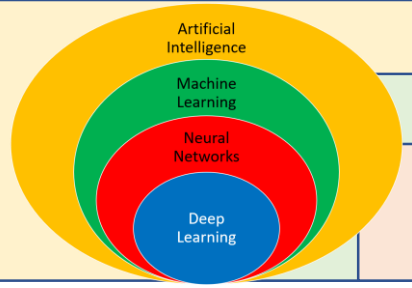


Artificial Intelligence (AI)

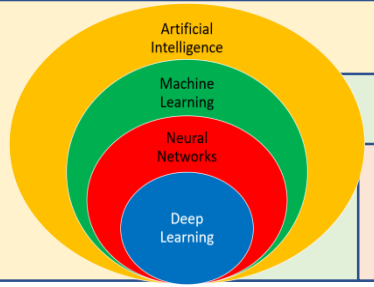
Machine Learning (ML)

Neural Networks (NNs)

Deep Learning (DL)

Advanced Artificial Intelligence

Dr. Rastgoo



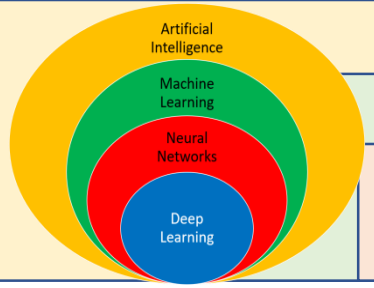
Artificial Intelligence (AI)

Machine Learning (ML)

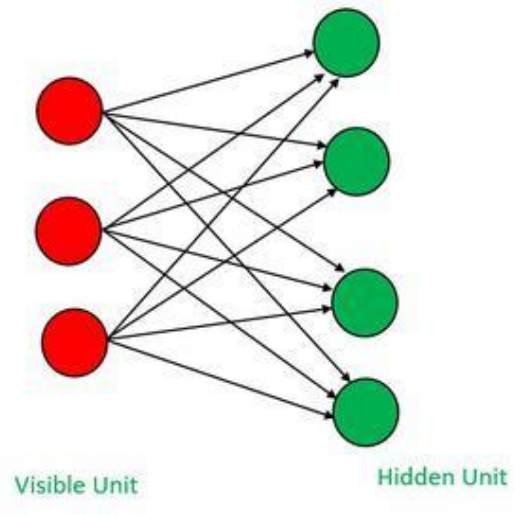
Neural Networks (NNs)

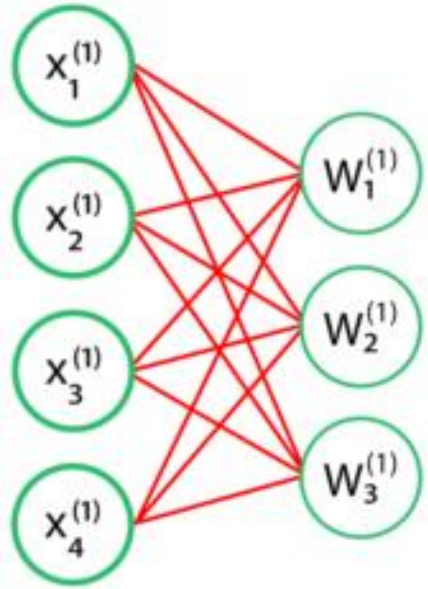
Deep Learning (DL)

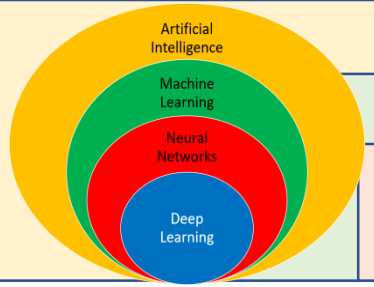
Generative models



Part 3: Restricted Boltzmann Machine (RBM)

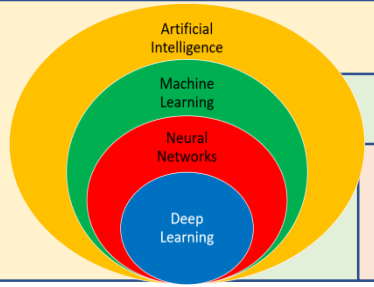






Introduction

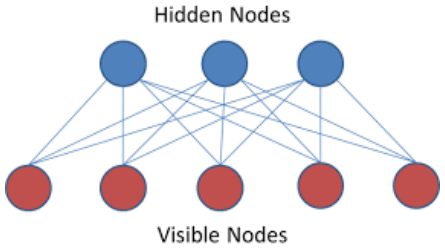
- RBM is a variant of Boltzmann Machine.
- RBM was invented in 1986.
- In the mid-2000, Geoffrey Hinton and collaborators invented fast learning algorithms which were commercially successful.

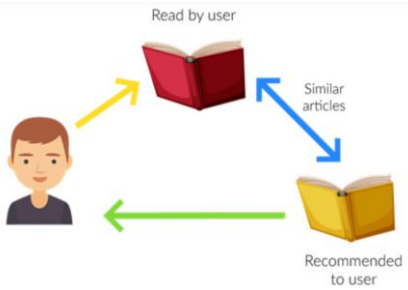


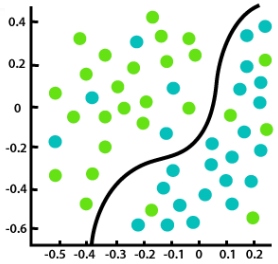
Introduction

The most significant difference between regression vs classification is that while regression helps predict a continuous quantity, classification predicts discrete class labels.

