St Xavier's College (Autonomous), Ahmedabad

M.Sc. Artificial Intelligence

Program Specific Outcomes (PSOs)

1. **AI Fundamentals:** Understand core AI concepts and mathematical foundations.
2. **Machine Learning Techniques:** Apply supervised and unsupervised learning.
3. **Ethical AI:** Consider ethical implications and societal norms.
4. **NLP and Computer Vision:** Explore language processing and image analysis.
5. **Reinforcement Learning and Robotics:** Study control systems and robotics.
6. **Industry Projects and Internships:** Gain practical experience.
7. **Research and Collaboration:** Foster innovation and interdisciplinary work.

Semester I

CORE Paper: Search Methods in Problem Solving
Course Code: PAI-1801
No. of Credits: 04 (3 Theory + 1 Practical)
Learning Hours: 75 (45 Theory + 30 Practical)

Course Outcomes (COs)

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

1. Apply blind search techniques to configuration and planning problems.
2. Use heuristic and local search methods for AI problem spaces such as SAT, TSP, and 8-Puzzle.
3. Implement stochastic and informed search algorithms for optimization and pathfinding.
4. Design problem decompositions and game-playing strategies using minimax and alpha-beta pruning.

Course Structure

Unit 1: Introduction and Blind Search

Introduction to AI and state-space search; configuration and planning problems; solution space generation; depth-first, best-first, and depth-bound search; iterative deepening; evaluation of search methods.

Unit 2: Heuristic Search

Heuristic functions for SAT, TSP, and 8-Puzzle; best-first and local search methods—hill climbing, variable neighborhood descent, beam search, tabu search, iterated hill climbing.

Unit 3: Stochastic Local Search and A*

Random walk and stochastic hill climbing; simulated annealing; genetic algorithms; TSP representations; ant colony optimization; A* and variants—branch and bound, Dijkstra's, IDA*, recursive best-first search.

Unit 4: Problem Decomposition and Game Playing

Goal trees and AO*; game theory; minimax, alpha-beta pruning, and SSS* algorithms.

References

1. Deepak Khemani, *Search Methods in Artificial Intelligence*, Cambridge University Press, 2024.
2. Stuart Russell & Peter Norvig, *Artificial Intelligence: A Modern Approach*, 4th Edition, Prentice Hall, 2024.
3. Deepak Khemani, *A First Course in Artificial Intelligence*, McGraw Hill, 2014.

Semester I

CORE Paper: Linear Algebra and Optimization
Course Code: PAI-1802
No. of Credits: 04 (3 Theory + 1 Practical)
Learning Hours: 75 (45 Theory + 30 Practical)

Course Outcomes (COs)

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

1. Apply vector and matrix concepts to model and analyze linear systems.
2. Analyze abstract linear algebra notions such as norms, orthogonality, and eigenstructure.
3. Use matrix decomposition techniques (eigen, SVD) for data reduction and ML applications.
4. Apply differential calculus and constrained optimization methods to AI problems.

Course Structure

Unit 1: Linear Algebra Essentials

Vectors, matrices, vector spaces, subspaces, basis, dimension, and linear transformations.

Unit 2: Abstract Linear Algebra

Metric and norm spaces, L1–L_∞ norms, orthogonal projection, eigenvalues, eigenvectors, and positive semidefinite matrices.

Unit 3: Matrix Decompositions and Applications

Eigen and spectral decomposition, SVD, low-rank approximations, and machine learning applications.

Unit 4: Calculus and Optimization

Differentiation, integration, partial derivatives, gradients, convex and non-convex optimization, Lagrange multipliers, and duality.

References

1. Marc Peter Deisenroth, A. Aldo Faisal, Cheng Soon Ong, *Mathematics for Machine Learning*, Cambridge University Press, 2020.
2. Sheldon Axler, *Linear Algebra Done Right*, Springer, 2015.
3. Stephen Boyd & Lieven Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

Semester I

CORE Paper: Python Programming
Course Code: PAI-1803
No. of Credits: 04 (2 Theory + 2 Lab)
Learning Hours: 60 (30 Theory + 30 Lab)

Course Outcomes (COs)

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

1. Write and debug basic Python programs using core language constructs.
2. Use Python data structures and file handling for practical tasks.
3. Apply advanced programming features such as OOP, recursion, and exceptions.
4. Utilize Python libraries and interfaces for data science and web-related tasks.

Course Structure

Unit 1: Introduction to Python and Basic Programming Concepts

Importance of Python, variables, statements, data types, control structures, loops, functions, and algorithms.

Unit 2: Data Structures and File Handling

Lists, strings, dictionaries, tuples, sets, file handling, string formatting, modules (OS, Random), and basic simulations.

Unit 3: Advanced Programming Concepts

Recursion, decorators, comprehensions, exceptions, object-oriented programming, classes, inheritance, interfaces, and generators.

Unit 4: Python for Data Science and Web

Databases, web scraping, GUI using Tkinter, and introduction to NumPy, Pandas, SciPy, and Matplotlib.

References

1. Mark Lutz, *Learning Python*, 5th ed., O'Reilly, 2013.
2. Wes McKinney, *Python for Data Analysis*, O'Reilly, 2018.
3. Allen B. Downey, *Think Python*, 2nd ed., O'Reilly, 2015.

Semester I

CORE Paper: Statistical Methods
Course Code: PAI-1804
No. of Credits: 04 (3 Theory + 1 Lab)
Learning Hours: 60 (45 Theory + 15 Lab)

Course Outcomes (COs)

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

1. Design and preprocess datasets for statistical analysis.
2. Summarize and interpret data using descriptive statistics and visualization.
3. Analyze categorical data using contingency tables and tests of association.
4. Identify temporal patterns using basic time-series techniques.

Course Structure

Unit 1: Data Collection & Visualization

Measurement scales, data collection, cleaning, preprocessing, and graphical representation of data.

