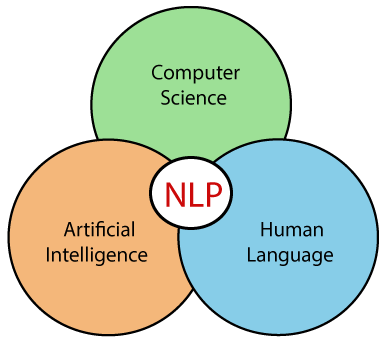
**UNIT-V**

**INTERACTIVE APPILICATIONS OF DEEP LEARNING**

**Natural Language Processing**

NLP stands for **Natural Language Processing**, which is a part of **Computer Science, Human language,** and **Artificial Intelligence**. It is the technology that is used by machines to understand, analyse, manipulate, and interpret human's languages. It helps developers to organize knowledge for performing tasks such as **translation, automatic summarization, Named Entity Recognition (NER), speech recognition, relationship extraction,** and **topic segmentation**.



## History of NLP

**(1940-1960) - Focused on Machine Translation (MT)**

The Natural Languages Processing started in the year 1940s.

**1948** - In the Year 1948, the first recognisable NLP application was introduced in Birkbeck College, London.

**1950s** - In the Year 1950s, there was a conflicting view between linguistics and computer science. Now, Chomsky developed his first book syntactic structures and claimed that language is generative in nature.

In 1957, Chomsky also introduced the idea of Generative Grammar, which is rule based descriptions of syntactic structures.

**(1960-1980) - Flavored with Artificial Intelligence (AI)**

In the year 1960 to 1980, the key developments were:

**Augmented Transition Networks (ATN)**

Augmented Transition Networks is a finite state machine that is capable of recognizing regular languages.

**Case Grammar**

Case Grammar was developed by **Linguist Charles J. Fillmore** in the year 1968. Case Grammar uses languages such as English to express the relationship between nouns and verbs by using the preposition.

In Case Grammar, case roles can be defined to link certain kinds of verbs and objects.

**For example:** "Neha broke the mirror with the hammer". In this example case grammar identify Neha as an agent, mirror as a theme, and hammer as an instrument.

In the year 1960 to 1980, key systems were:

**SHRDLU**

SHRDLU is a program written by **Terry Winograd** in 1968-70. It helps users to communicate with the computer and moving objects. It can handle instructions such as "pick up the green boll" and also answer the questions like "What is inside the black box." The main importance of SHRDLU is that it shows those syntax, semantics, and reasoning about the world that can be combined to produce a system that understands a natural language.

**LUNAR**

LUNAR is the classic example of a Natural Language database interface system that is used ATNs and Woods' Procedural Semantics. It was capable of translating elaborate natural language expressions into database queries and handle 78% of requests without errors.

**1980 - Current**

Till the year 1980, natural language processing systems were based on complex sets of hand-written rules. After 1980, NLP introduced machine learning algorithms for language processing.

In the beginning of the year 1990s, NLP started growing faster and achieved good process accuracy, especially in English Grammar. In 1990 also, an electronic text introduced, which provided a good resource for training and examining natural language programs. Other factors may include the availability of computers with fast CPUs and more memory. The major factor behind the advancement of natural language processing was the Internet.

Now, modern NLP consists of various applications, like **speech recognition, machine translation,** and **machine text reading**. When we combine all these applications then it allows the artificial intelligence to gain knowledge of the world. Let's consider the example of AMAZON ALEXA, using this robot you can ask the question to Alexa, and it will reply to you.

## Advantages of NLP

* NLP helps users to ask questions about any subject and get a direct response within seconds.
* NLP offers exact answers to the question means it does not offer unnecessary and unwanted information.
* NLP helps computers to communicate with humans in their languages.
* It is very time efficient.
* Most of the companies use NLP to improve the efficiency of documentation processes, accuracy of documentation, and identify the information from large databases.

## Disadvantages of NLP

A list of disadvantages of NLP is given below:

* NLP may not show context.
* NLP is unpredictable
* NLP may require more keystrokes.
* NLP is unable to adapt to the new domain, and it has a limited function that's why NLP is built for a single and specific task only.

## Components of NLP

There are the following two components of NLP -

**1. Natural Language Understanding (NLU)**

Natural Language Understanding (NLU) helps the machine to understand and analyse human language by extracting the metadata from content such as concepts, entities, keywords, emotion, relations, and semantic roles.

