Selection sort, Insertion sort, Merge sort and Quicksort.
2. Implementation and Time analysis of linear and binary search algorithm.
3. Write a program for Strassen's Matrix Multiplication.
4. To implement Huffman coding and analyse its time complexity.
5. Write a program for travelling salesman problem.

6. Implementation of chain matrix multiplication using dynamic programming.
7. Implementation of making a change problem using dynamic programming.
8. Implementation of a knapsack problem using greedy algorithm
9. Implementation of Graph and Searching (DFS and BFS).
10. Implement prim's algorithm.
11. Implement Kruskal's algorithm.
12. Implement LCS problem.
13. To implement following string-matching algorithms and analyse time complexities: a. Naïve, b. Rabin Karp, c. Knuth Morris Pratt
14. Write a program for Floyd-Warshall algorithm.

COMP-DSC-103: Artificial Intelligence

UNIT-I: Introduction: Introduction to Artificial Intelligence, various definitions of AI, AI Applications and Techniques, Turing Test and Reasoning - forward & backward chaining.

UNIT-II: Intelligent Agents: Definitions of a rational agent, reflex, model-based, goal-based, and utility-based agents, the environment in which a particular agent operates.

UNIT-III: Problem Solving and Search Techniques: State Space search, State Space Graphs, Uninformed search: breadth first search, depth first search, iterative deepening; Uniform cost algorithm, Informed search: use of heuristics, Greedy, A* algorithm, IDA* algorithm, AO* algorithm, Best first search, hill climbing, simulated annealing. Adversarial Search: Two agent games, AND/OR graphs, Minimax procedure, and game trees, Alpha – Beta pruning procedure, learning evaluation functions. Game Playing. Constrained Satisfaction Search.

UNIT-IV: Knowledge Representation: Introduction to First Order Predicate Logic, Resolution Principle, Unification, Semantic Nets, Conceptual Dependencies, Frames, and Scripts, Production Rules, Conceptual Graphs.

UNIT-V: Dealing with Uncertainty and Inconsistencies: Introduction to uncertainty in Artificial Intelligence; Truth Maintenance Systems; Default Reasoning and non-monotonic reasoning; Probabilistic Reasoning; Bayes theorem and Bayesian probabilistic inference; basic concepts of Bayesian Networks; Possible World Representations and reasoning under incomplete information.

Text/ Reference Books:

1. Elaine Rich and Kelvin Knight: Artificial Intelligence, Tata McGraw Hill, 2002
2. Stuart Russell and Peter Norvig: Artificial Intelligence: A Modern Approach (2nd ed.), Pearson Education,2006.
3. Dan W. Patterson, Introduction to Artificial Intelligence and Expert Systems: Prentice Hall of India, 2006.
4. Nils J Nilson, Artificial Intelligence: A New Synthesis, Morgan Kaufmann Publishers, Inc., San Francisco, California, 2000.
5. R. Akerkar: Introduction to Artificial Intelligence, Prentice-Hall of India, 2005
6. Nils J. Nilson: Principles of Artificial Intelligence, Narosa Publishing House, 2001

COMP-DSC-104: Artificial Intelligence Lab

The laboratory course is designed to provide practical exposure to the fundamental concepts and techniques of Artificial Intelligence through implementation, experimentation and analysis using Python and basic AI programming tools.

Suggested List of Assignments:

1. Formulate the 8-puzzle problem as a state space and represent it programmatically.
2. Model the Water Jug problem using state space representation and define valid operators.

3. Implement Breadth First Search (BFS) to solve a simple graph traversal problem.
4. Implement Depth First Search (DFS) and compare its behaviour with BFS.
5. Implement Iterative Deepening Search and analyze its advantages over DFS.
6. Implement Uniform Cost Search (UCS) for a weighted graph.
7. Compare BFS, DFS, and UCS in terms of nodes expanded, completeness, optimality and complexity..
8. Implement Greedy Best First Search using a heuristic function.
9. Implement the A* algorithm for pathfinding (e.g., grid or maze problem).
10. Design a suitable heuristic function and test its impact on A* performance.
11. Implement Hill Climbing algorithm and demonstrate the problem of local maxima.
12. Modify Hill Climbing to include random restart and compare results.
13. Implement Simulated Annealing for optimization problems.
14. Implement Minimax algorithm for a two-player game (e.g., Tic-Tac-Toe).
15. Enhance the above using Alpha - Beta pruning and compare efficiency.
16. Design a simple evaluation function for a game-playing agent and analyze its effectiveness.
17. Solve the N-Queens problem using backtracking.
18. Solve a Map Colouring problem using constraint satisfaction techniques.
19. Implement a rule-based expert system using production rules.
20. Represent family relationships using First Order Logic.
21. Implement unification for logical expressions.
22. Demonstrate resolution principle for theorem proving.
23. Create a semantic network representation for a real-world scenario.
24. Design a frame-based knowledge representation model.
25. Implement Bayesian probabilistic inference for simple decision making.
26. Develop a small application demonstrating probabilistic reasoning under uncertainty.
27. Demonstrate default reasoning using a rule-based example.
28. Develop a mini-AI application such as a chatbot, puzzle solver, pathfinding system, recommendation system or intelligent game agent integrating multiple AI techniques.

COMP-DSE-105 (A): Software Engineering

UNIT-I: Introduction: Software Engineering – A Layered Approach; Software Process – Process Framework, Umbrella Activities; Process Models – Waterfall Model, Incremental Model, and Evolutionary process Model (Prototyping, Spiral Model); Introduction to Agile – Agility Principles, Agile Model – Scrum.

UNIT-II: SRS: Software Requirements Analysis and Specifications: Use Case Approach, Software Requirement Specification Document, Flow oriented Modelling, Data Flow Modelling, Sequence Diagrams.

UNIT-III: Design Modelling: Translating the Requirements model into the Design Model, The Design Process, Design Concepts – Abstraction, Modularity and Functional Independence; Architectural Mapping using Data Flow.

UNIT-IV: Software Metrics and Project Estimations: Function based Metrics, Software Measurement, Metrics for Software Quality; Software Project Estimation (FP based estimations, COCOMO II Model); Project Scheduling (Timeline charts, tracking the schedule).

