Unit 2

Certainly! Let's explore Propositional Logic, Predicate Logic, and Resolution in detail:

1. **Propositional Logic:**

   - **Definition:** Propositional logic deals with statements that can be either true or false. It uses logical connectives like AND, OR, NOT, XOR, and implies to express relationships between these statements.

   - **Example:** If \(P \) is "It is raining," and \(Q \) is "I will take an umbrella," then \(P \rightarrow Q \) means "If it is raining, then I will take an umbrella."

2. **Predicate Logic:**

   - **Definition:** Predicate logic extends propositional logic by introducing variables, predicates, and quantifiers. Predicates are functions that return true or false for specific input values, and quantifiers allow statements about all or some elements in a domain.

   - **Example:** Let \(P(x) \) be "x is a student," and \(Q(x) \) be "x passed the exam." The statement \(\forall x (P(x) \rightarrow Q(x)) \) means "All students passed the exam."

3. **Resolution:**

   - **Definition:** Resolution is an inference rule used in logic to resolve conflicting clauses. It's particularly crucial in automated reasoning. The rule allows us to combine or simplify logical statements.

   - **Example:** Given clauses \(P \lor Q \) and \(\neg Q \), resolution enables us to derive \(P \).

4. **Resolution in Propositional Logic:**

   - **Definition:** This is the application of resolution specifically to propositions. If one clause contains \(P \lor Q \) and another \(\neg Q \), resolution lets us infer \(P \).

   - **Example:** With \((P \lor Q) \) and \((\neg Q \lor R) \), resolution allows us to deduce \(P \lor R \).

5. **Resolution in Predicate Logic:**

- **Definition:** Resolution adapted to the complexities of predicate logic, involving handling variables and quantifiers. Techniques like Skolemization may be used for this purpose.

- **Example:** Given \(\forall x P(x) \) and \(\neg P(a) \), resolution enables us to derive \(\exists x \neg P(x) \).

6. **Clause Form:**

- **Definition:** Clause form is a way of representing logical formulas as a disjunction of conjunctions. It simplifies the expression for ease of understanding and manipulation.

- **Example:** Instead of \((P \land Q) \lor (\neg R \land S) \), we write it in clause form as \((P \lor \neg R) \land (Q \lor \neg R) \land (P \lor S) \land (Q \lor S) \).

7. **Unification Algorithm:**

- **Definition:** The unification algorithm finds substitutions for variables in two expressions to make them identical. It's often used in logic programming and artificial intelligence.

- **Example:** For \(P(x, a) \) and \(P(b, y) \), the unification algorithm gives \(x=b \) and \(y=a \) as a substitution to make the expressions equivalent.

Answering a 5-mark question on each of these topics would involve providing the definitions, explaining the significance or application of each concept, and possibly offering an example or two to illustrate your points.

intelligence (AI). Hill Climbing is a local search algorithm used for mathematical optimization problems. Let's explore the Hill Climbing algorithm and its characteristics:

Hill Climbing Algorithm:

1. **Initialization:**
   - Start with an initial solution (a random or predefined state).
2. **Evaluate:**
   - Evaluate the current solution using an objective function or a heuristic.
3. **Generate Neighbors:**
   - Generate neighboring solutions by making small modifications to the current solution. These modifications depend on the problem at hand.
4. **Select Best Neighbor:**
   - Choose the neighbor with the highest (or lowest, depending on the optimization goal) evaluation value.
5. **Update:**
   - If the selected neighbor is better than the current solution, update the current solution with the selected neighbor.
6. **Repeat:**
   - Repeat steps 2-5 until a stopping criterion is met (e.g., a maximum number of iterations, a satisfactory solution is found, etc.).