NLU mainly used in Business applications to understand the customer's problem in both spoken and written language.

NLU involves the following tasks -

* It is used to map the given input into useful representation.
* It is used to analyze different aspects of the language.

**2. Natural Language Generation (NLG)**

Natural Language Generation (NLG) acts as a translator that converts the computerized data into natural language representation. It mainly involves Text planning, Sentence planning, and Text Realization.

#### Note: The NLU is difficult than NLG.

|  |  |
| --- | --- |
| **NLU** | **NLG** |
| NLU is the process of reading and interpreting language. | NLG is the process of writing or generating language. |
| It produces non-linguistic outputs from natural language inputs. | It produces constructing natural language outputs from non-linguistic inputs. |

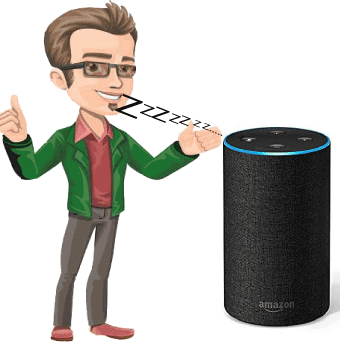
**Difference between NLU and NLG**

## Applications of NLP

There are the following applications of NLP -

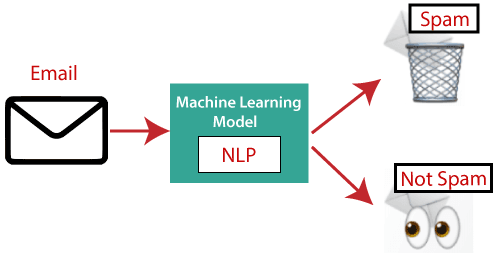
**1. Question Answering**

Question Answering focuses on building systems that automatically answer the questions asked by humans in a natural language.



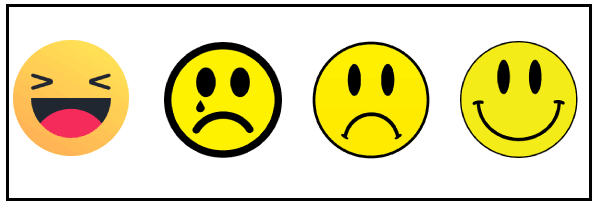
**2. Spam Detection**

Spam detection is used to detect unwanted e-mails getting to a user's inbox.



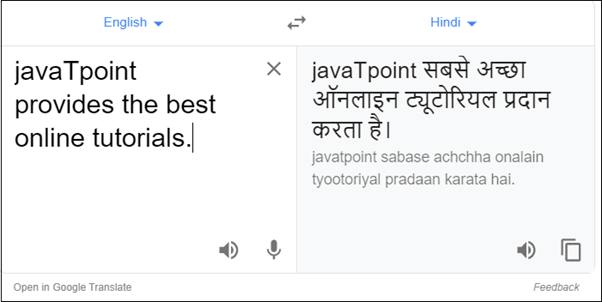
**3. Sentiment Analysis**

Sentiment Analysis is also known as **opinion mining**. It is used on the web to analyse the attitude, behaviour, and emotional state of the sender. This application is implemented through a combination of NLP (Natural Language Processing) and statistics by assigning the values to the text (positive, negative, or natural), identify the mood of the context (happy, sad, angry, etc.)



**4. Machine Translation**

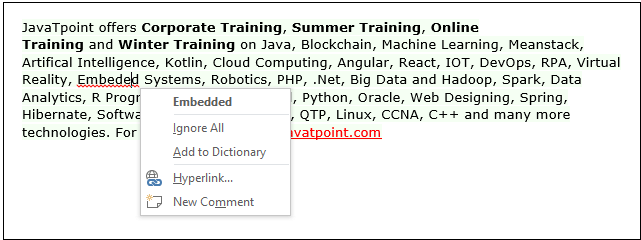
Machine translation is used to translate text or speech from one natural language to another natural language.



**Example:** Google Translator

**5. Spelling correction**

Microsoft Corporation provides word processor software like MS-word, PowerPoint for the spelling correction.



**6. Speech Recognition**

Speech recognition is used for converting spoken words into text. It is used in applications, such as mobile, home automation, video recovery, dictating to Microsoft Word, voice biometrics, voice user interface, and so on.

