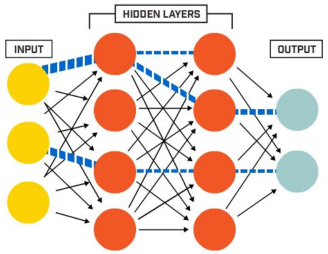
ARTIFICIAL NEURAL NETWORK AND DEEP LEARNING
21AI63



Dr S Anupama Kumar
Associate Professor
Dept of AIML
RVCE

Semester: VI						
ARTIFICIAL NEURAL NETWORK AND DEEP LEARNING **Category: Professional Core Course** **(Theory & Lab)**						
Course Code	:	**21AI63**		**CIE Marks**	:	**100+50 Marks**
Credits: L: T:P	:	**3:0:1**		**SEE Marks**	:	**100+50 Marks**
Total Hours	:	**45L+30P**		**SEE Duration**	:	**3 Hours**

Course Outcomes: After completing the course, the students will be able to:-	
CO1	Describe basic concepts of neural network, its applications and various learning models
CO2	Analyze different Network Architectures, learning tasks, convolutional networks, and deep learning models
CO3	Investigate and apply neural networks model and learning techniques to solve problems related to society and industry
CO4	Demonstrate a prototype application developed using any NN tools and APIs
CO5	Appraise the knowledge of Neural Networks and Deep Learning as an Individual /as a team member

Unit – I	9 Hrs
Neural Networks: **Introduction:** What is a Neural Network? Models of a Neuron, Network Architectures **Learning Processes:** Error-correction learning, memory-based learning, Hebbian learning, Competitive learning and Boltzmann learning, Learning with a teacher, Learning without a teacher, Learning tasks, Memory and adaptation. Statistical Learning Theory, VC dimension	
Unit – II	**9 Hrs**
Single-layer Perceptron: Adaptive Filtering Problem, Unconstrained Optimization Techniques, Steepest Descent, Least-Mean-Square Algorithm, Learning Curves, Learning rate annealing techniques, Perceptron and Convergence theorem **Multilayer Perceptron:** Back-propagation Algorithm, Sequential and Batch Modes of training, Stopping Criteria, XOR problem, Heuristics for BP algorithm to perform better	
Unit – III	**10 Hrs**
Convolutional Neural Networks: **Introduction:** Historical Perspective and Biological Inspiration, Broader Observations About Convolutional Neural Networks **The Basic Structure of a Convolutional Network:** Padding, Strides, Typical Settings, The ReLU Layer, Pooling, Fully Connected Layers , The Interleaving Between Layers , Local Response Normalization , Hierarchical Feature Engineering **Training a Convolutional Network:** Backpropagating Through Convolutions , Backpropagation as Convolution with Inverted/Transposed Filter, Convolution/Backpropagation as Matrix Multiplications, Data Augmentation **Applications of Convolutional Networks:** Content-Based Image Retrieval , Object Localization, Object Detection , Natural Language and Sequence Learning , Video Classification	

Unit – IV	10 Hrs

Recurrent Neural Networks
Introduction: Expressiveness of Recurrent Networks,
The Architecture of Recurrent Neural Networks: Language Modeling Example of RNN, Generating a Language Sample, Backpropagation Through Time, Bidirectional Recurrent Networks, Multilayer Recurrent Networks

Echo-State Networks, Long Short-Term Memory (LSTM) , Gated Recurrent Units (GRUs)
Applications of Recurrent Neural Networks: Application to Automatic Image Captioning, Sequence-to-Sequence Learning and Machine Translation , Question-Answering Systems, Application to Sentence-Level Classification , Token-Level Classification with Linguistic Features, Time-Series Forecasting and Prediction, Temporal Recommender Systems, Secondary Protein Structure Prediction
End-to-End Speech Recognition Handwriting Recognition

Unit – V	10 Hrs

Deep Reinforcement Learning : Introduction
Stateless Algorithms: Multi-Armed Bandits: Naïve Algorithm, Greedy Algorithm, Upper Bounding Methods
The Basic Framework of Reinforcement Learning: Challenges of Reinforcement Learning, Simple Reinforcement Learning for Tic-Tac-Toe, Role of Deep Learning and a Straw-Man Algorithm
Bootstrapping for Value Function Learning: Deep Learning Models as Function Approximators, Example: Neural Network for Atari Setting, On-Policy Versus Off-Policy Methods: SARSA, Modeling States Versus State-Action Pairs
Monte Carlo Tree Search
Case Studies: AlphaGo: Championship Level Play at Go, Alpha Zero: Enhancements to Zero Human Knowledge , Self-Learning Robots, Deep Learning of Locomotion Skills, Deep Learning of Visuomotor Skills, Building Conversational Systems: Deep Learning for Chatbots, Self-Driving Cars

Laboratory Component

Group of two students belongs to same batch are required to implement an engineering application using any one of the deep learning techniques, CNN and architectures, RNN or Reinforcement learning.

Examples:

CNN: Biometric authentication using CNN, Object identification and recognition, Emotion recognition, Auto translation, document classification, etc.

RNN: Language translation, Generating image descriptions, Speech recognition, etc,

Reinforcement learning: Real-time bidding, Recommendation Systems, Traffic Control Systems, etc.