UNIT-V: Quality Control and Risk Management: Quality Control and Quality Assurance, Software Process Assessment and Improvement Capability Maturity Model Integration (CMMI); Software Risks, Risk Identification, Risk Projection and Risk Refinement, Risk Mitigation, Monitoring and Management.

UNIT-VI: Software Testing: Strategic Approach to Software Testing, Unit Testing, Integration Testing, Validation Testing, System Testing; Black-Box and White Box Testing, Basis Path Testing, etc.

Text/ Reference Books:

1. Roger S.Pressman, Software engineering- A practitioner's Approach, McGraw-Hill International Editions
2. Ian Sommerville, Software engineering, Pearson education Asia
3. Pankaj Jalote, Software Engineering – A Precise Approach Wiley
4. Software Engineering Fundamentals by Ali Behhforoz & Frederick Hudson OXFORD
5. Rajib Mall, Fundamentals of software Engineering, Prentice Hall of India.
6. Engineering Software as a Service An Agile Software Approach, Armando Fox and David Patterson
7. John M Nicolas, Project Management for Business, Engineering and Technology, Elsev

COMP-DSE-105 (B): Human Computer Interaction (HCI)

UNIT-I: Introduction: Human–Computer Interaction: Definition and scope; Goals of HCI; Interdisciplinary nature of HCI; Human factors in computing; Usability and User Experience (UX); Principles of usability – learnability, efficiency, memorability, errors, satisfaction; Interaction paradigms.

UNIT-II: Human Aspects in Interaction: Human cognition, perception, memory, attention, problem solving; Mental models; Human information processing; Ergonomics and physical interaction; Human errors and user behavior; Cognitive load.

UNIT-III: Interaction Design and Process: User-Centered Design (UCD); Interaction design lifecycle; Requirement gathering techniques (interviews, surveys, observation); Personas and scenarios; Task analysis; Storyboarding; Iterative design process.

UNIT-IV: Design Principles and Models: Design principles, consistency, feedback, constraints, affordances; Shneiderman's Eight Golden Rules; Norman's design principles; Conceptual models; Interface metaphors; Navigation design; Information architecture.

UNIT-V: User Interface Design: Graphical User Interface (GUI) design; Web and mobile interface design principles; Menu design, form design, dialog design; Color, typography, and layout; Responsive design; Accessibility and universal design.

UNIT-VI: Prototyping Techniques: Low-fidelity and high-fidelity prototyping; Paper prototyping; Wireframing tools; Interactive prototypes; Rapid prototyping methods; Evaluation of prototypes.

UNIT-VII: Evaluation Techniques: Usability testing; Heuristic evaluation; Cognitive walkthrough; A/B testing; Surveys and questionnaires; Experimental evaluation; Metrics for usability (task time, error rate, satisfaction).

UNIT-VIII: Advanced Topics in HCI: Natural User Interfaces (NUI); Voice and gesture-based interaction; Virtual Reality (VR) and Augmented Reality (AR); HCI in AI systems; Affective computing; Social and ethical issues in HCI.

Text/ Reference Books:

1. Human-Computer Interaction by Alan Dix, Janet Finlay, Gregory D. Abowd, and Russell Beale
2. Designing the User Interface: Strategies for Effective Human-Computer Interaction by Ben Shneiderman et al.
3. Interaction Design: Beyond Human-Computer Interaction by Yvonne Rogers, Helen Sharp, and Jenny Preece
4. The Design of Everyday Things by Don Norman
5. About Face: The Essentials of Interaction Design by Alan Cooper, Robert Reimann, and David Cronin
6. Research Methods in Human-Computer Interaction by Jonathan Lazar, Jinjuan Heidi Feng, and Harry Hochheiser
7. Measuring the User Experience: Collecting, Analyzing, and Presenting Usability Metrics by Thomas Tullis and William Albert

COMP-DSE-105 (C): Agile and DevOps Engineering

UNIT-I: Introduction: Software Engineering evolution; Limitations of traditional models; Introduction to Agile: Agile Manifesto and principles; Overview of Agile methodologies; Introduction to DevOps: concept, goals, and benefits; Relationship between Agile and DevOps; DevOps lifecycle.

UNIT-II: Agile Methodologies: Scrum framework: roles (Product Owner, Scrum Master, Development Team), artifacts (Product Backlog, Sprint Backlog, Increment), events (Sprint, Daily Scrum, Sprint Review, Sprint Retrospective); Extreme Programming (XP): practices (pair programming, test-driven development, continuous integration); Kanban: principles and workflow visualization; Lean software development.

UNIT-III: Agile Planning and Management: User stories and acceptance criteria; Story estimation techniques (Planning Poker, story points); Release planning and sprint planning; Agile project tracking: burndown and burnup charts; Agile metrics (velocity, lead time, cycle time); Risk management in Agile.

UNIT-IV: DevOps Fundamentals: DevOps culture and practices; Continuous Integration (CI); Continuous Delivery and Continuous Deployment (CD); Infrastructure as Code (IaC); Configuration management; Version control systems; Build automation.

UNIT-V: DevOps Tools and Technologies: Overview of tools such as Git, Jenkins, Docker, Kubernetes; CI/CD pipeline design; Containerization and virtualization; Cloud platforms and deployment models.

UNIT-VI: Continuous Testing and Monitoring: Automated testing in DevOps; Test-driven development (TDD) and behavior-driven development (BDD); Performance and security testing; Monitoring and logging; Feedback loops; Incident management.

UNIT-VII: Microservices and Cloud-Native Development: Monolithic vs microservices architecture; Service decomposition; API management; Scalability and resilience; Cloud-native applications; Serverless computing basics.

UNIT-VIII: Security and DevSecOps: Introduction to DevSecOps; Secure coding practices; Security testing in CI/CD pipelines; Vulnerability assessment; Compliance and governance.