**7. Chatbot**

Implementing the Chatbot is one of the important applications of NLP. It is used by many companies to provide the customer's chat services.

**8. Information extraction**

Information extraction is one of the most important applications of NLP. It is used for extracting structured information from unstructured or semi-structured machine-readable documents.

**9. Natural Language Understanding (NLU)**

It converts a large set of text into more formal representations such as first-order logic structures that are easier for the computer programs to manipulate notations of the natural language processing.

**How to build an NLP pipeline**

There are the following steps to build an NLP pipeline -

**Step1: Sentence Segmentation**

Sentence Segment is the first step for building the NLP pipeline. It breaks the paragraph into separate sentences.

**Example:** Consider the following paragraph -

**Independence Day is one of the important festivals for every Indian citizen. It is celebrated on the 15th of August each year ever since India got independence from the British rule. The day celebrates independence in the true sense.**

**Sentence Segment produces the following result:**

1. "Independence Day is one of the important festivals for every Indian citizen."
2. "It is celebrated on the 15th of August each year ever since India got independence from the British rule."
3. "This day celebrates independence in the true sense."

**Step2: Word Tokenization**

Word Tokenizer is used to break the sentence into separate words or tokens.

**Example:**

JavaTpoint offers Corporate Training, Summer Training, Online Training, and Winter Training.

Word Tokenizer generates the following result:

"JavaTpoint", "offers", "Corporate", "Training", "Summer", "Training", "Online", "Training", "and", "Winter", "Training", "."

**Step3: Stemming**

Stemming is used to normalize words into its base form or root form. For example, celebrates, celebrated and celebrating, all these words are originated with a single root word "celebrate." The big problem with stemming is that sometimes it produces the root word which may not have any meaning.

**For Example,** intelligence, intelligent, and intelligently, all these words are originated with a single root word "intelligen." In English, the word "intelligen" do not have any meaning.

**Step 4: Lemmatization**

Lemmatization is quite similar to the Stamming. It is used to group different inflected forms of the word, called Lemma. The main difference between Stemming and lemmatization is that it produces the root word, which has a meaning.

**For example:** In lemmatization, the words intelligence, intelligent, and intelligently has a root word intelligent, which has a meaning.

**Step 5: Identifying Stop Words**

In English, there are a lot of words that appear very frequently like "is", "and", "the", and "a". NLP pipelines will flag these words as stop words. **Stop words** might be filtered out before doing any statistical analysis.

**Example:** He **is a** good boy.

#### Note: When you are building a rock band search engine, then you do not ignore the word "The."

**Step 6: Dependency Parsing**

Dependency Parsing is used to find that how all the words in the sentence are related to each other.

**Step 7: POS tags**

POS stands for parts of speech, which includes Noun, verb, adverb, and Adjective. It indicates that how a word functions with its meaning as well as grammatically within the sentences. A word has one or more parts of speech based on the context in which it is used.

**Example: "Google"** something on the Internet.

In the above example, Google is used as a verb, although it is a proper noun.

**Step 8: Named Entity Recognition (NER)**

Named Entity Recognition (NER) is the process of detecting the named entity such as person name, movie name, organization name, or location.

**Example: Steve Jobs** introduced iPhone at the Macworld Conference in San Francisco, California.

**Step 9: Chunking**

Chunking is used to collect the individual piece of information and grouping them into bigger pieces of sentences.

# Generative Adversarial Network (GAN)

Generative Adversarial Networks (GANs) can be described as extremely powerful kinds of neural networks that are employed to aid in Unsupervised Learning. They were created and first introduced in 2014 by Ian J. Goodfellow 2014. GANs are comprised of two neural networks that are in competition with one another and can analyse the changes within a set of data.

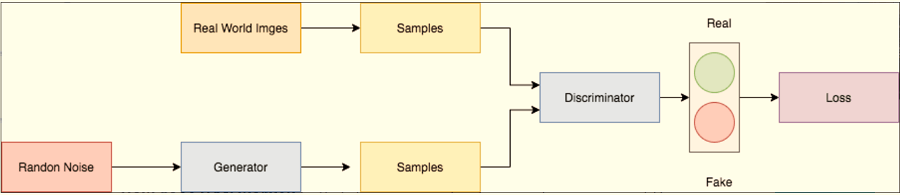
GANs are a method for generative modelling that uses deep learning methods like CNN (Convolutional Neural Network). Generative modelling is an unsupervised learning method that automatically discovers and learns patterns in input data so that the model can be used for new examples from the original dataset.

