U-2 {Deep Learning}.

* Biological Neuron -
- dendrites → accept inputs
- soma → process input.
- axon → turns processed inputs into output.
- synapses → electrochemical contact b/w neurons

- Basic unit of brain, comprising a cell body, dendrites & an axon.
- receives signals through dendrites & transmit signal via axon.
- communication occurs through electrical impulses & chemical signals.
- Integrates incoming signals & generates output based on thresholds.
- Forms complex networks to process info & produce responses.

* Perceptron -
- simple artificial neural network.
- single layer of processing units → neurons.
- fundamental building block of more complex neural networks.

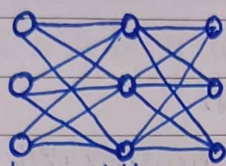
key components → 1) Inputs 2) Weights 3) Bias 4) Activation Function 5) Output

Training → 1) Initialization (start with random weights & bias)
2) Forward Pass (calculate weight sum of input → training example)
3) error calculation (compared predicted output with actual & calculate error)
4) Backpropagation (use error to adjust weights & bias → reduce error)
5) Repeat (Repeat step 2-4 for all training until overall error converges)

Types → • single layer → learn only linearly seperable patterns
• Multiple layer → learn about two or more layers → greater processing power

DOMS Page No. Date / /

* Multilayered feed-forward / Multilayer Preceptron-



Input Hidden Output.

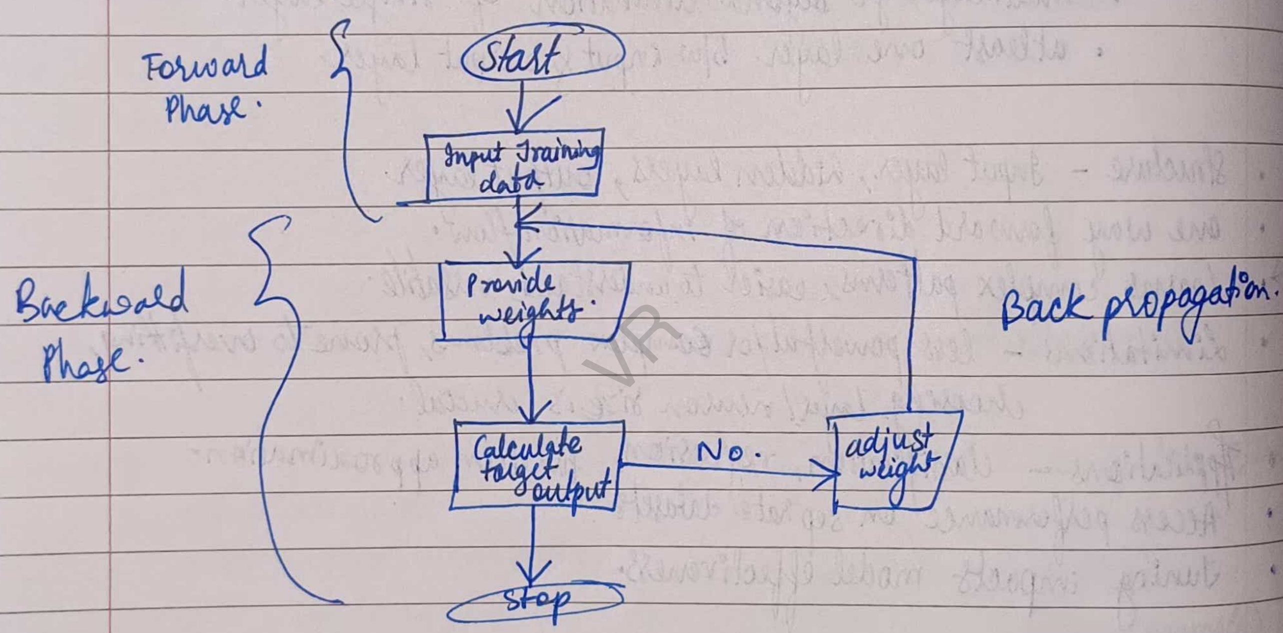
Adv → • complex design making → create mutiple layers of preception
• intricate info is too complex for layer to handle.
• multilayer go beyond limination of single layer.
• atleast one layer b/w input & output layer.

