**Module 2 assignment ML with graphs**

**1.“Deep learning models are capable enough to focus on the accurate features themselves by requiring a little guidance from the programmer” Critically analyse this sentence**

Ans.

Deep learning is based on the branch of machine learning, which is a subset of artificial intelligence. Since neural networks imitate the human brain and so deep learning will do. In deep learning, nothing is programmed explicitly. Basically, it is a machine learning class that makes use of numerous nonlinear processing units so as to perform feature extraction as well as transformation. The output from each preceding layer is taken as input by each one of the successive layers.

Deep learning models are capable enough to focus on the accurate features themselves by requiring a little guidance from the programmer and are very helpful in solving out the problem of dimensionality. Deep learning algorithms are used, especially when we have a huge no of inputs and outputs. Since deep learning has been evolved by the machine learning, which itself is a subset of artificial intelligence and as the idea behind the artificial intelligence is to mimic the human behavior, so same is "the idea of deep learning to build such algorithm that can mimic the brain".

Deep learning is implemented with the help of Neural Networks, and the idea behind the motivation of Neural network is the biological neurons, which is nothing but a brain cell Deep learning is a collection of statistical techniques of machine learning for learning feature hierarchies that are actually based on artificial neural networks. So basically, deep learning is implemented by the help of deep networks, which are nothing but neural networks with multiple hidden layers.

**2. Critically analyse various Deep Learning Network, and explain the working of Artificial Neural Networks.**

Ans.

Neural networks, also known as artificial neural networks (ANNs) or simulated neural networks (SNNs), are a subset of machine learning and are at the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another.

Artificial neural networks (ANNs) are comprised of a node layers, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.



Neural networks rely on training data to learn and improve their accuracy over time. However, once these learning algorithms are fine-tuned for accuracy, they are powerful tools in computer science and artificial intelligence, allowing us to classify and cluster data at a high velocity. Tasks in speech recognition or image recognition can take minutes versus hours when compared to the manual identification by human experts. One of the most well-known neural networks is Google’s search algorithm.

Think of each individual node as its own linear regression model, composed of input data, weights, a bias (or threshold), and an output. The formula would look something like this:

Mathematical formula used to determine summation

∑wixi + bias = w1x1 + w2x2 + w3x3 + bias

Mathematical formula used to determine the output

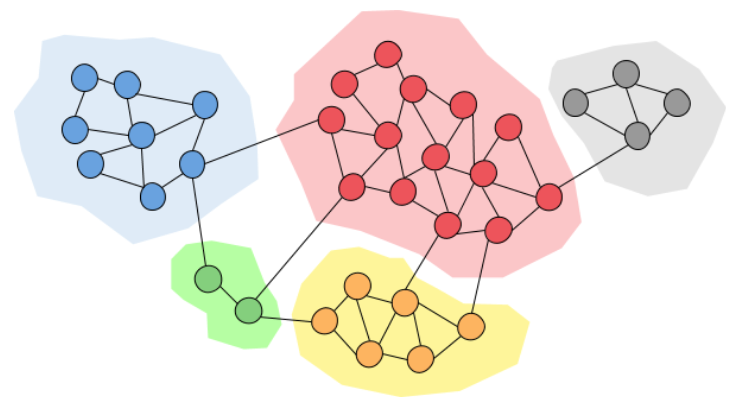
output = f(x) = 1 if ∑w1x1 + b>= 0; 0 if ∑w1x1 + b < 0

Once an input layer is determined, weights are assigned. These weights help determine the importance of any given variable, with larger ones contributing more significantly to the output compared to other inputs. All inputs are then multiplied by their respective weights and then summed. Afterward, the output is passed through an activation function, which determines the output. If that output exceeds a given threshold, it “fires” (or activates) the node, passing data to the next layer in the network. This results in the output of one node becoming in the input of the next node. This process of passing data from one layer to the next layer defines this neural network as a feedforward network.

**3. Specify the need for Community detection and discuss the various techniques for the same.**

Ans.

Community Detection Algorithms



Many of you are familiar with networks, right? You might be using social media sites such as Facebook, Instagram, Twitter, etc. They are social networks. You might be dealing with stock exchanges. Either you might be buying new stocks, selling what you already have, etc. They are networks. Not only in the technological field but also in our day to day social life, we deal with many networks. Communities are a property of many networks in which a particular network may have multiple communities such that nodes inside a community are densely connected. Nodes in multiple communities can overlap. Think of your Facebook or Instagram account and consider who you interact with daily. You might be heavily interacting with your friends, colleagues, family members and a few other important people in your life. They form a very dense community inside your social network.

When analyzing different networks, it may be important to discover communities inside them. Community detection techniques are useful for social media algorithms to discover people with common interests and keep them tightly connected. Community detection can be used in machine learning to detect groups with similar properties and extract groups for various reasons. For example, this technique can be used to discover manipulative groups inside a social network or a stock market.