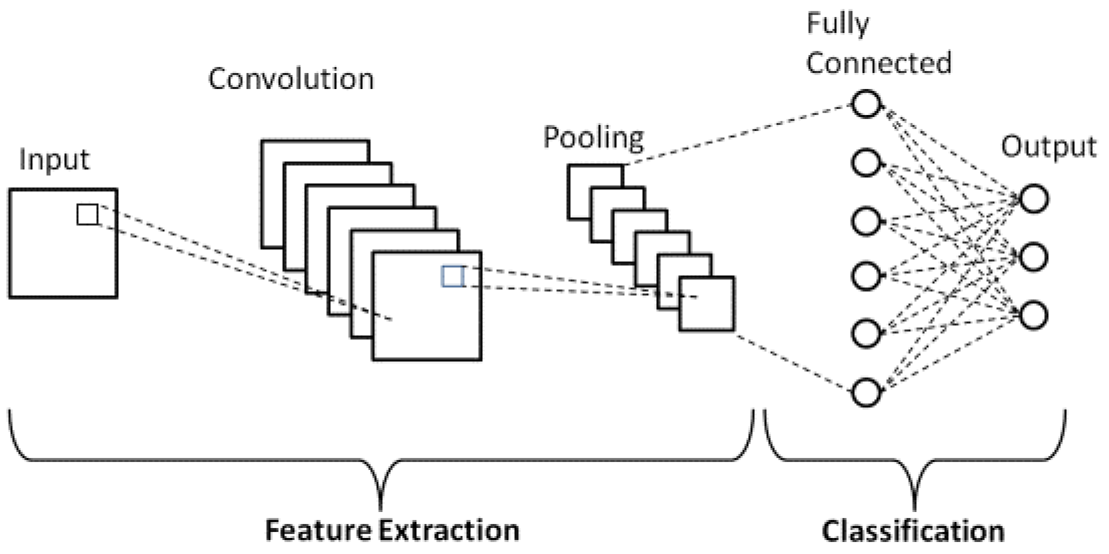
plaintext Copy code

```plaintext
current_solution = initialize()
while not stopping_criterion():
    evaluate current_solution
    neighbors = generate_neighbors(current_solution)
    best_neighbor = select_best_neighbor(neighbors)
    if evaluate(best_neighbor) > evaluate(current_solution):
        current_solution = best_neighbor
```

UNIT 3

CNN

Arch:



ResNet (Residual Neural Network):

- **Definition:**
  - *ResNet is a type of convolutional neural network (CNN) that introduces residual connections, allowing for the training of very deep networks.*
- **Key Features:**
  - *Residual Blocks: Introduces skip connections to jump over some layers, helping to avoid the vanishing gradient problem.*
  - *Facilitates the training of extremely deep networks, even exceeding 100 layers.*
- **Applications:**
  1. **Image Classification:**
     - *ResNet has been highly successful in image classification tasks, achieving state-of-the-art results on datasets like ImageNet.*
  2. **Object Detection:**
     - *Utilized in object detection tasks, where the network needs to identify and localize objects within an image.*

3. **Semantic Segmentation:**
    - *Used for segmenting images into different classes or regions, aiding in tasks like scene understanding.*

The working is ResNet is pretty simple, it adds the original input back to the output feature map obtained by passing the input through one or more convolutional layers.