Unit 2: Basic Statistics

Frequency tables, measures of central tendency, dispersion, skewness, kurtosis, percentiles, box plots, and correlation.

Unit 3: Contingency Tables

Two-way tables, measures of association, and chi-square tests for independence.

Unit 4: Introduction to Time Series

Components, decomposition, smoothing, autocorrelation, and stationarity in time-series data.

References

1. Sheldon M. Ross, *Introductory Statistics*, Academic Press, 2020.
2. S. C. Gupta & V. K. Kapoor, *Fundamentals of Mathematical Statistics*, Sultan Chand & Sons, 2020.
3. G. Jay Kerns, *Introduction to Probability and Statistics Using R*, CRC Press, 2010.

Semester I

CORE Paper: Probability and Stochastic Processes
Course Code: PAI-1805
No. of Credits: 04 (3 Theory + 1 Lab)
Learning Hours: 75 (45 Theory + 30 Lab)

Course Outcomes (COs)

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

1. Apply basic principles of probability, combinatorial reasoning, and conditional probability.
2. Model real-world phenomena using standard univariate probability distributions.
3. Analyze joint, marginal, and conditional properties of bivariate distributions.
4. Understand and apply stochastic process concepts, including Markov chains and Poisson processes.

Course Structure

Unit 1: Basic Probability

Concepts of experiments, outcomes, sample space, and events. Combinatorial probability and the birthday paradox. Principle of inclusion and exclusion. Conditional probability, independence, and Bayes' theorem.

Unit 2: Univariate Probability Distributions

Introduction to random variables with discrete and continuous probability models. Overview of key probability distributions: Binomial, Poisson, Geometric, Uniform, Normal, and Exponential distributions.

Unit 3: Bivariate Probability Distributions

Understanding bivariate random variables and joint probability distributions (pmf and pdf). Cumulative distribution functions, marginal probability distributions (marginal pmf and marginal pdf), and conditional distribution functions. Independence of random variables, conditional mean, and conditional variance.

Unit 4: Stochastic Processes

Introduction to Markov chains, classification of states, stationary distribution, and limit theorems. Poisson process and its applications with relevant illustrations.

References

1. Sheldon M. Ross, *Introduction to Probability Models*, Academic Press, 12th Edition, 2019.
2. Hossein Pishro-Nik, *Introduction to Probability, Statistics, and Random Processes*, Kappa Research, 2014.
3. Oliver C. Ibe, *Markov Processes for Stochastic Modeling*, Elsevier, 2nd Edition, 2013.

Semester I

CORE Paper: Data Structures and Algorithms
Course Code: PAI-1806
No. of Credits: 04 (3 Theory + 1 Lab)
Learning Hours: 75 (45 Theory + 30 Lab)

Course Outcomes (COs)

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

1. Understand fundamental data structures, their types, and algorithmic complexity analysis.
2. Implement linear data structures such as stacks, queues, and linked lists for problem-solving.
3. Apply tree, heap, and hashing techniques for efficient data organization and retrieval.
4. Design and analyze algorithms using greedy, divide and conquer, and dynamic programming approaches.

Course Structure

Unit 1: Fundamentals of Data Structures and Algorithm Analysis

Definition and importance of data structures. Types of data structures: Linear and Non-Linear. Algorithm complexity analysis: Big-$\mathcal{O}$, Big-Ω, Big-Θ notations. Introduction to arrays: one-dimensional and two-dimensional arrays, dynamic arrays, advantages, and limitations.

Unit 2: Linear Data Structures – Stacks, Queues, and Linked Lists

Stack operations: Push, Pop, Peek; applications such as expression evaluation and backtracking. Queues: FIFO, Circular Queue, and Deque (Double-Ended Queue). Linked Lists: singly, doubly, and circular linked lists with operations (insert, delete, traverse, reverse). Memory management in linked lists.

Unit 3: Trees, Heaps, and Hashing

Introduction to trees: binary trees and binary search trees (BST). Tree traversal techniques: inorder, preorder, postorder. Heap structures: min-heap and max-heap (priority queue implementation). Hashing: hash tables, hash functions, and collision resolution techniques (chaining, open addressing).

Unit 4: Algorithmic Techniques – Greedy, Divide & Conquer, and Dynamic Programming

Greedy algorithms: activity selection, Huffman coding, Kruskal's algorithm. Divide and conquer: merge sort, quick sort, matrix multiplication. Dynamic programming: Fibonacci sequence, 0/1 knapsack problem, and longest common subsequence (LCS).

References

1. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, *Introduction to Algorithms*, 4th Edition, MIT Press, 2022.

2. Mark Allen Weiss, *Data Structures and Algorithm Analysis in C++*, Pearson, 4th Edition, 2014.

3. Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman, *Data Structures and Algorithms*, Pearson, 1983.

Semester II

CORE Paper: Knowledge Representation and Reasoning
Course Code: PAI-2801
No. of Credits: 04 (3 Theory + 1 Lab)
Learning Hours: 75 (45 Theory + 30 Lab)

Course Outcomes (COs)

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

1. Apply rule-based systems and problem decomposition using And–Or graphs for structured knowledge representation.
2. Understand automated planning paradigms including STRIPS, Graphplan, and SAT-based approaches.
3. Perform logical deduction using propositional and first-order logic through methods like forward chaining, backward chaining, and resolution.
4. Explore advanced logical frameworks such as description logics, non-monotonic reasoning, and epistemic logic for intelligent system design.

Course Structure

Unit 1: Problem Decomposition and Rule-Based Systems

Pattern-directed inference systems and rule-based production systems. Working memory in OPS5, patterns, actions, inference engine. Conflict resolution strategies, Rete networks, and the Rete algorithm. Problem decomposition with And–Or graphs. Case studies: DENDRAL, symbolic integration, AO* algorithm.

