Assignment No :: 1

- Name :: Kisan Yadav
- Roll No :: A61
- Batch :: A4

raisoni EDUCATION

* Title :

Write a program Logistic Regression with Neural Network mindset.

* Objective :

Build the general architecture of a learning algorithm including.

- Initializing parameters
- Calculating the cost function and its gradient.
- Using an optimization algorithm (gradient)

Gather all three functions above into a main model function in the right order.

* Theory :

Welcome to the first (required) programming exercise of the deep learning specialization. In this notebook you will build your first image recognition algorithm. You will build a cat classifier that recognizes cats with 70% accuracy.

As you keep learning new techniques you will increase it to 80+ % accuracy on cat vs non-cat datasets. By completing this assignment you will :

- Work with logistic regression in a way that builds intuition relevant to neural network

Learn how to minimize the cost function.

Understand how derivatives of the cost are used to update parameters.

Packages :-

numpy :- is the fundamental package for scientific computing which Python

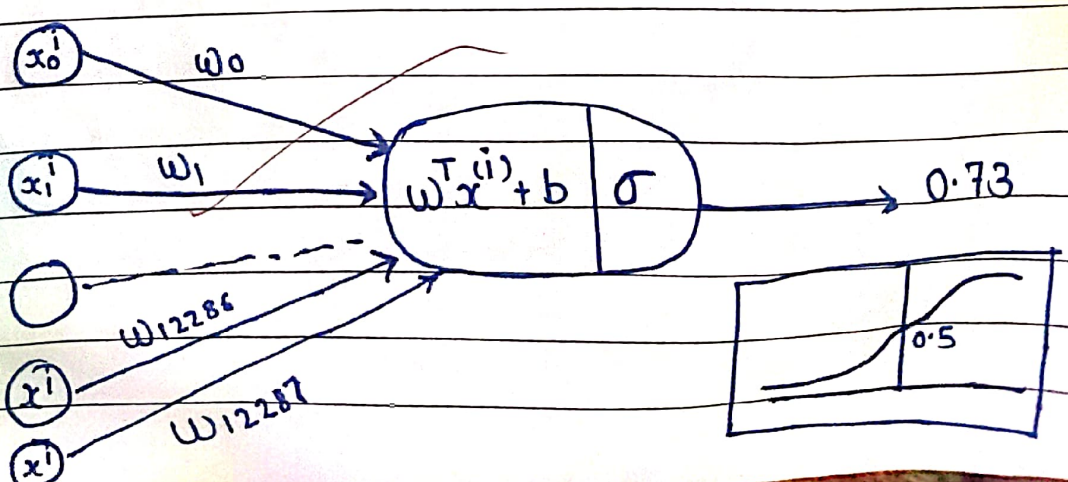
h5py :- is a common package to interact with a dataset that is stored on an H5 file.

matplotlib :- is a famous library to plot graphs in python.

PIL are used here to test your model with your own picture at the end.

General Architecture of the learning algorithm :-

It's time to design a simple algorithm to distinguish cat images from non-cat images. You will build a LR using a Neural Network mindset. The following Figure exptains why Logistic Regression is actually a very simple Neural Network.



Mathematical expression of the algorithm

$$z^{(i)} = w^T x^{(i)} + b$$

Key Steps : In this exercise, you will carry out the following steps :

Initialize the parameters of the model

Learn the parameters for the model by minimizing the cost

Use the learned parameters to make predictions (on the set)

Analyse the results and conclude

Building the part of our algorithm

The main steps for building a Neural Network are :

Define the model structure (such as number of input features)

Initialize the model's parameters

Loop :
- calculate current loss (forward propagation)
- Calculate current gradient (backward propagation
- Update parameters (gradient descent)

Optimization :

You have initialized your parameters

You are also able to compute a cost function and its gradient.

Now, you want to update the parameters using gradient descent.

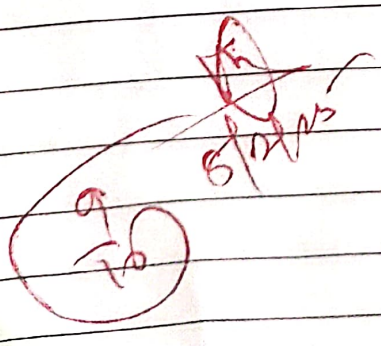
Platform Required :

Operating System : Windows

Software or Tools : Google Colab

Conclusion :

Hence we studied the concept of logistic Regression with Neural Network mindset along with accuracy is 70% for the dataset.