Text / Reference Books:

1. Agile Software Development with Scrum – Ken Schwaber, Mike Beedle
2. The DevOps Handbook – Gene Kim, Jez Humble, Patrick Debois, John Willis
3. Continuous Delivery – Jez Humble, David Farley
4. Accelerate – Nicole Forsgren, Jez Humble, Gene Kim
5. Lean Software Development – Mary Poppendieck, Tom Poppendieck
6. Site Reliability Engineering – Betsy Beyer, Chris Jones, Jennifer Petoff, Niall Richard Murphy

COMP-DSE-105 (D): Software Engineering for Big Data Systems

UNIT-I: Introduction: Overview of Big Data; Characteristics of Big Data (Volume, Velocity, Variety, Veracity, Value); Limitations of traditional software systems; Need for specialized software engineering practices for data-intensive systems; Big Data lifecycle.

UNIT-II: Big Data Architecture and Design: Architectural styles for Big Data systems; Distributed system fundamentals; Data storage architectures; Lambda and Kappa architectures; Data lakes and data warehouses; Scalability and elasticity; Fault tolerance and high availability.

UNIT-III: Big Data Processing Models: Batch processing vs stream processing; Parallel and distributed computing models; MapReduce paradigm; In-memory processing; Event-driven architectures.

UNIT-IV: Big Data Technologies and Frameworks: Overview and working of Hadoop ecosystem; Apache Spark for fast data processing; NoSQL databases (key-value, document, column-family, graph databases); Data ingestion tools; Workflow orchestration.

UNIT-V: Software Engineering Principles for Big Data: Requirement engineering for data-intensive systems; Design patterns for Big Data applications; Microservices architecture; API design; Data modeling for large-scale systems; Testing challenges in distributed environments.

UNIT-VI: Data Management and Storage: Distributed file systems (e.g., HDFS); Data partitioning and replication; Consistency models (CAP theorem basics); Indexing and querying large datasets; Data integration and cleaning.

UNIT-VII: Performance Optimization and Scalability: Load balancing; Resource scheduling; Performance tuning; Caching strategies; Monitoring and profiling; Handling large-scale concurrent workloads.

UNIT-VIII: Reliability, Fault Tolerance, and Security: Fault detection and recovery; Checkpointing; Replication strategies; Security challenges in Big Data (authentication, authorization, encryption); Data privacy and compliance.

UNIT-IX: Big Data Analytics and Machine Learning Integration: Data analytics pipelines; Integration with machine learning workflows; Real-time analytics; Visualization of large datasets; Use cases in recommendation systems, fraud detection.

UNIT-X: DevOps and Deployment for Big Data Systems: Continuous Integration and Deployment (CI/CD) for data systems; Containerization (e.g., Docker); Cluster management; Cloud platforms and Big Data services; Monitoring and logging.

Text / Reference Books:

1. Big Data Fundamentals – Thomas Erl, Wajid Khattak, Paul Buhler
2. Hadoop The Definitive Guide – Tom White
3. Spark The Definitive Guide – Bill Chambers, Matei Zaharia
4. Designing Data Intensive Applications – Martin Kleppmann
5. NoSQL Distilled – Pramod J. Sadalage, Martin Fowler

COMP-DSE-105 (E): SWAYAM MOOCs Course

A pool of relevant **SWAYAM MOOCs Courses** consistent with the curriculum and learning outcomes of the programme will be notified at the commencement of the semester. Each aspirant student shall be permitted to opt for **only one SWAYAM MOOCs Course** from the approved pool.

COMP-SEC-106: Professional Skill Development through Indian Knowledge Systems (IKS)

UNIT I: Introduction to Indian Knowledge Systems and Communication

Indian Knowledge Systems

- Bharatavarsha as a land of unique geographical, cultural, and intellectual heritage.
- Overview of Indian civilization and its contributions to world knowledge.
- Introduction to the Vedas, Itihasas, Puranas, and major philosophical traditions.
- Overview of streams of Indian Knowledge Systems: Ayurveda, Arthashastra, Sthapatya, Natyashastra, Jyotisha, and Dharmashastra.

Communication Skills

- Communication process and barriers.
- Verbal and non-verbal communication.
- Interpersonal communication.

- Professional etiquette.
- Listening skills.
- Confidence building and personality development.

UNIT II: Indian Language Sciences and Spoken Communication

Indian Language Sciences

- Language sciences in India and preservation of knowledge traditions.
- Structure of Varnamala and scientific classification of sounds.
- Introduction to Siksha, Vyakarana, Nirukta, and Chandas.
- Panini and the science of grammar.
- Continuity of Indian linguistic traditions.

Spoken Communication

- Public speaking techniques.
- Prepared speeches on IKS themes.
- Extempore speeches on Indian scientific and cultural contributions.
- Group discussions and debates.
- Audience interaction and communication effectiveness.

UNIT III: Reading, Comprehension and Critical Interpretation of IKS Texts

Reading Skills

- Reading strategies: skimming, scanning, intensive and extensive reading.
- Reading and comprehension of adapted excerpts from:
    - Vedas
    - Ramayana and Mahabharata
    - Puranas
    - Mathematical and linguistic texts

Critical Interpretation

- Interpretation of reports, charts, timelines, and knowledge maps.
- Analytical reading and summarization.
- Comparative analysis of traditional and modern knowledge systems.

UNIT IV: Indian Mathematics and Technical Communication

Indian Mathematics

- Numbers and geometry in Vedic literature.
- Decimal place value system.
- Zero and infinity.
- Sulba-sutra constructions.
- Kerala School of Mathematics.
- Contributions of Srinivasa Ramanujan.
- Overview of algebra, trigonometry, and calculus traditions in India.

Technical and Business Writing

- Principles of effective writing.
- Precis writing.
- Technical report writing.
- Proposal writing.
- Business correspondence.
- Formal letters and professional emails.
- Documentation of mathematical and scientific ideas.

UNIT V: Professional Documentation through IKS Themes

Documentation Skills

- Resume and Curriculum Vitae preparation.
- Statement of Purpose (SOP) writing.
- Cover letter writing.
- Preparation of project reports.
- Technical specifications and user documentation.

- Documentation standards and formatting.