Unit 2: Automated Planning

Representations of time and change in planning domains. STRIPS and the blocks world domain. State space planning: forward and backward planning, heuristic methods. Goal stack planning: linear planning, Sussman anomaly. Partial-order planning and applications (two-armed robot). Graphplan algorithm: planning graphs, heuristic derivation, solution extraction. Planning as satisfiability: direct encoding, planning graphs as SAT. Richer planning domains: durative actions, metric domains, conditional effects, contingent planning, multi-agent coordination, and epistemic planning.

Unit 3: Deduction as Search (Propositional & FOL)

Logical connectives and truth tables. Entailment, proof methods, soundness, completeness. First Order Logic (FOL): terms, atomic formulas, quantifiers, sentences. Deduction in FOL: implicit quantifier form, unification, forward chaining, backward chaining, Prolog basics. Incompleteness of forward/backward chaining, resolution refutation method, equality in FOL.

Unit 4: Advanced Logics and Reasoning Frameworks

Horn clause logic and Prolog extensions. Description logics and OWL foundations. Default reasoning and non-monotonic logic. Event calculus, epistemic logic, and reasoning about knowledge. Applications in knowledge representation and intelligent systems.

References

1. Brachman, R. J., & Levesque, H. J., *Knowledge Representation and Reasoning*, Morgan Kaufmann, 2004.

2. Michael Gelfond & Yulia Kahl, *Knowledge Representation, Reasoning, and the Design of Intelligent Agents*, Cambridge University Press, 2014.

3. Stuart Russell & Peter Norvig, *Artificial Intelligence: A Modern Approach*, 4th Edition, Pearson, 2024.

Semester II

CORE Paper: Introduction to Machine Learning
Course Code: PAI-2802
No. of Credits: 04 (3 Theory + 1 Lab)
Learning Hours: 75 (45 Theory + 30 Lab)

Course Outcomes (COs)

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

1. Explain the foundations of machine learning including model evaluation, bias-variance tradeoff, and dimensionality reduction.
2. Apply classification algorithms such as k-NN, decision trees, logistic regression, and naïve Bayes to real-world datasets.
3. Implement advanced ML models including support vector machines and neural networks for prediction and classification tasks.
4. Understand the theoretical foundations of learning and ensemble techniques such as bagging, boosting, and random forests.

Course Structure

Unit 1: Foundations of Machine Learning

Basic definitions: supervised, unsupervised, and reinforcement learning. Hypothesis space and inductive bias. Model evaluation, train-test splits, cross-validation. Overfitting and underfitting, bias-variance trade-off. Linear regression: normal equations, ordinary least squares, batch, stochastic, and mini-batch gradient descent, probabilistic interpretation. Feature reduction methods: PCA (eigen decomposition, SVD, applications in face recognition), LDA (linear discriminant analysis, Fisher's criterion), introduction to SVD and feature selection.

Unit 2: Classification and Dimensionality Reduction

Instance-based learning: k-NN and various distance metrics. Decision trees: ID3, CART, pruning. Collaborative filtering and recommendation systems. Logistic regression: decision boundary, maximum likelihood estimation. Naïve Bayes: Gaussian, multinomial, Laplace smoothing, handling imbalanced data.

Unit 3: Advanced Models

Support Vector Machines: hard margin, soft margin, slack variables, dual problem, kernel trick (polynomial, RBF, Gaussian), SMO algorithm, multiclass classification. Neural Networks: perceptron, activation functions, multi-layer perceptron, XOR problem, backpropagation, training strategies (batch, mini-batch, stochastic GD), introduction to deep neural networks.

Unit 4: Learning Theory and Ensemble Learning

Computational learning theory, PAC learning model, sample complexity, VC dimension. Ensemble learning: bagging, boosting, random forests.

References

1. Tom M. Mitchell, *Machine Learning*, McGraw Hill, 1997.
2. Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.
3. Kevin P. Murphy, *Machine Learning: A Probabilistic Perspective*, MIT Press, 2012.

Semester II

CORE Paper: Introduction to Computer Vision and Image Processing
Course Code: PAI-2803
No. of Credits: 04 (4 Theory)
Learning Hours: 60 (60 Theory)

Course Outcomes (COs)

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

1. Explain the fundamentals of image formation, radiometry, and geometric transformations in computer vision.
2. Apply core image processing techniques such as filtering, enhancement, and segmentation.
3. Extract and analyze visual features and descriptors for image understanding and object recognition.
4. Implement computer vision applications using neural networks, gesture recognition, motion estimation, and object tracking.

Course Structure

Unit 1: Introduction to Computer Vision and Image Formation

Introduction and goals of computer vision and image processing. Image formation concepts, radiometry, geometric transformations, geometric camera models, camera calibration, image formation in a stereo vision setup, and image reconstruction from a series of projections.

Unit 2: Image Processing Concepts

Image transforms, image enhancement, image filtering, color image processing, and image segmentation.

Unit 3: Image Descriptors and Features

Texture descriptors, color features, edges and boundaries, object boundary and shape representations, interest or corner point detectors, histogram of oriented gradients (HOG), scale invariant feature transform (SIFT), speeded-up robust features (SURF), and saliency.

Unit 4: Applications of Computer Vision

Artificial neural networks for pattern classification, convolutional neural networks, autoencoders, gesture recognition, motion estimation, and object tracking. Programming assignments and implementation exercises.