Assignment No : 2

Name :- Kisan Yadav
Roll No :- A61
Batch :- A4

raisoni EDUCATION

* Title :-

Implement Planner data classification with one hidden layer

* Objective (Problem Statement)

•) Develop an intuition of back-propagation and see it work on data

•) Recognize that the more hidden layers you have the more complex structure you could capture.

•) Capture the cross entropy loss

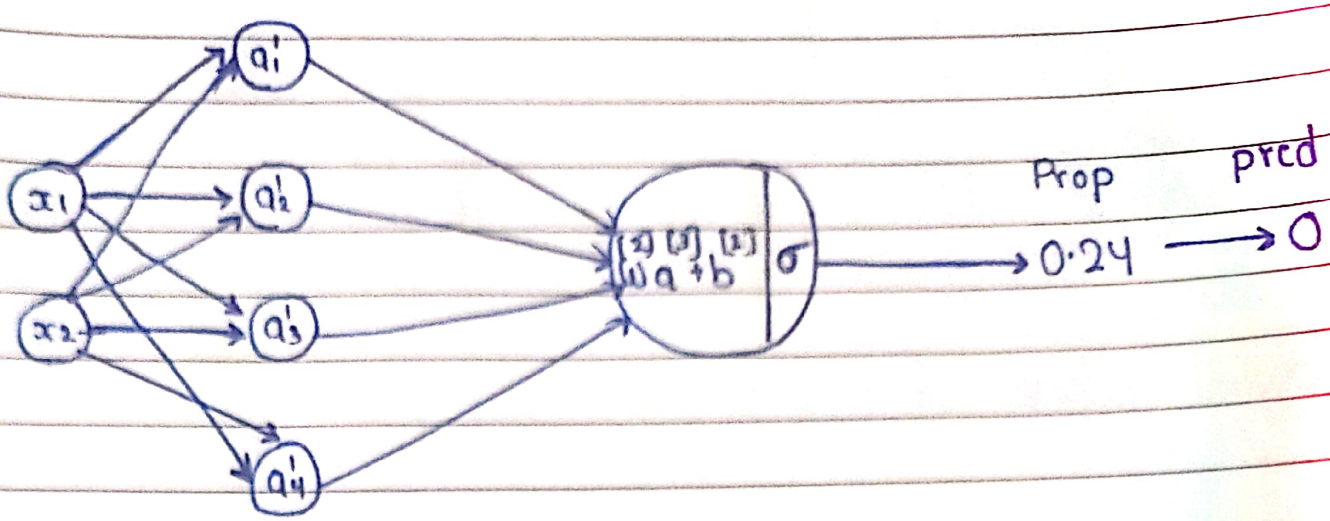
•) Implement forward and backward propagation.

* Theory of Objective :-

i) Define the neural network structure
ii) Initialize the model's parameters
iii) Loop :- Implement forward propagation - Compute loss - Implement backward propagation to get the gradients

* Theory :-

Neural Network model Logistic regression did not work well on the "flower dataset". You are going to train a neural network with a sigle hidden layer.



(Neural Network using Sigmoid Function)

Packages :-

numpy :- is the fundamental package for scientific comp

sk learn :- provides the simple and efficient tool for data mining and data analysis.

matplotlib :- is famous library to plot graph in Python.

test Cases :- provides some text examples to asses the correctness of your functions.

test Cases :- provides simple and efficient tools for data mining and data analysis.

Defining the nural network structure :-

Define three variables n_x : the size of input layer

n_h : the size of the hidden layer (set this to 4)

n_y : the size of the output layer.

Initialize the model's parameters :-

Make sure your parameter size are right. Refer to the neural network figure above if needed.

You will initialize the weighs matrices with the random values

You will initialize the bias vectors as zeros

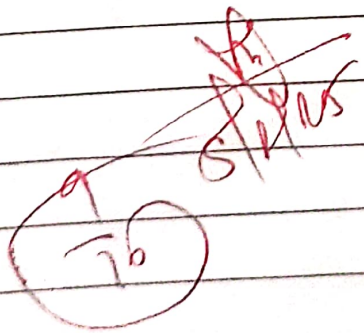
Platform Required :-

Operating System :-

Software Tool :- Google Colab

Conclusion :-

Hence we studied the concept of Planner data classification with one hidden layer with 91% accuracy.



