COEP Technological University

(Unitary Public University of Government of Maharashtra)

Wellesely Road, Shivajinagar, Pune - 411005

Department of Computer Science and Engineering

Curriculum

(Structure, Evaluation Scheme and Course Content)

For

Post Graduate Program

Master of Technology

In

Artificial Intelligence and Machine Learning

With Effect From

Academic Year 2025-2026

Master of Technology
Artificial Intelligence and Machine Learning

Program Educational Objectives (PEOs)

PEO 1. To make students eligible to take up higher studies/research

PEO 2. To build competency among students to take up jobs that require technical expertise and problem-solving ability

PEO 3. To inculcate readiness among students for self-learning

PEO 4. To build competency among students in applying technology to solve real-life socio-economic problems

Program Outcomes (POs)

The post-graduate students will demonstrate:

PO1. Knowledge Acquisition: Acquire in-depth knowledge of Artificial Intelligence and Machine Learning to pursue research and higher studies.

PO2. Problem Analysis and Modeling: Identify, analyze, and model complex problems using AI and ML techniques to arrive at feasible solutions.

PO3. Design and Development of Solutions: Design AI-driven systems that address real-world engineering and socio-economic challenges.

PO4. Research and Innovation Aptitude: Demonstrate competence in formulating and conducting research using appropriate methodologies, tools, and techniques.

PO5. Lifelong Learning: Engage in independent, life-long learning for personal and professional development in a rapidly evolving technological landscape.

PO6. Ethics and Societal Impact: Understand professional, ethical, legal, and societal responsibilities and apply AI/ML knowledge to benefit society at large.

Correlation between the PEOs and the POs

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6
PEO 1	√	√	√		√	
PEO 2	√		√	√		√
PEO 3			√		√	
PEO 4		√	√	√	√	√

Master of Technology
Artificial Intelligence and Machine Learning
Curriculum Structure

List of Abbreviations

Abbreviation	Title	No of courses	Credits	% of Credits
PSMC	Program Specific Mathematics Course	1	4	5.00%
PSBC	Programme Specific Bridge Course	1	3	3.75%
PCC + LC	Programme Core Course + Laboratory Course	6	24	30.00%
PEC	Programme Elective Course	3	9	11.25%
OJT	On Job Training	1	3	3.75%
OE	Open Elective	1	3	3.75%
LLC	Liberal Learning Course	1	1	1.25%
SLC	Self Learning Course	2	6	7.50%
RM	Research Methodology	1	3	3.75%
AEC	Ability Enhancement Course	1	2	2.50%
Project	Project	2	22	27.5
	Total	**20**	**80**	**100%**

Master of Technology
Artificial Intelligence and Machine Learning
Curriculum Structure

Semester I

Sr. No.	Course Type	Course Code	Course Name	L	T	P	S	Cr	Evaluation Scheme (Weightages in %)				
									Theory			Laboratory	
									MSE	TA	ESE	ISE	ESE
1	PSMC	<tbd>	Probability, Statistics and Queuing Theory	3	1	-	1	4	30	20	50	-	-
2	PSBC	<tbd>	Algorithms and Complexity Theory	2	-	2	1	3	30	20	50	50	50
3	PCC	<tbd>	Artificial Intelligence	3	-	2	1	4	30	20	50	50	50
4	PCC	<tbd>	Machine Learning	3	-	2	1	4	30	20	50	50	50
5	PCC	<tbd>	Data Visualization Techniques	3	-	2	1	4	30	20	50	50	50
6	PEC-1	<tbd>	Department Elective -I 1. Time Series Analysis 2. Data Security and Privacy 3. Data Analytics	3	-	-	1	3	30	20	50	-	-
7	RM	<tbd>	Research Methodology and Intellectual Property Rights	3	-	-	1	3	30	20	50	-	-
Total Credits				**25**									

Legends:

L-Lecture, **T**-Tutorial, **P**-Practical, **S**-Self Study, **Cr**-Credits, **ISE**: In-Semester-Evaluation, **ESE**: End-Semester-Evaluation, **MSE**: Mid-Semester Evaluation, **TA**: Teacher's Assessment, **CIE**: Continuous-Internal-Evaluation

Master of Technology
Artificial Intelligence and Machine Learning
Curriculum Structure

Semester II

Sr. No.	Course Type	Course Code	Course Name	L	T	P	S	Cr	Evaluation Scheme (Weightages in %)				
									Theory			Laboratory	
									MSE	TA	ESE	ISE	ESE
1	OE	<tbd>	Open Elective	3	-	-	1	3	30	20	50	-	-
2	PCC	<tbd>	Deep Learning	3	-	2	1	4	30	20	50	50	50
3	PCC	<tbd>	Generative Adversarial Network	3	-	2	1	4	30	20	50	50	50
4	PCC	<tbd>	Optimization Techniques	3	-	2	1	4	30	20	50	50	50
5	PEC-2	<tbd>	Department Elective -II 1. Reinforcement Learning 2. Explainable Artificial Intelligence 3. ML-OPS	3	-	-	1	3	30	20	50	-	-
6	PEC-3	<tbd>	Department Elective -III 1. Natural Language Processing 2. Graph Nural Network 3. Federated Artificial Intelligence	3	-	-	1	3	30	20	50	-	-
7	AEC	<tbd>	Effective Technical Communication Skills and Self Awareness	1	-	2	1	2	50	50	-	100	
8	LLC	<tbd>	Liberal Learning Course	-	-	2	2	1	-	-	-	100	-
			Total Credits			**24**							

- The department offers “Data Structures” as Open Elective for students of other departments

- Exit option to qualify for PG Diploma in Computer Engineering:
    - Eight weeks domain-specific industrial internship in the month of June-July after successfully completing the first year of the program

Master of Technology
Artificial Intelligence and Machine Learning
Curriculum Structure

Semester III

Sr. No.	Course Type	Course Code	Course Name	L	T	P	S	Cr	Evaluation Scheme (Weightages in %)				
									Theory			Laboratory	
									MSE	TA	ESE	ISE	ESE
1	SLC	<tbd>	Massive Open Online Course –I	3	-	-	1	3	-	-	100	-	-
2	SLC	<tbd>	Massive Open Online Course –II	3	-	-	1	3	-	-	100	-	-
3	OJT	<tbd>	Internship	-	-	-	-	3	-	-	100	-	-
4	Project	<tbd>	Dissertation Phase – I	-	-	22	12	11	-	-	-	70	30
Total Credits				**20**									

Semester IV

Sr. No.	Course Type	Course Code	Course Name	L	T	P	S	Cr	Evaluation Scheme (Weightages in %)				
									Theory			Laboratory	
									MSE	TA	ESE	ISE	ESE
1	Project	<tbd>	Dissertation Phase – II	-	-	22	12	11	-	-	-	70	30
Total Credits				**11**									

Probability, Statistics and Queuing Theory

Course Code:		**Credit:**	4
Teaching Scheme		**Examination Scheme**	
Lectures:	3 Hrs	MSE:	30
Self-Study:	1 Hrs	TA:	20
Tutorial:	1 Hrs	ESE:	50

Course Outcomes: Students will be able to:

1. Solve problems related to basic probability theory.
2. Solve problems related to basic concepts and commonly used techniques of statistics.
3. Model a given scenario using continuous and discrete distributions appropriately and estimate the required probability of a set of events.
4. Apply the theory of probability and statistics to solve problems in domains such as machine learning, data mining, computer networks etc.

Basic Probability Theory **[2 Hrs]**

Probability axioms, conditional probability, independence of events, Bayes' rule, Bernoulli trials.

Random Variables and Expectation **[10 Hrs]**

Discrete random variables: Random variables and their event spaces, Probability Mass Function, Discrete Distributions such as Binomial, Poisson, Geometric etc., Indicator random variables, **Continuous random variables:** Distributions such as Exponential, Erlang, Gamma, Normal etc., Functions of a random variable, **Expectation:** Moments, Expectation based on multiple random variables, transform methods, Moments and Transforms of some distributions such as Binomial, Geometric, Poisson, Gamma, Normal.

Stochastic Processes **[6 Hrs]**

Introduction and classification of stochastic processes, Bernoulli process, Poisson process, Renewal processes.

Markov chains **[8 Hrs]**

Discrete-Time Markov chains: computation of n-step transition probabilities, state classification and limiting probabilities, distribution of time between time changes, M/G/1 queuing system. **Continuous-Time Markov chains:** Birth-Death process (M/M/1 and M/M/m queues), non-birthdeath processes, Petri nets.

Statistical Inference **[8 Hrs]**

Parameter Estimation – sampling from normal distribution, exponential distribution, estimation related to Markov chains, Hypothesis testing.

Regression and Analysis of Variance **[6 Hrs]**

Least square curve fitting, Linear and non-linear regression, Analysis of variance.

Textbooks:

[1] Ronald Walpole, Probability and Statistics for Engineers and Scientists, Pearson, ISBN13: 978-0321629111

Reference Books:

[1] Kishor Trivedi, Probability and Statistics with Reliability, Queuing, and Computer Science

Algorithms and Complexity Theory

Course Code:		**Credit:**	3		
Teaching Scheme		**Examination Scheme**			
Lectures:	2 Hrs	MSE:	30	Lab ISE:	50
Self-Study:	1 Hrs	TA:	20	Lab ESE:	50
Lab:	2 Hrs	ESE:	50		

Course Outcomes: Students will be able to:

1. Understand the fundamental concepts of algorithm design and time complexity analysis.
2. Apply suitable algorithm design techniques such as divide and conquer, greedy, and dynamic programming to solve well-known problems.
3. Analyze the time and space complexity of algorithms using asymptotic notation.
4. Evaluate computational problems by classifying them into complexity classes and justifying the classification.
5. Create efficient algorithms for given computational tasks using appropriate design strategies.

Mathematical Foundation **[4 Hrs]**

Growth of functions – Asymptotic notation, Standard notation and common functions, Summations, solving recurrences.

Analysis of Algorithms **[4 Hrs]**

Necessity of time and space requirement analysis of algorithms, Worst-case analysis of common algorithms and operations on elementary data structures (e.g. Heapsort), Average case analysis of Quicksort.

Standard Design Techniques-I **[6 Hrs]**

Divide and Conquer, Greedy method.

Standard Design Techniques-II **[6 Hrs]**

Dynamic programming, Network flow.

Standard Design Techniques-III **[6 Hrs]**

Backtracking, Branch-and-bound.

Complexity Theory **[4 Hrs]**

Introduction to NP-Completeness, Reducibility (SAT, 3VC, Independent Set, Subset Sum, Hamiltonian Circuit etc.)

Self-Study

Sorting in linear time, Elementary graph algorithms, Minimum spanning tree, Number -Theoretic algorithms: GCD algorithm, Chinese remainder theorem, Primality testing, String Matching Algorithms

Textbooks:

[1] Ellis Horowitz, Sartaj Sahni and Sanguthevar Rajasekaran, "Fundamentals of Computer Algorithms", Universities Press, 2nd edition (2008), ISBN-13: 978- 8173716126

[2] Thomas Cormen, Charles Leiserson, Ronald Rivest and Cliford Stein, "Introduction to Algorithms", PHI, 3rd edition, ISBN-13: 978-8120340077

Reference Books:

[1] Gilles Brassard and Paul Bratley, "Fundamentals of Algorithmics", PHI, ISBN-13: 978-8120311312

[2] Jon Kleinberg and Éva Tardos, "Algorithm Design", Pearson Education India, ISBN13: 978-9332518643

Suggested List of Assignments in the Laboratory:

[1] Recurrence Relations: Study and solve recurrence relations using formal methods such as substitution, recursion tree, and the Master Theorem to determine algorithmic time complexities.

[2] Sorting Algorithm Analysis: Implement, trace, and analyze the performance of advanced

sorting algorithms including Heap Sort, Quick Sort, and Merge Sort, with emphasis on time complexity in best, worst, and average cases.

[3] Greedy Strategy Implementation: Design and implement greedy algorithms to solve problems such as Fractional Knapsack, Job Sequencing with Deadlines, Huffman Coding, and Optimal Merge Pattern. Analyze correctness and efficiency.

[4] Graph Algorithms: Develop and evaluate graph-based solutions including: Single-source shortest path using Dijkstra's algorithm, Minimum Spanning Tree construction using Prim's and Kruskal's algorithms.

[5] Dynamic Programming Techniques: Implement dynamic programming solutions for classical problems:Matrix Chain Multiplication, Longest Common Subsequence (LCS), 0/1 Knapsack, All-Pairs Shortest Paths using Floyd-Warshall algorithm, Bellman-Ford algorithm for single-source shortest paths

[6] String Matching Algorithms: Implement and analyze the performance of pattern matching algorithms including the Naive approach and Knuth-Morris-Pratt (KMP) algorithm.

