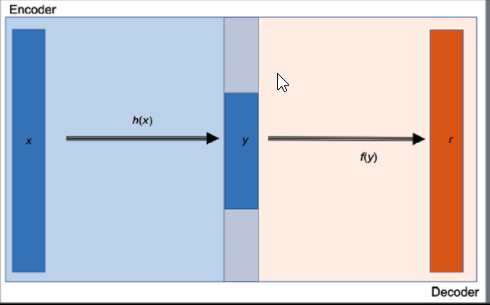
**1. What are Vanilla autoencoders ?**

**Ans:** The Vanilla autoencoder, as proposed by Hinton in his 2006 paper Reducing the Dimensionality of Data with Neural Networks, consists of one hidden layer only. The number of neurons in the hidden layer are less than the number of neurons in the input (or output) layer.

This results in producing a bottleneck effect in the flow of information in the network. The hidden layer in between is also called the "bottleneck layer." Learning in the autoencoder consists of developing a compact representation of the input signal at the hidden layer so that the output layer can faithfully reproduce the original input.

In the following diagram, you can see the architecture of Vanilla autoencoder:

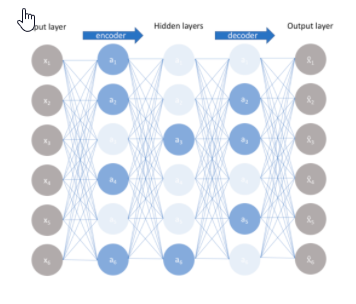


**2. What are Sparse autoencoders ?**

**Ans:** A Sparse Autoencoder is a type of autoencoder that employs sparsity to achieve an information bottleneck. Specifically the loss function is constructed so that activations are penalized within a layer. The sparsity constraint can be imposed with L1 regularization or a KL divergence between expected average neuron activation to an ideal distribution P.

Or A sparse autoencoder is one of a range of types of autoencoder artificial neural networks that work on the principle of unsupervised machine learning. Autoencoders are a type of deep network that can be used for dimensionality reduction – and to reconstruct a model through backpropagation.

Autoencoders seek to use items like feature selection and feature extraction to promote more efficient data coding. Autoencoders often use a technique called backpropagation to change weighted inputs, in order to achieve dimensionality reduction, which in a sense scales down the input for corresponding results. A sparse autoencoder is one that has small numbers of simultaneously active neural nodes.



**3. What are Denoising autoencoders ?**

**Ans:** A denoising autoencoder is a specific type of autoencoder, which is generally classed as a type of deep neural network. The denoising autoencoder gets trained to use a hidden layer to reconstruct a particular model based on its inputs.

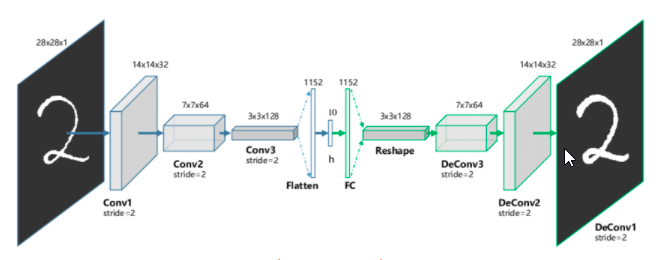
In general, autoencoders work on the premise of reconstructing their inputs. Autoencoders are generally unsupervised machine learning programs deriving results from unstructured data.

To achieve this equilibrium of matching target outputs to inputs, denoising autoencoders accomplish this goal in a specific way – the program takes in a corrupted version of some model, and tries to reconstruct a clean model through the use of denoising techniques. Engineers may apply noise in a particular amount as a percentage of the model and try to force the hidden layer to work from the corrupted version to produce a clean version. Denoising autoencoders can also be stacked on each other to provide iterative learning toward this key goal.