Assignment No : 3

raisoni EDUCATION

Aim :

Implement Neural Network with one hidden layer

Problem Statement :

Develop an intuition of forward-propagation and see it work on data

Reconize that the one hidden layers you have the more complex structure you could capture.

Build all the helper function to implement a full model with one hidden layer

Use unit with a non-linear activation function, such as tanh.

Objective :

Define the neural network structure (# of input units, # of hidden units, etc)

Initialize the model's parameter.

Theory :

Different activation function used in neural network

Activation Function :

The main objective of the activation function is to perform a mapping of a weighted sum upon the output. The transformation function comprises of activation function such as tanh, ReLU, sigmoid, etc.

The activation function is categorized into two main parts

Linear Activation Function
Non Linear Activation Function

Linear Activation function :-

In the linear activation function, the output of functions is not restricted in between any range. Its range is specified from -infinity to infinity.

Non-linear function

These are one of the model is most widely used activation function. It solves the following problems faced by linear ~~fun~~ activation functions :-
Since the non-linear function comes up with derivative function so the problem related to back propagation has been successfully solved

For the creation of deep neural network, it permits the stacking up of several layers of the neurons

The non-linear activation function is further divided in following parts :-

Sigmoid or Logistic Activation Function :-

It provides a smooth gradient by preventing sudden jump in the output values. It has an output value range between 0 and 1 that helps in the normalization of each neuron's output. For x, if it has a value above 2 or below -2, then the value of y will be much steeper. In simple language, it means that even a small change in the x can bring a lot of change in y.

It's value ranges between 0 to 1 due to which it is highly preferred by Binary classification whose result is either 0 or 1.

Tanh or Hyperbolic Tangent Activation Function.

The tanh activation function work much better that of the sigmoid function, or simply we can say it is an advanced version of the sigmoid activation function. Since it has a value range between -1 to 1, so it is utilized by the hidden layer in neural network & and because of this reason, it has made the process of learning much easier.

Relu Activation Function

Relu is one of the most widely used activation function by hidden layer in the neural network. It value range from 0 to infinity.

Softmax Function

It is one of sigmoid function whereby solving the problems of classifications. If is mainly used to handle multiple classes for which it squeezes the output of each class between 0 and 1, followed by dividing it by divi sum of output layer.

Packages :

Numpy :- It is the fundamental package for scientific computing with python.

Sk learn :- Provide simple and efficient tools for data mining and data analysis

Matplotlib :- It is a famous library to plot graphs in python.

planar_utils provide :- various useful functions used in this assignment

Define three variables :- n_x :- the size of the input layer

n_h :- the size of the hidden layer

n_y :- the size of the o/p layer

* Platform Required

OS : Windows
Software Tools : Google Colab

Conclusion :

Hence we studiud the concept of Planner data classification with one hidden layer with 67.7% accuracy.

Assignment No : 4

raisoni EDUCATION

Title :

To build deep neutral network step by step

Problem Statements :

- Develop an intuition of the over all structure of neutral network.
- Write function that would help you decompose your code1 and ease the process of building a neural network
- Intialize / update parameter according to your desired structure.

Objective :

- Use non-linear units like Rellu to improve your model
- Build a deeper neural network
- Implement an easy-to-use neural network class

Theory :

i) Intidlize the parameter for a two-layer network
ii) Implement the an easy-to-use neural network class
iii) Implement an easy-to-use neural network class

1) Intialize the parameter for a two-layer network and for an L-layer network
2) Implement the forward propagation.

Complete the Linear part of a layer's forward propagation step

We give you the Activation Function
Combime the previous two steps into a new forward funt"
Stack the forward L-1 time and add a at the end
This give you a new L_model_forward. function

Compute the loss

Implement the backward propagation module

Complete the Linear part of a layer's backward propagation step.

We give you the gradient of the Activate function

Combine the previous two steps into a new Backward function.

Stack backward L-1 times and add backward in a new L_model_backward function.

Packages :

numpy
matplotlib
dnn_until provide some necessary functions for this notebook
testCases
np.random.seed()

raisoni
EDUCATION

* Platform Required

- OS → Windows
- Software tool :- Google Colab

* Conclusion :-

Hence we studied the steps required to implement Deep Neural Network.