IKS-Based Documentation Activities

- Report on Indian mathematical heritage.
- Documentation of Paninian grammar as a knowledge system.
- Case study reports on Indian scientific traditions.
- Project documentation on selected IKS themes.

UNIT VI: Presentation, Visual and Digital Communication

Presentation Skills

- Presentation planning and organization.
- Multimedia presentation techniques.
- Professional presentation delivery.
- Handling audience interaction and questions.

Visual Communication

- Poster presentation.
- Infographics based on Indian Knowledge Systems.
- Timelines of Indian scientific developments.
- Knowledge visualization techniques.

Digital Communication

- Use of presentation software and collaborative tools.
- Online presentation etiquette.
- Digital communication ethics.
- Professional networking and online communication.

Text / Reference Books:

1. M. Ashraf Rizvi, Effective Technical Communication, McGraw-Hill Education.
2. Meenakshi Raman and Sangeeta Sharma, Technical Communication: Principles and Practice, 3rd Edition, Oxford University Press.
3. Business Correspondence & Report Writing, Sharma, TMH.
4. English for Technical communication, Laxminarayanan, Scitech.
5. Kapil Kapoor and Avadesh Kumar Singh (Eds.), Indian Knowledge Systems: Vol. I & II, Indian Institute of Advanced Study (IIAS), Shimla.
6. Mahadevan, Vinayak Rajat Bhat, and Nagendra Pavana R. N., Introduction to Indian Knowledge System: Concepts and Applications, PHI Learning, 2022.
7. B. Mahadevan, Indian Knowledge Systems, PHI Learning.

Semester-II

COMP-DSC-201: Deep Learning

UNIT-I: Introduction to Deep Learning: Overview of artificial neural networks, perceptron and multi-layer perceptron (MLP), activation functions, representation learning, and the evolution and applications of deep learning.

UNIT-II: Backpropagation and Optimization: Backpropagation algorithm; gradient descent and its variants; loss functions; learning rate scheduling; optimization algorithms including SGD, Adam, and RMSProp; challenges in optimization.

UNIT-III: Datasets, Learning and Regularization: Training, validation, and test datasets; overfitting and underfitting; regularization methods including dropout, batch normalization, data augmentation, and early stopping.

UNIT-IV: Training Deep Neural Networks: Weight initialization strategies; vanishing and exploding gradient problems; hyperparameter tuning; optimization challenges in deep networks; strategies for efficient training of deep models.

UNIT-V: Convolutional Neural Networks (CNNs): Convolution operations; kernels and feature maps; padding and stride; pooling layers; feature extraction; transfer learning; state-of-the-art CNN architectures including LeNet, AlexNet, VGGNet, and ResNet.

UNIT-VI: Recurrent Neural Networks (RNNs) and Sequence Modelling: Sequence learning; recurrent neural networks (RNNs); long short-term memory (LSTM); gated recurrent units (GRU); sequence-to-sequence learning; applications in text processing, speech recognition, and time-series analysis.

UNIT-VII: Transformer Architecture and Attention Mechanisms: Attention mechanism; self-attention; encoder-decoder architecture; positional encoding; Transformer models; applications of Transformers in sequence learning and natural language processing.

UNIT-VIII: Generative Deep Learning: Introduction to generative models; autoencoders and variational autoencoders (VAEs); generative adversarial networks (GANs); diffusion models; applications in image synthesis, data generation, and creative AI.

UNIT-IX: Large Language Models (LLMs): Foundations of large language models; pre-training and fine-tuning; transfer learning in NLP; prompt engineering; instruction tuning; applications in natural language understanding and generation.

UNIT-X: Vision-Language Models and Multimodal AI: Introduction to multimodal learning; image-text alignment models; large vision-language models (LVLMs); multimodal reasoning; applications in visual question answering, caption generation, and AI assistants.

Text/ Reference Books:

1. Deep Learning – Ian Goodfellow, Yoshua Bengio, and Aaron Courville, MIT Press.
2. Pattern Recognition and Machine Learning – Christopher M. Bishop, Springer.
3. Neural Networks and Deep Learning – Michael Nielsen, Determination Press.
4. Deep Learning with Python – François Chollet, Manning Publications.
5. Speech and Language Processing – Daniel Jurafsky and James H. Martin, Pearson.
6. Dive into Deep Learning – Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola, Cambridge University Press.
7. Generative Deep Learning – David Foster, O'Reilly Media.
8. Natural Language Processing with Transformers – Lewis Tunstall, Leandro von Werra, and Thomas Wolf, O'Reilly Media.

COMP-DSC-202: Deep Learning Lab

Implement using Python and suitable deep learning frameworks. Students will complete assignments involving the implementation, training, and evaluation of neural networks, convolutional neural networks, recurrent neural networks, transformer models, and other deep learning architectures related to the topics covered in the theory component of the course. Empirical studies on model performance using standard datasets will also be conducted. The practical work will also explore applications of deep learning in areas such as computer vision, speech recognition, natural language processing, healthcare, robotics, and multimedia analysis.

The questions listed below are illustrative examples and not an exhaustive list. They serve as a starting point to cover various aspects of the course.

1. Write a Python program to implement a Multi-Layer Perceptron (MLP) for classification using a standard dataset such as Iris or MNIST.

Tasks

- Design an MLP with at least one hidden layer.
- Train the model using appropriate optimization techniques.
- Evaluate model performance.

Display

- Training and testing accuracy
- Loss values during training

2. Write a Python program to implement and visualize the following activation functions:
   - Sigmoid
   - ReLU
   - Tanh

   Tasks
   - Plot the activation functions using graphs.
   - Compare their mathematical properties and practical applications.

   Display
   - Graphs of activation functions
   - Comparative analysis

3. Implement a simple neural network using backpropagation to classify a dataset such as the Iris dataset.

   Tasks
   - Initialize network parameters.
   - Implement forward and backward propagation.
   - Train the network using gradient descent.

   Display
   - Training loss
   - Classification accuracy

4. Design and train a Convolutional Neural Network (CNN) for image classification using the MNIST dataset.

   Tasks
   - Define CNN architecture with convolution, pooling, and fully connected layers.
   - Train and evaluate the CNN model.