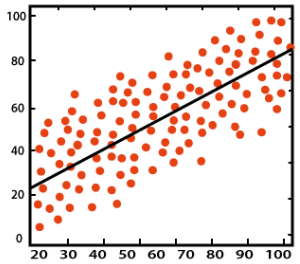
- RBM can be use in many applications like Dimensionality reduction , Collaborative Filtering, Feature Learning, Regression Classification, and Topic Modeling.



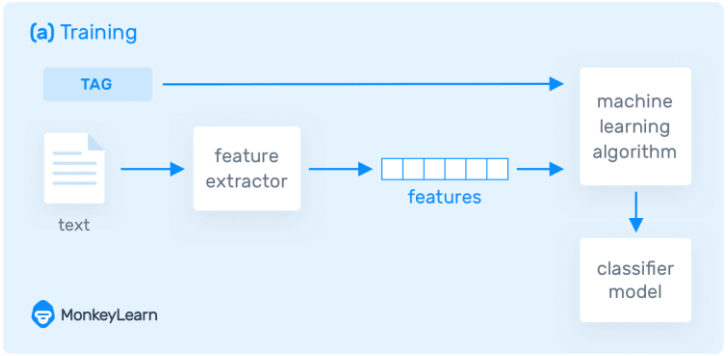




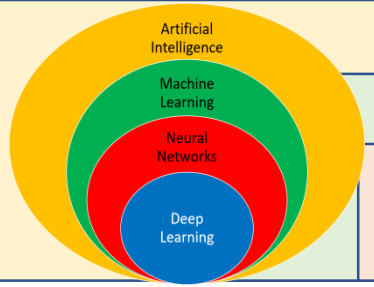
Classification



Regression

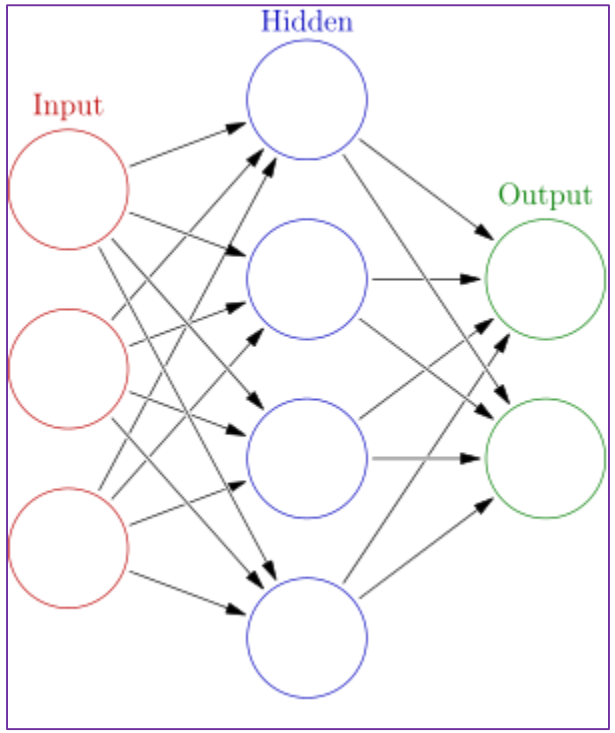


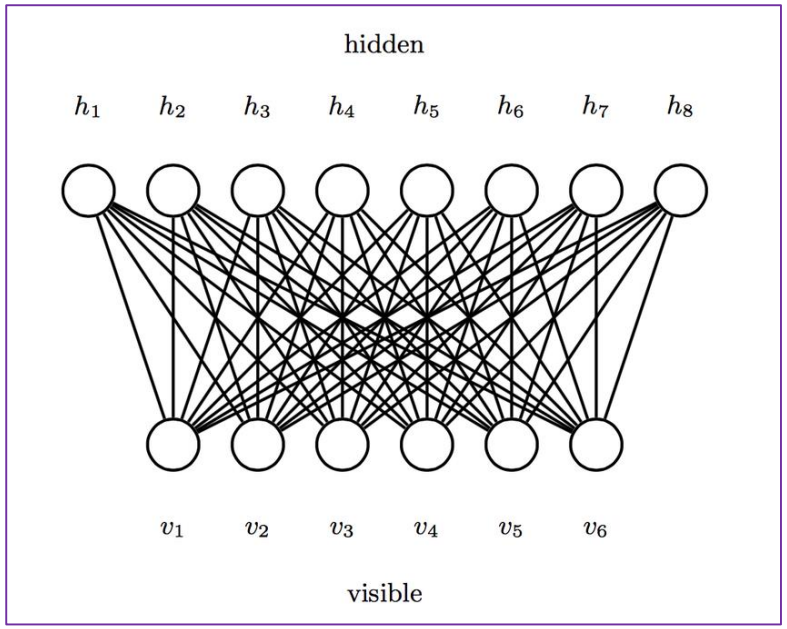
Topic modeling is a machine learning technique that automatically analyzes text data to determine cluster words for a set of documents. This is known as 'unsupervised' machine learning because it doesn't require a predefined list of tags or training data that's been previously classified by humans.