Assignment No : 5

raisoni EDUCATION

Title :

Implement the concept of regularization, gradient checking and optimization in convolutional model step by step.

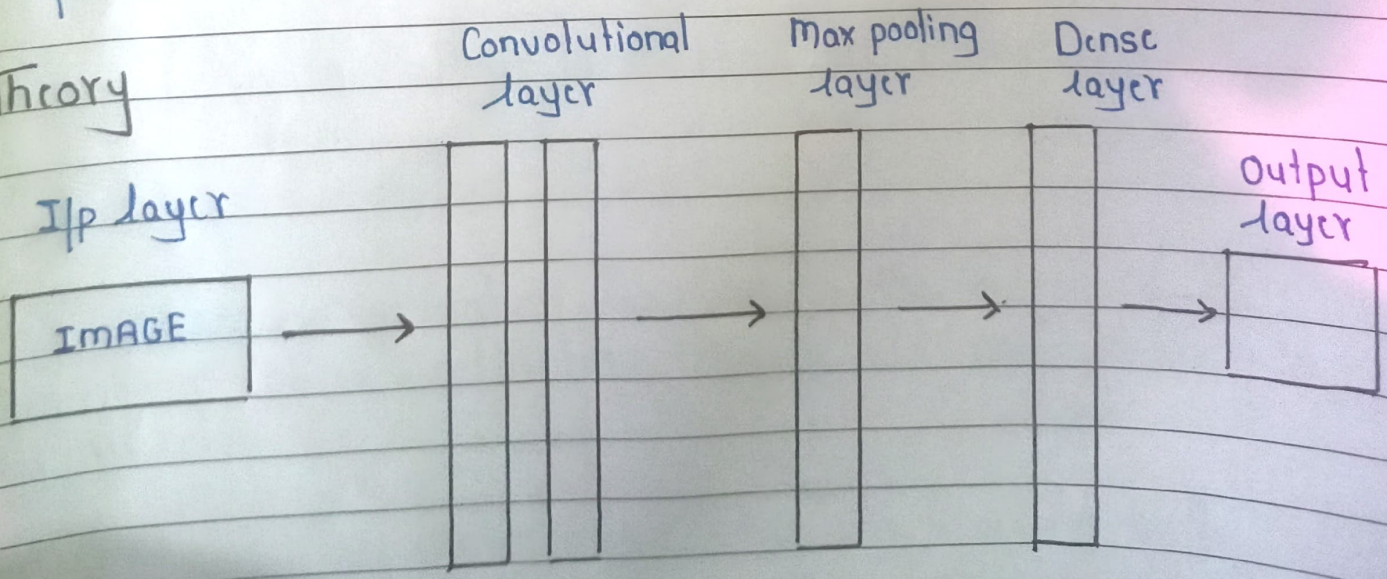
Problem Statement :

Develop an intuition of the over all structure of a neural network
Write function that would help you decompose your code and ease the process of building a neural network
Initialize / update parameter according to your desired structure.

Objective :

Take an input ~~value~~. volume.
Applies a filter at every position of the input.
Outputs another volume.

Theory



raisoni EDUCATION

Input layer

It's the layer in which we give input to our model
In CNN, Generally, input will be image or a sequence of images
This layers hold the raw input of the image ~~with~~ width 32, height 32, and depth 3.

Convolutional Layer

This is the layer, which is used to extract the feature from the input datasets.
It applies a set of learnable filters known as the kernals to the input images.
The filters / kernals are the smaller matrices usually 2x2, 3x3 or 5x5 shape.

It slides over the input image data and computes the dot product between kernel weight and the corresponding input image patch.

The output of this layer is referred ad feature maps.

Activation layer :

By adding an activation function to the output of the preceding layer, activation layers add non linearity to the network.

It will apply an element-wise activation function to the output of the convolution layer. Some common activation function are Relu : $\max(0, x)$, Tanh, Leaky ReLu, etc.

raisoni EDUCATION

Pooling Layer

- This layer is periodically inserted in the ~~convent~~ convnets and main function is reduce the size of volume which makes the computation fast reduces memory and also prevents overfitting.
- Two common types of pooling layer are max pooling and average pooling. If we use a max pool with 2x2 filter and stride 2, the resultant volume will be of dimension 16 x 16 x 12.