GANs are a method of training generative modelling by framing the problem as a supervised learning problem and using two sub-models. GANs have two components:

Generator This is a program that generates new data from real-world images.

Discriminator This compares the images with real-world examples to classify fake and real images.

Example The Generator generates random images (e.g., The Generator generates some random images (e.g., tables), and then the Discriminator compares these images with real-world table images. Finally, the Generator sends the feedback directly to Generator. See the GAN structure in the following figure.



## What is the Reason GANs were Invented in the First Place?

It is well-known that most of the neural nets used in mainstream research are easily misled into misclassifying items by introducing a tiny amount of noise to the data. Surprisingly, the model modified after adding noise has a higher probability of making a mistake than when it has made a good prediction. The reason for this is that machines learn from the smallest amount of data. This is a main drawback as it can be overfitted. The relationship between input and output is almost linear. While the lines of separation between different classes could be linear, they are comprised of linearities, and even minor changes in one point of the feature space could cause data to be classified incorrectly.

## How does GAN works?

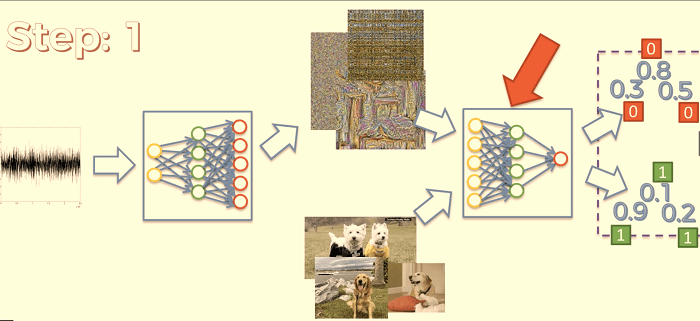
Generative Adversarial Networks (GANs) can be broken down into three parts:

* **Generative:** To learn more about a dynamic model that explains how data are generated using a probabilistic model.
* **Adversarial:** The process of training models, is conducted in an adversarial environment.
* **Networks:** Make use of deep neural networks to create Artificial Intelligence (AI) algorithms to train for purposes.

Let's look at an example: generating images of Dogs.

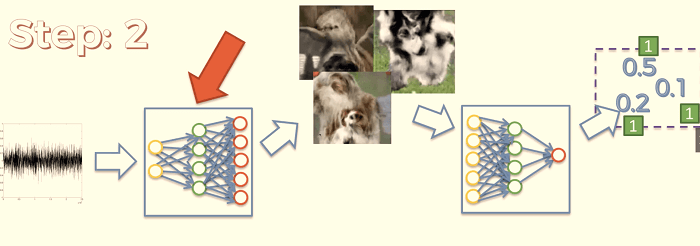
### Step 1: Training of Discriminator

1. First, a random noise signal is sent through a generator. This generates useless images that contain noise.
2. (See fig. 2)
3. Discriminator has two inputs. The first is the Generator sample output images, and the second is real-world dog image samples.
4. After comparing the images in fig., The Discriminator then populates certain values (probability). 2. It calculates 0.8 to 0.3 and 0.0.5 for generator output images and 0.1, 9.9, and 0.2 for real-world images.
5. An error can be calculated by comparing the probabilities of generated images with 0 (Zero) and the probabilities of real-word images with 1. (Ex. (Ex.
6. After it has calculated individual errors, it will calculate cumulative loss(loss), which is backpropagated. The weights of the Discriminator can then be adjusted. This is how a Discriminator gets trained.



### Step 2 - Training the Generator

1. The loss is propagated back to the Discriminator in step 1. This will allow it to adjust its weights. We must also backpropagate an error, so it can adjust its weights and train itself to produce better images.
2. The Generator generates images that are used to input the Generator.
3. The Discriminator will now use the newly generated images as input. It calculates probabilities such as 0.5, 0.01, and 0.2. (See fig. 2)
4. An error can be calculated by comparing the probabilities of images generated with 1 (One).
5. After it has calculated individual errors, it will calculate cumulative loss(loss), which is backpropagated. The weights of the Generator then are adjusted. This is how Generator gets trained.



After a few more iterations, the Generator will start generating images similar to real-world images.