[7] Branch and Bound Applications: Apply the branch-and-bound strategy to solve computationally hard problems such as the 0/1 Knapsack and Travelling Salesperson Problem (TSP), focusing on pruning and bounding techniques.

[8] Backtracking Approaches: Solve combinatorial problems using backtracking techniques, including N-Queens Problem, Graph Coloring, Hamiltonian Path, Travelling Salesperson Problem (exact approach)

[9] Network Flow Algorithms: Implement and analyze network flow algorithms such as: Ford-Fulkerson method, Push-Relabel algorithm and evaluate their time complexity and applicability.

[10] NP-Completeness and Reductions: Apply polynomial-time reduction techniques to prove the NP-completeness of selected decision problems, understanding the significance of complexity classes and intractability.

This list serves as a guideline and is intended to be continuously refined by the instructor.

Artificial Intelligence

Course Code:		**Credit:**	4			
Teaching Scheme		**Examination Scheme**				
Lectures:	3 Hrs	MSE:	30	Lab ISE:	50	
Self-Study:	1 Hrs	TA:	20	Lab ESE:	50	
Lab:	2 Hrs	ESE:	50			

Course Outcomes: Students will be able to:

1. Explain the fundamental concepts, paradigms, and scope of Artificial Intelligence.
2. Apply search techniques, game-solving methods, and CSP frameworks to real-world problems.
3. Analyze the architecture and working of Artificial Neural Networks, including regression, classification, CNNs, and RNNs.
4. Evaluate AI models in the domains of computer vision and reinforcement learning for robustness, efficiency, and adaptability.
5. Design and create AI-based solutions integrating search, learning, perception, and reasoning to solve complex applications.

Introduction **[8 Hrs]**

The Foundations of Artificial Intelligence, The History of Artificial Intelligence, The State of the Art, Risks and Benefits of AI, Intelligent Agents: Agents and Environments, The Concept of Rationality, The Nature of Environments, The Structure of Agents.

Solving Problem by Searching **[8 Hrs]**

Problem Solving Agents, Uninformed Search: Breadth-First Search (BFS), Depth-First Search (DFS), Iterative Deepening DFS (IDDFS). Informed Search: Heuristic search, A* search, Greedy Best-First Search, and Local Search Algorithms. Search in Partially Observable Environments, Online Search Agents and Unknown Environments.

Game Solver and Constraint Satisfaction Problems (CSP) **[8 Hrs]**

Game Theory, Optimal Decisions in Games, Heuristic Alpha–Beta Tree Search, Monte Carlo Tree Search, Stochastic Games, Partially Observable Games. Defining CPSs, Constraint Propagation, Backtracking Search, Local Search for CSPs, Min-conflicts, heuristics (MRV, LCV).

Artificial Neural Network **[8 Hrs]**

Perceptron, Neural Network, Simple Feedforward Networks, Activation and Loss function, Backpropagation, Gradient descent, Linear Regression and Classification, Convolutional Networks, Generalization and Regularization, Recurrent Neural Networks.

Computer Vision **[8 Hrs]**

Image Formation, Simple Image Features, Classifying Images, Detecting Objects, Modern vision approaches with CNNs and Transformers, Encoder decoder function. Applications in face recognition, surveillance, medical imaging.

Self-Study

Reinforcement Learning: Learning from Rewards, Passive Reinforcement Learning, Active Reinforcement Learning, Generalization in Reinforcement Learning, Policy Search, Applications of Reinforcement Learning. Philosophy, Ethics, and Safety of AI.

Textbooks:

[1] Russell, S. & Norvig, P. (2020), Artificial Intelligence: A Modern Approach, 4th Ed. Pearson.
[2] Goodfellow, I., Bengio, Y., & Courville, A (2016), Deep Learning, MIT Press.
[3] Perry Xiao (2022), Artificial Intelligence Programming with Python, First Edition, Wiley.

Reference Books:

[1] David L. Poole, Alan K. Mackworth, "Artificial Intelligence: Foundations of Computational Agents", Third Edition, 2023, Cambridge University Press.
[2] Bishop, C. (2006). Pattern Recognition and Machine Learning. Springer.
[3] Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction. (2nd ed.). A Bradford Book.

Web Resources:

[1] Stanford Course webpage: https://stanford-cs221.github.io/spring2024/

[2] Barckley Course webpage: https://inst.eecs.berkeley.edu/~cs188/su25/
[3] MIT Course web page: https://ocw.mit.edu/courses/6-034-artificial-intelligence-fall-2010/pages/syllabus/
[4] NPTL Course webpage: https://nptel.ac.in/courses/106102220
[5] Coursera: Andrew Ng's courses on Machine Learning and Deep Learning: https://www.coursera.org/specializations/machine-learning-introduction
[6] Papers with Code: A repository of academic papers with corresponding code implementations: https://paperswithcode.com/
[7] TensorFlow Official Website: Excellent documentation and tutorials for the leading Deep learning framework: https://www.tensorflow.org/
[8] PyTorch Official Website: Excellent documentation and tutorials for the leading deep learning framework: https://pytorch.org/
[9] Google AI: Research and education resources from Google's AI team: https://cloud.google.com/ai
[10] Fast.ai: Practical, code-first courses on deep learning: https://www.fast.ai/

Suggested List of Assignments in the Laboratory:

[1] AI Agents & Search: Implement BFS, DFS, and UCS for pathfinding in a maze.
[2] Heuristic Search: Implement A* algorithm with different heuristics on an 8-puzzle problem.
[3] Game Solver: Implement Minimax and Alpha-beta pruning for Tic-Tac-Toe or Connect-4.
[4] CSP Solver: Implement a Sudoku solver using backtracking + MRV heuristic.
[5] Linear Models: Implement linear regression and logistic regression using gradient descent from scratch.
[6] Neural Networks (FFNN): Build a feedforward neural network for MNIST digit classification.
[7] CNNs for Vision: Implement and train a CNN on CIFAR-10 dataset; test with transfer learning (ResNet).
[8] RNNs for Sequence Data: Implement an RNN/LSTM for sentiment analysis on IMDB reviews.
[9] Reinforcement Learning: Implement Q-learning for OpenAI Gym environments (e.g., FrozenLake or CartPole).
[10] Advanced Project Implementation: Choose a topic from the self-study unit (e.g., NLP with Transformers, creating a simple GAN, or a ethical AI analysis) and build a small-scale project to demonstrate your understanding.

This list serves as a guideline and is intended to be continuously refined by the instructor.

Machine Learning

Course Code:		**Credit:**	4		
Teaching Scheme		**Examination Scheme**			
Lectures:	3 Hrs	MSE:	30	Lab ISE:	50
Self-Study:	1 Hrs	TA:	20	Lab ESE:	50
Lab:	2 Hrs	ESE:	50		

Course Outcomes: Students will be able to:

1. Understand fundamental machine learning concepts, including various types of learning paradigms.
2. Implement core machine learning algorithms from scratch using programming languages like Python and apply them to solve real-world problems.
3. Analyze the performance of different models, interpret their results, and evaluate trade-offs between model complexity, bias, and variance.
4. Critically assess and compare the suitability of various machine learning models for a given problem.
5. Design and develop an end-to-end machine learning pipeline, from data preprocessing and feature engineering to model deployment and maintenance.

Introduction **[6 Hrs]**
AI and ML Basics, AI Evolution, ML Systems Engineering, Defining ML Systems, Types of ML, Lifecycle of ML Systems, Emerging Trends, Practical Applications and Challenges.

Supervised Learning **[10 Hrs]**
Regression: Linear Regression, best fit regression line using least square and gradient descent method, performance metrics for regression. Classification: Logistic Regression, Naive Bayes, Decision Trees, K-Nearest Neighbors, Support Vector Machine, Performance Metrics: Confusion Matrix, Accuracy, Precision, Recall, F1-Score, ROC Curve, and AUC.

Ensemble Learning **[8 Hrs]**
Bagging, Boosting (AdaBoost, Gradient Boosting), and Random Forests. Model selection, hyperparameter tuning, feature selection.

Unsupervised Learning **[8 Hrs]**
Clustering: K-Means, DBSCAN, Hierarchical Clustering, Gaussian Mixture Models. Evaluating clustering performance (Silhouette Score).

Dimensionality Reduction **[8 Hrs]**
Dimensionality Reduction: Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), and t-SNE. Anomaly Detection: Isolation Forest and One-Class SVM.

Self-Study
Semi-Supervised Learning, MLOps: Concepts of model deployment, monitoring, and scaling, Federated Learning. Ethical considerations in Machine Learning: Bias, fairness, accountability, and privacy.

Textbooks:

[1] Introduction to Machine Learning by Ethem Alpaydin
[2] Machine Learning by Tom M. Mitchell.
[3] Pattern Recognition and Machine Learning by Christopher M. Bishop.

Reference Books:

[1] Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow by Aurelien Géron. (Latest Edition)
[2] Introduction to Machine Learning System by Vijay Janapa Reddi
[3] Practical Machine Learning with Python by Dipanjan Sarkar, Raghav Bali, Tushar Sharma

Web Resources:

[1] **Stanford CS229 lecture notes (Andrew Ng) — comprehensive lecture notes.** https://cs229.stanford.edu/lectures-spring2022/main_notes.pdf
[2] GeeksforGeeks - Machine Learning: https://www.geeksforgeeks.org/machine-learning/
Google Developers - Machine Learning Crash Course:

[3] https://developers.google.com/machine-learning/crash-course
Probabilistic Machine Learning (Kevin Murphy) — official project / book site.
[4] https://probml.github.io/pml-book/
scikit-learn — User guide & tutorials (official). https://scikit-
[5] learn.org/stable/user_guide.html
Keras guide (tf.keras): https://www.tensorflow.org/guide/keras
[6] TensorFlow Tutorials: https://www.tensorflow.org/tutorials
[7] **Andrew Ng — Machine Learning / Deep Learning courses (Coursera).**
[8] https://www.coursera.org/collections/machine-learning
UCI Machine Learning Repository — datasets for experiments.
[9] https://archive.ics.uci.edu/ml/datasets
Kaggle Datasets & Competitions (practical projects & datasets).
[10] https://www.kaggle.com/datasets

Suggested List of Assignments in the Laboratory:

[1] Data preprocessing: Clean, visualize and perform EDA on a tabular dataset (e.g., UCI Heart Disease / Titanic). Tasks: missing value handling, feature encoding, scaling, EDA plots, correlation, train/test split.
[2] Implement Linear Regression on a Boston Housing dataset, including model training, evaluation (MSE, R-squared), and visualization of fitted lines.
[3] Implementation of Logistic Regression for binary classification on a Breast Cancer dataset. Analyze performance using a confusion matrix, accuracy, precision, and recall.
[4] Building a Decision Tree Classifier for the Iris dataset. Visualize the tree and analyze the impact of pruning.
[5] Predicting heart disease using a Random Forest or Gradient Boosting Classifier (e.g., XGBoost/LightGBM). Compare the performance with a single Decision Tree. Demonstrate feature importances and effect of depth/regularization.
[6] Implementing a multi-class classification model using a Support Vector Machine (SVM) with different kernels on a digit recognition dataset. visualize decision boundaries and perform hyperparameter tuning (C, gamma).
[7] Performing clustering on a customer segmentation dataset using K-Means and Hierarchical Clustering. Visualize the clusters and interpret the results.
[8] Applying Principal Component Analysis (PCA) for dimensionality reduction on a high-dimensional dataset (e.g., face recognition).
[9] Applying a Naive Bayes classifier for spam detection on an email dataset.
[10] End-to-end pipeline: data ingestion, model training, evaluation, model versioning, export and a simple deployment demo (e.g., Flask or TensorFlow SavedModel + REST endpoint). Dataset: Choose a classification/regression dataset of interest (e.g., Bike Sharing, House prices). Deliverable: reproducible pipeline, Dockerfile or deployment notes, final report and project presentation.

This list serves as a guideline and is intended to be continuously refined by the instructor.

Data Visualization Techniques

Course Code:		**Credit:**	4		
Teaching Scheme		**Examination Scheme**			
Lectures:	3 Hrs	MSE:	30	Lab ISE:	50
Self-Study:	1 Hrs	TA:	20	Lab ESE:	50
Lab:	2 Hrs	ESE:	50		

Course Outcomes: Students will be able to:

1. Understand the fundamentals of Data Visualization.
2. Study and apply data collection and preprocessing techniques for visualizing data.
3. Identify and apply suitable visualizations techniques to represent distributions, amounts, and proportions in data.
4. Critically evaluate the quality and effectiveness of existing visualizations and dashboards.
5. Create end-to-end, reproducible visualization systems integrated with ML workflows.