Introduction

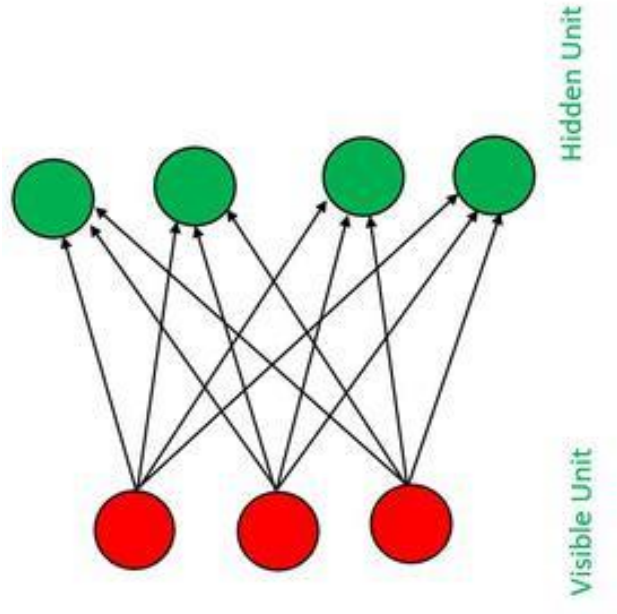
- It can be trained in either Supervised or Unsupervised ways, depending on the task.





RBM's Architecture

- This Restricted Boltzmann Machine (RBM) have an input layer (also referred to as the visible layer) and one single hidden layer and the connections among the neurons are restricted.



RBM's Architecture

- Neurons are connected only to the neurons in other layers but not to neurons within the same layer.
- There are no connections among visible neurons to visible neurons.
- There are no connections among hidden neurons to hidden neurons.
- In RBM visible and hidden neurons connections form a bipartite graph.
- An RBM is considered "restricted" because no two nodes in the same layer share a connection

How does RBM work?

- RBM has two phases:
- Forward Pass,
- Backward Pass or Reconstruction.

How does RBM work?

- RBM takes the inputs and translates them to a set of numbers that represents them (forward pass). Then, these numbers can be translated back to reconstruct the inputs (backward pass).
- In the forward pass, an RBM takes the inputs and translates them into a set of numbers that encode the inputs.
- In the backward pass, it takes this set of numbers and translates them back to form the re-constructed inputs.

How does RBM work?

- Through several forward and backward passes, an RBM is trained to reconstruct the input data.
- Three steps are repeated over and over through the training process.
- (a) With a forward pass, every input is combined with an individual weight and one overall bias, and the result is passed to the hidden layer which may or may not activate.
- (b) Next, in a backward pass, each activation is combined with an individual weight and an overall bias, and the result is passed to the visible layer for reconstruction.
- (c) At the visible layer, the reconstruction is compared against the original input to determine the quality of the result.

How does RBM work?

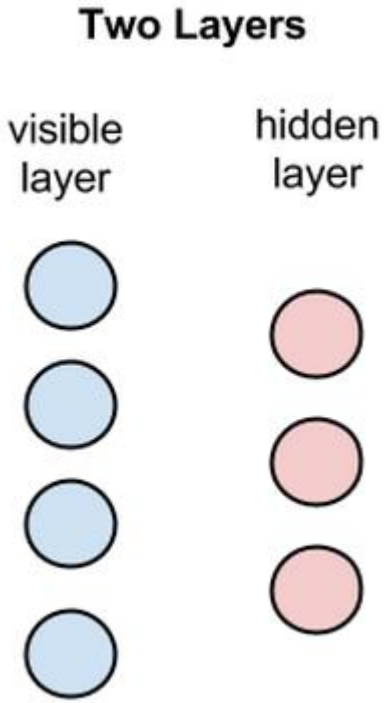
- A trained RBM can reveal which features are the most important ones when detecting patterns.
- A well-trained net will be able to perform the backwards translation with a high degree of accuracy.
- In both steps, the weights and biases have a very important role.
- They allow the RBM to decipher the interrelationships among the input features, and they also help the RBM decide which input features are the most important when detecting patterns.

How does RBM work?

- An important note is that an RBM is actually making decisions about which input features are important and how they should be combined to form patterns.
- In other words, an RBM is part of a family of feature extractor neural nets, which are all designed to recognize inherent patterns in data.
- These nets are also called autoencoders, because in a way, they have to encode their own structure.
- Note : RBMs use a stochastic approach to learning the underlying structure of data, whereas autoencoders, for example, use a deterministic approach.

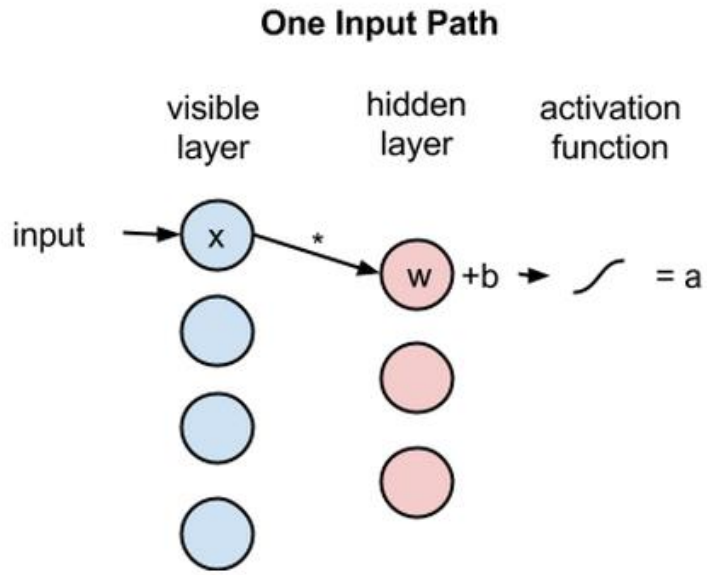
How does RBM work?

- Each visible node takes a low-level feature from an item in the dataset to be learned.
- For example, from a dataset of grayscale images, each visible node would receive one pixel-value for each pixel in one image.
- (MNIST images have 784 pixels, so neural nets processing them must have 784 input nodes on the visible layer.)