References

1. M. K. Bhuyan, *Computer Vision and Image Processing: Fundamentals and Applications*, CRC Press, 2019.

2. Richard Szeliski, *Computer Vision: Algorithms and Applications*, Springer, 2010.

3. Forsyth & Ponce, *Computer Vision: A Modern Approach*, Pearson Education, 2003.

Semester II

CORE Paper: Introduction to Natural Language Processing
Course Code: PAI-2804
No. of Credits: 04 (4 Theory)
Learning Hours: 60 (60 Theory)

Course Outcomes (COs)

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

1. Understand the fundamental principles of NLP, including finite state methods, morphology, and probabilistic models for text analysis.
2. Apply syntactic and parsing techniques such as part-of-speech tagging, context-free grammars, and probabilistic parsing for sentence structure analysis.
3. Implement semantic and information extraction techniques including entity linking, semantic analysis, and representation of meaning.
4. Develop NLP applications such as text classification, sentiment analysis, and machine translation using modern language models and generation techniques.

Course Structure

Unit 1: Foundations and Basic Text Processing

Introduction to NLP, regular expressions, finite state automata, morphology and finite state transducers, probabilistic models and N-gram models, spelling correction techniques.

Unit 2: Syntax and Parsing

Word classes and part-of-speech tagging, context-free grammars for English, parsing with context-free grammar, syntax features and unification, lexicalized and probabilistic parsing, constituency parsing.

Unit 3: Semantics and Information Extraction

Representing meaning and meaning structure, first-order predicate calculus, syntax-driven semantic analysis, semantic attachments, robust analysis, lexical semantics, entity linking, information extraction.

Unit 4: Applications and Analysis

Text summarization, text classification, sentiment analysis and opinion mining, dependency parsing, distributional semantics, topic models, natural language generation, and machine translation.

References

1. Daniel Jurafsky & James H. Martin, *Speech and Language Processing*, Pearson Education, 3rd Edition, 2023.

2. Christopher D. Manning & Hinrich Schütze, *Foundations of Statistical Natural Language Processing*, MIT Press, 1999.

3. Steven Bird, Ewan Klein, & Edward Loper, *Natural Language Processing with Python*, O'Reilly Media, 2009.

Semester II

CORE Paper: Machine Learning Lab
Course Code: PAI-2805L
No. of Credits: 02 (Laboratory)
Learning Hours: 60 (Lab-Based)

Lab Outcomes (LOs)

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

1. Perform data preprocessing tasks including handling missing values, outliers, encoding, scaling, and cross-validation.
2. Implement and evaluate core supervised learning algorithms such as linear regression, logistic regression, k-NN, and decision trees from scratch and using Python libraries.
3. Apply advanced ML techniques such as SVMs, perceptrons, and neural networks with backpropagation for predictive modeling.
4. Analyze learning theory concepts through experiments on bias-variance trade-off, PAC learning, and ensemble methods.

Course Structure

Unit 1: Foundations of ML & Data Preprocessing

Handling missing values (imputation, deletion, advanced methods). Outlier detection and treatment (IQR, z-score, isolation forest). Feature scaling methods: Min-Max, standardization, robust scaling. Encoding categorical variables: one-hot, label encoding, target encoding. Train-test split, k-fold cross-validation, and stratified sampling. Bias-variance analysis with synthetic datasets. Linear regression (normal equation and OLS) from scratch. Gradient descent (batch, stochastic, mini-batch) for regression (from scratch and using in-built functions). PCA with eigen decomposition and SVD – visualization in 2D. LDA for dimensionality reduction and Fisher's criterion. Face recognition using PCA (Olivetti dataset).

Unit 2: Classification & Dimensionality Reduction

Logistic regression using sklearn and from scratch. k-NN classification and regression (with different distance metrics). Decision tree classifier and regressor with pruning. Naïve Bayes (Gaussian and multinomial) on text and tabular data. Recommender system with collaborative filtering (MovieLens dataset).

Unit 3: Advanced Models

Support Vector Machines: hard-margin vs. soft-margin (code from scratch, code using sklearn, and tuning hyperparameters). SVM with kernel trick (RBF, polynomial, Gaussian) on synthetic data. SMO algorithm (implement simple version). Perceptron learning rule on linearly separable data. XOR problem: failure of perceptron, solution with MLP. Backpropagation implementation for MLP. Training strategies comparison: batch vs. mini-batch vs. SGD.

Unit 4: Learning Theory & Ensemble Models

PAC learning simulation: sample complexity vs. accuracy. VC dimension estimation for linear classifiers. Bagging with decision trees. Boosting (AdaBoost, gradient boosting). Random forest implementation and feature importance. Model comparison report: linear, logistic, SVM, NN, ensemble on one dataset.

References

1. Aurélien Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, O'Reilly Media, 3rd Edition, 2023.

2. Sebastian Raschka & Vahid Mirjalili, *Python Machine Learning*, Packt Publishing, 3rd Edition, 2019.

3. Andreas C. Müller & Sarah Guido, *Introduction to Machine Learning with Python*, O'Reilly Media, 2016.

Semester II

CORE Paper: Computer Vision and Natural Language Processing Lab
Course Code: PAI-2806L
No. of Credits: 04 (Laboratory)
Learning Hours: 120 (Lab-Based)

Lab Outcomes (LOs)

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

1. Apply fundamental computer vision techniques for image formation, transformation, and processing using OpenCV and deep learning tools.
2. Implement advanced feature extraction and object recognition techniques using CNNs, autoencoders, and tracking systems.
3. Develop NLP pipelines for text preprocessing, parsing, semantic analysis, and classification using classical and transformer-based models.
4. Apply machine learning and deep learning methods to solve vision and language tasks through end-to-end implementation.