Purpose of Data Visualization **[8 Hrs]**
Data, Data Taxonomy: 1D, temporal, 2D, 3D, 4D, ND, Trees, Graphs, Nominal, Ordinal, Quantitative; Visualization, Importance of Data Visualization, Goals of Visualization, Characteristics of Good Visualizations; Exploratory Data Analysis; Time series data and Visualization; Text and Visualization; Tools for Visualization

Collecting, pre-processing and Visualizing Data **[8 Hrs]**
Collection: Single source, Multiple sources, Web scraping; Data Cleaning and Aggregation; Mapping data onto aesthetics: Aesthetics and types of data, Scales map data values onto aesthetics; Coordinate systems and axes: Cartesian coordinates, Nonlinear axes, Coordinate systems with curved axes

Visualizing Amounts, Distributions, Proportions **[8 Hrs]**
Visualizing amounts: Bar plots; Grouped and stacked bars; Dot plots and heat maps; Visualizing distributions: Histograms and density plots, visualizing a single distribution, Visualizing multiple distributions at the same time, Empirical cumulative distribution functions and q-q plots, Empirical cumulative distribution functions, highly skewed distributions, Quantile–quantile plots.

Advanced Visualization Design **[8 Hrs]**
Visualizing many distributions at once: Visualizing distributions along the vertical axis, visualizing distributions along the horizontal axis; Visualizing proportions: pie charts, side-by-side bars, stacked bars and stacked densities, visualizing proportions separately as parts of the total; Visualizing nested proportions; Visualizing associations among two or more quantitative variables: Scatter plots, Correlograms, Dimension reduction, Paired data

Visualizing time series, trends and geospatial data **[8 Hrs]**
Time series: Individual time series, Multiple time series and dose–response curves, Time series of two or more response variables; Trends: Smoothing, Showing trends with a defined functional form, Detrending and time-series decomposition; Geospatial data: Projections, Layers, Choropleth mapping, Cartograms

Self-study
Text Data, n-gram, techniques for text data visualization: word cloud, bar chart, Bigram Network, Word frequency distribution plot, network graph, case study for text data visualization

Textbooks:

[1] Kieran Healy, “Data Visualization: A Practical Introduction". Princeton University Press, with ISBN-13: 978-0691185064
[2] Claus O. Wilke, “Fundamentals of Data Visualization: A Primer on Making Informative and Compelling Figures”, O'Reilly Media. The ISBN-13: 978-1492031086

Reference Books:

[1] Edward R. Tufte, “The Visual Display of Quantitative Information”, Graphics Pr, ISBN-13: 978-1930824133
[2] Alberto Cairo – The Functional Art: An Introduction to Information Graphics and Visualization, New Riders, ISBN-13: 978-0321834737

Web Resources:

[1] D3.js Gallery: https://d3js.org/
[2] From Data to Viz: https://www.data-to-viz.com/
[3] Information is Beautiful: https://www.informationisbeautiful.net/
[4] Storytelling with Data: https://www.storytellingwithdata.com/
[5] Nightingale - The Journal of the Data Visualization Society: https://nightingaledvs.com/
Tableau Public Gallery: https://public.tableau.com/en-us/gallery
[6] Data Viz Project: https://datavizproject.com/
[7] **Microsoft Power BI docs & training modules.** https://learn.microsoft.com/sr-latn-rs/power-
[8] bi/fundamentals/?utm_source=chatgpt.com
[9] Google AI's People + AI Research: https://pair.withgoogle.com/
[10] Matplotlib Gallery: https://matplotlib.org/stable/gallery/index.html

Suggested List of Assignments in the Laboratory:

[1] Create a single Python script that generates a scatter plot, a line plot, and a bar chart using Matplotlib for a given dataset, ensuring each plot has proper labels and a title.
[2] Using a public dataset (e.g., Titanic), create a pairplot to visualize relationships between numerical features and a heatmap to show feature correlations using Seaborn.
[3] Build an interactive scatter plot of the Iris dataset using Plotly, allowing users to hover over points to see details and click to highlight data points.
[4] Design and implement a simple dashboard using Dash (a framework for building web apps with Python), displaying at least three linked plots for a chosen dataset.
[5] Choose a real-world dataset and create a series of 3-4 plots that tell a clear story, annotating the plots with text to guide the viewer through the narrative.
[6] Apply PCA or t-SNE to a high-dimensional dataset (e.g., a handwritten digits dataset) and create a 2D scatter plot of the reduced dimensions to visualize clusters.
[7] Use a Python library like NetworkX and Matplotlib to visualize a social network or an airline route network as a graph, with nodes and edges representing entities and their connections.
[8] Create a choropleth map of India showing population density by state, using a library like Folium or GeoPandas.
[9] Implement a KPI dashboard in Tableau or Power BI: parameterized filters, actions/drill-through, and at least one forecast/explainability visual from an ML model. Provide a short usability test & heuristics report.
[10] For a trained model (classification or regression), build explainability views: feature importance, PDP/ICE or SHAP-inspired charts, calibration plot, error analysis. Package as an interactive report (Altair/Plotly or BI).

This list serves as a guideline and is intended to be continuously refined by the instructor.

Time Series Analysis

Course Code:		**Credit:**	3
Teaching Scheme		**Examination Scheme**	
Lectures:	3 Hrs	MSE:	30
Self-Study:	1 Hrs	TA:	20
		ESE:	50

Course Outcomes: Students will be able to:

1. Understand the fundamental concepts and components of time series data and its relevance in AI/ML applications.
2. Apply various statistical and machine learning techniques to model and forecast time series data.
3. Analyze the behavior of time series models using diagnostic tools and performance metrics.
4. Evaluate different time series models for real-world datasets and justify model selection.
5. Create advanced time series solutions using deep learning and hybrid approaches for complex temporal data.

Introduction **[8 Hrs]**

Definition, components (trend, seasonality, cycle, noise), and types of time series data. Time Series Visualization and Descriptive Statistics, Concept of stationarity, methods for testing stationarity, and transformations to achieve stationarity. White Noise and Random Walks.

Linear Models for Forecasting **[8 Hrs]**

Autoregressive (AR), Moving Average (MA), and ARMA Models, ARIMA and SARIMA Models, SARIMAX model, model identification using ACF/PACF plots, and model selection, Exponential smoothing, forecast combinations: hierarchical & grouped reconciliation.

Multivariate & Causal Time Series **[8 Hrs]**

Univariate Time Series Forecasting with Prophet. State-Space Models and Kalman Filtering, Vector Autoregression (VAR) models for multivariate time series, Cointegration and Granger Causality, Hidden Markov Models (HMM) for time series segmentation and classification.

Volatility, Anomalies & Probabilistic Forecasting **[8 Hrs]**

Conditional heteroscedasticity: ARCH/GARCH variants; volatility forecasting, Anomaly/Change-point detection (residual, density, and Bayesian methods), Evaluation beyond point forecasts: prediction intervals, coverage, Winkler score, CRPS, Probabilistic forecasting & simulation; backtesting and rolling-origin evaluation

Machine Learning and Deep Learning for Time Series **[8 Hrs]**

Time Series Forecasting as a Supervised Learning Problem. Feature Engineering for Time Series, Traditional ML Models for Time Series, Deep Learning Models: Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNNs). Temporal Fusion Transformers (TFT) and Informer models.

Self-Study

Validation Strategies for Time Series, Evaluation Metrics, Case Studies: Financial time series, sensor data, and sales forecasting, energy load,

Textbooks:

[1] Rob J Hyndman and George Athanasopoulos (2021), Forecasting: Principles and Practice (3rd Edition)

[2] Manu Joseph (2022), Modern Time Series Forecasting with Python

[3] Jason Brownlee (2018), Deep Learning for Time Series Forecasting: Predict the Future with MLPs, CNNs and LSTMs in Python

Reference Books:

[1] Robert H. Shumway and David S. Stoffer (2017), Time Series Analysis and Its Applications: With R Examples (4th Edition)

[2] Ben Auffarth (2021), Machine Learning for Time-Series with Python

[3] Brockwell, P.J., & Davis, R.A, Introduction to Time Series and Forecasting, 3rd ed., Springer (latest widely used edition).

Web Resources:

[1] Hyndman's Forecasting: Principles and Practice: The free online version of the textbook. (https://otexts.com/fpp3/)

[2] Machine Learning Mastery - Time Series: A practical blog with detailed tutorials on time series forecasting. (https://machinelearningmastery.com/time-series-forecasting/)

[3] Towards Data Science: Articles on a wide range of time series topics. (https://towardsdatascience.com/tagged/time-series-analysis)

[4] statsmodels Documentation: Official documentation for the core Python statistical library. (https://www.statsmodels.org/stable/tsa.html)

[5] Prophet by Meta: The official documentation and tutorials for the Prophet forecasting tool. (https://facebook.github.io/prophet/)

[6] sktime Documentation: A unified framework for machine learning with time series. (https://www.sktime.org/en/stable/)

[7] darts Library: A Python library for easy manipulation and forecasting of time series. (https://unit8co.github.io/darts/docs/userguide/)

[8] Google Developers - Time Series Forecasting: Tutorials and best practices. (https://developers.google.com/machine-learning/time-series-forecasting)

[9] TensorFlow Time Series Tutorial: Official tutorial on building a time series model with TensorFlow and Keras. (https://www.tensorflow.org/tutorials/structured_data/time_series)

[10] Papers with Code - Time Series Forecasting: A great resource for finding the latest research papers and their implementations. (https://paperswithcode.com/task/time-series-forecasting)

Data Security and Privacy

Course Code:		**Credit:**	3
Teaching Scheme		**Examination Scheme**	
Lectures:	3 Hrs	MSE:	30
Self-Study:	1 Hrs	TA:	20
		ESE:	50

Course Outcomes: Students will be able to:

1. Define cryptography and its principles
2. Explain Cryptography algorithms
3. Illustrate Public and Private key cryptography
4. Explain Key management, distribution, and certification
5. Explain authentication protocols.

Classical Encryption Techniques **[8 Hrs]**

Symmetric Cipher Model, Cryptography, Cryptanalysis and Brute-Force Attack, Substitution Techniques, Caesar Cipher, Monoalphabetic Cipher, Playfair Cipher, Hill Cipher, Polyalphabetic Cipher, One Time Pad.

Block Ciphers and the data encryption standard **[8 Hrs]**

Traditional block Cipher structure, stream Ciphers and block Ciphers, Motivation for the Feistel Cipher structure, the Feistel Cipher, The data encryption standard, DES encryption, DES decryption, A DES example, results, the avalanche effect, the strength of DES, the use of 56-Bit Keys, the nature of the DES algorithm, timing attacks, Block cipher design principles, number of rounds, design of function F, key schedule algorithm.

Public-Key Cryptography and RSA **[8 Hrs]**

Principles of public-key cryptosystems. Public-key cryptosystems. Applications for public-key cryptosystems, requirements for public-key cryptosystems. public-key cryptanalysis. The RSA algorithm, description of the algorithm, computational aspects, the security of RSA. **Other Public-Key Cryptosystems:** Diffie-Hellman key exchange, The algorithm, key exchange protocols, man in the middle attack, ElGamal Cryptographic systems.

Elliptic curve arithmetic **[8 Hrs]**

Elliptic curve arithmetic, abelian groups, elliptic curves over real numbers, elliptic curves over Zp, elliptic curves over GF(2m), Elliptic curve cryptography, Analog of Diffie-Hellman key exchange, Elliptic curve encryption/ decryption, security of Elliptic curve cryptography, Pseudorandom number generation based on an asymmetric cipher, PRNG based on RSA. **Key Management and Distribution:** Symmetric key distribution using Symmetric encryption, A key distribution scenario, Hierarchical key control, session key lifetime, a transparent key control scheme, Decentralized key control, controlling key usage, Symmetric key distribution using asymmetric encryption, simple secret key distribution, secret key distribution with confidentiality and authentication, A hybrid scheme, distribution of public keys, public announcement of public keys, publicly available directory, public key authority, public keys certificates.