**4. What are Convolutional autoencoders ?**

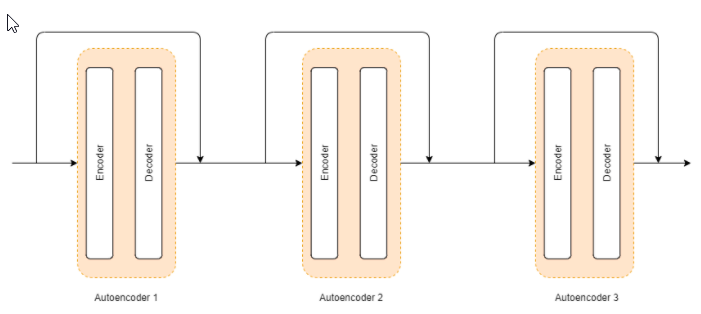
**Ans:** Convolutional Autoencoder is a variant of Convolutional Neural Networks that are used as the tools for unsupervised learning of convolution filters. They are generally applied in the task of image reconstruction to minimize reconstruction errors by learning the optimal filters.

The block diagram of a Convolutional Autoencoder is given in the below figure.



**5. What are Stacked autoencoders ?**

**Ans:** Some datasets have a complex relationship within the features. Thus, using only one Autoencoder is not sufficient. A single Autoencoder might be unable to reduce the dimensionality of the input features. Therefore for such use cases, we use stacked autoencoders. The stacked autoencoders are, as the name suggests, multiple encoders stacked on top of one another. A stacked autoencoder with three encoders stacked on top of each other is shown in the following figure.



According to the architecture shown in the figure above, the input data is first given to autoencoder

1. The output of the autoencoder 1 and the input of the autoencoder 1 is then given as an input to autoencoder
2. Similarly, the output of autoencoder 2 and the input of autoencoder 2 are given as input to autoencoder
3. Thus, the length of the input vector for autoencoder 3 is double than the input to the input of autoencoder 2.

This technique also helps to solve the problem of insufficient data to some extent.

**6. Explain Extractive summarization ?**

**Ans:** Extractive summarization aims at identifying the salient information that is then extracted and grouped together to form a concise summary. Abstractive summary generation rewrites the entire document by building internal semantic representation, and then a summary is created using natural language processing.

**7. Explain Abstractive summarization ?**

**Ans:** Abstractive Summarization is a task in Natural Language Processing (NLP) that aims to generate a concise summary of a source text. Unlike extractive summarization, abstractive summarization does not simply copy important phrases from the source text but also potentially come up with new phrases that are relevant, which can be seen as paraphrasing. Abstractive summarization yields a number of applications in different domains, from books and literature, to science and R&D, to financial research and legal documents analysis.

To date, the most recent and effective approach toward abstractive summarization is using transformer models fine-tuned specifically on a summarization dataset.

**8. Explain Beam search ?**

**Ans:** Beam search is a heuristic search algorithm that explores a graph by expanding the most optimistic node in a limited set. Beam search is an optimization of best-first search that reduces its memory requirements.

Best-first search is a graph search that orders all partial solutions according to some heuristic. But in beam search, only a predetermined number of best partial solutions are kept as candidates. Therefore, it is a greedy algorithm.

Beam search uses breadth-first search to build its search tree. At each level of the tree, it generates all successors of the states at the current level, sorting them in increasing order of heuristic cost. However, it only stores a predetermined number (β), of best states at each level called the beamwidth. Only those states are expanded next.

The greater the beam width, the fewer states are pruned. No states are pruned with infinite beam width, and beam search is identical to breadth-first search. The beamwidth bounds the memory required to perform the search. Since a goal state could potentially be pruned, beam search sacrifices completeness (the guarantee that an algorithm will terminate with a solution if one exists). Beam search is not optimal, which means there is no guarantee that it will find the best solution.

In general, beam search returns the first solution found. Once reaching the configured maximum search depth (i.e., translation length), the algorithm will evaluate the solutions found during a search at various depths and return the best one that has the highest probability.

The beam width can either be fixed or variable. One approach that uses a variable beam width starts with the width at a minimum. If no solution is found, the beam is widened, and the procedure is repeated.

**9. Explain Length normalization ?**

**Ans:** Document length normalization adjusts the term frequency or the relevance score in order to normalize the effect of document length on the document ranking.

One simple length normalization formula is to divide the number of occurrences by the length of the document.

**10. Explain ROUGE metric evaluation ?**

**Ans:** ROUGE, or Recall-Oriented Understudy for Gisting Evaluation, is a set of metrics and a software package used for evaluating automatic summarization and machine translation software in natural language processing