Course Structure

Unit 1: Computer Vision Fundamentals and Image Processing

1.	Reading, displaying, saving images	OpenCV basics
2.	Converting RGB ↔ GRAY ↔ HSV	cv2.cvtColor
3.	Drawing shapes and text on images	cv2.line, rectangle, putText
4.	Image negative, brightness & contrast adjustment	Pixel operations
5.	Geometric transforms – translation, rotation, scaling	cv2.warpAffine
6.	Affine and perspective transforms	cv2.warpPerspective
7.	Camera calibration using chessboard images	cv2.calibrateCamera
8.	Stereo image disparity & depth mapping	cv2.StereoBM
9.	Image reconstruction from multiple projections	Sinogram simulation
10.	DFT & frequency domain filtering	np.fft, cv2.dft
11.	DCT transform & compression	cv2.dct
12.	Histogram equalization & CLAHE	cv2.equalizeHist
13.	Smoothing filters (mean, Gaussian, median)	cv2.filter2D
14.	Edge detection (Sobel, Canny)	cv2.Canny
15.	Colour image processing & masking	cv2.inRange, bitwise operations
16.	Thresholding & contour detection	cv2.threshold, findContours
17.	Region-based segmentation (Watershed)	cv2.watershed
18.	K-means clustering segmentation	cv2.kmeans
19.	Texture analysis using GLCM & LBP	skimage.feature
20.	Shape descriptors (Hu moments, contour features)	cv2.moments

Unit 2: Feature Detection, Deep Models, and Vision Applications

21.	Harris & FAST corner detection	cv2.cornerHarris
22.	HOG feature extraction	skimage.hog
23.	SIFT/SURF feature matching	cv2.SIFT_create
24.	Saliency detection	cv2.saliency
25.	Dimensionality reduction (PCA on image dataset)	sklearn.decomposition
26.	Basic image classifier using ANN	keras Sequential
27.	CNN for MNIST/CIFAR dataset	keras CNN
28.	Autoencoder-based image reconstruction	keras Autoencoder
29.	Motion estimation using optical flow (Farneback)	cv2.calcOpticalFlow
30.	Object tracking (Mean-Shift / Kalman filter)	cv2.Tracker

Unit 3: Natural Language Processing – Text Processing and Syntax Analysis

1.	Tokenization of sentences and words	nltk, spacy
2.	Stopword removal and stemming	nltk.corpus, PorterStemmer
3.	Lemmatization using spaCy	spacy.lemmatizer
4.	Frequency distribution of words	nltk.FreqDist
5.	POS tagging and visualization	nltk.pos_tag, spacy
6.	Named Entity Recognition (NER)	spacy, stanza
7.	Text normalization and cleaning pipeline	regex, string ops
8.	N-gram model creation	nltk.ngrams
9.	Simple spell corrector	edit distance, symspell
10.	Bag-of-Words and TF-IDF representation	sklearn.feature_extraction
11.	Text similarity (cosine, Jaccard)	sklearn.metrics.pairwise
12.	Text classification using Naïve Bayes	sklearn.naive_bayes
13.	Logistic regression for sentiment classification	sklearn.linear_model
14.	Decision tree and random forest classifiers	sklearn.tree, ensemble
15.	Topic modeling with LDA	gensim.models.LdaModel

Unit 4: Natural Language Processing – Semantics, Deep Learning, and Applications

16.	Word embeddings using Word2Vec	gensim.models.Word2Vec
17.	Visualizing word vectors using PCA/t-SNE	sklearn.decomposition
18.	Document similarity using embeddings	cosine similarity
19.	Dependency parsing visualization	spacy.displacy
20.	Coreference resolution	spacy.neuralcoref
21.	Information extraction using regex and POS	regex + POS tags
22.	Named Entity Linking	spaCy, DBpedia Spotlight API
23.	Sentiment analysis using pre-trained models	transformers, HuggingFace
24.	Text summarization (extractive)	sumy, gensim
25.	Abstractive summarization (transformer-based)	HuggingFace Transformers
26.	Machine translation (English–Hindi)	transformers MarianMT
27.	Chatbot with rule-based intent classification	regex, nltk
28.	Simple language generation using GPT API	openai / transformers
29.	Text clustering using K-means	sklearn.cluster
30.	End-to-end NLP pipeline (Preprocessing → Model → Evaluation)	sklearn, spacy, transformers

---

References

1. Abhinav Dadhich, *Practical Machine Learning for Computer Vision*, Packt Publishing, 2022.

2. Martin Görner, Ryan Gillard, & Valliappa Lakshmanan, *Practical Machine Learning for Computer Vision*, O'Reilly Media, 2023.

3. Steven Bird, Ewan Klein, & Edward Loper, *Natural Language Processing with Python*, O'Reilly Media, 2009.

Semester II

CORE Paper: Ethics and Value Thinking
Course Code: PAI-2807
No. of Credits: 02 (1 Theory + 1 Tutorial)
Learning Hours: 45

Course Outcomes (COs)

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

1. Explore the foundations of ethics and moral reasoning through diverse cultural and philosophical perspectives.
2. Analyze human values, dilemmas, and decision-making through literary and cinematic representations.
3. Reflect critically on contemporary issues of science, technology, and society using ethical frameworks.
4. Cultivate empathy, self-awareness, and reflective thinking through dialogue, discussion, and experiential learning.

Course Structure

This is an open and thematic course designed to engage students with questions of ethics, human values, and critical reasoning through selected films and literary works. The learning process emphasizes reflection, dialogue, and moral inquiry rather than formal instruction. Discussions are centered around themes such as justice, truth, human dignity, moral conflict, and the relationship between knowledge and responsibility.

Learning Resources

Films

1. *Twelve Angry Men* (1957)
2. *Rashomon* – Akira Kurosawa
3. *Judgment at Nuremberg* (1961)
4. *The Mahabharata* – Peter Brook
5. *Jurassic Park* (1993)
6. *Interstellar* (2014)

Books

1. Arthur Conan Doyle, *The Hound of the Baskervilles*
2. Agatha Christie, *Five Little Pigs*
3. Edgar Allan Poe, *The Purloined Letter*
4. Erle Stanley Gardner, *The Case of the Substitute Face*
5. Isaac Asimov, *The Caves of Steel*
6. Ray Bradbury, *Fahrenheit 451*

Semester III

CORE Paper: Sequential Decision Making and Reinforcement Learning
Course Code: PAI-3801
No. of Credits: 04 (4 Theory)
Learning Hours: 60 (60 Theory)

Course Outcomes (COs)

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

1. Understand the foundations of sequential decision-making and explore the multi-armed bandit framework for balancing exploration and exploitation.
2. Formulate reinforcement learning tasks as Markov Decision Processes (MDPs) and apply dynamic programming algorithms for prediction and control.
3. Implement Monte Carlo and Temporal Difference (TD) methods for model-free learning and control in tabular settings.
4. Apply function approximation methods—both linear and non-linear—to estimate value functions and extend reinforcement learning to large or continuous state spaces.