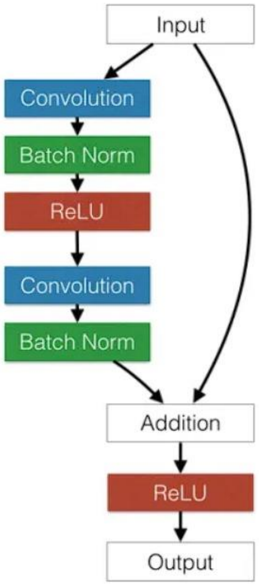


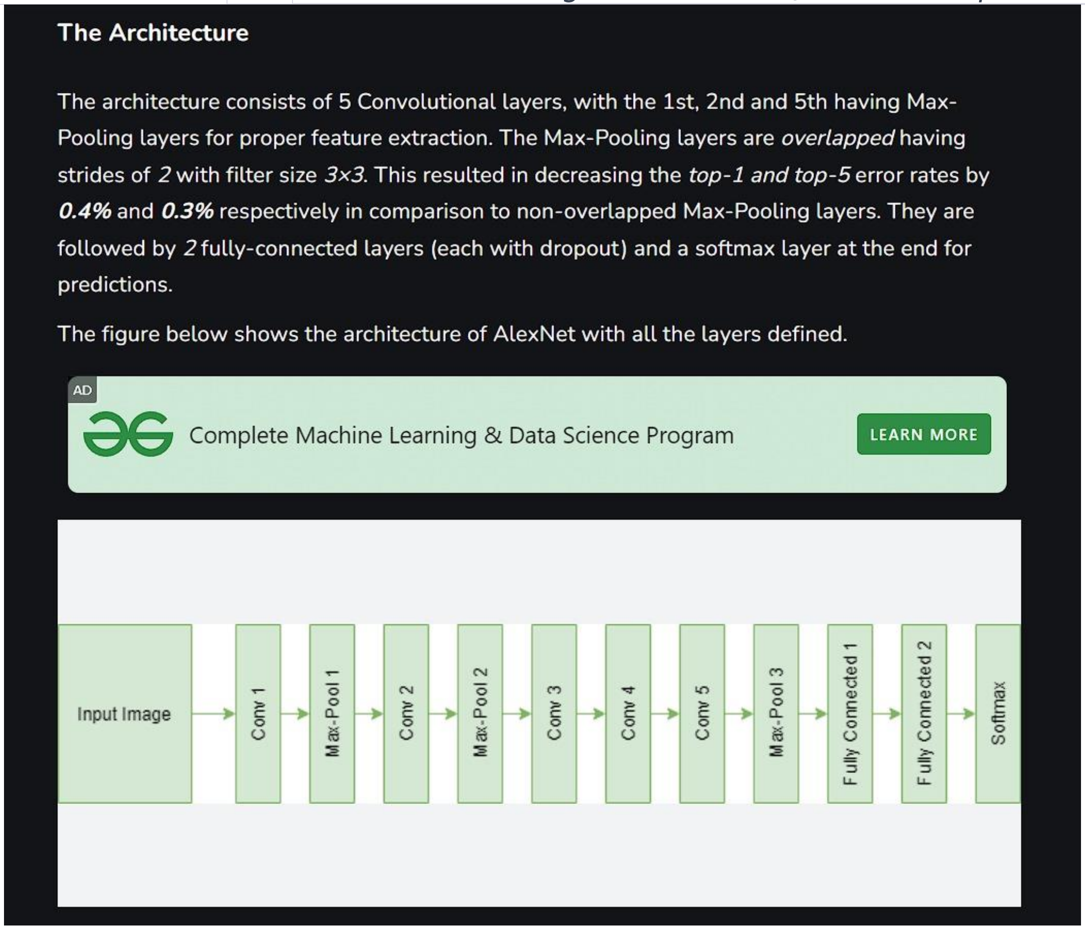
Figure 1. A RestNet basic block

AlexNet:

- **Definition:**
    - *AlexNet is a convolutional neural network designed for large-scale image classification tasks.*
- **Key Features:**
    - *Convolutional Layers: AlexNet consists of multiple convolutional layers, capturing hierarchical features.*
    - *Utilizes local response normalization and dropout to prevent overfitting.*
- **Applications:**
    1. **Image Classification:**

- *AlexNet gained prominence by winning the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012, significantly advancing image classification accuracy.*

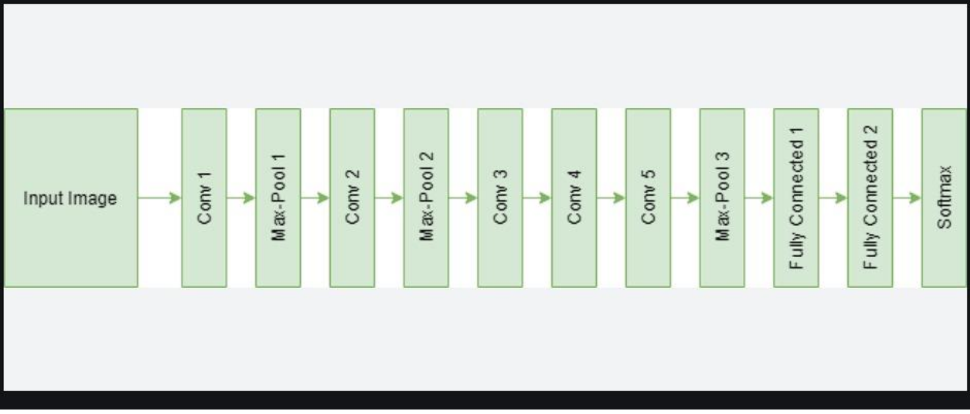
2. **Object Localization:**
   - *Applied in tasks where the goal is not only to classify objects but also to locate them within an image.*
3. **Feature Extraction:**
   - *Used as a feature extractor in transfer learning scenarios, where pre-trained AlexNet weights are utilized for other computer vision tasks.*



The Architecture

The architecture consists of 5 Convolutional layers, with the 1st, 2nd and 5th having Max-Pooling layers for proper feature extraction. The Max-Pooling layers are *overlapped* having strides of *2* with filter size *3x3*. This resulted in decreasing the *top-1 and top-5* error rates by ***0.4%*** and ***0.3%*** respectively in comparison to non-overlapped Max-Pooling layers. They are followed by *2* fully-connected layers (each with dropout) and a softmax layer at the end for predictions.

The figure below shows the architecture of AlexNet with all the layers defined.