Flattening

The resulting feature maps are flattered in 1D vector after the convolution and pooling layer so they can they can be passed into a completely liked layer for categorization or regression.

Fully connected layer

It takes the input from the previous layer and computes the final classification or regression task.

Output layers

The output from the fully connected layers is then fed into a logistic function for classification tasks like sigmoid or softmax which converts the output of each class into the probability score of each class.

raisoni
EDUCATION

Procedure

1) import the necessary libraries.
2) set the parameter
3) define the kernal
4) Load the image and plot it
5) Reformat the image
7) Apply ~~con~~ activation layer operation and plot the output image.
6) Apply convolution layer operation and plot the output image.
8) Apply pooling layer operation and plot the output image

Conclusion :

Hence we studied the step required to implement CNN

Assignment No : 6

raisoni EDUCATION

Title : Implementation and Demonstration of the usages of the YOLO algorithm for the detection of the surrounding cars and the objects.

Problem Statement :

You are working on self-driving car. As a critical component of this project, you'd like to first build a car detection system.

To collect data, we have mounted a camera to the hood of the car, which take pictures of the road ahead every few seconds while you drive around.

Objective : YOLO algorithm Is capable of accurate vehicle detection with near real-time performance in the variety of different driving conditions

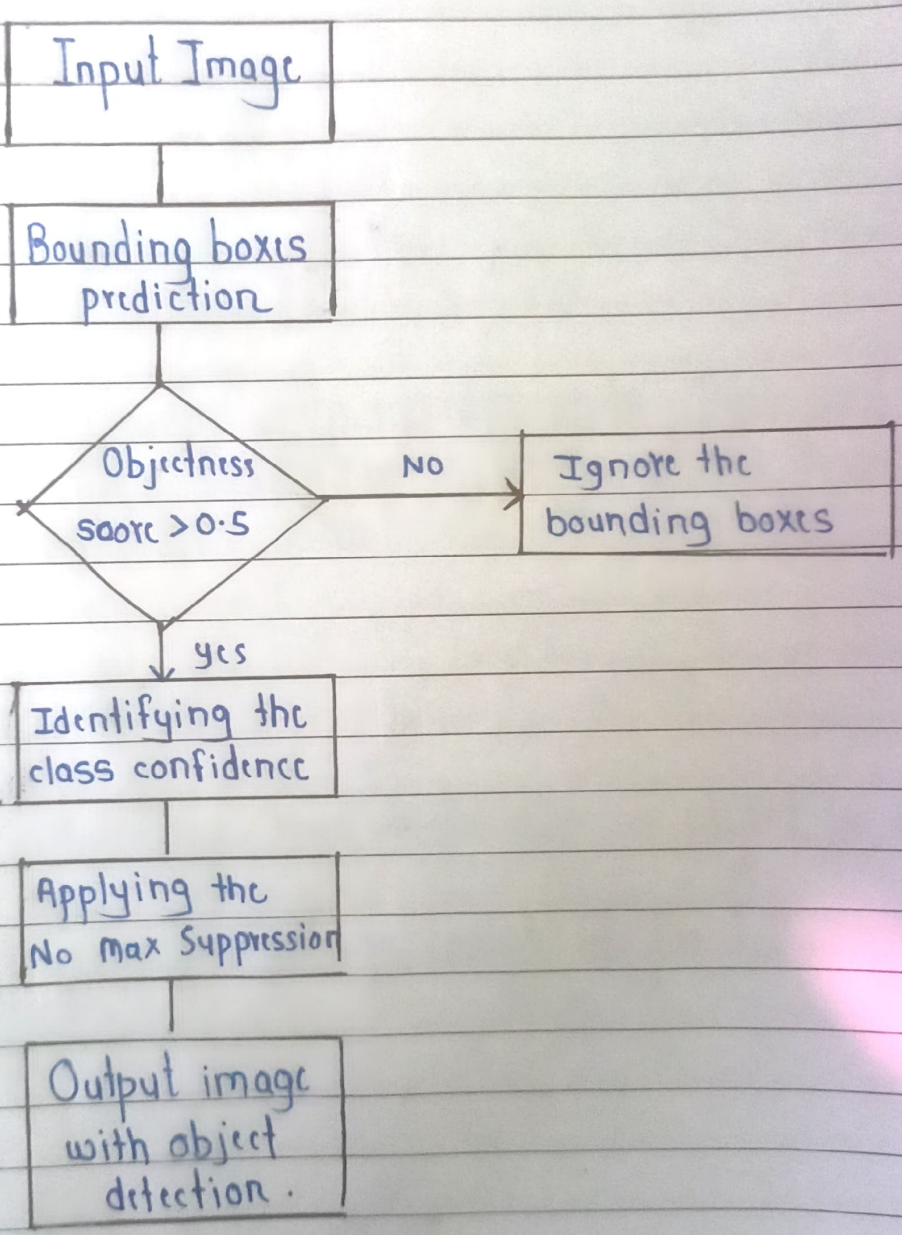
Theory :