Course Structure

Unit 1: Introduction and Multi-Armed Bandits

Philosophy of sequential decision-making; real-world motivation for reinforcement learning; reward-based learning; exploration vs. exploitation dilemma; k-armed bandit problem; action-value estimation; incremental implementation; handling non-stationary problems; optimistic initialization; upper-confidence bound (UCB) action selection; gradient bandit algorithms; contextual (associative) bandits.

Unit 2: Markov Decision Processes and Dynamic Programming

Agent–environment interaction framework; goals and rewards; returns and episodic tasks; policies and value functions; optimal policies and Bellman optimality equations; value iteration and policy iteration; policy evaluation and improvement; asynchronous and generalized policy iteration; efficiency and limitations of dynamic programming.

Unit 3: Monte Carlo and Temporal Difference Methods

Monte Carlo prediction and control; on-policy and off-policy learning; importance sampling; incremental implementation; TD prediction (TD(0)); on-policy TD control (SARSA); off-policy TD control (Q-learning); Expected SARSA; maximization bias and double Q-learning, comparison of MC and TD approaches.

Unit 4: Function Approximation in Reinforcement Learning

Value-function approximation; prediction objectives (VE); stochastic and semi-gradient methods; linear function approximation using polynomial, Fourier, coarse coding, tile coding, and radial basis function features; step-size selection; non-linear function approximation with artificial neural networks; least-squares TD and kernel-based methods; stability and generalization in approximate reinforcement learning.

References

1. Richard S. Sutton & Andrew G. Barto, *Reinforcement Learning: An Introduction* (2nd ed.), MIT Press, 2018.

2. Csaba Szepesvári, *Algorithms for Reinforcement Learning*, Morgan & Claypool, 2010.

3. Dimitri P. Bertsekas & John N. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, 1996.

Semester III

CORE Paper: Unsupervised Learning and Deep Learning
Course Code: PAI-3802
No. of Credits: 04 (4 Theory)
Learning Hours: 60 (60 Theory)

Course Outcomes (COs)

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

1. Apply key clustering and density estimation algorithms for unsupervised learning, including k-means, hierarchical, and Gaussian mixture models.
2. Design, train, and optimize feedforward and convolutional neural networks (CNNs) using modern architectures and regularization techniques.
3. Implement recurrent and sequence modeling networks such as RNNs, LSTMs, and attention-based encoder-decoder frameworks for temporal or sequential data.
4. Understand and develop generative deep learning models such as autoencoders, VAEs, and GANs to learn data representations and generate new samples.

Course Structure

Unit 1: Unsupervised Learning

Partitional clustering: k-means and k-medoids; hierarchical clustering—agglomerative and divisive methods; density-based clustering (DBSCAN); Gaussian Mixture Models (GMMs); Expectation–Maximization (EM) algorithm; comparison of clustering paradigms; model evaluation and selection for unsupervised learning.

Unit 2: Convolutional Neural Networks (CNNs) and Optimization

Introduction to deep learning; perceptrons and gradient descent; multilayer perceptrons (MLPs); back-propagation and feedforward networks; convolutional architectures (LeNet, AlexNet, VGGNet, ResNet); optimization algorithms—momentum, RMSProp, Adam; regularization techniques—dropout, batch normalization; visualization and guided backpropagation for interpretability.

Unit 3: Recurrent Neural Networks (RNNs) and Applications

Foundations of recurrent networks; challenges of vanishing and exploding gradients; Long Short-Term Memory (LSTM) networks; encoder–decoder architectures; attention mechanisms; sequence-to-sequence learning; applications to natural language and vision—semantic segmentation, object detection, image denoising.

Unit 4: Generative Models and Beyond

Representation learning using autoencoders and PCA; variational autoencoders (VAEs) for probabilistic generation; generative adversarial networks (GANs) and adversarial training dynamics; evaluation of generative models; emerging trends in self-supervised, multimodal, and foundation models; ethical considerations and future directions in deep learning.

References

1. Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, 2016.
2. Christopher Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.
3. Aurélien Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, O'Reilly Media, 2022.

Semester III

CORE Paper: Generative AI and Large Language Models
Course Code: PAI-3803
No. of Credits: 04 (4 Theory)
Learning Hours: 60 Theory

Course Outcomes (COs)

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

1. Understand the evolution, structure, and foundational principles of generative AI and large language models (LLMs).
2. Analyze and implement transformer architectures and apply fine-tuning and post-training methods such as RLHF, SFT, and DPO.
3. Explore practical applications of LLMs across domains, leveraging frameworks like Hugging Face for contextual embeddings, optimization, and evaluation.
4. Critically examine the ethical, social, and economic implications of LLMs, including responsible deployment and regulatory frameworks.

Course Structure

Unit 1: Introduction to Generative AI and Large Language Models

Overview and history of Generative AI; definition, key terminologies, and evolution of language models; key architectures—GPT, BERT, T5; representation learning and word embeddings; from word vectors to sequence models; RNNs and the shift to transformer-based models; early LLM development milestones.

