ASSIGNMENT – 01

Q.1. Consider a scenario, resources are not available to train the model from scratch. In this situation, pre-trained model should be used. Describe the history and applications of these pretrained models.

<u>Ans1</u>

Pre-trained models have transformed the fields of artificial intelligence and machine learning by allowing researchers and developers to use the skills and expertise of already trained models rather than having to start from scratch. These models have a long history and have been used in many different fields. Pre-trained models are particularly useful in situations where resources are few since they can give users a head start on a variety of activities.

<u>History of Pre-trained Models:</u>

Pre-trained models have their roots in the early days of machine learning, when researchers first began to explore the possibilities of transfer learning. Transfer learning is the process of teaching a model to perform one task and then using that information to perform another, related activity. But it was the development of deep learning and neural networks that marked the actual breakthrough.

The creation of Word2Vec by Tomas Mikolov and his team at Google in 2013 was one of the seminal events in the history of pre-trained models. Word2Vec learned word embeddings from a sizable corpus of text using a shallow neural network, which could then be applied to other NLP applications. This signaled the start of NLP's pre-trained model era.

Convolutional neural networks (CNNs) for computer vision were first used for transfer learning in 2015 by Stanford University researchers. The model, known as VGG-Face, produced remarkable performance on face recognition tasks after being pre-trained on a sizable dataset of human faces.

In 2018, Google AI introduced large-scale pre-trained models like BERT (Bidirectional Encoder Representations from Transformers), which marked a significant advancement. BERT, which had been pre-trained on a sizable corpus of text, displayed exceptional performance on a variety of NLP tasks, pave the way for the explosion of pre-trained models in a variety of fields.

Applications of Pre-trained Models:

Natural Language Processing (NLP):

For tasks including text classification, sentiment analysis, machine translation, and question-answering, pre-trained models like BERT, GPT-3, and RoBERTa have been widely employed. They have transformed chatbots and virtual assistants, giving them the ability to comprehend and produce content that is more human-like.

Computer Vision:

Models like VGG-Face, ResNet, and Inception have been pre-trained on enormous datasets and optimized for certain tasks like image classification, object identification, and picture segmentation in the field of computer vision. The performance of computer vision systems has substantially increased thanks to these pre-trained models.

Speech Recognition:

Pre-trained models, like DeepSpeech, have been used for speech recognition tasks, making it possible to create highly accurate voice assistants and transcription services.

Recommendation Systems:

In a variety of industries, including as e-commerce, content streaming, and social media, pre-trained models can improve recommendation systems. They are able to pick up on intricate user preferences and provide more specialized recommendations.

Healthcare:

Pre-trained models have been used in the medical industry for things like disease diagnosis, drug discovery, and medical picture analysis. These models can expedite diagnosis while also increasing their precision.

Autonomous Vehicles:

For functions like object detection, lane detection, and pedestrian recognition, self-driving cars use pre-trained models, making the technology safer and more dependable.

Q.2. Describe loss function, backpropagation, hyperparameters etc. in Neural Networks.

Ans2

Loss function:

A loss function is a mathematical function that measures how well a neural network model performs on a given task. It is used to calculate the error between the model's predicted outputs and the actual target outputs. The goal of training a neural network is to minimize the loss function.

There are many different loss functions that can be used for neural networks, but the most common ones are:

Mean squared error (MSE):

This loss function is used for regression tasks, where the model is trying to predict a continuous value. It is calculated by averaging the squares of the errors between the predicted and target outputs.

Cross-entropy loss:

This loss function is used for classification tasks, where the model is trying to predict a category. It is calculated by measuring the difference between the predicted probability distribution and the actual target distribution.

Backpropagation:

Backpropagation is an algorithm used to train neural networks. It works by calculating the gradient of the loss function with respect to the model's parameters (weights and biases). This gradient is then used to update the parameters in a way that reduces the loss function.

Backpropagation works by propagating the error from the output layer of the network to the input layer. At each layer, the error is used to calculate the gradient of the loss function with respect to the parameters of that layer. The parameters are then updated using the gradient descent algorithm.

Hyperparameters:

Hyperparameters are parameters of a neural network that are not learned from the data, but are instead set by the user before training.

Some common hyperparameters include:

Learning rate: This controls how quickly the model updates its parameters during training.

Batch size: This controls how many training examples are processed at once during training.
Number of hidden layers: This controls how complex the model is.
Number of neurons per hidden layer: This controls how much capacity the model has to learn complex patterns.

How loss function, backpropagation, and hyperparameters work together

The loss function, backpropagation, and hyperparameters all work together to train a neural network. The loss function measures how well the model is performing on the training data, backpropagation calculates the gradient of the loss function with respect to the model's parameters, and the hyperparameters control how the model is updated during training.

The training process for a neural network is as follows:

A batch of training examples is fed into the network.
The network's output is calculated.
The loss function is calculated to measure how well the model performed.
Backpropagation is used to calculate the gradient of the loss function with respect to the model's parameters.
The model's parameters are updated using the gradient descent algorithm.
Steps 1-5 are repeated until the model converges, meaning that the loss function is no longer decreasing significantly.

Example:

Suppose we are training a neural network to classify images of cats and dogs. We have a training dataset of images, each of which is labeled as either a cat or a dog.

We first need to choose a loss function. Since we are solving a classification problem, we will use the cross-entropy loss function.

We then need to initialize the model's parameters. This can be done randomly or using a pre-trained model.

We can now start the training process. In each iteration of the training process, we will feed a batch of training images to the network and calculate its output. We will then calculate the loss function to measure how well the model performed.

Next, we will use backpropagation to calculate the gradient of the loss function with respect to the model's parameters. Finally, we will update the model's parameters using the gradient descent algorithm.

We will repeat this process until the model converges, meaning that the loss function is no longer decreasing significantly.

Once the model has been trained, we can use it to classify new images of cats and dogs. The model will take an input image and calculate its output, which is the predicted category (cat or dog).