How does RBM work?

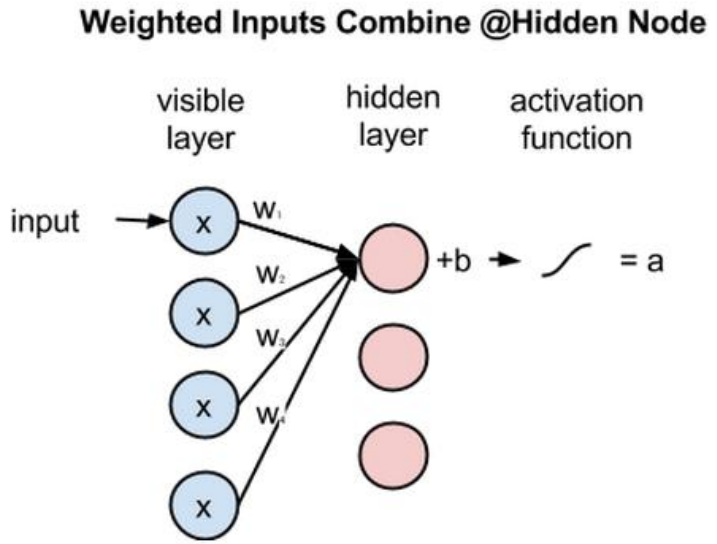
- Now let's follow that single pixel value, x, through the two-layer net.
- At node 1 of the hidden layer, x is multiplied by a weight and added to a so-called bias.
- The result of those two operations is fed into an activation function, which produces the node's output, or the strength of the signal passing through it, given input x.



activation f((weight w * input x) + bias b) = output a

How does RBM work?

- Next, let's look at how several inputs would combine at one hidden node.
- Each x is multiplied by a separate weight, the products are summed, added to a bias, and again the result is passed through an activation function to produce the node's output.



How does RBM work?

- Because inputs from all visible nodes are being passed to all hidden nodes, an RBM can be defined as a symmetrical bipartite graph.
- Symmetrical means that each visible node is connected with each hidden node.
- Bipartite means it has two parts, or layers, and the graph is a mathematical term for a web of nodes.

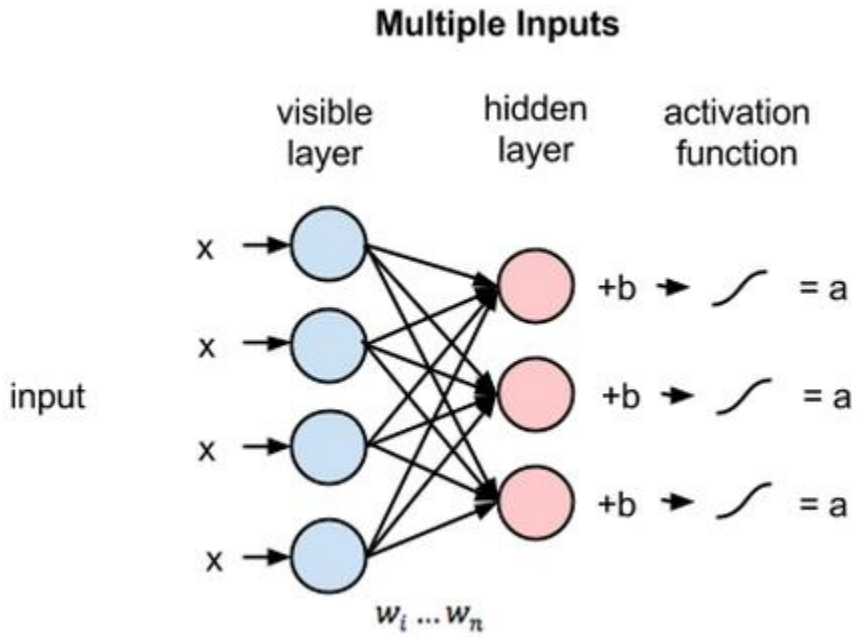
How does RBM work?

- At each hidden node, each input x is multiplied by its respective weight w.
- That is, a single input x would have three weights here, making 12 weights altogether (4 input nodes x 3 hidden nodes).
- The weights between two layers will always form a matrix where the rows are equal to the input nodes, and the columns are equal to the output nodes.

How does RBM work?

- Each hidden node receives the four inputs multiplied by their respective weights.

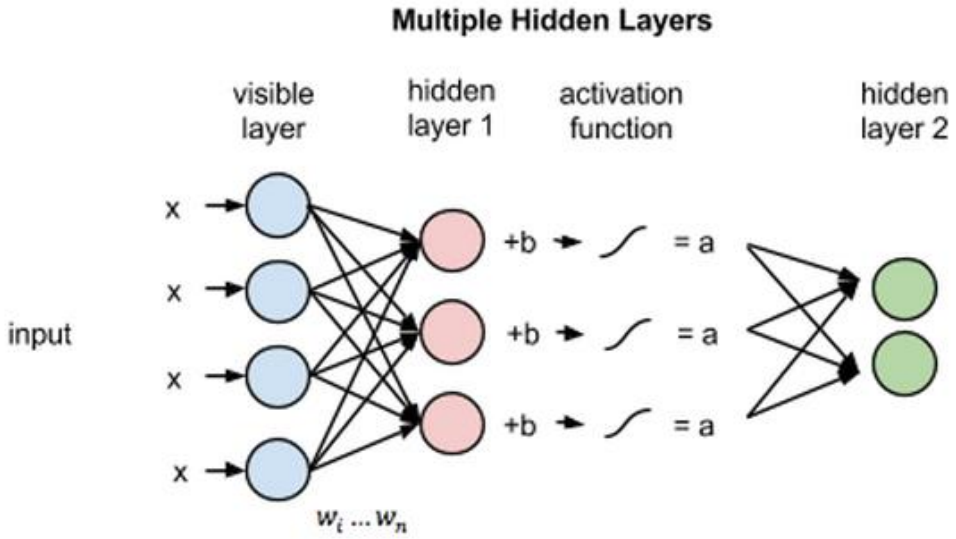
- The sum of those products is again added to a bias (which forces at least some activations to happen), and the result is passed through the activation algorithm producing one output for each hidden node.