Unit 2: Transformer Architecture and Model Training

Review of RNN limitations and advanced variants (LSTM, GRU); the "Attention Is All You Need" paradigm; transformer architecture—self-attention, multi-head attention, positional encoding; training and scaling of large language models; pretraining objectives (masked language modeling, causal LM); fine-tuning and post-training techniques—Supervised Fine-Tuning (SFT), Reinforcement Learning from Human Feedback (RLHF), Direct Preference Optimization (DPO); basics of reinforcement learning as applied to LLMs.

Unit 3: Applications and Advanced Techniques

Practical applications of LLMs in text generation, summarization, reasoning, and dialogue; implementation using Hugging Face Transformers; contextual embeddings and long-range dependency handling; prompt engineering, in-context learning—zero-shot, one-shot, and few-shot paradigms; model compression, distillation, and optimization; evaluation and benchmarking of LLM performance.

Unit 4: Ethical and Societal Implications

Retrieval-Augmented Generation (RAG) and hybrid reasoning frameworks; incentivizing reasoning and factual grounding via reinforcement learning; ethical considerations—bias, misinformation, privacy; societal and economic impacts of LLM deployment; AI alignment, accountability, and interpretability; global regulatory initiatives; real-world case studies and classroom debates on responsible AI.

References

1. Lewis Tunstall, Leandro von Werra, Thomas Wolf, *Natural Language Processing with Transformers*, O'Reilly Media, 2022.

2. Andrej Karpathy et al., *Deep Learning and the Transformer Revolution*, MIT OpenCourse Notes, 2023.

3. Sebastian Raschka, Yuxi (Hayden) Liu, *Build a Large Language Model (From Scratch)*, O'Reilly Media, 2024.

Semester III

CORE Paper: Time Series and Forecasting
Course Code: PAI-3804
No. of Credits: 04 (3 Theory + 1 Lab)
Learning Hours: 60 (30 Theory + 30 Lab)

Course Outcomes (COs)

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

1. Differentiate between stationary and non-stationary time series and understand their implications in modeling.
2. Develop and analyze AR, MA, ARMA, and ARIMA models for univariate time series data.
3. Apply statistical tests for non-stationarity and implement forecasting techniques with error analysis.
4. Model seasonal variations and handle missing data in practical time series forecasting tasks.

Course Structure

Unit 1: Stationary and Non-Stationary Time Series

Definition and classification of time series; characteristics and properties of stationary processes; trend and seasonality; concept of ergodicity; transformation of non-stationary to stationary series through differencing and detrending; visualization and exploratory analysis of time series data.

Unit 2: Time Series Models

Autoregressive (AR) models; Moving Average (MA) models; Autoregressive Moving Average (ARMA) models; Autoregressive Integrated Moving Average (ARIMA) models; model identification using Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF); estimation of model parameters and diagnostics; model adequacy checking and residual analysis.

Unit 3: Non-Stationarity and Forecasting

Tests for non-stationarity: Unit Root Tests (ADF, KPSS); concept of cointegration and long-term equilibrium; forecasting with ARIMA and exponential smoothing methods; Minimum Mean Squared Error (MSE) forecast; forecast intervals and uncertainty quantification; evaluation of forecasting accuracy and model comparison.

Unit 4: Seasonal Time Series and Missing Data

Modeling seasonal patterns using Seasonal ARIMA (SARIMA) and decomposition methods; trend–cycle–season decomposition; exponential smoothing with seasonality; dealing with missing data—imputation, interpolation, and smoothing approaches; case studies on real-world forecasting problems using Python or R.

References

1. George E. P. Box, Gwilym M. Jenkins, Gregory C. Reinsel, Greta M. Ljung, *Time Series Analysis: Forecasting and Control*, Wiley, 2016.

2. Robert H. Shumway, David S. Stoffer, *Time Series Analysis and Its Applications: With R Examples*, Springer, 2017.

3. Chris Chatfield, *The Analysis of Time Series: An Introduction*, Chapman and Hall/CRC, 2016.

Semester III

CORE Paper: Reinforcement Learning and Deep Learning Lab
Course Code: PAI-3805L
No. of Credits: 04 (8 Hours/Week)
Learning Hours: 120 (60 RL + 60 DL)

Course Outcomes (COs)

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

1. Implement and analyze classical reinforcement learning algorithms using tabular and approximate methods.
2. Design and conduct experiments to evaluate exploration–exploitation trade-offs and policy optimization.
3. Build and train deep neural networks using modern frameworks for vision and sequence data.
4. Integrate reinforcement and deep learning approaches for control, perception, and representation tasks.
5. Evaluate model performance, convergence, and stability through experimentation and visualization.

Course Structure

Unit 1: Classical and Multi-Armed Bandit Algorithms (Reinforcement Learning – Part 1)

1. Introduction to RL environments (Gridworld, Bandit setup) — OpenAI Gym basics.
2. K-armed Bandit implementation (ϵ-greedy, softmax, optimistic initialization).
3. UCB (Upper Confidence Bound) exploration strategy.
4. Gradient Bandit algorithm and preference updates.
5. Contextual Bandits — simulation of state-dependent arms.
6. Markov Decision Process (MDP) formulation for discrete tasks.
7. State transition probabilities, rewards, and return estimation.
8. Policy evaluation using Monte Carlo sampling.
9. Value Iteration and Policy Iteration methods.
10. Dynamic Programming for prediction and control (evaluation vs improvement).
11. Policy visualization and convergence analysis.

Unit 2: Temporal-Difference and Control Algorithms (Reinforcement Learning – Part 2)

12. Monte Carlo control without exploring starts.
13. TD(0) prediction and incremental updates.
14. SARSA (On-policy TD control).
15. Q-Learning (Off-policy TD control).
16. Expected SARSA.