X-509 certificates **[6 Hrs]**

X-509 certificates. Certificates, X-509 version 3, public key infrastructure. **User Authentication:** Remote user Authentication principles, Mutual Authentication, one way Authentication, remote user Authentication using Symmetric encryption, Mutual Authentication, one way Authentication, Kerberos, Motivation, Kerberos version 4, Kerberos version 5, Remote user Authentication using Asymmetric encryption, Mutual Authentication, one way Authentication. **Electronic Mail Security:** Pretty good privacy, notation, operational; description, S/MIME, RFC5322, Multipurpose internet mail extensions, S/MIME functionality, S/MIME messages, S/MIME certificate processing, enhanced security services, Domain keys identified mail, internet mail architecture, E-Mail threats, DKIM strategy, DKIM functional flow.

IP Security **[6 Hrs]**

IP Security: IP Security overview, applications of IPsec, benefits of IPsec, Routing applications, IPsec documents, IPsec services, transport and tunnel modes, IP Security policy, Security associations, Security associations database, Security policy database, IP traffic processing, Encapsulating Security payload, ESP format, encryption and authentication algorithms, Padding, Anti replay service. Transport and tunnel modes, combining security associations, authentication plus confidentiality, basic combinations of security associations, internet key exchange, key determinations protocol, header and payload formats, cryptographic suits.

Textbooks:

[1] William Stallings: Cryptography and Network Security, Pearson 6th edition.

Reference Books:

[1] V K Pachghare: Cryptography and Information Security, PHI 2nd edition

Data Analytics

Course Code:	**Credit:** 3
Teaching Scheme	**Examination Scheme**
Lectures: 3 Hrs	MSE: 30
Self-Study: 1 Hrs	TA: 20
	ESE: 50

Course Outcomes: Students will be able to:

1. Understand the foundational concepts of data analytics, including data types, preprocessing, and statistical analysis.
2. Apply analytical techniques to extract insights from structured and unstructured data.
3. Analyze real-world datasets using machine learning and visualization tools.
4. Evaluate the performance of data models and interpret results for decision-making.
5. Create end-to-end data analytics pipelines using modern tools and frameworks.

Fundamentals of Data Analytics **[8 Hrs]**
Data Analytics Basics, Data Types, Analytics Types, Data Analytics Steps: Data Pre-Processing, Data Imputation, Data Cleaning, Data Transformation, Data Visualization, and Data Engineering. Descriptive, Predictive, and Prescriptive Analytics.

Data Analytics with Python **[8 Hrs]**
Data Analytics using Python, Statistical Procedures, Web Scraping in Python, Advanced analytics, NumPy, Pandas, SciPy, Matplotlib.

Correlated Data Analysis **[8 Hrs]**
Analysis of Variance and Co-Variance, ANOVA results, Chi-Square Statistical Test, Examine Regression results, Regressing Analysis, Linear Regression and its analysis, Logistic Regression and its analysis.

Decision Tree and Cluster Analysis **[8 Hrs]**
Decision Tree Problem Analysis, Decision tree Construction, Decision Tree Algorithms; Applications of Cluster Analysis, Definition of Cluster, representing clusters, Clustering Techniques, K-Means Algorithm for Clustering, Advantages and Disadvantages of K-Means Clustering.

Social Media Analytics **[8 Hrs]**
Datasets, Analysis of Social Network Dataset Features, Learning Models and Validation, Association Rule Mining, artificial Neural Networks for web analytics.

Self-study
Case Study based the real-world application such as personalized recommendations, fraud detection and prevention in finance, healthcare with predictive analytics, agricultural yield prediction, agile retail with real-time data analytics.

Textbooks:

[1] Anil Maheshwari (2014), Data Analytics made accessible, Amazon Digital Publication
[2] Glenn J.Myatt,Wayne P.Johnson (2009), Making Sense of Data I: A Practical Guide to Exploratory Data Analysis and Data Mining, Wiley
[3] Reis & Housley (2022), Fundamentals of Data Engineering, O'Reilly

Reference Books:

[1] Wes McKinney (2017), Python for Data Analysis, O'Reilly
[2] Bruce & Bruce (2023), Practical Statistics for Data Scientists, 3rd ed., O'Reilly
[3] Thomas H. Davenport, Jeanne G. Harris and Robert Morison (2010), Analytics at Work: Smarter Decisions, Better Results, Harvard Business Press

Web Resources:

[1] Towards Data Science: https://towardsdatascience.com/
[2] Coursera (Google Data Analytics Professional Certificate): https://www.coursera.org/professional-certificates/google-data-analytics
[3] Pandas Documentation: Official documentation for the key Python data manipulation library. https://pandas.pydata.org/docs/

[4] Scikit-learn Documentation: https://scikit-learn.org/stable/
[5] Stack Overflow: A massive Q&A community for programmers and data professionals. https://stackoverflow.com/
[6] Tableau Public: A platform to create and share interactive data visualizations. https://public.tableau.com/en-us/s/
[7] Great Expectations Docs: docs.greatexpectations.io
[8] GeeksforGeeks AI/ML/Data Science Tutorial: https://www.geeksforgeeks.org/machine-learning/ai-ml-and-data-science-tutorial-learn-ai-ml-and-data-science/

Research Methodology and Intellectual Property Rights

Course Code:		**Credit:**	3
Teaching Scheme		**Examination Scheme**	
Lectures:	3 Hrs	MSE:	30
Self-Study:	1 Hrs	TA:	20
		ESE:	50

Course Outcomes: Students will be able to:

1. Understand research problem formulation and approaches of investigation of solutions for research problems
2. Learn ethical practices to be followed in research.
3. Discover how IPR is regarded as a source of national wealth and the mark of economic leadership in the context of the global market scenario.
4. Study the national & International IP system.

Research Problem **[8 Hrs]**

Meaning of research problem, Sources of research problem, Criteria Characteristics of a good research problem, Errors in selecting a research problem, Scope and objectives of research problem. Approaches of investigation of solutions for research problems, data collection, analysis, interpretation, and necessary instrumentations.

Literature Studies **[8 Hrs]**

Effective literature studies approaches, analysis, Use Design of Experiments /Taguchi Method to plan a set of experiments or simulations or build prototype Analyze your results and draw conclusions or Build Prototype, Test and Redesign.

Developing a Research Proposal **[8 Hrs]**

Plagiarism, Research ethics, Effective technical writing, how to write report, Paper. Developing a Research Proposal, Format of research proposal, a presentation and assessment by a review committee.

Intellectual Property Rights **[8 Hrs]**

Introduction to the concepts Property and Intellectual Property, Nature and Importance of Intellectual Property Rights, Objectives and Importance of understanding Intellectual Property Rights.

Types of Intellectual Property Rights **[8 Hrs]**

Patents-Indian Patent Office and its Administration, Administration of Patent System – Patenting under Indian Patent Act, Patent Rights and its Scope, Licensing and transfer of technology, Patent information and database. Provisional and Non-Provisional Patent Application and Specification, Plant Patenting, Idea Patenting, Integrated Circuits, Industrial Designs, Trademarks, Copyrights

Self-study

Case studies and new developments in IPR, Process of Patenting and Development: technological research, innovation, patenting, development, International Scenario: WIPO, TRIPs, Patenting under PCT.

Textbooks:

[1] B L Wadehra, Law Relating to Patents, Trademarks, Copyright, Designs and Geographical Indications, 2004
[2] Satyawrat Ponkse, The Management of Intellectual Property, 1991

Reference Books:

[1] Manual of Patent Office Practice and Procedure, 2019
[2] Niebel, Benjamin W, Product Design and Process Engineering, McGraw-Hill, 1974

Data Structure

Course Code:		**Credit:**	3
Teaching Scheme		**Examination Scheme**	
Lectures:	3 Hrs	MSE:	30
Self-Study:	1 Hrs	TA:	20
		ESE:	50

Course Outcomes: Students will be able to:

1. Understand and use the basic concepts for algorithm complexity
2. Use advanced data structures for efficient searching
3. Employ various data structures for implementing priority queues.
4. Analyse the time and space complexity string data structures.
5. Use different data structures to solve graph problems.
6. Design solutions for real-life problems using appropriate data structures.

Algorithm Concepts **[8 Hrs]**

Abstract data types, Data structures, Algorithms, Big Oh, Small Oh, Omega and Theta notations, solving recurrence equations, Master theorems, Generating function techniques, Constructive induction.

Advanced Search Structures for Dictionary ADT **[8 Hrs]**

Splay trees, Amortized analysis, 2-3 trees, 2-3-4 trees, Red-black trees, Randomized structures, Skip lists, Treaps, Universal hash functions.

Advanced Structures for Priority Queues **[6 Hrs]**

Binary Heap, Min Heap, Max Heap, Binomial heaps, Leftist heaps, Skewed heaps, Fibonacci heaps and its amortized analysis, Applications to minimum spanning tree algorithms.

Data Structures for Strings **[8 Hrs]**

Introduction to string data structures, Huffman coding tree, Tries, Compressed Tries, Suffix Trees, Suffix Arrays; Applications-Search Engines, Bioinformatics, Pattern Matching: KMP algorithm; Internet Packet Forwarding.

Graph Algorithms **[6 Hrs]**

BFS, DFS, Disjoint set union problem, Network flow; Maximum-Flow / Minimum-Cut; Ford–Fulkerson algorithm, Hamiltonian Path and circuit problem, Introduction to Hypergraphs.

Self-study

Geometric data structures, Cut vertices, Plane sweep paradigm, Hamiltonian Path and circuit problem.

Textbooks:

[1] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, Introduction to Algorithms, 3rd Edition, PHI Learning Pvt. Ltd.; ISBN-13: 978-0262033848 ISBN-10: 0262033844

[2] Robert Sedgewick and Kevin Wayne, Algorithms, Pearson Education, 4th Edition, ISBN-13: 978-0321573513

Reference Books:

[1] S. Dasgupta, C.H. Papadimitriou, and U. V. Vazirani, Algorithms, McGraw-Hill, 2006; ISBN-13: 978-0073523408 ISBN-10: 007352340, 2

[2] J. Kleinberg and E. Tardos, Algorithm Design, Addison-Wesley, 2006; ISBN-13: 978-0321295354 ISBN-10: 0321295358

Deep Learning

Course Code:		**Credit:**	4		
Teaching Scheme		**Examination Scheme**			
Lectures:	3 Hrs	MSE:	30	Lab ISE:	50
Self-Study:	1 Hrs	TA:	20	Lab ESE:	50
Lab:	2 Hrs	ESE:	50		

Course Outcomes: Students will be able to:

1. Understand the foundational concepts of neural networks and deep learning architectures.
2. Apply various deep learning models to solve real-world problems in vision, NLP, and structured data.
3. Analyze the performance of deep learning algorithms using appropriate metrics and techniques.
4. Evaluate different optimization strategies and regularization techniques for model improvement.
5. Create innovative deep learning solutions using frameworks like TensorFlow and PyTorch.

Feedforward Networks **[8 Hrs]**

Artificial Neurons, Neural networks, Activation Functions, Feedforward Neural Networks, Loss Functions, Backpropagation derivation and computational graph, Deep learning frameworks (PyTorch and TensorFlow)

Optimization Techniques **[8 Hrs]**

Gradient Descent Variants: Stochastic Gradient Descent (SGD), Mini-Batch GD. Advanced Optimizers: Momentum, RMSprop, Adam, and AdamW. Regularization Techniques: L1/L2 regularization, Dropout, and Batch Normalization. Hyperparameter Tuning: Grid Search, Random Search, and automated methods. Monitoring and debugging deep neural networks.

Convolutional Neural Networks (CNN) **[8 Hrs]**

Convolutional and pooling layers, architectures (LeNet, AlexNet, VGG, GoogLeNet, ResNet, EfficientNet), transfer learning, fine-tuning, batch size, normalization, augmentation, learning rate.

Recurrent Neural Networks (RCNN) **[8 Hrs]**

RNNs, LSTM, GRU, teacher forcing, sequence-to-sequence models, attention mechanisms.
Vanishing and Exploding Gradients, Truncated BPTT, Applications of RNNs: Machine translation, sentiment analysis, and time-series forecasting.

Advanced Topics **[8 Hrs]**

Autoencoders, Regularization in autoencoders, Denoising autoencoders, Sparse autoencoders, Contractive autoencoders, Variational Autoencoders (VAEs), Transformers: Self-attention mechanism. Key Models: BERT, GPT, and their applications in NLP. Generative Adversarial Networks (GANs) Applications: Image generation, style transfer, and data augmentation.