   Display
   - Accuracy
   - Loss curve
   - Confusion matrix and other performance metrics

5. Train a neural network model and demonstrate overfitting. Apply one of the following regularization techniques:
   - Dropout
   - Batch Normalization
   - Early Stopping

   Tasks
   - Compare model performance before and after regularization.

   Display
   - Training and validation accuracy
   - Loss comparison graphs

6. Implement an RNN or LSTM model to predict the next value in a simple time-series dataset such as stock prices or sine wave data.

   Tasks
   - Preprocess sequential data.
   - Design and train the recurrent model.
   - Predict future values.

   Display

- Actual vs predicted values
- Prediction error

2. Write a Python program to demonstrate a simple attention mechanism using matrix operations.
   Tasks
   - Compute attention scores and weights.
   - Apply attention to input sequences.

   Display
   - Attention weight matrix
   - Visualization of attention weights

3. Implement a simple Generative Adversarial Network (GAN) or Variational Autoencoder (VAE) to generate images using the MNIST dataset.
   Tasks
   - Train the generative model.
   - Generate synthetic images.

   Display
   - Generated image samples
   - Generator and discriminator loss (for GAN)

4. Use a text dataset such as movie reviews and build a text classification model using RNN or LSTM.
   Tasks
   - Preprocess text data.
   - Convert text into numerical representation.
   - Train and evaluate the model.

   Display
   - Classification accuracy
   - Sample predictions

5. Use a pre-trained Transformer model such as BERT with the HuggingFace library to perform one of the following tasks:
   - Sentiment analysis
   - Text summarization
   - Question answering

   Tasks
   - Load a pre-trained Transformer model.
   - Perform inference on sample inputs.

   Display
   - Model outputs for sample inputs
   - Performance observations

Text/ Reference Books:

1. Deep Learning – Ian Goodfellow, Yoshua Bengio, and Aaron Courville, MIT Press.
2. Pattern Recognition and Machine Learning – Christopher M. Bishop, Springer.
3. Neural Networks and Deep Learning – Michael Nielsen, Determination Press.
4. Deep Learning with Python – François Chollet, Manning Publications.
5. Speech and Language Processing – Daniel Jurafsky and James H. Martin, Pearson.
6. Dive into Deep Learning – Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola, Cambridge University Press.
7. Generative Deep Learning – David Foster, O'Reilly Media.
8. Natural Language Processing with Transformers – Lewis Tunstall, Leandro von Werra, and Thomas Wolf, O'Reilly Media.

COMP-DSC-203: Secure Web Engineering Lab

UNIT-I: Foundations of Web Technologies and Security: Evolution of web technologies; Client-server architecture; Internet and WWW; Web protocols – HTTP/HTTPS, DNS, FTP; Web clients and web servers; Web application architecture; Static vs dynamic web applications; Overview of cybersecurity principles: confidentiality, integrity, availability, authentication, non-repudiation; Security challenges in web systems.

UNIT-II: Advanced HTML and CSS: HTML5 semantic elements; Advanced forms and validation; Multimedia integration; Responsive web design; CSS3 advanced styling techniques; Flexbox and Grid layout; CSS animations and transitions; Media queries; UI/UX principles for modern web applications; Accessibility and cross-browser compatibility.

UNIT-III: Client-Side Programming and Dynamic Interfaces: Advanced JavaScript concepts: functions, closures, callbacks, promises, async programming; DOM manipulation; Event handling; AJAX and Fetch API; JSON and XML processing; Dynamic HTML (DHTML); Client-side validation; Introduction to front-end frameworks and component-based design.

UNIT-IV: Server-Side Web Programming: Server-side scripting concepts; Advanced PHP programming; Session and cookie management; File handling; Exception handling; Database connectivity and CRUD operations; PHP with MySQL/PostgreSQL; RESTful web services; MVC architecture basics; API development concepts.

UNIT-V: Database and Web Application Integration: Database-driven web applications; SQL and prepared statements; Transaction management; Data validation and sanitization; ORM concepts; Handling large-scale web data; Web application deployment basics.

UNIT-VI: Fundamentals of Information Security: Overview of information security; Threats and vulnerabilities in web applications; Malware: worms, viruses, Trojan horse, trapdoor, ransomware; Buffer overflow and injection attacks; Intrusion concepts; Risk assessment and security policies.

UNIT-VII: Secure Web Application Development: Common web vulnerabilities: SQL Injection, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), session hijacking, clickjacking; Secure coding practices; Authentication and authorization mechanisms; Password security and hashing; Secure API development; Input validation and sanitization.

UNIT-VIII: Security Mechanisms and Monitoring: Intrusion Detection Systems (IDS); Logging and auditing; Firewall concepts; Web application firewalls; Security testing and vulnerability assessment; Penetration testing basics; System-call monitoring; Security compliance and ethics.

UNIT-IX: Modern Trends in Web and Security: Cloud-based web applications; Secure deployment practices; DevSecOps concepts; Container security; Web security in mobile and IoT environments; Emerging trends in AI-driven cybersecurity.

Laboratory / Practical Component:

- Design responsive web pages using HTML5 and CSS3
- Implement dynamic client-side applications using JavaScript
- Develop database-driven web applications using PHP and MySQL
- Implement authentication and session management
- Perform input validation and secure coding practices
- Conduct vulnerability analysis on sample web applications
- Development of a secure web application

Text / Reference Books:

1. Web Technologies – Uttam K. Roy
2. Internet and World Wide Web How to Program – Harvey Deitel, Paul Deitel
3. PHP and MySQL Web Development – Luke Welling, Laura Thomson

4. Cryptography and Network Security – William Stallings
5. Web Application Security – Andrew Hoffman
6. Computer Security Principles and Practice – William Stallings, Lawrie Brown
7. The Web Application Hackers Handbook – Dafydd Stuttard, Marcus Pinto

COMP-DSC-204: Advanced Compiler Design

UNIT-I: Introduction to Compiling: Compilers, Analysis of the source program, The phases of the compiler, Cousins of the compiler.