How does RBM work?

- If these two layers were part of a deeper neural network, the outputs of hidden layer no. 1 would be passed as inputs to hidden layer no. 2, and from there through as many hidden layers as you like until they reach a final classifying layer.

- (For simple feed-forward movements, the RBM nodes function as an autoencoder and nothing more.)



Reconstructions

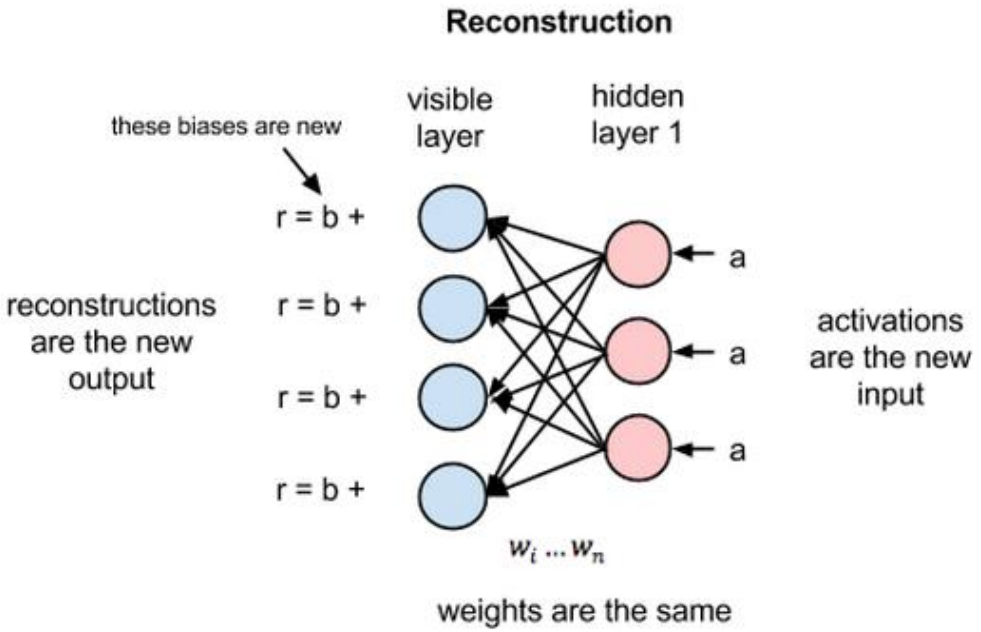
- Here, we'll focus on how they learn to reconstruct data by themselves in an unsupervised fashion (unsupervised means without ground-truth labels in a test set), making several forward and backward passes between the visible layer and hidden layer no. 1 without involving a deeper network.

- In the reconstruction phase, the activations of hidden layer no. 1 become the input in a backward pass.

- They are multiplied by the same weights, one per internode edge, just as x was weight-adjusted on the forward pass.

Reconstructions

- The sum of those products is added to a visible-layer bias at each visible node, and the output of those operations is a reconstruction; i.e. an approximation of the original input. This can be represented by the following diagram:



Reconstructions

- Because the weights of the RBM are randomly initialized, the difference between the reconstructions and the original input is often large.

- You can think of reconstruction error as the difference between the values of r and the input values, and that error is then backpropagated against the RBM's weights, again and again, in an iterative learning process until an error minimum is reached.

Reconstructions

- As you can see, on its forward pass, an RBM uses inputs to make predictions about node activations, or the probability of output given a weighted x: p(a | x; w).

- But on its backward pass, when activations are fed in and reconstructions, or guesses about the original data, are spit out, an RBM is attempting to estimate the probability of inputs x given activations a, which are weighted with the same coefficients as those used on the forward pass. This second phase can be expressed as p(x | a; w).

- Together, those two estimates will lead you to the joint probability distribution of inputs x and activations a, or p(x, a).

Reconstructions

- Reconstruction does something different from regression, which estimates a continuous value based on many inputs, and different from classification, which makes guesses about which discrete label to apply to a given input example.

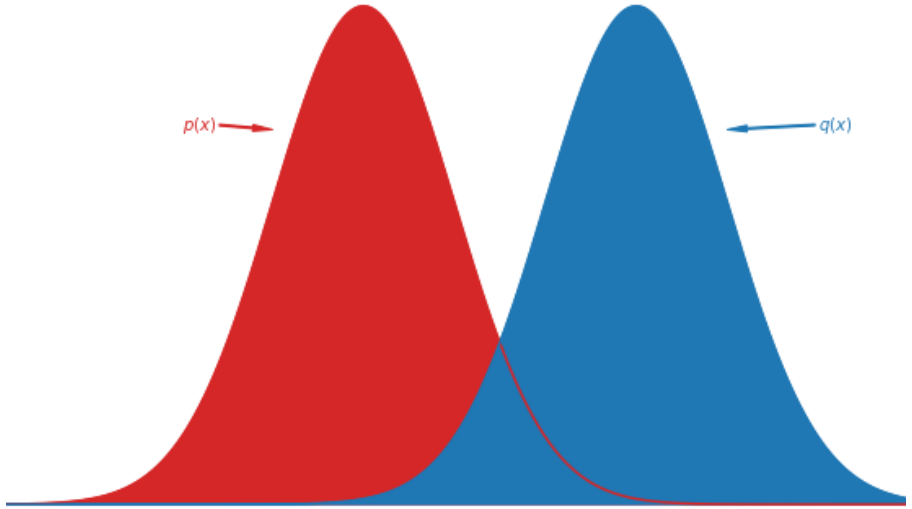
- Reconstruction is making guesses about the probability distribution of the original input; i.e. the values of many varied points at once.