YOLO is a clever convolutional neural network (CNN) for doing object detection in real-time. The algorithm applies a single neural network to full image, and then divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probilities.

YOLO is clever convolutional neural network (CNN) for doing object detection in real-time. The algorithm applies a single neural network to the full image and then divides (them) the image into region and predict bounding bo

and probabilities for each region. These bounding boxes are weighted by the predicted probabilities.

Object Detection Flowchart



Object Detection Flow Chart : 1

raisoni EDUCATION

* How the YOLO algorithm works :

YOLO algorithm work using the following three techniques

- Residual blocks
- Bounding box regression
- Intersection Over Union (IOU)

⇨ Residual Blocks :

First, the image is divided into various grids. Each grid has a demension of S×S. The following image show how an input image is divided into grids

⇨ Bounding Box Regression

A bounding is an outline that highlights an object in an image

Every bounding box in the image consist of the following attributes :

- Width (bw)
- Height (bh)
- Class (Ex : person, car, traffic, etc)
- Bounding box center

YOLO uses a single bounding box reggression to predict the height, width, center, and class of objects.

Intersection over union (IOU)

Intersection over union (IOU) is a phenomenon in object detection that describes how boxes overlap. YOLO uses IOU to provide an output box that surrounds the object perfectly.

Each grid cell is responsible for predicting the bounding boxes and their confidence scores. The eliminates bounding boxes that are not equal to the real box.

Applications of YOLO :-

YOLO algorithm can be applied in the following fields :-

Autonomous driving

Wildlife

Security

Packages :-
- numpy
- matplotlib
- np.random.seed()

Platform Required :-
- OS :- Windows
- Tools :- Google Colab

Conclusion :- Hence we studied the steps required to implement CNN

Assignment No : 7

raisoni EDUCATION

Title : Case Study on recent Deep Learning Application

Objective :

To analyze and understand AlphaFold, a deep learning-based application for protein structure prediction, its working (application) principles, impact on biological research and healthcare, and its comparsion with traditional methods.

Theory :

Deep Learning is a subset of machine learning that utilize artificial neural network to process and anatyze large dataset. It has gained popularity due to its ability to extract patterns and features from complex data without explicit programming.

Recent advancements in deep learning have led to application in various domains such as healthcare, finance, autonomous system, and natural language processing

Introduction to AlphaFold :

Alphafold, developed by DeepMind, is a deep learning based model that predicts the 3D-structure of proteins with high accuracy. This breakthrough has significantly impacted the fields of bio informatics and molecular biology.

Traditional methods like X-ray crystallography and cryo-electron microscopy are expensive and time-consuming, whereas, AlphaFold provides faster and cost-effective protein structure predictions.

Key Features of AlphaFold :

Uses deep learning models trained on protein sequences and structural databases.

Predicts highly accurate 3D protein structures.

Helps in drug discovery and disease research.

Has been recognized as a solution to the long-standing protein-folding problem.

Methodology :

Understanding Deep Learning in AlphaFold

AlphaFold is based on neural network model that learns patterns from known protein structures.

It uses a deep residual convolutional network along with attention mechanisms to predict protein structures.

Training Data and Model Architecture :

AlphaFold is trained using large dataset such as the Protein Data Bank (PDB) and genetic sequence databases.

The model applies evolutionary data and multiple sequence alignments (MSAs) to improve accuracy.

Comparsion with Traditional Methods :

Traditional methods like x ray crystallography and NMR spectroscopy take months or years to determine the protein structures.

AlphaFold provides accurate predictions in a matter of hours, reducing both cost and time.

Real - World Application of AlphaFold :

Drug Discovery
Disease Research
Synthetic Biology
Vaccine Development

Observations :

AlphaFold achieves over 90% accuracy in predicting protein structures compared to laboratory - based techniques

The model has been successfully used in predicting structures of unknown proteins, aiding in vaccine development.