UNIT-II: Lexical Analysis: The role of the lexical analyzer, Tokens, Patterns, Lexemes, Input buffering, Specifications of a token, Recognition of a tokens, Finite automata, From a regular expression to DFA, Design of a lexical analyzer generator (Lex).

UNIT-III: Syntax Analysis: The role of a parser, Context free grammars, Writing a grammar, Top-down Parsing, Non-recursive Predictive parsing (LL), Bottom-up parsing, Handles, Viable prefixes, Operator precedence parsing, LR parsers (SLR, LALR)

UNIT-IV: Syntax directed translation: Syntax director definitions, Construction of syntax trees, Bottom-up evaluation of S attributed definitions, L attributed definitions, Bottom-up evaluation of inherited attributes.

UNIT-V: Type checking: Type systems, Specification of a simple type checker, Equivalence of type expressions, Type
conversions

UNIT-VI: Code optimization: Introduction, Basic blocks & flow graphs, Transformation of basic blocks, DAG representation of basic blocks, The principal sources of optimization, Loops in flow graph, Peephole optimization.

UNIT-VII: Code generations: Issues in the design of code generator, a simple code generator, Register allocation & assignment.

Text / Reference Books:

1. Compilers Principles Techniques and Tools – Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman
2. Engineering a Compiler – Keith D. Cooper, Linda Torczon
3. Modern Compiler Implementation in C – Andrew W. Appel
4. Advanced Compiler Design and Implementation – Steven S. Muchnick
5. Compiler Construction Principles and Practice – Kenneth C. Louden
6. LLVM Essentials – Suyog Sarda, Mayur Pandey

COMP-DSE-205 (A): Interpretable and Responsible AI

Course Overview:

As AI systems are deployed in high-stakes domains such as healthcare, law, and finance, the ability to explain, audit, and ethically govern them becomes critical. This course equips students with the theoretical foundations and practical tools to build AI systems that are transparent, fair, accountable, and technically verifiable.

UNIT-I: Foundations:

- Interpretability vs. explainability vs. transparency: formal definitions and scope
- Taxonomy of explanation types: by audience (developer, regulator, end user) and by scope (local vs. global)
- The AI accountability stack: Model level, system level, organisational level, regulatory level
- Regulatory context: EU AI Act (risk tiers), GDPR Article 22, ISO/IEC 42001, NIST AI RMF

UNIT-II: Interpretability Methods:

- Intrinsically interpretable models: Decision trees, rule lists, logistic regression, scoring systems
- SHAP (SHapley Additive exPlanations): Game-theoretic foundations, TreeSHAP, KernelSHAP, complexity
- LIME: local surrogate models, sampling strategy, fidelity–interpretability trade-off
- Gradient-based methods: Saliency maps, integrated gradients, GradCAM, SmoothGrad
- Example-based explanations: Influence functions, counterfactual generation (DiCE), prototypes
- Model-agnostic vs. model-specific approaches: when each is appropriate

UNIT-III: Fairness and Bias:

- Sources of bias: Data collection, label noise, aggregation, and proxy variables
- Fairness metrics: Demographic parity, equalised odds, predictive parity, individual fairness
- Impossibility theorems: Chouldechova's theorem and inherent metric conflicts
- Bias detection tools: Fairlearn, AI Fairness 360 - API usage and metric computation
- Mitigation techniques: Reweighting and resampling (pre-processing); constrained optimisation, adversarial debiasing (in-processing); threshold calibration (post-processing)
- Case study: COMPAS recidivism score - technical audit methodology

UNIT-IV: Robustness and Safety:

- Adversarial examples: types of attacks, threat models, and physical-world implications
- Robustness certification: verification approaches and their trade-offs
- Distribution shift and out-of-distribution detection: causes, detection strategies, and mitigation
- Uncertainty quantification: probabilistic and ensemble-based approaches
- AI safety: reward misspecification, goal mis-generalization, and alignment challenges

UNIT-V: Privacy-Preserving and Governed AI:

- Privacy-preserving techniques: anonymization, differential privacy, and federated learning
- Attacks on ML models: privacy threats and defensive strategies
- Accountability artefacts: model cards, datasheets, audit trails, and impact assessments
- Human oversight patterns: design choices for keeping humans in the decision loop

Text / Reference Books:

1. Christoph Molnar: Interpretable Machine Learning, 2nd ed., 2022 (open access)
2. Solon Barocas, Moritz Hardt, Arvind Narayanan: Fairness and Machine Learning, MIT Press, 2023
3. Ian Goodfellow et al.: Explaining and Harnessing Adversarial Examples, ICLR, 2015
4. Cynthia Dwork, Aaron Roth: The Algorithmic Foundations of Differential Privacy, Foundations and Trends, 2014
5. European Commission Ethics Guidelines for Trustworthy AI, High-Level Expert Group on AI, 2019

COMP-DSE-205 (B): Computational Intelligence

Course Overview

Computational Intelligence (CI) is a branch of artificial intelligence that draws inspiration from biological and natural systems to solve complex, imprecise, or uncertain problems. This course covers five core paradigms: foundational concepts, artificial neural networks, evolutionary computation, fuzzy logic systems, and rough set theory.

UNIT-I: Introduction:

- What is Computational Intelligence? - Biological inspiration, soft computing paradigm.
- CI vs. Artificial Intelligence: Tolerance for imprecision, uncertainty, and approximation.

- Overview of constituent paradigms: neural networks, evolutionary computation, fuzzy systems, rough set.
- Application domains and historical context.

UNIT-II: Artificial Neural Networks:

- Biological neuron model: The McCulloch–Pitts unit, activation functions.
- Perceptron: Single-layer learning, convergence theorem, limitations.
- Multilayer Neural Networks: Architecture, forward pass, universal approximation.
- Supervised learning networks: Backpropagation, gradient descent, weight update rules.
- Unsupervised learning networks: Self-organizing maps (SOM), Hebbian and competitive learning.
- Radial Basis Function (RBF) networks: Architecture, training, comparison with MLP.
- Reinforcement learning: Reward-based paradigm, Q-learning, temporal difference methods.
- Deep Learning: Training, optimization, evaluations.