Community detection methods can be broadly categorized into two types; Agglomerative Methods and Divisive Methods. In Agglomerative methods, edges are added one by one to a graph which only contains nodes. Edges are added from the stronger edge to the weaker edge. Divisive methods follow the opposite of agglomerative methods. In there, edges are removed one by one from a complete graph.

There can be any number of communities in a given network and they can be of varying sizes. These characteristics make the detection procedure of communities very hard. However, there are many different techniques proposed in the domain of community detection. Four popular community detection algorithms are explained below. All of these listed algorithms can be found in the python cdlib library.

**1. Louvain Community Detection**

Louvain community detection algorithm was originally proposed in 2008 as a fast community unfolding method for large networks. This approach is based on modularity, which tries to maximize the difference between the actual number of edges in a community and the expected number of edges in the community. However optimizing modularity in a network is NP-hard, therefore have to use heuristics. Louvain algorithm is divided into iteratively repeating two phases;

1. Local moving of nodes
2. Aggregation of the network

The algorithm starts with a weighted network of N nodes. In the first phase, the algorithm assigns a different community to each node of the network. Then for each node, it considered the neighbours and evaluate the gain of modularity by removing the particular node from the current community and placing in the neighbour’s community. The node will be placed in the neighbour’s community if the gain is positive and maximized. The node will remain in the same community if there is no positive gain. This process is applied repeatedly and for all nodes until no further improvement is there. The first phase of the Louvain algorithm stops when a local maxima of modularity is obtained. In the second phase, the algorithm builds a new network considering communities found in the first phase as nodes. Once the second phase is completed, the algorithm will reapply the first phase to the resulting network. These steps are repeated until there are no changes in the network and maximum modularity is obtained.

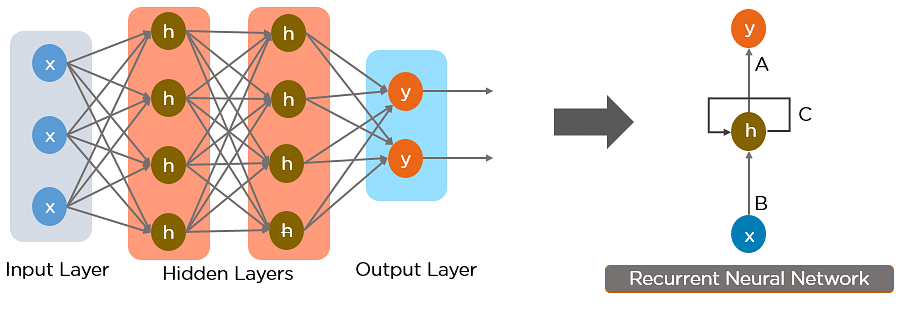
**2. Surprise Community Detection**

Due to limitations of the modularity, a measure based on classical probabilities known as Surprise has been introduced to evaluate the quality of a partition of a network into communities. The algorithm is almost similar to the Louvain community detection algorithm except that it uses surprises instead of modularity. Nodes are moved from one community to another such that surprises are greedily improved. This approach considers the probability that a link lies within a community. The use of surprises works well in the limit of many small communities and the use of modularity works well in the limit of a few large communities.

**4. Explain the Recurrent Neural Networks with the help of real life example, also describe how to train RNN?**

Ans.

RNN works on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer. Below is how you can convert a Feed-Forward Neural Network into a Recurrent Neural Network:



Here, “x” is the input layer, “h” is the hidden layer, and “y” is the output layer. A, B, and C are the network parameters used to improve the output of the model. At any given time t, the current input is a combination of input at x(t) and x(t-1). The output at any given time is fetched back to the network to improve on the output.

Diagram

Description automatically generated

RNN were created because there were a few issues in the feed-forward neural network:

Cannot handle sequential data

Considers only the current input

Cannot memorize previous inputs

The solution to these issues is the RNN. An RNN can handle sequential data, accepting the current input data, and previously received inputs. RNNs can memorize previous inputs due to their internal memory.

**Two Issues of Standard RNNs**

1. Vanishing Gradient Problem

Recurrent Neural Networks enable you to model time-dependent and sequential data problems, such as stock market prediction, machine translation, and text generation. You will find, however, RNN is hard to train because of the gradient problem.

RNNs suffer from the problem of vanishing gradients. The gradients carry information used in the RNN, and when the gradient becomes too small, the parameter updates become insignificant. This makes the learning of long data sequences difficult.

2. Exploding Gradient Problem

While training a neural network, if the slope tends to grow exponentially instead of decaying, this is called an Exploding Gradient. This problem arises when large error gradients accumulate, resulting in very large updates to the neural network model weights during the training process.