- This is known as generative learning, which must be distinguished from the so-called discriminative learning performed by classification, which maps inputs to labels, effectively drawing lines between groups of data points.

Reconstructions

- Let's imagine that both the input data and the reconstructions are normal curves of different shapes, which only partially overlap.

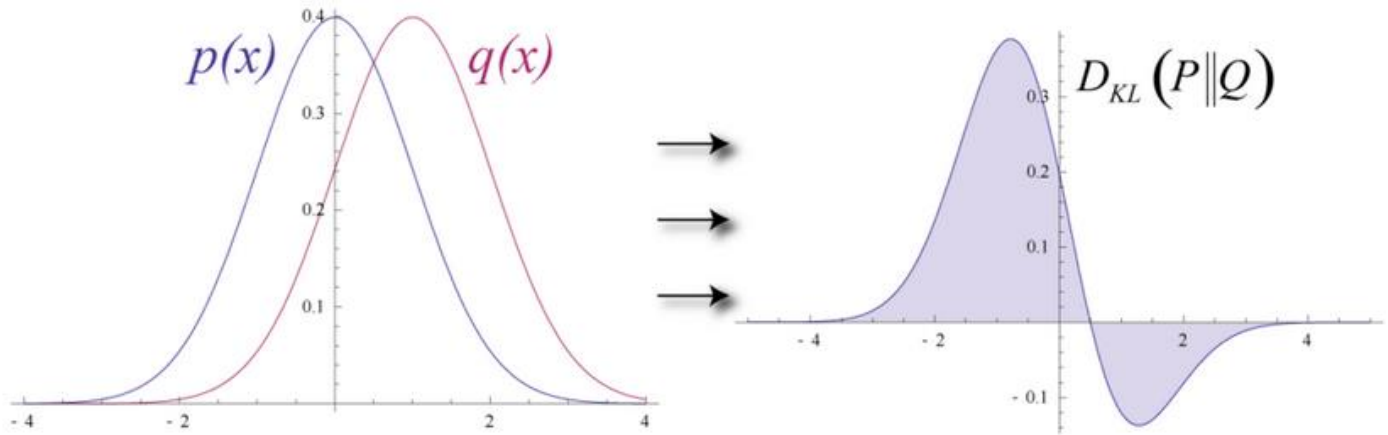
- To measure the distance between its estimated probability distribution and the ground-truth distribution of the input, RBMs use Kullback Leibler Divergence.



Reconstructions

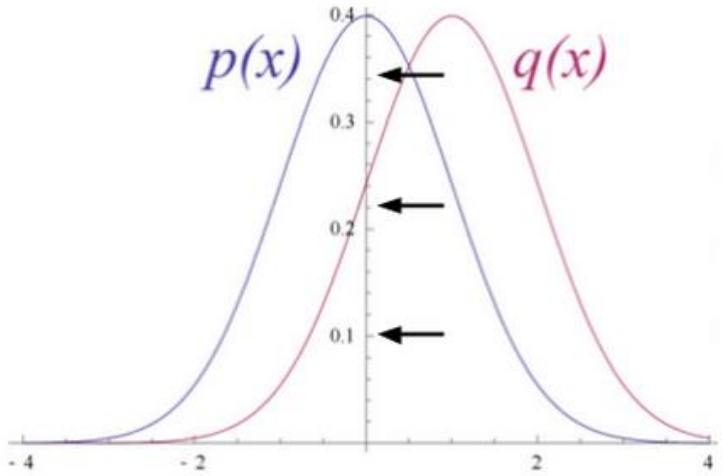
- KL-Divergence measures the non-overlapping, or diverging, areas under the two curves, and an RBM's optimization algorithm attempts to minimize those areas so that the shared weights, when multiplied by activations of hidden layer one, produce a close approximation of the original input.

- On the left is the probability distribution of a set of original input, p, juxtaposed with the reconstructed distribution q; on the right, the integration of their differences.