UNIT-III: Evolutionary Computation:

- Introduction to evolutionary computation: Natural selection analogy, fitness landscape
- Genetic operators: Selection, crossover (single-point, multi-point), mutation, elitism
- Building block hypothesis: Schema theorem, implicit parallelism
- Evolution of structure: Variable-length representations, genetic programming trees
- Tree-based GAs: Parse-tree encoding, subtree crossover and mutation
- Linear graph-based GAs: Path encoding, edge assembly crossover
- Applications: Function optimisation, scheduling, neural architecture search, engineering design

UNIT-IV: Fuzzy Logic Systems:

- Notion of fuzziness: Membership functions, linguistic variables
- Fuzzy modeling: Representation of vague knowledge
- Operations on fuzzy sets: Union, intersection, complement
- T-norms and aggregation operators: Triangular norms, S-norms, OWA
- Approximate reasoning: Compositional rule of inference (CRI)
- Fuzzy rule-based systems: Mamdani–Assilian model, Takagi–Sugeno (T-S) model
- Fuzzy system pipeline: Fuzzification schemes, inferencing engines, defuzzification (centroid, MOM)
- Fuzzy clustering: Fuzzy C-Means (FCM), validity indices
- Fuzzy rule-based classifier: Classification rules, confidence and certainty factors

UNIT-V: Rough Sets:

- Introduction: motivation, Pawlak's rough set model
- Indiscernibility relation: Equivalence classes and knowledge granularity
- Set approximation: Lower and upper approximations, boundary regions
- Decision tables: Attributes, reductions, rule induction from data
- Applications in data mining and feature selection

Text/Reference Books:

1. Leszek Rutkowski — Computational Intelligence: Methods and Techniques, Springer, 2008
2. Andries P. Engelbrecht — Computational Intelligence: An Introduction, Wiley, 2007
3. K. H. Lee — First Course on Fuzzy Theory and Applications, Springer, 2005
4. E. Goldberg — Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, 1989
5. Alpaydin — Introduction to Machine Learning, Prentice-Hall of India, 2004
6. Amit Konar — Computational Intelligence: Principles, Techniques and Applications, Springer, 2005

COMP-DSE-205 (C): Computational Sustainability

Course Overview

Computational Sustainability applies optimisation, machine learning, simulation, and data science to challenges in environmental science, ecology, energy systems, and sustainable development. This course

develops both the computational toolkit and the domain knowledge needed to model, optimise, and evaluate real-world sustainability problems algorithmically.

UNIT-I: Introduction:

- Scope of computational sustainability: problem classes, data types, and computational challenges
- Mapping sustainability challenges to CS paradigms: search, optimisation, prediction, simulation
- UN Sustainable Development Goals (SDGs): quantitative targets and data-driven monitoring
- Carbon footprint of computing: energy-efficient AI, green software engineering principles

UNIT-II: Optimisation for Sustainability

- Integer linear programming (ILP): formulation, branch-and-bound, solver interfaces (OR-Tools, Gurobi)
- Multi-objective optimisation: Pareto dominance, NSGA-II algorithm, hypervolume indicator
- Stochastic optimisation: scenario-based programming, chance constraints for uncertain natural systems
- Markov Decision Processes (MDPs): Bellman equations, value iteration, policy gradient for resource management
- Case study: conservation planning with Marxan — reserve selection as a set-cover ILP

UNIT-III: Machine Learning for Environmental Science

- Remote sensing: CNN-based land cover classification, change detection, SAR image processing
- Species distribution modelling: MaxEnt as regularised logistic regression; ensemble SDMs
- Spatiotemporal prediction: graph neural networks for sensor networks, kriging for spatial interpolation
- Precision agriculture: crop yield regression, soil health time-series; data fusion from IoT sensors
- Climate downscaling: statistical bias correction, neural operator emulators (FNO)

UNIT-IV: Energy Systems and Smart Grids

- Renewable generation forecasting: LSTM and Transformer models for solar/wind time series
- Unit commitment and economic dispatch: MILP formulation, Lagrangian relaxation
- Demand response: mechanism design, multi-agent reinforcement learning for load scheduling
- Battery storage optimisation: dynamic programming for charge/discharge scheduling
- Electric vehicle fleet scheduling: vehicle-to-grid (V2G) as a stochastic optimisation problem

UNIT-V: Climate Modelling and Disaster Response

- Earth system models: numerical PDE solvers, parameterisation schemes, model ensembles
- AI-accelerated simulation: physics-informed neural networks (PINNs), neural operator surrogates
- Extreme event detection: anomaly detection on reanalysis data; causal attribution methods
- Early warning systems: probabilistic flood forecasting with ensemble Kalman filtering
- Disaster logistics: vehicle routing problem (VRP) variants for evacuation and supply distribution

Books and References

1. Carla P. Gomes — Computational Sustainability, The Bridge, National Academy of Engineering, 2009
2. David Rolnick et al. — Tackling Climate Change with Machine Learning, ACM Computing Surveys, 2022
3. Dimitri P. Bertsekas — Dynamic Programming and Optimal Control, Athena Scientific, 2017
4. IPCC — Sixth Assessment Report (AR6), Cambridge University Press, 2021–2022
5. Tom Mitchell — Machine Learning, McGraw-Hill, 1997 (foundational methods)

COMP-DSE-205 (D): AI and Society

Course Overview

This course examines the societal impact of AI systems through a technical lens. Students analyse how algorithmic design choices, data pipelines, and deployment contexts produce measurable social outcomes and learn to apply computational auditing, impact assessment, and responsible design methods to real-world AI systems.