Pharmaceutical companies and researchers are leveraging AlphaFold to accelerate the drug discovery process.

raisoni EDUCATION

- It has also contributed to agriculture by helping in protein engineering for improved crop yield and resistance to diseases.

* Challenges and limitations :-

- Although highly accurate, AlphaFold is not perfect and may produce incorrect predictions in some cases.
- Requires extensive computational resources and high-end GPUs for model training and execution.
- Needs further validation before its predictions can fully replace traditional lab-based methods.

* References :-

- Jumper, J., et al. (2021). "Highly accurate protein structure prediction with AlphaFold" Nature.
- Deep Mind Official Blog on Alphafold.
- Various research papers and online resources on deep learning applications in bioinformatics.

* Conclusion :- AlphaFold has revolutionized protein structure prediction, making it faster and more accurate. Its application extend to drug discovery, disease research, synthetic biology, and agriculture.

Assignment No : 8

raisoni EDUCATION

Title :

Implement DevOps and MLOPS using Cloud

Objective :

To understand the concept of DevOps and MLOps, their implementation using cloud technologies, and their impact on software development and machine learning life cycle management

Theory.

Introduction to DevOps and MLOps :

DevOps is a combination of development and operations praticks aimed at automating and streamlining software delivery. It promotes collaboration between development and IT Operations to enhance efficiency, reduce deployment time, and ensure continuous integration and delivery

MLOps (Machine Learning Operations) extends the principles of DevOps to machine learning workflows. It involves automating model training, deployment, monitoring and governance to ensure scalable and reliable ML Applications.

Key Components of DevOps and MLOps :

DevOps :

- Continuous Integration and Continuous Deployment (CI/CD)
- Infrastructure as Code (IaC)
- Automated Testing
- Monitoring and Logging

→ MLOps :

- Model Training and Experimentation
- Model Versioning and Management
- Model Deployment and Monitoring
- Data Pipeline Automation

* Cloud - Based Implementation of DevOps and MLOps :

Cloud platforms like AWS, Google Cloud, and Azure provide various services to facilitate DevOps and MLOps practices.

- For DevOps :

→ AWS Code Pipeline, Azure DevOps, Google Cloud Build of. CI/CD
→ Terraform and AWS CloudFormation for Iac.

Kubernetes for container orchestration
Prometheus and Grafana for monitoring.

For MLOps :

AWS SageMaker, Google Vertex AI, Azure Machine Learning for model training and deployment.
Kubeflow and MLflow for model versioning and tracking
Dataflow and Apache Airflow for automating ML Pipelines
Tensorflow Extended (TFX) for production-grade ML workflows.

Advantages of Cloud-based DevOps and MLOps :

Scalability : Cloud services offer auto scaling and dynamic resource allocation for efficient processing

Cost Efficiency : Pay-as-you-go models reduce infrastructure costs and optimize resource utilization

Automation : CI/CD pipelines and automated model deployment improve efficiency and reduce manual intervention.

Monitoring and Logging : Continuous monitoring and real time logging help track performance and detect anomalies.

Challenges in Cloud-based DevOps and MLOps :

raisoni EDUCATION

- Complexity :- Managing cloud resources and tools requires expertise.
- Security Risks :- Handling sensitive data in the cloud requires robust security measures.
- Cost Management :- Unmonitored cloud usage can lead to high expenses.
- Integration Issues :- Ensuring seamless integration across multiple cloud services can be challenging.

* Future of DevOps and MLOps in the Cloud.

i) AI driven automation :-
ii) Severless Computing
iii) Edge Computing
iv) Compliance - focused MLOps

* Conclusion :-

The implementation of DevOps and MLOps using cloud technologies has revolutionized software development and machine learning workflow

Assignment No : 9

raisoni EDUCATION

Title :

Implement a car detection model using CNN

Objective :

To develop and understand a Convolutional Neural Network (CNN) - based car detection model, explore its architecture and analyze its implementation using deep learning frameworks.

Theory :

Introduction to CNN in Car Detection :

Convolutional Neural Network (CNNs) are class of deep learning models specifically designed for image processing tasks, including object detection model using and classification. A car detection model using CNN can identify and locate cars in images or video streams, making it valuable for applications like autonomous driving, traffic monitoring, and security surveillance.