Reconstructions

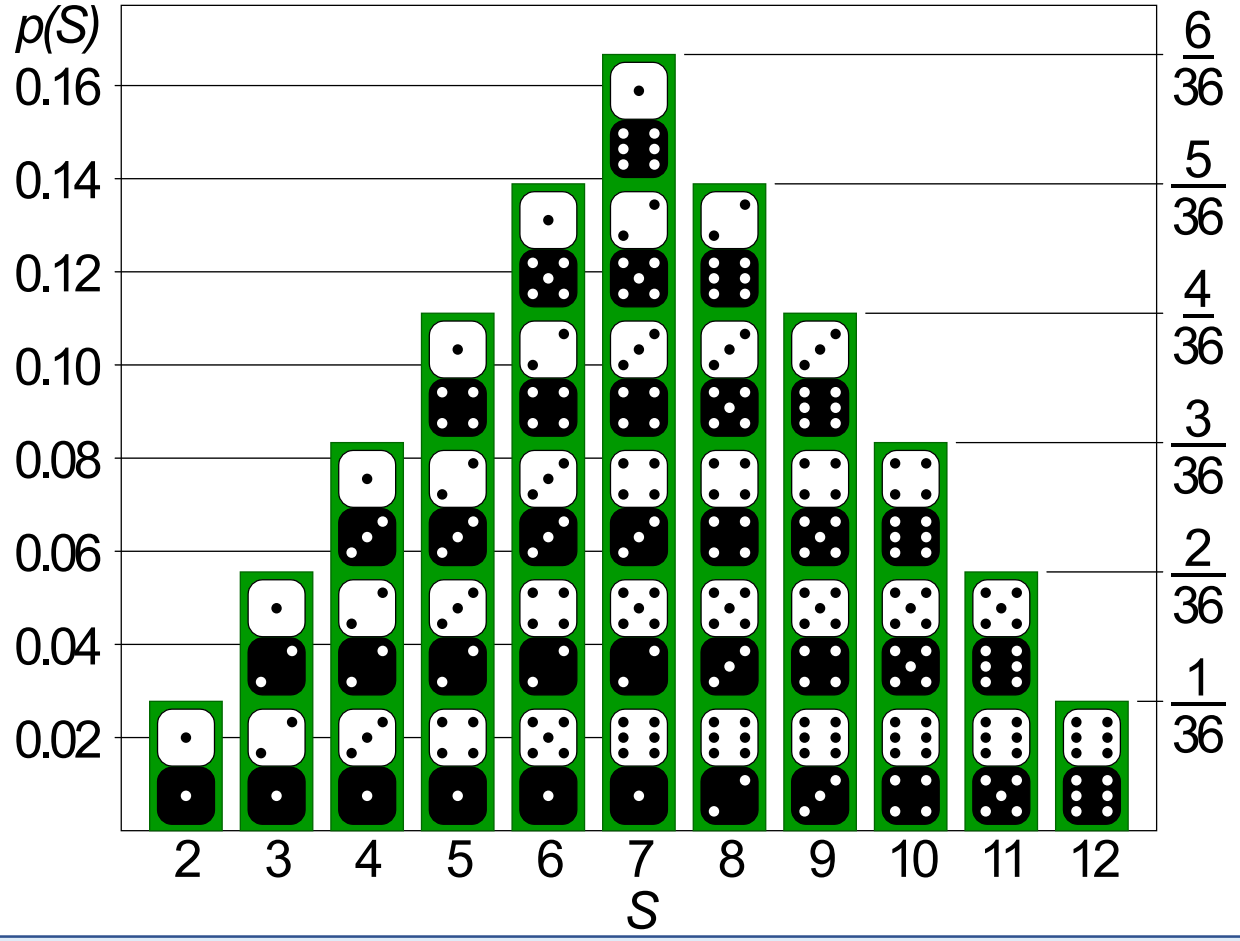
- By iteratively adjusting the weights according to the error they produce, an RBM learns to approximate the original data.
- You could say that the weights slowly come to reflect the structure of the input, which is encoded in the activations of the first hidden layer.
- The learning process looks like two probability distributions converging, step by step.



Probability Distributions

- Let's talk about probability distributions for a moment.

- If you're rolling two dice, the probability distribution for all outcomes looks like this:



Probability Distributions

- That is, 7s are the most likely because there are more ways to get to 7 (3+4, 1+6, 2+5) than there are ways to arrive at any other sum between 2 and 12.

- Any formula attempting to predict the outcome of dice rolls needs to take seven's greater frequency into account.

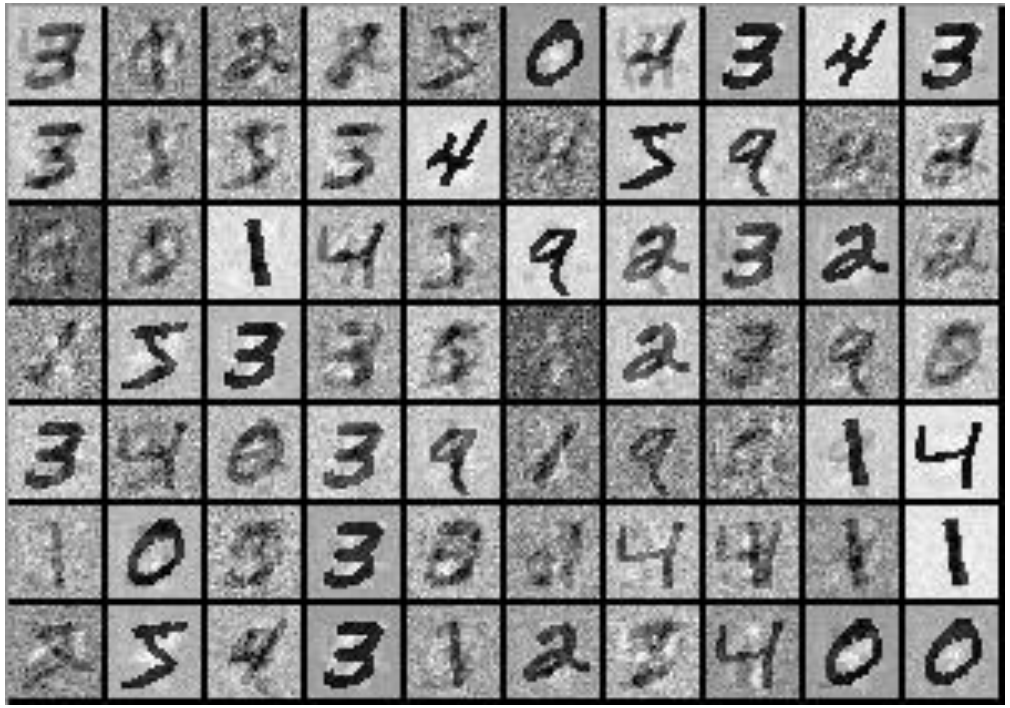
Probability Distributions

- Take another example: Languages are specific in the probability distribution of their letters, because each language uses certain letters more than others.
- In English, the letters e, t and a are the most common, while in Icelandic, the most common letters are a, r and n.
- Attempting to reconstruct Icelandic with a weight set based on English would lead to a large divergence.