UNIT-I: AI as a Sociotechnical System:

- Sociotechnical systems theory: How technical artefacts encode social choices
- AI system anatomy: Data pipeline, model, deployment context, feedback loops
- Algorithmic amplification: Formalizing how recommender systems optimise engagement metrics
- Power asymmetries: Who controls training data, compute infrastructure, and model access

UNIT-II: Algorithmic Labour and Economic Impact:

- Task automation risk: Computational models of job susceptibility (Frey & Osborne, O*NET task database)
- Algorithmic management: Optimisation objectives in gig-economy dispatch systems; worker surveillance metrics
- Platform economics: Network effects, winner-takes-all dynamics, and data moats
- Policy levers: Algorithmic transparency requirements, portability, and interoperability mandates

UNIT-III: Surveillance, Disinformation, and Democratic Systems:

- Recommender system mechanics: Collaborative filtering, engagement optimisation, and filter bubble formation
- Computational propaganda: Bot detection, coordinated inauthentic behaviour, network analysis methods
- Facial recognition: System accuracy disparities across demographic groups; legal and technical limits
- Predictive policing algorithms: Risk score construction, feedback loops, and civil liberties implications
- Technical countermeasures: Watermarking, synthetic media detection, algorithmic transparency

UNIT-IV: Bias, Representation, and Generative AI:

- Training data auditing: Dataset documentation, representational bias measurement, participatory data collection
- Embedding geometry and bias: Word vector bias metrics (WEAT), debiasing methods and their limitations
- Generative AI outputs: Stereotype amplification in text-to-image models; benchmarks and red-teaming
- Machine translation and linguistic equity: BLEU score limitations, low-resource language challenges

UNIT-V: AI in Critical Public Services:

- Clinical AI: diagnostic model validation, regulatory pathways (FDA SaMD), distribution shift in deployment
- Algorithmic welfare systems: automated eligibility scoring, appeals mechanisms, explainability requirements
- Automated grading and proctoring: technical accuracy, adversarial circumvention, equity implications
- Technical audit methodology: black-box testing, disparate impact analysis, documentation standards

UNIT-VI: Governance, Regulation, and Standards:

- AI regulation architecture: EU AI Act risk tiers (unacceptable, high, limited, minimal)
- Auditing frameworks: NIST AI RMF, ISO/IEC 42001 — technical requirements and implementation
- Algorithmic impact assessment: methodologies, stakeholder engagement, and public disclosure
- Technical standards for AI transparency: model cards, datasheets, system cards

- Open research problems: measuring real-world algorithmic harm, causal inference for policy evaluation

Text/Reference Books:

1. Kate Crawford — Atlas of AI, Yale University Press, 2021
2. Safiya Umoja Noble — Algorithms of Oppression, NYU Press, 2018
3. Cathy O'Neil — Weapons of Math Destruction, Crown, 2016
4. Solon Barocas, Moritz Hardt, Arvind Narayanan — Fairness and Machine Learning, MIT Press, 2023
5. OECD — Artificial Intelligence in Society, OECD Publishing, 2019

COMP-DSE-205 (E): SWAYAM MOOCs Course

A pool of relevant **SWAYAM MOOCs Courses** consistent with the curriculum and learning outcomes of the programme will be notified at the commencement of the semester. Each aspirant student shall be permitted to opt for **only one SWAYAM MOOCs Course** from the approved pool.

COMP-SEC-206: AI Tools and Their Use Cases

Course Overview

This practitioner-oriented course surveys the AI tools ecosystem and develops students' ability to select, deploy, evaluate, and critically assess AI tools across real-world domains. Students gain hands-on experience with APIs, frameworks, and deployment pipelines, and develop the judgment to match tools to problems effectively and efficiently.

UNIT-I: The AI Tools Landscape:

- Taxonomy of AI tools: general-purpose assistants, domain-specific tools, APIs, and no-code/low-code platforms
- Tool evaluation criteria: performance, cost, privacy, scalability, and vendor considerations
- Working with AI APIs: authentication, rate limiting, and error handling
- Setting up an AI development environment: tools, containers, and cloud notebooks

UNIT-II: Language, Reasoning, and General-Purpose AI:

- How large language models work: architecture overview, training objectives, and alignment
- Prompt engineering: zero-shot, few-shot, chain-of-thought, and structured output strategies
- AI for reasoning and problem solving: logical inference, mathematical reasoning, and question answering
- AI for writing and communication: drafting, summarisation, translation, and editing
- Retrieval-Augmented Generation (RAG): grounding AI outputs in external knowledge
- Evaluation: automated metrics, human judgement, and benchmark suites

UNIT-III: Vision, Audio, and Multimodal AI:

- AI for image understanding and generation: capabilities, use cases, and limitations
- AI for video: generation, editing, summarisation, and analysis
- AI for audio: speech recognition, text-to-speech, and music/sound generation
- AI for presentations and visual content: slide generation, diagram creation, and design assistance
- Multimodal models: combining text, image, audio, and document inputs
- Responsible use of generative media: provenance, detection, and synthetic data

UNIT-IV: AI for Software Engineering:

- AI coding assistants: capabilities, workflows, and evaluation
- Evaluating AI-generated code: correctness, security, and maintainability
- Agentic coding: multi-step problem solving, tool use, and automated debugging

- AI across the software development lifecycle: testing, review, and static analysis

UNIT-V: Domain-Specific AI Tools:

- Healthcare AI: clinical text, medical imaging, and regulatory considerations
- Finance AI: fraud detection, risk assessment, and compliance automation
- Legal AI: contract analysis, legal research, and document intelligence
- Scientific AI: literature mining, hypothesis generation, and experiment support
- Education AI: intelligent tutoring, automated feedback, and personalised learning

UNIT-VI: Building and Deploying AI Applications:

- Orchestration and agent frameworks: chaining models, managing context, and building pipelines
- MLOps essentials: experiment tracking, model registry, and CI/CD for ML
- Deployment options: cloud platforms, managed endpoints, and self-hosted models
- Production monitoring: drift detection, logging, latency, and cost management

Text /Reference Books:

1. Chip Huyen — Designing Machine Learning Systems, O'Reilly, 2022
2. Sebastian Raschka et al. — Machine Learning with PyTorch and Scikit-Learn, Packt, 2022
3. Andrej Karpathy — Neural Networks: Zero to Hero (lecture series), 2023 (open access)
4. Hugging Face — The Hugging Face Course (NLP, Diffusion, RL), huggingface.co/learn (open access)
5. Peter Norvig and Stuart Russell — Artificial Intelligence: A Modern Approach, 4th ed., Pearson, 2021