- Structure - Input layer, hidden layers, output layer
- one way forward direction of information flow.
- Learns complex patterns, easier to understand, versatile.
- Limitations - less powerful for complex problems, prone to overfitting, choosing layer/neuron size is crucial.
- Applications - classification, regression, function approximation.
- Access performance on seprate datasets.
- Tuning impacts model effectiveness.

Training Neural Networks-

- Backpropagation

  - algo for supervised learning for ANN.
  - keeps adjusting weights of connecting neurons with an object to reduce deviation of output signal with target output.
  - reach global loss minimising backpropagation
  - consists of mutiple iterations known as epochs.



Features → gradient descent method → case of single perception network with diff
→ weights are calculate in learning period of network.
→ feedforward of input training pattern
→ calculation of back propagation of error updation of weight

Adv → • simple fast & easy • only no. of input architecture are tuned, not any parameter
• flexible & efficient • • No need for user to learning any special function

Disadv → • sensitive to noisy data & irregularities • performance → dependent → data
• too much time training • matrix based on approach preffered never min batch.

- Forward Propagation -
  - process input data through network to produce predictions.
  - calculates output of a neural network for a given input.

- Input Layer, Weighted Sum, Activation Function & Output layer

Advantages -
- efficiently compute predictions for given input data.
- forms basis of training & inference.
- supports parallel processing.
- straightforward to implement.
- efficient for calculating outputs.

Disadvantages -
- doesn't directly contribute to training network.
- limited role on its own - needs backpropagation.
- limited in handling sequential.
- may be computationally expensive for deep networks.

* Activation Function - decides whether artificial neural network for a given set of inputs.

1) Linear Activation
   - → $f(x) = x$
   - → outputs input value directly without any modification.
   - → easy to compute
   - → stacks of linear layers create a linear model, not commonly used in deep neural network.

2) Sigmoid Activation
   - → $f(x) = \frac{1}{1+e^{-x}}$
   - → squeezes input value between 0 & 1.
   - → outputs 0 & 1 → interpretations in output layers
   - → suffers from vanishing gradient problem.
   - → where gradients becomes very small during backpropagation.

DOMS Page No.
Date / /

3) Tanh (Hyberbolic Tangent) –

- $f(x) = (\exp(x) - \exp(-x)) / (\exp(x) + \exp(-x))$
- outputs values between -1 & 1.
- Over sigmoid due to zero centering.
- gradient can be small in deep networks.

4) Hard Tanh:

- $f(x) = \{ \max(0, \min(x, \text{threshold})) \}$ | threshold is a user-defined value.
- clips output b/w a predefined threshold
- faster computation compared to sigmoid.
- can introduce dead spots where neurons never fire if threshold

5) Softmax –

- $f(x) = \exp(x_i) / \text{sum}(\exp(x_j))$ for all j.
- normalizes output of a layer into a probability distribution b/w 0 & 1.
- ensures all output sum to 1.
- Not suitable for regression tasks.

6) Rectified Linear Unit (ReLU) –

- $f(x) = m(0, x)$
- output the input directly if its positive or 0.
- Mitigates vanishing gradient problem as gradients.
- neurons become inactive if they receive negative inputs.

Loss Function -

- LF Notation → quantify difference between predicted & actual values.
  - → minimize loss to optimize model parameters
  - → example → MSE, Cross Entrophy Loss, Reconstruction.

- LF Regression → MSE- Measures average square difference b/w predicted & actual values.
  - → Min Variance b/w predicted values & actual.
  - → sensitive to outliers, differentiable with respect to predictions.

- LF classification → Cross Entropy Loss - Measures dissimilarity b/w predicted & actual class.
  - → Penalizes incorrect class probabilities.
  - → commonly used in multiclass classification tasks.