Applications of Emerging NN Architectures:

1. **Image Recognition and Classification:**
   - *Both ResNet and AlexNet excel in image recognition tasks, where the goal is to categorize images into different classes.*
2. **Object Detection:**
   - *ResNet and AlexNet are employed in object detection applications, identifying and localizing objects within images.*
3. **Semantic Segmentation:**
   - *ResNet is commonly used for semantic segmentation tasks, dividing images into meaningful regions.*
4. **Transfer Learning:**
   - *AlexNet, with its pre-trained weights, is often utilized as a feature extractor in transfer learning applications.*
5. **Medical Image Analysis:**
   - *ResNet has found applications in the analysis of medical images, assisting in tasks such as tumor detection.*
6. **Video Analysis:**
   - *Both architectures can be adapted for video analysis tasks, such as action recognition and tracking.*
7. **Natural Language Processing (NLP):**
   - *While primarily designed for computer vision, these architectures' principles have inspired developments in NLP, such as image captioning.*

UNIT-4

Vanishing and Exploding Gradients:

- Vanishing gradients occur when the gradients become extremely small during backpropagation through time (BPTT), hindering the learning process.

- Exploding gradients happen when the gradients become too large, causing instability during training.

Gradient Clipping:

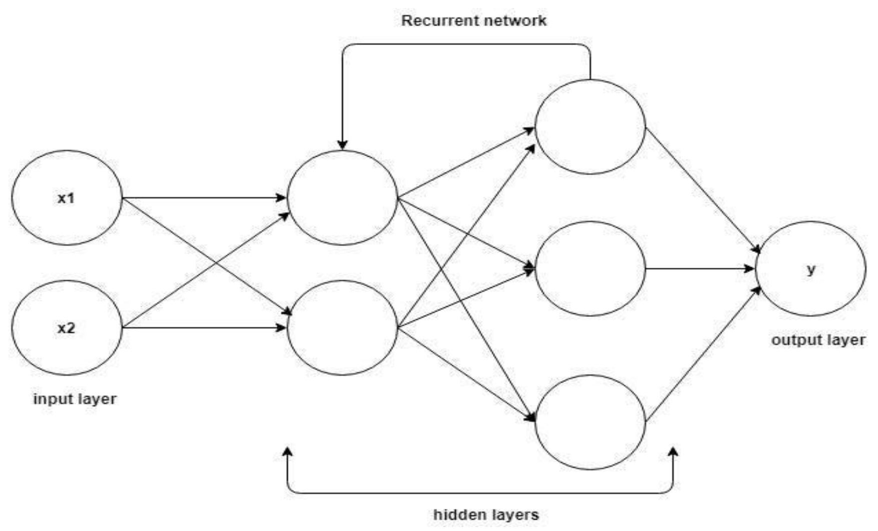
- Gradient clipping involves setting a threshold for gradients during training.

- If gradients exceed the threshold, they are scaled down to prevent exploding gradients without losing the information.

Recurrent Neural Networks (RNNs):

- **Definition:**
    - *RNNs are neural networks designed for sequential data, maintaining a hidden state to capture dependencies across time.*

Arch:



Building Recurrent NN:

1. **Architecture:**
    - *Involves an input layer, hidden layer with recurrent connections, and an output layer.*
    - *Captures sequential dependencies through feedback loops.*
2. **Vanishing and Exploding Gradients:**
    - *Issues where gradients become too small (vanishing) or too large (exploding) during training.*
    - *Addressed using techniques like gradient clipping.*

Long Short-Term Memory (LSTM):

- **Definition:**
    - *LSTMs are a type of RNN designed to overcome the vanishing gradient problem and better capture long-term dependencies.*

Time Series Forecasting:

- *Application of RNNs or LSTMs to predict future values based on historical data.*

Bidirectional RNNs:

- *RNNs processing sequences in both forward and backward directions to capture information from both past and future.*

Encoder-Decoder Sequence-to-Sequence Architectures:

- *Used for tasks like machine translation, involving an encoder processing input sequences and a decoder generating output sequences.*

BPTT for Training RNN:

- *Backpropagation Through Time (BPTT) is an extension of backpropagation used to train RNNs, involving unfolding the network over time.*

Computer Vision, Speech Recognition, Natural Language Processing:

- *Applications of neural networks in different domains.*

Case Studies:

1. **Classification:**
   - *Categorizing data into classes using deep networks.*
2. **Regression:**
   - *Predicting numerical values, often applied in financial forecasting.*
3. **Deep Networks:**
   - *Utilized for complex tasks requiring multiple layers, e.g., image generation or style transfer.*

Summary:

- *RNNs capture sequential dependencies, but face challenges like vanishing/exploding gradients.*
- *LSTMs address RNN issues, especially in tasks like time series forecasting.*
- *Bidirectional RNNs provide richer context by processing sequences in both directions.*
- *Encoder-decoder architectures are used in tasks like machine translation.*
- *BPTT facilitates training RNNs over time.*
- *Applications span computer vision, speech recognition, and natural language processing.*
- *Case studies showcase the versatility of neural networks in classification, regression, and deep learning tasks.*