Self-study

Large Language Models (fine-tuning, RLHF overview), Deep Reinforcement Learning (DRL), Diffusion Models for image and video generation, Federated Learning and its applications

Textbooks:

[1] Ian Goodfelllow, Yoshua Benjio and Aaron Courville, Deep Learning, The MIT Press, 2016.
[2] François Chollet, Deep Learning with Python, Manning, Second Edition, 2021
[3] Trask, A. W. (2019). Grokking Deep Learning. Manning Publications.

Reference Books:

[1] Josh Patterson and Adam Gibson, Deep Learning: A Practitioner's Approach, O'Reilly Media, Inc., 2017
[2] Christopher Bishop, Pattern Recognition and Machine Learning, Springer, 2007
[3] Aggarwal, C. C., Neural Networks and Deep Learning: A Textbook, Springer, 2018

Web Resources:

[1] DeepLearning.AI (Coursera Specialization): https://www.deeplearning.ai
[2] Fast.ai Course: https://course.fast.ai/
[3] TensorFlow Tutorials: https://www.tensorflow.org/tutorials
[4] PyTorch Tutorials: https://pytorch.org/tutorials

[5] The Batch Newsletter by DeepLearning.AI: https://www.deeplearning.ai/the-batch/
[6] Distill.pub: https://distill.pub/
[7] CS231n: Convolutional Neural Networks for Visual Recognition (Stanford): http://cs231n.stanford.edu/
[8] Stanford CS231n GitHub repos and assignments (practical code): https://cs231n.github.io/?utm_source=chatgpt.com
[9] The Unofficial PyTorch and TensorFlow Book (A compilation of resources): https://d2l.ai/
Hugging Face: transformers library & tutorials (fine-tuning Transformers).
[10] https://huggingface.co/docs/transformers/

Suggested List of Assignments in the Laboratory:

[1] Manually implement a two-layer MLP (forward, backward) with ReLU/sigmoid, train on MNIST, also backprop, gradient checking, numerical stability, initialization. Compare the results with the framework's automatic differentiation.
[2] Implement training loop, dataset/dataloader, optimizer, scheduler, weight decay, and early stopping. Evaluate on MNIST.
[3] Develop a CNN model to classify images from the CIFAR-10 dataset. Use a pre-trained ResNet model on a new image dataset (e.g., a flower classification dataset) by fine-tuning the last layers.
[4] Build an LSTM model to generate new text, such as song lyrics or a simple story, character by character.
[5] Implement a simplified sequence-to-sequence model with an attention mechanism for a short translation task.
[6] Construct and train a simple GAN to generate new images of handwritten digits.
[7] Train the same model on the same dataset using different optimizers (e.g., SGD vs. Adam) and analyze their convergence and performance.
[8] Use Hugging Face Transformers to fine-tune BERT (or DistilBERT) for text classification (e.g., SST2/IMDB). Evaluate and report.
[9] Implement a VAE on MNIST and a basic DCGAN on CelebA / small image dataset. Compare generated samples and training stability.
[10] Students will select a project from a provided list or propose their own. The project should involve a complex problem requiring a deep learning solution, with a focus on problem formulation, implementation, and a comprehensive final report. Deliver code, trained model, evaluation, ablation study (1–2 variables), and short demo (colab notebook or REST endpoint).

This list serves as a guideline and is intended to be continuously refined by the instructor.

Generative Adversarial Network

Course Code:		**Credit:**	4		
Teaching Scheme		**Examination Scheme**			
Lectures:	3 Hrs	MSE:	30	Lab ISE:	50
Self-Study:	1 Hrs	TA:	20	Lab ESE:	50
Lab:	2 Hrs	ESE:	50		

Course Outcomes: Students will be able to:

1. Understand the foundational concepts of generative modeling and adversarial learning.
2. Apply GAN architectures to solve real-world problems in image, text, and audio generation.
3. Analyze the performance and limitations of various GAN models.
4. Evaluate advanced GAN variants and their suitability for specific applications.
5. Create novel GAN architectures and optimize them for custom tasks.

Introduction **[8 Hrs]**

Generative Adversarial Networks (GANs), mathematics underpinning, Random noise vector, Generator network, Discriminator network, Iterative training/tuning, Adversarial training, Nash equilibrium, Introduction to generative modelling, Encoder network, Latent space, Decoder network, Generation using an autoencoder, Variational autoencoder.

Deep Convolutional GAN **[8 Hrs]**

Convolutional neural networks, Convolutional filters, Parameter sharing, Batch normalization, Training challenges (Mode collapse, Slow convergence, Overgeneralization), Non-Saturating GAN.

Various types of GANs **[14 Hrs]**

Semi-Supervised GAN: Introduction to Semi-Supervised GAN, Architecture, Training process, Training objective. Conditional GAN: Motivation, CGAN Generator, CGAN Discriminator, Architecture of CGAN CycleGAN: Image-to-image translation, Cycle-consistency loss, Adversarial loss, Identity loss, CycleGAN architecture: building the network Object-oriented design of GANs.

Real-World Applications for GANs **[10 Hrs]**

Human Faces Generation, Deep Fake, Image-to-Image Translation/Restoration, Text to Image Generation, Enhancing Image Resolution, Semantic Image Inpainting, Text to Speech, Speech Enhancement with GANs.

Self-study

Reviewing the latest research papers, Case studies on Deepfake detection and mitigation, medical images for data augmentation, Image-to-image translation for style transfer and creative applications etc.

Textbooks:

[1] David Foster, Generative Deep Learning, 2nd edition, OREILLY, 2024
[2] Kuntal Ganguly, Learning Generative Adversarial Networks, 1st edition, BPB Publications, 2017
[3] Marija Jegorova, Hands-On Generative Adversarial Networks with PyTorch 2.x, 2nd edition, Packt, 2024

Reference Books:

[1] Sanaa Kaddoura, A Primer on Generative Adversarial Networks, Springer, 2023
[2] Kartik Chaudhary, The GAN Book: Train stable Generative Adversarial Networks using TensorFlow2, Keras and Python, 2024,
[3] Anish Khobragade, From Code to Creativity: The Generative AI Revolution, Xoffencer, 2024

Web Resources:

[1] GAN Lab: Interactive visualization of GAN training in a browser. https://poloclub.github.io/ganlab/
[2] The Original GAN Paper: "Generative Adversarial Networks" by Goodfellow et al. (2014). https://arxiv.org/abs/1406.2661
[3] DCGAN paper (Radford et al.): https://arxiv.org/abs/1511.06434

[4] Pix2Pix (Isola et al., CVPR 2017): https://openaccess.thecvf.com/content_cvpr_2017/papers/Isola_Image-To-Image_Translation_With_CVPR_2017_paper.pdf
[5] CycleGAN (Jun-Yan Zhu et al.): https://openaccess.thecvf.com/content_ICCV_2017/papers/Zhu_Unpaired_Image-To-Image_Translation_ICCV_2017_paper.pdf
[6] StyleGAN (Karras et al.): https://arxiv.org/abs/1812.04948
[7] Fréchet Inception Distance & TTUR (Heusel et al.): https://arxiv.org/abs/1706.08500
[8] TensorFlow Core: Official documentation and guides on GANs in TensorFlow. https://www.tensorflow.org/tutorials/generative/dcgan
[9] The State of the Art in GANs: A blog post that provides a timeline of GAN research. https://lilianweng.github.io/lil-log/2017/08/20/generative-adversarial-networks.html
[10] GANs in Action (Manning) - companion/livebook and code: https://www.manning.com/books/gans-in-action

Suggested List of Assignments in the Laboratory:

[1] Setup Python + PyTorch/TensorFlow environment. Implement a simple GAN to generate 1D data points or low-resolution images from a random noise vector. Visualize generator outputs during training; log generator/discriminator losses.
[2] Build a DCGAN using TensorFlow or PyTorch to generate images of a specific dataset (e.g., MNIST, CIFAR-10). Experiment with batchnorm, LeakyReLU, and different latent dimensions; report on sample quality.
[3] Explore GAN training instability by intentionally modifying the learning rates and observing the effects like mode collapse. Document the results and the loss curves.
[4] Implement a cGAN to generate images based on class labels on the MNIST dataset.
[5] Build an image-to-image translation model (e.g., Pix2Pix) to transform grayscale images into color images.
[6] Replicate the StyleGAN architecture to generate high-resolution, realistic human faces from the CelebA dataset. Explore latent space editing and attribute control.
[7] Implement a GAN for super-resolution, taking a low-resolution image and upscaling it while adding realistic detail.
[8] Create a GAN-based data augmentation pipeline to generate synthetic images for a small training dataset to improve a classifier's performance.
[9] Develop a simple GAN-based music generator that creates short musical sequences or MIDI files.
[10] Analyze and reproduce a simple GAN model from a research paper of your choice, not covered in the class, and prepare a presentation on its architecture and results.

This list serves as a guideline and is intended to be continuously refined by the instructor.

Optimization Techniques

Course Code:		**Credit:**	4		
Teaching Scheme		**Examination Scheme**			
Lectures:	3 Hrs	MSE:	30	Lab ISE:	50
Self-Study:	1 Hrs	TA:	20	Lab ESE:	50
Lab:	2 Hrs	ESE:	50		

Course Outcomes: Students will be able to:

1. Illustrate the concepts of optimization and its terminologies.
2. Apply and compare bio-inspired and physics-based optimization techniques.
3. Solve constrained and unconstrained optimization problems.
4. Design efficient hybrid and parallel algorithms for real-world problems.
5. Analyse the performance of algorithms in various engineering and ML applications.

Introduction to Optimization **[8 Hrs]**
Optimization Process, Mathematical Formulation, Minima, Optimality Conditions. Derivatives and Gradients, Local Decent

Deterministic Methods **[8 Hrs]**
First Order Methods: Gradient Descent, Conjugate Gradient, Momentum, Nesterov, Momentum, AdaGrad, RMSProp, Adadelta, Adam. Second-Order Methods: Newton's Method, Secant Method, Levenberg-Marquardt Algorithm, Levenberg-Marquardt for Sum of Squares, Quasi-Newton Methods

Stochastic and Population Methods **[8 Hrs]**
Noisy Descent, Mesh Adaptive Direct Search, Memory-Efficient Zeroth-Order Optimization, Simulated Annealing, Cross-Entropy Method, Natural Evolution Strategies, Covariance Matrix Adaptation. Population Methods: Population Iteration, Genetic Algorithms, Differential Evolution, Particle Swarm Optimization, Firefly Algorithm, Cuckoo Search

Probabilistic Models **[8 Hrs]**
Gaussian Distribution, Gaussian Processes, Prediction, Gradient Measurements, Noisy Measurements, Fitting Gaussian Processes.

Constraints Handling **[8 Hrs]**
Constrained Optimization, Constraint Types, Transformations to Remove Constraints, Removing Affine Equality Constraints, Lagrange Multipliers, Inequality Constraints, Slack Variables, Penalty Methods, Method of Multipliers, Interior Point Methods

Self-study
Recent optimizer papers (e.g., modifications to Adam/SGD), ADMM for constrained ML (sparsity, fairness constraints), Bayesian Optimization for Hyperparameter Tuning, Federated or distributed optimization algorithms, Automatic differentiation and implicit differentiation in hyperparameter tuning.