Probability Distributions

- In the same way, image datasets have unique probability distributions for their pixel values, depending on the kind of images in the set.

- Pixels values are distributed differently depending on whether the dataset includes MNIST's handwritten numerals:



Probability Distributions

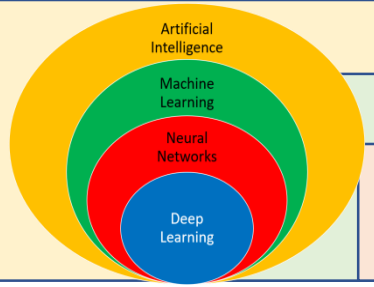
- Imagine for a second an RBM that was only fed images of elephants and dogs, and which had only two output nodes, one for each animal.
- The question the RBM is asking itself on the forward pass is: Given these pixels, should my weights send a stronger signal to the elephant node or the dog node?
- And the question the RBM asks on the backward pass is: Given an elephant, which distribution of pixels should I expect?

Probability Distributions

- That's joint probability: the simultaneous probability of x given a and of a given x, expressed as the shared weights between the two layers of the RBM.
- The process of learning reconstructions is, in a sense, learning which groups of pixels tend to co-occur for a given set of images.
- The activations produced by nodes of hidden layers deep in the network represent significant co-occurrences; e.g. "nonlinear gray tube + big, floppy ears + wrinkles" might be one.

Probability Distributions

- One last point: You'll notice that RBMs have two biases.
- This is one aspect that distinguishes them from other autoencoders.
- The hidden bias helps the RBM produce the activations on the forward pass (since biases impose a floor so that at least some nodes fire no matter how sparse the data), while the visible layer's biases help the RBM learn the reconstructions on the backward pass.



Artificial Intelligence (AI)

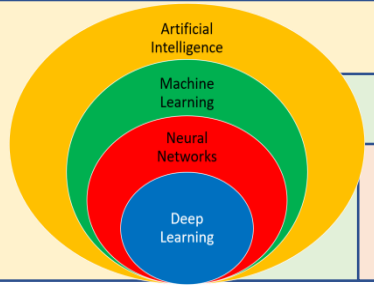
Machine Learning (ML)

Neural Networks (NNs)

Deep Learning (DL)

Multiple Layers

- Once this RBM learns the structure of the input data as it relates to the activations of the first hidden layer, then the data is passed one layer down the net.
- Your first hidden layer takes on the role of visible layer.
- The activations now effectively become your input, and they are multiplied by weights at the nodes of the second hidden layer, to produce another set of activations.



Artificial Intelligence (AI)

Machine Learning (ML)

Neural Networks (NNs)

Deep Learning (DL)

Multiple Layers

- This process of creating sequential sets of activations by grouping features and then grouping groups of features is the basis of a feature hierarchy, by which neural networks learn more complex and abstract representations of data.

- With each new hidden layer, the weights are adjusted until that layer is able to approximate the input from the previous layer.

- This is greedy, layerwise and unsupervised pre-training. It requires no labels to improve the weights of the network, which means you can train on unlabeled data, untouched by human hands, which is the vast majority of data in the world.

Multiple Layers

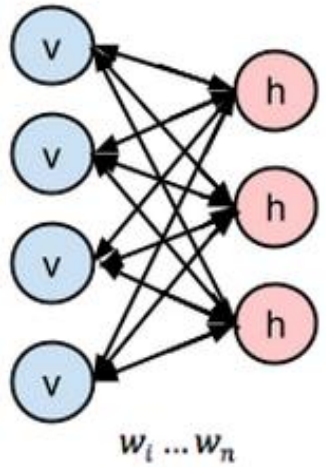
- As a rule, algorithms exposed to more data produce more accurate results, and this is one of the reasons why deep-learning algorithms are kicking butt.

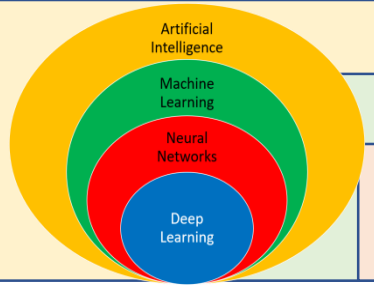
- Because those weights already approximate the features of the data, they are well positioned to learn better when, in a second step, you try to classify images with the deep-belief network in a subsequent supervised learning stage.

Multiple Layers

- While RBMs have many uses, proper initialization of weights to facilitate later learning and classification is one of their chief advantages.
- In a sense, they accomplish something similar to backpropagation: they push weights to model data well.
- You could say that pre-training and backprop are substitutable means to the same end.

A Symmetrical, Bipartite, Bidirectional Graph with Shared Weights





Artificial Intelligence (AI)

Machine Learning (ML)

Neural Networks (NNs)

Deep Learning (DL)