## Applications to GAN:

* Generating images
* Super Resolution
* Image Modification
* Photos of real people
* Face Ageing

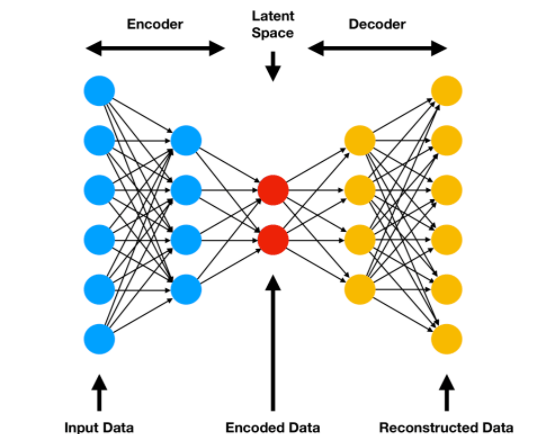
**DEEP LEARNING RSEARCH**

**AUTOENCODERS:**

### Autoencoders

Autoencoders are a special type of neural network where inputs are outputs are found usually identical. It was designed to primarily solve the problems related to unsupervised learning. Autoencoders are highly trained neural networks that **replicate** the data. It is the reason why the input and output are generally the same. They are used to achieve tasks like **pharma discovery, image processing,** and **population prediction.**

Autoencoders constitute three components namely the **encoder**, the **code**, and the **decoder.** Autoencoders are built in such a structure that they can receive inputs and transform them into various representations. The attempts to copy the original input by reconstructing them is more accurate. They do this by encoding the image or input, reduce the size. If the image is not visible properly they are passed to the neural network for clarification. Then, the clarified image is termed a reconstructed image and this resembles as accurate as of the previous image. To understand this complex process, see the below-provided image.



# DEEP GENERATIVE MODELS

# Boltzmann Machines

Boltzmann machine refers to an association of uniformly associated neuron-like structure that make hypothetical decisions about whether to be **on** or **off**. Boltzmann Machine was invented by renowned scientist **Geoffrey Hinton** and **Terry Sejnowski** in 1985. Boltzmann Machines have a fundamental learning algorithm that permits them to find exciting features that represent complex regularities in the training data. The learning algorithm is usually slow in networks with various layers of feature detectors, but it is quick in "**Restricted Boltzmann Machines**" that has a single layer of feature detectors. Many hidden layers can be adapted efficiently by comprising Boltzmann Machines, utilizing the feature activations of one as the training data for the next.

Boltzmann Machines are utilized to resolve two different computational issues. First, **for a search problem**, the weight on the associations are fixed and are used to represent a cost function. The stochastic dynamics of a Boltzmann Machine permit it to sample binary state vectors that have minimum values of the cost function. Second, for a learning issue, the Boltzmann Machine has indicated a set of binary data vectors, and this must figure out how to generate these vectors with high probability. To solve this, it must discover weights on the associations so that relative to other possible binary vectors, the data vectors have minimum values of the cost function. For solving a learning issue, Boltzmann machines make numerous small updates to their weights, and each update expects them to tackle a wide range of search issues.

## The Stochastic Dynamics of a Boltzmann Machine:

When unit **i** is given a chance to update its binary state, it initially computes its absolute input, **pi**, which is the sum of its own bias, **qi** , and the weights on associations coming from other active units:

**Pi = qi + ?jmj wij**

Where,

**wij** = It is the weight on the association between **i**and **j**, and **mj** is **1** when unit **j** is on. Unit **i** turns on with a probability given by the logistic function:

Boltzmann Machines

If the units are updated successively in any order that does not rely on their total inputs, the network will eventually reach a Boltzmann distribution (also known as equilibrium or stationary distribution) in which the probability of the given state vector **k** is determined exclusively by the "energy" of that state vector compared to the energies of all possible binary state vectors:

Boltzmann Machines

As in Hopfield networks, the energy of state vector **k** is defined as

Boltzmann Machines

Where, sik is the binary state appointed to unit **i**by state vector **k**. If the weights on the associations are chosen so that the energies of the state vectors represent the cost of those state vector, the stochastic dynamics of a Boltzmann machine can be seen as a method for getting away from poor local optima while looking for low-cost solutions. The total input of unit **i** , **pi** , represents the difference in energy relying upon whether the units are off or on, and the way that unit **i** sometimes turns on even if **pi** is negative implies that the energy can occasionally increase during the search, therefore permitting the search to jump over energy barriers. The search can be upgraded by using simulated annealing. It scales down all of the weights and energies by a factor **T**, which is equivalent to the temperature of a physical network. By minimizing **T** from a considerable initial value to small final value, it is possible to benefit from the fast equilibrium at high temperature and still have a final equilibrium distribution that makes minimal solutions considerably more probable than high-cost ones. At a zero temperature, the update rule becomes deterministic, and a Boltzmann Machines transforms into a Hopefield network.