Textbooks:

[1] Mykel J. Kochenderfer, Tim A. Wheeler, Algorithms for Optimization. 2nd edition, MIT Press, 2025.
[2] Jorge Nocedal & Stephen J. Wright, Numerical Optimization, 2nd edition, Springer, 2006
[3] E. K. P. Chong & S. H. Żak, An Introduction to Optimization, 4th edition, Wiley, 2011

Reference Books:

[1] Debashish Das, Ali Safaa Sadiq, Seyedali Mirjalili, Optimization Algorithms in Machine Learning: A Meta-heuristics Perspective, Springer, 2025
[2] Suvrit Sra, Sebastian Nowozin & Stephen J. Wright, Optimization for Machine Learning. MIT Press, 2012
[3] Charu Aggarwal, Linear Algebra and Optimization for Machine Learning, Springer, 2020

Web Resources:

[1] Optimization Modules in TensorFlow - https://www.tensorflow.org/api_docs/python/tf/keras/optimizers

[2] PyTorch Optimization / torch.optim docs & tutorial https://docs.pytorch.org/tutorials/beginner/basics/optimization_tutorial.htm
[3] SciPy's Optimize Module - https://docs.scipy.org/doc/scipy/reference/optimize.html
[4] The Matrix Calculus You Need for Deep Learning - http://www.matrixcalculus.org/
[5] Coursera: Optimization for Machine Learning Specialization - https://www.coursera.org/specializations/optimization-for-machine-learning
[6] Stanford — CS231n (Optimization notes for deep learning) https://cs231n.github.io/optimization-1/
[7] MIT Press: Optimization for Machine Learning https://mitpress.mit.edu/9780262537766/optimization-for-machine-learning/
[8] SIAM / MOS: First-Order Methods in Optimization (Amir Beck) https://epubs.siam.org/doi/book/10.1137/1.9781611974997
[9] Numerical Optimization (Springer page) https://link.springer.com/book/10.1007/978-0-387-40065-5

Suggested List of Assignments in the Laboratory:

[1] Implement Gradient Descent from scratch to find the minimum of a simple 1D quadratic function. Visualize the descent path.
[2] Extend the Gradient Descent implementation to find the minimum of a 2D function like the Sphere or Beale function. Plot the contour lines and the convergence path.
[3] Implement Mini-batch Gradient Descent and compare its convergence speed with full-batch Gradient Descent on a simple linear regression problem.
[4] Implement a Newton's Method and compare its convergence rate with Gradient Descent on a convex function. Analyze the number of iterations required for both.
[5] Use the scipy.optimize library to solve an unconstrained problem. Compare the results of different built-in solvers (e.g., BFGS, CG, Newton-CG).
[6] Use a constrained optimization library (scipy.optimize.minimize) to solve a simple problem with a single equality constraint using the Lagrange Multiplier method.
[7] Implement a basic Genetic Algorithm from scratch to solve a simple function optimization problem. Experiment with different crossover and mutation rates.
[8] Apply a pre-built meta-heuristic solver (e.g., from pygad) to solve a combinatorial problem like the Traveling Salesperson Problem (TSP) for a small number of cities.
[9] Train a simple neural network using different built-in optimizers in TensorFlow or PyTorch (e.g., SGD, Adam, Adagrad). Compare and report the training and validation losses for each.
[10] Select an optimization technique from Unit VI (Self-Study) and apply it to a real-world dataset or a Kaggle competition problem. Document the problem formulation, chosen algorithm, and results in a concise report.

This list serves as a guideline and is intended to be continuously refined by the instructor.

Reinforcement Learning

Course Code:	**Credit:** 3
Teaching Scheme	**Examination Scheme**
Lectures: 3 Hrs	MSE: 30
Self-Study: 1 Hrs	TA: 20
	ESE: 50

Course Outcomes: Students will be able to:

1. Understand the **fundamental concepts of** Reinforcement Learning
2. Apply the RL algorithms for decision making in uncertain conditions
3. Analyze the performance and convergence of various RL techniques.
4. Evaluate the suitability of RL models for real-world problems in robotics, gaming, and finance.
5. Design and implement advanced RL systems using deep learning frameworks.

Introduction **[8 Hrs]**

Introduction to Reinforcement Learning: The Agent-Environment Interaction. The Reinforcement Learning Problem: Goals and Rewards. Elements of Reinforcement Learning: Policy, Reward Function, Value Function, and Model. Finite Markov Decision Processes (MDPs). Introduction to Multi-armed Bandits.

Dynamic Programming and Monte Carlo Methods **[8 Hrs]**

Dynamic Programming: Policy Evaluation, Policy Improvement, Policy Iteration, and Value Iteration. Monte Carlo Methods: Monte Carlo Prediction, Monte Carlo Control, and On-policy vs. Off-policy learning.

Temporal-Difference (TD) Learning **[8 Hrs]**

TD Prediction: TD(0), Sarsa, and Q-learning. On-policy Control with Sarsa. Off-policy Control with Q-learning. The concept of Eligibility Traces: Forward and Backward view of TD(λ) and Sarsa(λ), Q(λ).

Planning and Learning with Tabular Methods **[8 Hrs]**

Models and Plan, Integrating Planning, Acting, and Learn, Prioritized Sweeping, Full vs. Sample Backups, Trajectory Sampling, Heuristic Search, Monte Carlo Tree Search

Function Approximation **[8 Hrs]**

Value Prediction with Function Approximation, Gradient-Descent Method, Linear Method, Control with Function Approximation, Bootstrap. Actor–Critic Method, Eligibility Traces for Actor–Critic Method, R-Learning and the Average-Reward Setting.

Self-study

Students must select and study one or more advanced topics from: Deep Q-Networks (DQN), model-based RL (MBPO), offline RL (CQL, Conservative Q-learning), distributional RL (C51, QR- DQN), RLHF case studies, safety & constrained optimization in RL, multi-agent learning, or RL for robotics and sim2real). Submit a short survey and present it in class.

Textbooks:

[1] Rich Sutton and Andrew Barto (2018). Reinforcement Learning: An Introduction (2nd ed.). A Bradford Book. MIT Press.

[2] Lapan, M. (2020). Deep Reinforcement Learning Hands-On, 2nd ed. Packt Publishing.

[3] Morales, M. (2020). Grokking Deep Reinforcement Learning. Manning Publications.

Reference Books:

[1] Bertsekas, D.P. (2019). Reinforcement Learning and Optimal Control. Athena Scientific.

[2] Russell, S., & Norvig, P. (2020). Artificial Intelligence: A Modern Approach (4th ed.). Pearson.

[3] Shiyu Zhao (2025), Mathematical Foundations of Reinforcement Learning , Springer

Web Resources:

[1] Reinforcement Learning (Coursera Specialization): https://www.coursera.org/specializations/reinforcement-learning

[2] OpenAI Spinning Up in Deep RL: https://spinningup.openai.com/

[3] Hugging Face Deep Reinforcement Learning Course: https://huggingface.co/learn/deep-rl-course/
[4] DeepMind's RL Lectures: https://deepmind.com/learning-resources/reinforcement-learning-lectures
[5] RL Course by David Silver (UCL): https://www.davidsilver.uk/teaching/
[6] Andrej Karpathy's Blog: "Deep Reinforcement Learning: Pong from Pixels": https://karpathy.github.io/2016/05/31/rl/
[7] Sutton & Barto Book Online: http://incompleteideas.net/book/bookdraft2018jan1.pdf
[8] OpenAI Gym Environment: https://gym.openai.com/
[9] Stable Baselines3 Documentation: https://stable-baselines3.readthedocs.io/en/master/
[10] The Unreasonable Effectiveness of RL: https://www.oreilly.com/library/view/the-unreasonable-effectiveness/9781492039942/

Explainable Artificial Intelligence

Course Code:	**Credit: 3**
Teaching Scheme	**Examination Scheme**
Lectures: 3 Hrs	MSE: 30
Self-Study: 1 Hrs	TA: 20
	ESE: 50

Course Outcomes: Students will be able to:

1. Differentiate between black-box and transparent Artificial Intelligence systems
2. Understand the concept of explainability and interpretability in Artificial Intelligence
3. Apply interpretability techniques in Artificial Intelligence models
4. Design and use the explainable methods for applications
5. Evaluate the explainability in Artificial Intelligence models
6. Employ explainable techniques to state-of-the-art deep learning models in vision and language

---

Explainability **[8 Hrs]**

XAI: explainability, need, blackbox model, architecture, characteristics, advantages, disadvantages, challenges; AI vs XAI; Inherently interpretable models, post-hoc explanations, accuracy-explainability tradeoff; Transparency: types, dangers, fairness and discrimination

Inherently Interpretable Models **[8 Hrs]**

Interpretability: definitions, properties, motivations, challenges; Interpretable Models: rule-based approaches – Bayesian rule list, interpretable decision sets, risk scores, generalized additive models, prototype-based models, attention-based models

Post-hoc explanation methods **[8 Hrs]**

Classification of explanation methods; Local and global explanations; Local explanation methods - Feature importance, anchors, saliency maps - input gradient, smoothgrad; Global explanation methods – collection of local explanations, representation-based, model distillation, summaries of counterfactuals

Evaluation **[8 Hrs]**

Evaluation of correctness and interpretability of explanations; Evaluation methods for interpretable models and post-hoc explanation methods; Limitations of interpretable models and post-hoc explanation evaluation methods

Applications and Case studies **[8 Hrs]**

XAI in computer vision and natural language processing applications, Explainability in LSTM, LLMs

Self-study **[8 Hrs]**

Tools, frameworks, platforms for XAI; LIME, SHAP

Textbooks:

[1] Christoph Molnar, “Interpretable Machine Learning”, Leanpub, 2020

[2] Pethuru Raj, Utku Kose, Usha Sakthivel, Susila Nagarajan, Vijanth S, Asirvadam, “Explainable Artificial Intelligence (XAI): Concepts, enabling tools, technologies and applications”, ISBN: 978183953695

Reference Books:

[1] Mayuri Mehta, Vasile Palade, Indranath Chatterjee, “Explainable AI: Foundations, Methodologies and Applications”, Springer, ISBN 978-3-031-12806-6

[2] Wojciech Samek, Gregoire Montavon, Andrea Vedaldi, Lars Kai Hansen, Klaus Robert Muller, “Explainable AI: Interpreting, Explaining and Visualizing Deep Learning”, Springer

[3] Michael Munn, David Pitman, “Explainable AI for Practitioners: Designing and implementing Explainable ML solutions”, O’Reilly Media, Inc., ISBN: 9781098119133

Web Resources:

[1] Machine Learning Explainability Workshop, Stanford online, https://www.youtube.com/playlist?list=PLoROMvodv4rPh6wa6PGcHH6vMG9sEIPxL

[2] Doshi-Velez and Kim, 2017, Towards a Rigorous Science of Interpretable Machine Learning

[3] Weller, 2019, Transparency: Motivation and Challenges

[4] Lipton, 2017, The Mythos of Model Interpretability

[5] Hong et. al., 2020, Human Factors in Model Interpretability: Industry Practices, Challenges

[6] Kaur et. al., 2020, Interpreting Interpretability: Understanding Data Scientists' Use of Interpretability Tools for Machine Learning

ML-OPS

Course Code:	**Credit:** 3
Teaching Scheme	**Examination Scheme**
Lectures: 3 Hrs	MSE: 30
Self-Study: 1 Hrs	TA: 20
	ESE: 50

Course Outcomes: Students will be able to:

1. Explain the MLOps lifecycle, roles, and system components for production ML.
2. Apply tooling for data/version control, experiment tracking, CI/CD, and containerization to make ML work reproducible.
3. Design scalable training/serving pipelines with orchestration on cloud/Kubernetes, including batch, online, and streaming patterns.
4. Evaluate reliability, observability, drift, fairness, privacy, and security risks; choose appropriate monitoring and governance mechanisms.
5. Build and deploy an end-to-end ML/LLM system with automated testing, rollout strategies (canary/shadow/A-B), and cost controls.

MLOps Foundations & Reproducibility **[8 Hrs]**
MLOps motivation vs. model-centric workflows; org roles and interfaces. ML system design: datasets, features, models, services, feedback loops. Reproducibility: environments (conda/poetry), images (Docker), seeds; data & model versioning (DVC, Git-LFS); experiment tracking (MLflow/W&B). Artifact/metadata stores; model registries; basics of governance.

Pipelines & Orchestration (CI/CD for ML) **[8 Hrs]**
CI for ML: unit tests, data/feature tests, model tests; code quality gates. CD for ML: build, validate, and promote models via registries; infra as code (Terraform). Workflow engines: Airflow/Prefect/Kubeflow; feature stores; automated retraining. Data contracts and schema enforcement (Great Expectations).

Serving & Deployment Patterns **[8 Hrs]**
Packaging models (ONNX, TorchScript), inference servers (FastAPI, BentoML, KFServing/Triton). Deployment targets: VMs, containers, serverless (AWS Lambda/Cloud Run), and Kubernetes/EKS/GKE. Patterns: batch, request/response, streaming; real-time feature pipelines. Rollouts: blue-green, canary, shadow, A/B; traffic shaping; rollback strategies; performance tuning (p95 latency, throughput).

Observability, Reliability, & Responsible AI **[8 Hrs]**
Monitoring: data quality, data drift, concept drift, performance regression, skew; logging, tracing, metrics. Post-deployment evaluation: slices, confidence, calibration, and continuous evaluation. SRE for ML systems: SLIs/SLOs for accuracy & latency; incident response and playbooks. Responsible AI: bias/fairness checks, privacy (PII handling, DP basics), model cards; security for ML supply chain.

Scaling, Cost, and LLMOps **[8 Hrs]**
Distributed training & inference (data/model/pipeline parallelism; autoscaling; GPU scheduling). Feature & vector stores; retrieval-augmented generation (RAG) basics for LLMOps. LLMOps specifics: prompt/versioning, offline/online LLM evals, safety filters, guardrails; caching. Cost governance: right-sizing, spot instances, quantization/distillation; green AI considerations.