Long training time, poor performance, and bad accuracy are the major issues in gradient problems.

**5. Discuss about Backpropagation also analyse factors that are responsible for the training and performance of the Neural network**

Ans.

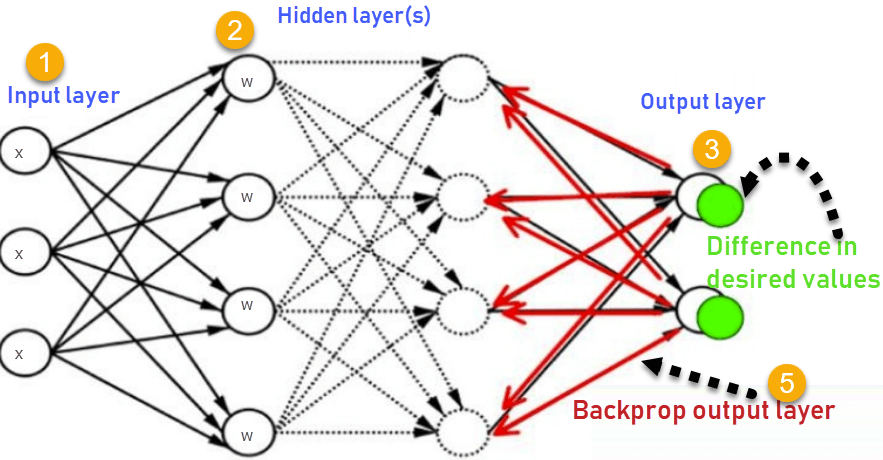
Backpropagation is the essence of neural network training. It is the method of fine-tuning the weights of a neural network based on the error rate obtained in the previous epoch (i.e., iteration). Proper tuning of the weights allows you to reduce error rates and make the model reliable by increasing its generalization.

Backpropagation in neural network is a short form for “backward propagation of errors.” It is a standard method of training artificial neural networks. This method helps calculate the gradient of a loss function with respect to all the weights in the network.

**How Backpropagation Algorithm Works**

The Back propagation algorithm in neural network computes the gradient of the loss function for a single weight by the chain rule. It efficiently computes one layer at a time, unlike a native direct computation. It computes the gradient, but it does not define how the gradient is used. It generalizes the computation in the delta rule.

Consider the following Back propagation neural network example diagram to understand:

How Backpropagation Algorithm Works

1. Inputs X, arrive through the preconnected path
2. Input is modeled using real weights W. The weights are usually randomly selected.
3. Calculate the output for every neuron from the input layer, to the hidden layers, to the output layer.
4. Calculate the error in the outputs

**6. Discuss the need and working of Convolutional Neural Networks with suitable diagram.**

Ans.

A Convolutional Neural Network, also known as CNN or ConvNet, is a class of [neural networks](https://datascience.hubs.vidyard.com/watch/CYfbzzj57RPfCwoMnEHD4M) that specializes in processing data that has a grid-like topology, such as an image. A digital image is a binary representation of visual data. It contains a series of pixels arranged in a grid-like fashion that contains pixel values to denote how bright and what color each pixel should be.

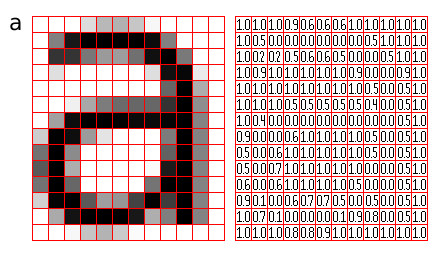


Figure 1: Representation of image as a grid of pixels ([Source](http://pippin.gimp.org/image_processing/images/sample_grid_a_square.png))

The human brain processes a huge amount of information the second we see an image. Each neuron works in its own receptive field and is connected to other neurons in a way that they cover the entire visual field. Just as each neuron responds to stimuli only in the restricted region of the visual field called the receptive field in the biological vision system, each neuron in a CNN processes data only in its receptive field as well. The layers are arranged in such a way so that they detect simpler patterns first (lines, curves, etc.) and more complex patterns (faces, objects, etc.) further along. By using a CNN, one can enable sights of computers.

**Convolutional Neural Network Architecture**

A CNN typically has three layers: a convolutional layer, a pooling layer, and a fully connected layer.



Figure 2: Architecture of a CNN ([Source](https://www.mathworks.com/videos/introduction-to-deep-learning-what-are-convolutional-neural-networks--1489512765771.html))

**Convolution Layer**

The convolution layer is the core building block of the CNN. It carries the main portion of the network’s computational load.

This layer performs a dot product between two matrices, where one matrix is the set of learnable parameters otherwise known as a kernel, and the other matrix is the restricted portion of the receptive field. The kernel is spatially smaller than an image but is more in-depth. This means that, if the image is composed of three (RGB) channels, the kernel height and width will be spatially small, but the depth extends up to all three channels.