**Different types of Boltzmann Machine**

The learning rule can hold more complex energy functions. For example, the quadratic energy function can be replaced by an energy function that has a common term **si sj     sk wijk**. The total input **i** is utilized to update rule must be replaced by

Boltzmann Machines

The significant change in the learning rule is that si sj is replaced by si sj sk. Boltzmann machines model the dispersion of the data vectors. However, there is a basic extension, the "conditional Boltzmann machine" for modeling conditional distributions. The significant difference between the visible and the hidden units is that when sampling (si   sj) data, the visible units are clamped, and the hidden units are possibly not. If a subset of the visible units is clamped when sampling ) (si   sj) model, this subset acts as "input" units, and the remaining visible units serve as "output" units.

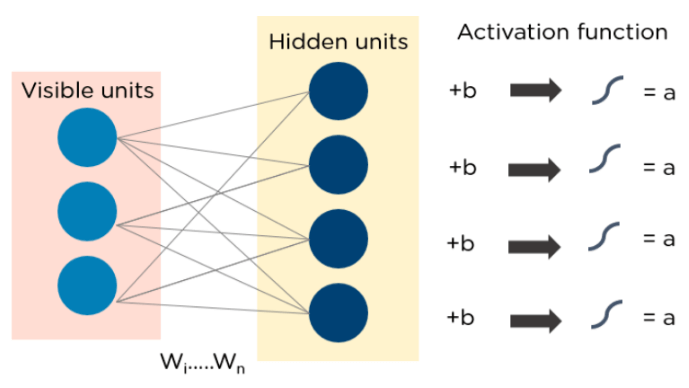
### The Speed of Learning

Learning is commonly very slow in Boltzmann machines with various hidden layers because the enormous networks can take quite a long time to approach their equilibrium distribution, particularly when the weights are huge and the equilibrium distribution is highly multimodal. When samples from the equilibrium distribution can be acquired, the learning signal is very noisy because it is the difference between the two sampled expectations. These issues can be overcome by confining the network, simplifying the learning algorithm, and learning one hidden layer at a time.

### Restricted Boltzmann Machines (RBMs)

RBMs were developed by **Geoffrey Hinton** and resemble stochastic neural networks that learn from the probability distribution in the given input set. This algorithm is mainly used in the field of dimension **reduction, regression** and **classification, topic modeling** and are considered the building blocks of DBNs. RBIs consist of two layers namely the **visible layer** and the **hidden layer**. Both of these layers are connected through hidden units and have bias units connected to nodes that generate the output. Usually, RBMs have two phases namely **forward pass** and **backward pass**.

The functioning of RBMs is carried out by accepting inputs and translating them to numbers so that inputs are encoded in the forward pass. RBMs take into account the weight of every input, and the backward pass takes these input weights and translates them further into reconstructed inputs. Later, both of these translated inputs, along with individual weights, are combined. These inputs are then pushed to the visible layer where the activation is carried out, and output is generated that can be easily reconstructed. To understand this process, consider the below image.



### Deep Belief Networks (DBNs)

DBNs are called generative models because they have various layers of latent as well as stochastic variables. The latent variable is called a **hidden unit** because they have binary values. DBNs are also called **Boltzmann Machines** because the **RGM** layers are stacked over each other to establish communication with previous and consecutive layers. DBNs are used in applications like video and image recognition as well as capturing motional objects.

DBNs are powered by **Greedy algorithms.** The layer to layer approach by leaning through a **top-down** approach to generate weights is the most common way DBNs function. DBNs use step by step approach of **Gibbs** sampling on the hidden **two-layer** at the top. Then, these stages draw a sample from the visible units using a model that follows the ancestral sampling method. DBNs learn from the values present in the latent value from every layer following the **bottom-up** pass approach.