Self-study
A comparative study of two orchestration stacks (e.g., Kubeflow vs. Prefect) on the same pipeline. A production case study (from an open-source repo or paper) dissecting deployment, monitoring, and rollback. An LLMOps mini-project: prompt/versioning, evaluation harness, and a safe rollout plan.

Textbooks:

[1] Practical MLOps: Operationalizing Machine Learning Models by Noah Gift and Alfredo Deza (latest edition).
[2] Machine Learning Production Systems: Engineering Machine Learning Models and Pipelines by Carl Osipov (latest edition).
[3] Building Machine Learning Pipelines by Hannes Hapke and Catherine Nelson (latest edition)

Reference Books:

[1] Designing Machine Learning Systems: An Iterative Process for Production-Ready Applications by Chip Huyen.
[2] MLOps Engineering at Scale by Carl Osipov.
[3] Introducing MLOps: How to Scale Machine Learning in the Enterprise by Mark Treveil and The Dataiku Team.

Web Resources:

[1] Coursera Specialization: Machine Learning Engineering for Production (MLOps) from DeepLearning.AI.
[2] Google Cloud: Machine Learning Operations (MLOps) Fundamentals (Official Documentation and Tutorials).
[3] DataCamp: MLOps and other related courses.
[4] Stanford CS329S: Machine Learning Systems Design (complete course site with projects/readings). stanford-cs329s.github.io
[5] Chip Huyen's MLOps Guide (updated Jan 2025) (curated roadmap from fundamentals to advanced). Chip Huyen
[6] Neptune.ai — How to Learn MLOps (2024+) (survey of tools, books, and learning paths). neptune.ai
[7] Awesome-MLOps (GitHub) (actively maintained, curated resources and tools). GitHub
[8] Coursera MLOps course catalog (2025) (up-to-date course options for extra practice). Coursera

Natural Language Processing

Course Code:		**Credit:**	3
Teaching Scheme		**Examination Scheme**	
Lectures:	3 Hrs	MSE:	30
Self-Study:	1 Hrs	TA:	20
		ESE:	50

Course Outcomes: Students will be able to:

1. Demonstrate the understanding of basic text processing techniques in NLP.
2. Design, implement and evaluate part-of-speech taggers and parsers for a language.
3. Build language models and demonstrate Word Sense Disambiguation using WordNet.
4. Analyze and build word embeddings for different languages.

Introduction **[8 Hrs]**

What is NLP, Fundamental and Scientific goals, Engineering goals, stages of NLP, problems in NLP, Applications of NLP, Empirical Laws of language, zipf's law, Heap's law..

Basic Text Processing **[8 Hrs]**

Tokenization, word token, word type, sentence segmentation, feature extraction, issues in tokenization for different languages, word segmentation, text segmentation, normalization, case folding, Spelling Correction, Morphology, Stemming, Porters Algorithm, , lemmatization, spelling correction - dynamic programming approach for finding edit distance, N-gram Language Modelling- context sensitive spelling correction, probabilistic language model, auto completion prediction, Evaluation and perplexity, Smoothing techniques.

POS Tagging **[8 Hrs]**

Sequence labelling tasks of NLP, POS tagging, POS tag sets, Hidden Markov Model- Introduction, Markov Processes, HMM characterization -Likelihood of a sequence (Forward Procedure, Backward Procedure), Best state sequence (Viterbi Algorithm), Re-estimation(Baum-Welch - Forward-Backward Algorithm), Models for Sequential tagging – Maximum Entropy, Conditional Random Field.

Syntax **[8 Hrs]**

Constituency and dependency parsing, Constituency parser -Syntactic structure, Parsing methodology, Different parsing algorithms, Parsing in case of ambiguity, Probabilistic parsing, CKY algorithm, Issues in parsing, Dependency parsing- Syntactic structure, Parsing methodology, Transition-Based Dependency Parsing, Graph-Based dependency parsing, Evaluation, Co-Reference Books resolution, Named-entity recognition.

Knowledge Base and Semantics **[8 Hrs]**

WordNet: Word Senses, Word relations, Word similarity and thesaurus methods, Word sense disambiguation, WordNet. Lexical and Distributional Semantics - Introduction, models of semantics, applications.

Self-study

Word Embeddings: Introduction, one-hot vectors, methods of generating word embeddings, Skip-gram, CBOW, Glove model, Fast Text model, evaluation measures-rough scores.

Textbooks:

[1] Daniel Jurafsky and James H. Martin, "Speech and Language Processing", Second Edition, Prentice Hall, 2008, ISBN: 978-0131873216.

[2] Allen James, "Natural Language Understanding", Second Edition, Benjamin/Cumming, 1994, ISBN: 978-0805303346.

[3] Chris Manning and Hinrich Schuetze, "Foundations of Statistical Natural Language Processing", MIT Press, ISBN: 978-0262133609.

Reference Books:

[1] Journals: Computational Linguistics, Natural Language Engineering, Machine Learning, Machine Translation, Artificial Intelligence.

[2] Conferences: Annual Meeting of the Association of Computational Linguistics (ACL), Computational Linguistics (COLING), European ACL (EACL), Empirical Methods in NLP (EMNLP), Annual Meeting of the Special Interest Group in Information Retrieval (SIGIR),

Human Language Technology (HLT).

Web Resources:

[1] Stanford University's CS224N: Natural Language Processing with Deep Learning, https://web.stanford.edu/class/cs224n/

[2] DeepLearning.AI's Natural Language Processing Specialization, https://www.coursera.org/specializations/natural-language-processing

[3] Hugging Face's NLP Course, https://huggingface.co/learn/nlp-course

[4] NLTK Book: Natural Language Processing with Python, https://www.nltk.org/book

[5] fast.ai: Practical Deep Learning for Coders, https://course.fast.ai/

[6] Swayam NPTEL: Deep Learning for Natural Language Processing, https://onlinecourses.nptel.ac.in/noc25_cs22/preview

[7] Jay Alammar's Blog, https://jalammar.github.io/

[8] Google Cloud: Natural Language Operations Fundamentals, https://www.coursera.org/learn/google-cloud-natural-language-processing

[9] Natural Language Processing [An Overview Guide] – https://www.deeplearning.ai/resources/natural-language-processing/?utm_source=chatgpt.com

Graph Neural Network

Course Code:		**Credit:**	3
Teaching Scheme		**Examination Scheme**	
Lectures:	3 Hrs	MSE:	30
Self-Study:	1 Hrs	TA:	20
		ESE:	50

Course Outcomes: Students will be able to:

1. Explain graph data models, message passing, and the GNN learning paradigm.
2. Apply core GNN architectures (GCN, GraphSAGE, GAT, graph transformers) to node/edge/graph-level tasks.
3. Analyze scalability techniques (sampling, mini-batching, distributed training) and evaluate models on standard benchmarks.
4. Evaluate robustness, fairness, explainability, and temporal aspects of GNNs in real applications.
5. Design and implement an end-to-end GNN pipeline using a modern framework.

Graph Learning Foundations **[8 Hrs]**

Graph Theory: nodes, edges, types of graphs, and graph properties. Challenges of applying deep learning to graph data, differences between tabular and graph data, graph-based learning, Graph Neural Network (GNN), GNN applications, Mental model for training a GNN, mechanisms of a GNN model, message passing.

Graph Embedding **[8 Hrs]**

Creating embeddings with Node2Vec: loading data, setting parameters, and creating embeddings, demystifying embeddings, transforming and visualizing the embeddings, constructing the embeddings, GNN vs. N2V embeddings, data preprocessing, random forest classification, embeddings in an end-to-end model, representations and embeddings, Transductive and inductive methods, N2V, random walks across graphs, Message passing as deep learning.

Graph Convolutional Network **[8 Hrs]**

Predicting consumer product categories, loading and processing the data, creating our model classes, model training, model performance analysis. Aggregation methods: neighborhood aggregation, advanced aggregation tools, practical considerations in applying aggregation, Optimizations and refinements, dropout, model depth. convolution methods, message passing, GCN aggregation function, GCN in PyTorch Geometric, Spectral vs. spatial convolution, GraphSAGE aggregation function, GraphSAGE in PyTorch Geometric

Graph Attention Networks **[8 Hrs]**

Detecting spam and fraudulent reviews, exploring the review spam dataset: explaining the node features, data analysis, graph structure, Training GAT models: neighborhood loader and GAT models, addressing class imbalance in model performance, deciding between GAT and XGBoost.

Graph Autoencoder **[8 Hrs]**

Generative models: generative and discriminative models, synthetic data, graph autoencoders for link prediction, defining a graph autoencoder, training a graph autoencoder to perform link prediction. Variational graph autoencoders, generating graphs using GNNs, understanding link prediction tasks, and inner product decoder.

Self-study

Data preparation and project planning, project definition, project objectives and scope, designing graph models, domain and use case, constructing the graph dataset and schemas, creating instance models, testing and refactoring, data pipeline, raw data, data exploration and visualization, preprocessing and loading data into PyG, Find graph data.

Textbooks:

[1] Broadwater, K., & Stillman, N. (2025), Graph Neural Networks in Action, Manning Publications.
[2] Lingfei Wu, Peng Cui, Jian Pei, Liang Zhao (2022), Graph Neural Networks: Foundations, Frontiers, and Applications, Springer Nature
[3] Labonne, M. (2023), Hands-On Graph Neural Networks Using Python, Packt Publishing.

Reference Books:

[1] Maxime Labonne (2023), Hands-On Graph Neural Networks Using Python, Packt Publishing.
[2] Pethuru Raj Chelliah, Pawan Whig, Susila Nagarajan, Usha Sakthivel, Nikhitha Yathiraju (2025), Graph Neural Networks: Essentials and Use Cases, Springer Nature
[3] Jude Max (2025), Graph Neural Networks in Practice, Amazon Digital Services

Web Resources:

[1] Stanford CS224W – Machine Learning with Graphs (course site) — lectures, notes, assignments. cs224w.stanford.edu
[2] CS224W YouTube playlist (full lectures) latest recorded lectures. YouTube
[3] Stanford Online CS224W/XCS224W page — official course overview & enrollment info. Stanford Online+1
[4] Geometric Deep Learning – Lectures — complete GDL lecture series (Bronstein et al.). geometricdeeplearning.com
[5] Geometric Deep Learning – Book chapters/proto-book — free chapters and blog. geometricdeeplearning.com
[6] PyTorch Geometric (PyG) documentation — tutorials, API, examples. pytorch-geometric.readthedocs.io
[7] PyG official site + GitHub repository — project home and example zoo. pyg.orgGitHub
[8] Deep Graph Library (DGL) docs — tutorials and user guide (incl. distributed training). DGL+1
[9] Open Graph Benchmark (OGB) website + dataset overview — datasets, loaders, leaderboards. Open Graph Benchmark+1
[10] Papers with Code – GNN task hub — curated papers + leaderboards for GNN methods. Papers with Code

Federated Artificial Intelligence

Course Code:	**Credit:** 3
Teaching Scheme	**Examination Scheme**
Lectures: 3 Hrs	MSE: 30
Self-Study: 1 Hrs	TA: 20
	ESE: 50

Course Outcomes: Students will be able to:

1. Explain the principles, architecture, and challenges of Federated Learning.
2. Apply algorithms and tools for federated training, model aggregation, and decentralized optimization.
3. Analyze issues related to privacy, security, scalability, and heterogeneity in federated systems.
4. Evaluate real-world applications of Federated AI in healthcare, finance, IoT, and edge intelligence.
5. Design and implement end-to-end federated AI systems with privacy-preserving techniques and efficient deployment strategies.

Introduction **[8 Hrs]**
Centralized vs decentralized ML, need for federated AI. Fundamental Principles: On-device data, collaborative training, and no raw data sharing. Federated Learning (FL) paradigms: horizontal FL, vertical FL, federated transfer learning. Architecture: client-server, peer-to-peer, hybrid. Core algorithms: federated averaging (FedAvg), FedSGD.

Fundamentals of Federated Learning **[8 Hrs]**
Federated Learning Architecture & System Design, Types of Federated Learning: Cross-device vs Cross-silo, Basic algorithms: Federated Averaging (FedAvg), Data heterogeneity and non-IID data challenges, Communication efficiency and compression techniques.

Privacy and Security in Federated Learning **[8 Hrs]**
Privacy risks in distributed learning, Differential Privacy (DP), Secure Multi-Party Computation (SMPC), Homomorphic Encryption, Adversarial attacks and robustness in FL. Optimization and Algorithms, Optimization challenges in FL, Gradient aggregation and client selection strategies, Handling stragglers and fault tolerance Personalization in federated learning.