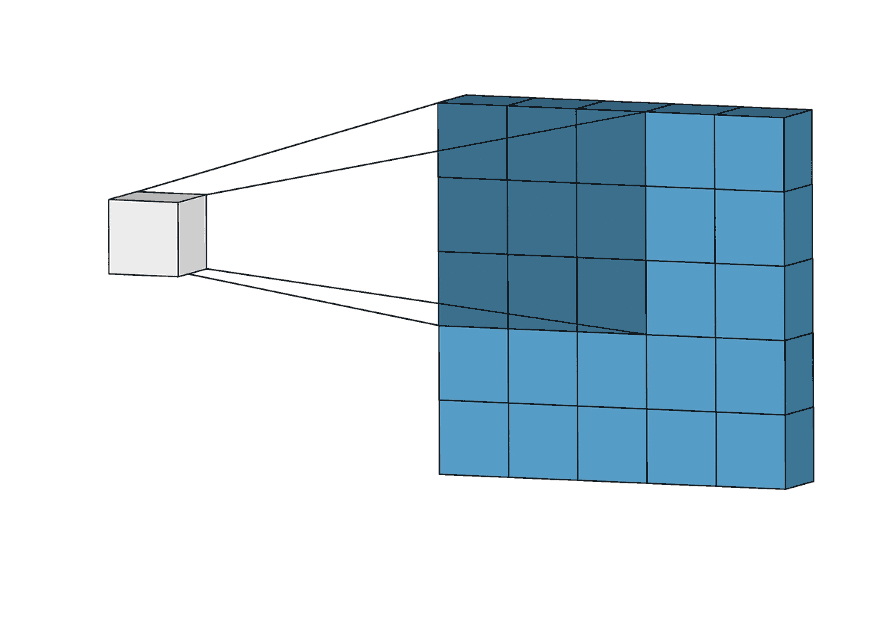


Illustration of Convolution Operation ([source](https://miro.medium.com/max/2340/1*Fw-ehcNBR9byHtho-Rxbtw.gif))

During the forward pass, the kernel slides across the height and width of the image-producing the image representation of that receptive region. This produces a two-dimensional representation of the image known as an activation map that gives the response of the kernel at each spatial position of the image. The sliding size of the kernel is called a stride.

**Pooling Layer**

The pooling layer replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs. This helps in reducing the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually.

There are several pooling functions such as the average of the rectangular neighborhood, L2 norm of the rectangular neighborhood, and a weighted average based on the distance from the central pixel. However, the most popular process is max pooling, which reports the maximum output from the neighborhood.

Table

Description automatically generated

**7. Explain the Training process of Recurrent Neural Network with proper steps**

Ans.

A single time step of the input is provided to the network.

Then calculate its current state using set of current input and the previous state.

The current ht becomes ht-1 for the next time step.

One can go as many time steps according to the problem and join the information from all the previous states.

Once all the time steps are completed the final current state is used to calculate the output.

The output is then compared to the actual output i.e the target output and the error is generated.

The error is then back-propagated to the network to update the weights and hence the network (RNN) is trained.

**8. Critically analyse the need of Community detection, also highlight the difference between Surprise Community detection and Louvain community detection algorithm**

**Ans.**

Community detection algorithms are used to evaluate how groups of nodes are clustered or partitioned, as well as their tendency to strengthen or break apart.

**Louvain community detection**

The Louvain method for community detection is an algorithm for detecting communities in networks. It maximizes a modularity score for each community, where the modularity quantifies the quality of an assignment of nodes to communities. This means that the algorithm evaluates how much more densely connected the nodes within a community are, compared to how connected they would be in a random network. On each iteration the Louvain algorithm recursively merges communities into a single node and executes the modularity clustering on the condensed graphs. It has the following use cases:

Providing recommendations for Reddit users to find similar subreddits, based on the general user behavior. For more details, see "Subreddit Recommendations within Reddit Communities".

Extracting topics from online social platforms, such as Twitter and Youtube, based on the co-occurence graph of terms in documents, as a part of Topic Modeling process. This process is described in "Topic Modeling based on Louvain method in Online Social Networks".

Investigating the human brain and finding hierarchical community structures within the brain’s functional network. The study mentioned is "Hierarchical Modularity in Human Brain Functional Networks"

**Surprise Community Detection**

Due to limitations of the modularity, a measure based on classical probabilities known as Surprise has been introduced to evaluate the quality of a partition of a network into communities. The algorithm is almost similar to the Louvain community detection algorithm except that it uses surprises instead of modularity. Nodes are moved from one community to another such that surprises are greedily improved. This approach considers the probability that a link lies within a community. The use of surprises works well in the limit of many small communities and the use of modularity works well in the limit of a few large communities.