Key Concepts of CNN for Car Detection

Convolutional Layers : Extract spatial features from input images.

Pooling Layers : Reduce dimensionality and retain essential features.

Fully Connected Layers :: Interpret features and make predictions.

Activation Functions :: Introduce non-linearity for complex decision-making.

Loss Function :: Measures model performance (e.g., cross entropy loss for classification, mean squared error, for bounding box regression

Implementation of car Detection Model

The implementation of car detection model using CNN involves

Data Preparation :: Collecting and annotating images of cars

Model Architecture :: Designing a CNN model or using pre-trained architectures like YOLO, Faster R-CNN, or SSD.

Training Process :: Training the model using ~~pre~~ labeled datasets with techniques like data augmentation and transfer learning.

Testing and Evaluation :: Measuring accuracy using metrics like Intersection over Union (IOU) and mean Average Precision (mAP)

- Deployment :- Deploying the trained model for real-time detection using TensorFlow, OpenCV, or cloud-based APIs.

* Challenges in Car Detection Using CNN

•) Computational Complexity :- Training deep CNN models requires high processing power and GPU acceleration.

•) Data Dependence :- Performance depends on the quality and diversity of training data.

•) Occlusion and Weather Conditions :- Cars may be partially hidden or affected by lighting conditions, impacting detection accuracy.

•) False Positives and Negatives :- The model may misclassify object due to insufficient training data or poor generalization.

* Conclusion :-

CNN-based ~~cardet~~) car detection model have revolutionized object detection, offering high accuracy and real-time capabilities. By leveraging deep learn techniques, these models have applications in autonomous driving, traffic monitoring, and surveillance.

Assignment No : 10

raisoni EDUCATION

Title :

Case Study on Image processing based on Deep Learning.

Objective :

To explore deep learning techniques used in image processing, understand their applications, an analyze their impact on real-word scenarios.

Theory :

Introduction to Deep Learning in Image Processing.

Deep learning has transformed image processing by enabling automatic feature extraction, improve accuracy and reducing manual effort.

Convolutional Neural Networks (CNNs) form the backbone of deep learning-based image processing, allowing models to recognize patterns, detect objects, and enhance images effectively.

Key Concepts in Deep Learning for Image Processing

Convolutional Neural Networks (CNNs) : Extract spatial features from images using convolutional layers.

raisoni EDUCATION

- Autoencoders : Used for image denoising, compression and reconstruction.
- Generative Adversarial Networks (GANs) : Generate realistic images and enhance image quality.
- Transfer Learning : Utilizing pre-trained deep learning models for efficient image processing
- Edge Detection and Feature Extraction : Techniques for detecting objects and patterns in images.

* Methodology :

To conduct this case study, the following methodology was used

1) Data Collection :-

- A dataset of image was obtained from open-source repositories such as ImageNet and Kaggle.
- Pre-processing techniques like resizing, normalization and augmentation were applied

2) Model Selection :-

- A CNN-based architecture such as ResNet, VGG or EfficientNet was chosen for image classification and detection tasks.
- For image enhancement, a GAN-based model was selected

2) Training and Evaluation :

- The models were trained using deep learning framework like TensorFlow and PyTorch
- Training was performed using GPUs to accelerate the process.
- Accuracy, precision, recall, and F-1 -score were used as performance evaluation metrics

3) Implementation and Testing :

- The trained model was deployed on a cloud-based edge computing platform for real-time inference.
- Test datasets were used to validate model predictions and asses generalization capability.

* Observation

1) Accuracy Improvement :

CNN-based models provided high accuracy for object detection achieving over 90% accuracy in certain cases.

- GANs were effective in generating high-resolution images but required extensive training time

2) Processing Time :

- Training deep learning models required significant computational power, but once trained, inference was real-time and efficient.

- GPU-based acceleration significantly reduced processing time compared to CPU-based execution.

3) Challenges Noted :

- Large dataset size increased storage and processing requirements.

- Model overfitting was observed when training datasets were limited or not diverse enough.

- Performance degradation was noticed in cases of poor lighting conditions or image noise.

* Conclusion :

Deep learning has significantly advanced image processing offering enhanced accuracy and automation in multiple applications, from healthcare to security. Despite challenges like data dependency and computational costs continuous research in AI-driven image processing is paving the way for smarter and more efficient image-based technologies.