The laboratory component will be evaluated in two phases :

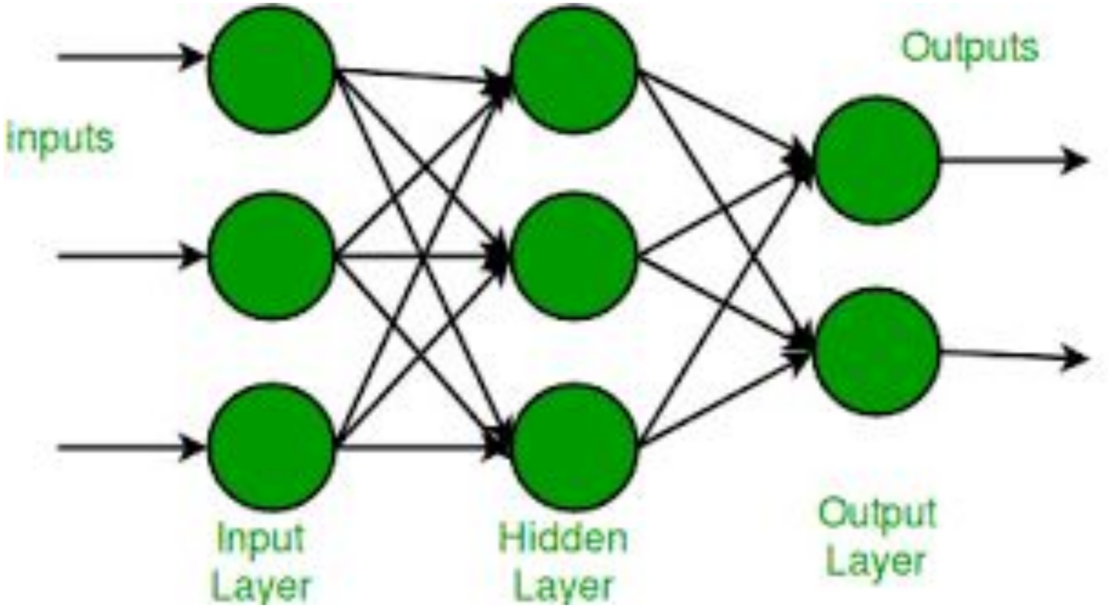
- **Phase I –** 25 marks which includes identification of the problem, data set collection and preprocessing, selection of appropriate algorithm and its justification
- **Phase II -** Implementation of the deep learning model with appropriate GUI, performance and evaluation of the model .

Unit -2

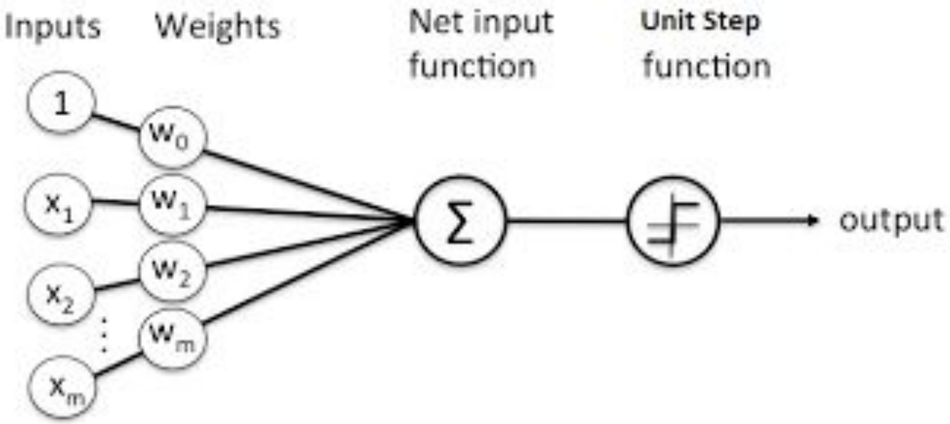
- **Single Layer Perceptron**
  - Introduction
  - Adaptive Filtering Problem
  - Unconstrained Optimization techniques
- **Multi Layer Perceptron**
  - Back propagation algorithm
  - Sequential and Batch modes of training
  - Stopping criteria
  - X-OR problem
  - Heuristics for BP algorithm

Unit -2

- **Perceptron Architecture**



Single Layer Perceptron



Unit -2

Perceptron networks :

- The perceptron network consists of three units, namely, sensory unit (input unit), associator unit (hidden unit), response unit (output unit).
- The sensory units are connected to associator units with fixed weights having values 1, 0 or -l, which are assigned at random.
- The binary activation function is used in sensory unit and associator unit.
- The response unit has an activation of l, 0 or -1. The binary step will be fixed threshold □

is used as activation for associator.

- The output signals that are sent from the $y = f(y_{in})$ tor unit to the response unit are only binary.
- The output of the percept

where $f(y_{in})$ is activation function and is defined as

$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > \theta \\ 0 & \text{if } -\theta \leq y_{in} \leq \theta \\ -1 & \text{if } y_{in} < -\theta \end{cases}$$

Unit -2

Perceptron networks :

- The perceptron learning rule is used in the weight updation between the associator unit and the response unit. For each training input, the net will calculate the response and it will determine whether or not an error has occurred.
- The error calculation is based on the comparison of the values of targets with those of the calculated outputs.
- The weights on the connections from the units that send the nonzero signal will get adjusted suitably.
- The weights will be adjusted on the basis of the learning rule an error has occurred for a particular training pattern

$$w_i(\text{new}) = w_i(\text{old}) + \alpha t x_i$$
$$b(\text{new}) = b(\text{old}) + \alpha t$$

Unit -2

Perceptron networks :

Perceptron learning rule is given by

The weight updation in case of perceptron learning is as shown.

If $y \neq t$, then

$$w(\text{new}) = w(\text{old}) + \alpha t x \quad (\alpha - \text{learning rate})$$

else, we have

$$w(\text{new}) = w(\text{old})$$

Unit -2

Perceptron networks :

Perceptron learning rule is given by

The weight updation in case of perceptron learning is as shown.

If $y \neq t$, then

$$w(\text{new}) = w(\text{old}) + \alpha t x \quad (\alpha - \text{learning rate})$$

else, we have

$$w(\text{new}) = w(\text{old})$$

Perceptron networks