- LF Reconstruction → Reconstruction Loss - Measures diff b/w input & output in autoencoder models
  - → depends on type of reconstruction.
  - → Minimize reconstruction error to recover input data.
  - → used in auto encoders, variational autoencoders, & generative models for feature extraction & data generation.

Hyperparameters-

1) Learning Rate → determine stepsize during optimization
→ controls rate at which model parameters are updated
→ too high can lead to instability too low can slow convergence

2) Regularization → Technique to prevent overfitting by penalizing large p[arameters]
→ Types include L1(Lasso) & L2 (Ridge) regularization.
→ Balances b/w fitting training data & generalizing to unseen da[ta]

3) Momentum → accelerates gradient descent by adding a fraction of previous up[date]
→ helps in navigating through saddle points & flat regions.
→ Prevents oscillations & speedup convergence.

4) Sparsity → encourages models to have fewer activated neurons
→ Improves interpretability
→ Reduces memory requirements
→ used in models where features selection is crucial.

5) Deep Feedforward Networks → generally overskilled for solving problems like XOR
→ more powerful & suited for complex tasks.

6) Example of ExOR → a neural network architecture comprising multiple lay[ers]
→ consisting of an input layer, hidden layers
→ used in models each neuron applies an activation function.
→ Trained using backpropagation to learn XOR function

DOMS Page No.
Date / /

* Hidden Unit → Neurons in hidden layers of neural network.
→ extract & transform features from input data
→ Increase network's capacity to learn complex patterns
→ Activation funct introduce nonlinearity
→ no. of hidden units affected model complexity & training time.

* Cost Functions → Measure discrepancy b/w predicted & actual values
→ examples include MSE for regression
→ Cross entropy loss for classification.
→ Min cost functions to optimize model parameters.
→ Diff task may require diff cost functions.

* Error Backpropagation → Algorithm to compute gradient cost function.
→ propagates error backward networks.
→ utilizes chain rule for effigent gradient
→ enables effucient parameter update using gradient
→ Backbone of training deep neural networks.

* Gradient Based learning → 1) Approach to optimizes model parameter using gradient
2) Updates parameters in direction of steepest descent
3) Repeat unit convergence or stopping criteria is met
4) Monitoring loss convergence is crucial
5) compute gradient of cost function (to parameters)

* Vanishing & Exploding Gradient Descent → Problems encountered during training deep neural networks
→ vanishing gradient slow learning
→ Exploding gradient cause instability
→ Mitigated using techniques like weight initialization
→ choice of activation function also influences gradient behaviour.

DOMS Page No.
Date / /

* Sentiment Analysis → NLP task to determine sentiment from text
  → classifies text into positive, negative, or neutral category
  → Appl'n include social media monitoring & product review analysis
  → deep learning models like RNN & Transformers are commonly used
  → Requires large labelled datasets for training.

* Jupyter → Interactive computing environment
  → creation & sharing of documents
  → support for data cleaning, transformation & visualization
  → multiple language → Python, R & Julia
  → enhances reproducibility by integrating code & output in single document
  → enables quick prototyping & iterative development.
  → facilitates sharing notebooks via email, dropbox, github, etc
  → supports wide range interactive data exploration & visualization
  → used in ML, data science & research.

* Colab → cloud based platform provided by Google for ml & data science
  → offers similar features as Jupyter
  → free access to GPU's & TPU's
  → supports collaborative & real time connecting
  → seamlessly integrates with google drive for storing & sharing.
  → Includes pre-installed libraries for data manipulation, visualization
  → Provide access to Googles vast computing resources.
  → supports Markdown & LATEX for creating rich text documents
  → popular choice for prototyping, experimenting & sharing ml projects

DOMS Page No.
Date / /

PyTorch → Deep Learning framework developed by Facebook's AI research lab.
- tensor computation with GPU acceleration.
- offers dynamic graph construction.
- supports automatic diff for gradient computation.
- Popular for research & production due to flexibility
- Includes modules for building & training neural networks.
- Rich ecosystem with libraries for CV, NLP & more
- utilizes Python for easy interaction with existing workflows
- continuous updates & improvements from an active community
- used by researchers, engineers & enthusiasts worldwide.

VR