Edge AI Fundamentals **[8 Hrs]**
Hardware constraints and AI model design for Edge devices, Model compression techniques: Pruning, Quantization, Knowledge Distillation, Lightweight neural network architectures (MobileNet, TinyML), Edge inference and deployment strategies.

Case Studies and Applications **[8 Hrs]**
Federated learning in healthcare, finance, and IoT, Edge AI in autonomous vehicles, smart cameras, and wearable devices, Real-world frameworks: TensorFlow Federated, PySyft, Flower.

Self-study
Survey of latest federated LLM architectures. Comparative analysis of privacy-preserving techniques (DP, SMC, HE). Reproduction of a research paper (NeurIPS/ICML/AAAI) on federated AI. Mini-project: Building a federated recommendation system or federated fraud detection.

Textbooks:

[1] Sahoo, J., Ouaissa, M., & Nair, A. K. (Eds.). (2024). Federated Learning: Principles, Paradigms, and Applications. Apple Academic Press.
[2] Yu, H., Li, X., Xu, Z., Goebel, R., & King, I. (Eds.). (2025). Federated Learning in the Age of Foundation Models. Springer Nature.
[3] Li, Q., & Liu, C. (2024). Federated Learning: From Algorithms to System Implementation. World Scientific Publishing.
[4] Qiang Yang, Yang Liu, Tianjian Chen, Yongxin Tong, Springer (2021), Federated Learning: Foundations and Applications

Reference Books:

[1] Nguyen, V., et al. (2024). Federated Learning: Theory and Practice. Elsevier.
[2] Konečnỳ, J., et al. (2016). Federated Optimization: Distributed Machine Learning for On-Device Intelligence. arXiv.

[3] Srivastava, P., & Singh, J. (2024). Blockchain for Secure Federated Learning: Applications in Edge Computing. CRC Press.

Web Resources:

[1] Stanford's CS329X: **Federated Learning: Research and Practice,** https://cs329x.stanford.edu
[2] Google AI Blog on Federated Learning, https://ai.googleblog.com/2017/04/federated-learning-collaborative.html
[3] TensorFlow Federated Documentation, https://www.tensorflow.org/federated
[4] PySyft (Privacy-preserving FL framework), https://github.com/OpenMined/PySyft
[5] Flower Framework for Federated Learning, https://flower.dev
[6] FedML: Research-Oriented FL Framework, https://fedml.ai
[7] OpenMined's Federated Learning Courses, https://courses.openmined.org
[8] MIT Deep Learning for Privacy & Federated Learning (Course notes), http://people.csail.mit.edu/tal/fedai/
[9] Papers With Code: Federated Learning Leaderboard, https://paperswithcode.com/task/federated-learning
[10] arXiv: Federated Learning Survey Papers Collection, https://arxiv.org/list/cs.LG/recent

Effective Technical Communication Skills and Self Awareness

Course Code:		**Credit:**	2	
Teaching Scheme		**Examination Scheme**		
Lectures:	1 Hrs	MSE:	50	Lab ISE: 100
Self-Study:	1 Hrs	TA:	50	
Lab:	2 Hrs			

Course Outcomes: Students will be able to:

1. Produce effective dialogue for business related situations
2. Use listening, speaking, reading and writing skills for communication purposes and attempt tasks by using functional grammar and vocabulary effectively
3. Analyze critically different concepts/principles of communication skills
4. To appreciate, analyze, and evaluate business reports and research papers

Fundamentals of Communication **[4 Hrs]**

7 Cs of communication, common errors in English, enriching vocabulary, styles, and registers.

Aural-Oral Communication **[4 Hrs]**

The art of listening, stress and intonation, group discussion, oral presentation skills.

Reading and Writing **[8 Hrs]**

Types of reading, effective writing, business correspondence, interpretation of technical reports and research papers.

Textbooks:

[1] Raman Sharma, "Technical Communication", Oxford University Press.

[2] Raymond Murphy "Essential English Grammar" (Elementary & Intermediate) Cambridge University Press.

[3] Markel, M., & Rosson, P. (2024). Technical Communication (14th ed.). Bedford/St. Martin's.

[4] Shirley Taylor, "Model Business Letters, Emails and Other Business Documents" (seventh edition), Prentice Hall

[5] Thomas Huckin, Leslie Olsen "Technical writing and Professional Communications for Non-native speakers of English", McGraw Hill.

Reference Books:

[1] Thomas Huckin, Leslie Olsen "Technical writing and Professional Communications for Non-native speakers of English", McGraw Hill.

[2] Pfeiffer, W., & Goodall, H. L. (2024). Technical Communication: A Practical Approach. Pearson.

Web Resources:

[1] MIT OpenCourseWare – Technical Communication, https://ocw.mit.edu/courses/technical-communication

[2] Purdue OWL (Online Writing Lab) – Technical Writing, https://owl.purdue.edu/owl/subject_specific_writing/technical_writing/index.html

[3] IEEE Author Center – Resources for Writing Technical Papers, https://authorcenter.ieee.org

[4] Nature Masterclasses – Scientific Writing and Publishing, https://masterclasses.nature.com

[5] Toastmasters International – Public Speaking Resources, https://www.toastmasters.org/resources

[6] GitHub: Awesome Technical Writing, https://github.com/maestroj/awesome-technical-writing

[7] The Hemingway App: A tool for clear writing, http://www.hemingwayapp.com/

Liberal Learning Course

Course Code:	**Credit:** 1
Teaching Scheme	**Examination Scheme**
Lab: 2 Hrs	Lab ISE: 100
Self-Study: 2 Hrs	

Guidelines:

Liberal Learning Courses began with a vision of expanding the horizons of knowledge in a variety of areas beyond Engineering. It provides opportunities to students of Engineering to foray into areas of their interest, to contribute to their overall personality development. The students are required to go through the areas of agriculture, Clay Art & Pottery, Dance (Contemporary), Dance (Indian), Film Appreciation, French, Geography, Holistic Health, Interior Design, Introduction to Indian Armed Forces, Music (Instrumental), Music (Vocal), Painting, Photography, Political Science, Theatre & Dramatics, Wood & Metal Art etc. Experts from respective areas conduct classes for each area on campus through activities, discussions, presentations, and lecture methods, and an evaluation out of 100 per area is done for each area throughout the semester. Evaluation patterns may differ according to the nature of each area. Although there is no pre-defined syllabus for LLC areas, there is an outline that experts normally develop and follow for the classes. However, students may approach the faculty to cover certain topics of interest in that area during classes based on students' interests and experts' areas of expertise.

Massive Open Online Course – I

Course Code: **Credit:** 3

Teaching Scheme		**Examination Scheme**	
Lectures:	3 Hrs		
Self Study:	1 Hr	ESE:	100

Course Outcomes: Students will be able to:

1. Acquire new skills or knowledge to enhance their personal and professional development.
2. Receive a flexible learning environment, allowing one to study at own pace and convenience.
3. Opportunity for lifelong learning.
4. Foster collaboration and networking among participants.

The students in consultation with the faculty advisor, opt for a single course of 12 weeks offered by the NPTEL in the current semester. The students need to register for the examination conducted by the NPTEL. For the students who secured a passing score in the NPTEL examination, the marks obtained for assignments (in 25 marks) will be upscaled to out of 50 marks of CIE and the marks obtained from the certificate examination (in 75 marks) will be downscaled 50 marks of ESE assessments.

Massive Open Online Course – II

Course Code: **Credit:** 3

Teaching Scheme		**Examination Scheme**	
Lectures:	3 Hrs		
Self Study :	1 hrs	ESE:	100

Course Outcomes: Students will be able to:

1. Acquire new skills or knowledge to enhance their personal and professional development.
2. Receive a flexible learning environment, allowing one to study at own pace and convenience.
3. Opportunity for lifelong learning.
4. Foster collaboration and networking among participants.

The students in consultation with the faculty advisor, opt for a single course of 12 weeks offered by the NPTEL in the current semester. The students need to register for the examination conducted by the NPTEL. For the students who secured a passing score in the NPTEL examination, the marks obtained for assignments (in 25 marks) will be upscaled to out of 50 marks of CIE and the marks obtained from the certificate examination (in 75 marks) will be downscaled 50 marks of ESE assessments.

Dissertation Phase – I

Course Code:	**Credit:** 11
Teaching Scheme	**Examination Scheme**
Lab: 22 Hrs	ISE: 70
Self-Study: 12 Hrs	ESE: 30

Course Outcomes: Students will be able to:

1. Demonstrate how to search the existing literature to gather information about a specific problem or domain.
2. Identify the state-of-the-art technologies and research in the chosen domain and highlight open problems that are relevant to societal or industrial needs.
3. Evaluate various solution techniques to determine the most feasible solution within the given constraints for the chosen dissertation problem.
4. Apply software engineering principles related to requirements gathering and design to produce relevant documentation.
5. Write a dissertation report that details the research problem, objectives, literature review, and solution architecture.
6. Deliver effective oral presentations to communicate the findings and outcomes of the research work.

Guidelines:

The dissertation is a year-long project, conducted and evaluated in two phases. It can be carried out either in-house or within an industry as assigned by the department. The project topic and internal advisor (a faculty member from the department) are determined at the beginning of Phase I.

Students are expected to complete the following activities in Phase-I:

1. Literature survey
2. Problem Definition
3. Motivation for study and Objectives
4. Preliminary design /feasibility / modular approaches

Deliverables:

1. A report having the following details: Abstract, Problem statement, Requirements specification, Literature survey, Proposed solution, High-level design description, Plan for implementation and testing in Phase-II
2. A presentation that covers the major points covered in the report.
3. A proof of concept (preferably, but not mandatory)

Evaluation:

Two independent assessments (Mid-Semester and End-Semester evaluations) will be made. In both the Examinations, the internal guide, along with a Senior Faculty member of the department, will evaluate the work. The marks obtained in these two assessments will be combined to get the final evaluation out of 100 marks. The course grading, like other courses, will be relative in nature.

The evaluation will take place based on criteria such as literature survey and well-defined project problem statement, proposed high level system design, concrete plan for implementation and result generation, presentation etc.

The panel (external examiner(s) and senior faculty) will provide a report about suggestions/changes to be incorporated during phase-II.

Dissertation Phase – II

Course Code:		**Credit:**	11
Teaching Scheme		**Examination Scheme**	
Lab:	22 Hrs	ISE:	70
Self-Study:	12 Hrs	ESE:	30

Course Outcomes: Students will be able to:

1. Achieve proficiency in the languages, tools, libraries, and technologies used in the dissertation work.
2. Apply project planning principles and techniques to ensure effective and efficient project execution.
3. Demonstrate an understanding of the entire lifecycle of a software product or solution.
4. Produce artifacts such as source code, test plans, and test results based on the dissertation work.
5. Write research paper(s) and a thesis in accordance with publication ethics.
6. Exhibit the presentation skills needed to effectively present the work at various platforms.

Guidelines:
Students are expected to complete the following activities in Phase-II:

1. Implementation of the proposed approach in the first stage
2. Testing and verification of the implemented solution
3. Writing of a report and presentation
4. Publish the work done at a suitable Scopus indexed conference/in a journal

Deliverables:
1. Source code (if the project is in-house)
2. Dissertation report that gives overview of the problem statement, literature survey, design, implementation details, testing strategy and results of testing
3. All the artifacts created throughout the duration of dissertation such as requirements specification, design, project plan, test cases etc
4. Presentation based on the dissertation report
5. Research Paper(s) based on the dissertation work

Evaluation:
Mid-Semester evaluation: In the MSE, the internal guide, along with a Senior Faculty of the department, will evaluate the work. In the End Semester Examination evaluation, the internal guide, along with an external expert (usually from an Industry) will evaluate the work. The marks obtained in these two assessments will be combined to get the final evaluation out of 100 marks. The course grading, like other courses, will be relative in nature. The assessment is done on the criteria such as concrete system design, implementation status and concrete plan for completion of remaining tasks, presentation etc. The purpose of Mid-Semester evaluation is also to check preparedness of students for the End-Semester evaluation. Examiners may give suggestions for changes/corrections to be incorporated before the final evaluation. If the work done till then may not lead to successful completion of the dissertation in the remaining time, the student may be asked to take an extension in time to complete the course.
End-Semester evaluation: The assessment of End-Semester evaluation will be done based on the criteria such as quality of implementation, result analysis, project outcomes (publications, patent, copyright, contribution to opensource community, participation in project competition etc.), quality of report, presentation etc. The total assessment of phase-II work is for 100 marks and the grading, like other courses, will be relative.