17. Double Q-Learning to reduce maximization bias.
18. N-step TD and TD(λ) (Lambda-returns).
19. Episodic vs continuous tasks — discounted returns.
20. Eligibility traces and backward view implementation.
21. Function approximation using linear architectures (Fourier, Polynomial, RBF).
22. Non-linear approximation using neural networks.
23. On-policy prediction with approximation (semi-gradient methods).
24. Least Squares TD and Gradient TD methods.
25. Deep Q-Networks (DQN): experience replay and target network.
26. Policy Gradient methods (REINFORCE algorithm).
27. Actor–Critic architectures.
28. Advantage Actor–Critic (A2C / A3C).
29. Proximal Policy Optimization (PPO).
30. Comparative study of tabular vs function-approximation RL methods.

Unit 3: Deep Neural Networks and CNN Architectures (Deep Learning – Part 1)

1. Introduction to Keras / PyTorch frameworks.
2. Building feed-forward neural networks for classification tasks.
3. Forward propagation, backpropagation, and loss functions.
4. Optimization algorithms (SGD, Momentum, RMSProp, Adam).
5. Regularization techniques (dropout, L2, batch normalization).
6. Activation functions (ReLU, Sigmoid, Tanh, Leaky ReLU).
7. Convolutional Neural Networks (CNN): basics of filters and kernels.
8. Pooling operations (max pool, average pool).
9. Architectures: LeNet, AlexNet, VGG, ResNet.
10. Transfer learning using pretrained models (VGG, ResNet, MobileNet).
11. Visualization of feature maps and filters.
12. Image augmentation techniques (OpenCV + Keras).
13. CNN for image classification (MNIST, CIFAR-10).
14. Object detection basics (R-CNN overview).
15. Autoencoder basics – encoding and reconstruction.

Unit 4: Recurrent, Generative, and Hybrid Deep Models (Deep Learning – Part 2)

16. Recurrent Neural Networks (RNNs) and sequence data.
17. LSTM and GRU architectures for time-series modeling.
18. Encoder–Decoder framework for sequence-to-sequence tasks.
19. Attention mechanism and self-attention overview.
20. Generative Adversarial Networks (GANs): generator and discriminator.
21. Conditional GAN and applications.
22. Variational Autoencoders (VAEs) – latent representation learning.
23. CNN for semantic segmentation (U-Net).
24. Image denoising using autoencoders.
25. Deep Reinforcement Learning integration (DQN + CNN).
26. Visual policy learning and feature embedding from raw pixels.
27. Model comparison: CNN vs RNN vs Hybrid architectures.
28. Performance metrics and confusion matrices.
29. Hyperparameter tuning and early stopping strategies.
30. Capstone Project – Integrating RL and DL for control or vision tasks.

References

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

2. Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, 2016.

3. Maxim Lapan, *Deep Reinforcement Learning Hands-On*, Packt Publishing, 2020.

Semester III

CORE Paper: AI and Entrepreneurship
Course Code: PAI-3806L
No. of Credits: 04 (3 Theory + 1 Lab)
Learning Hours: 75 (45 Theory + 30 Lab)

Course Outcomes (COs)

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

1. Apply design thinking, business model frameworks, and product development strategies to real-world challenges.
2. Collaborate effectively in teams to manage operations, marketing, finance, and innovation cycles.
3. Present and defend a complete entrepreneurial venture through reports, pitches, and prototypes.

Course Structure

This course bridges artificial intelligence and entrepreneurship through hands-on experience. Each student will become part of a start-up team, register their enterprise on the **Udyam** portal, and develop a minimum viable product (MVP) or AI-enabled solution.

Students will work through key entrepreneurial stages — ideation, validation, product development, financial planning, marketing strategy, and impact assessment. The lab component will include practical sessions on AI tools, data visualization, pitch design, and prototype creation.

At the end of the course, every team will present its start-up progress report, financial summary, and business pitch for evaluation.

Evaluation and Marking System

			Internal		External		
Code	Title	Grading will be based on	20%	25%	15%	40%	Bonus Marks
PAI 3806L	**AI and Entrepreneurship**	• Presentation of Case Study • A functional MVP with a demonstration of its AI capabilities.	Case Study Presentation	Internal pitch presentation	Business model development, customer validation, and problem formulation	MVP + Pitch presentation (evaluated by industry experts and college professors).	• Publish a Paper on product. • Sell a Product in the real market.

References

1. Eric Ries, *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*, Crown Business, 2011.

2. Andrew Ng, *AI Transformation Playbook: How to Lead Your Company into the AI Era*, Deeplearning.ai, 2022.

3. Peter Thiel, *Zero to One: Notes on Startups, or How to Build the Future*, Crown Business, 2014.

Semester IV

CORE Paper: Internship
Course Code: PAI-4801
No. of Credits: 20
Learning Hours: 20 Weeks (Approx. 600 Hours)

Course Outcomes (COs)

By the end of this internship, students will be able to:

1. Apply theoretical and practical knowledge of AI and data science to real-world industrial problems.
2. Collaborate with interdisciplinary teams to design, implement, and evaluate AI-driven solutions.
3. Develop professional skills in project management, documentation, and communication.
4. Demonstrate critical thinking and ethical decision-making in AI deployment.
5. Present findings and insights effectively through a technical report and viva-voce.

Course Structure

Students will undertake a **20-week industry internship** as part of their M.Sc. Artificial Intelligence program. Each student will be assigned two supervisors — one from the **academic institution** and one from the **industry partner**.

During this period, students are expected to:

- Identify a real-world AI problem relevant to the host organization.
- Design and implement data collection, analysis, and model development pipelines.
- Document methodologies, results, and recommendations in a professional project report.
- Present outcomes in an open seminar at the end of the internship.

The internship experience integrates academic learning with professional application, fostering research, innovation, and industry readiness.

Evaluation Scheme

Component	**Marks (Total: 500)**
Report from Academic Supervisor	100
Report from Industry Supervisor	100
Final Project Report	100
Grand Viva-Voce Examination	100
Final Presentation (Open Evaluation)	100
