PYTHON-BASED DEEP LEARNING FOR IMAGE CLASSIFICATION

Contents:

1. Introduction
2. Why deep learning for images and later python
3. Various algorithms available and why I choose the one
4. Dataset and the code discussion
5. Results and conclusion
6. References

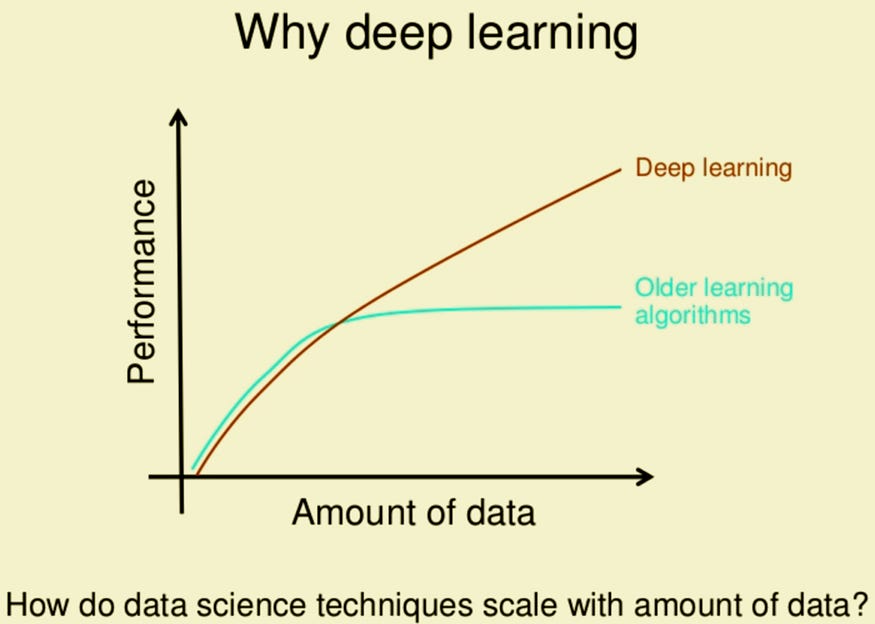
Chapter 1 : Introduction

* 1. OverView

Classification is a systematic arrangement of groups and categories based on their features. Image classification began to decrease the gap between computer vision and human vision by training the computer with the data. The image classification is achieved by differentiating the image into the prescribed category based on the content of the vision.

Machine Learning is the science of getting computers to learn without being explicitly programmed. It is closely related to computational statistics, which focuses on making prediction using computer. In its application across business problems, machine learning is also referred as predictive analysis. Machine Learning is closely related to computational statistics. Machine Learning focuses on the development of computer programs that can access data and use it to learn themselves. The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly

Deep learning is a method in artificial intelligence (AI) that teaches computers to process data in a way that is inspired by the human brain. Deep learning models can recognize complex patterns in pictures, text, sounds, and other data to produce accurate insights and predictions. You can use deep learning methods to automate tasks that typically require human intelligence, such as describing images or transcribing a sound file into text.



* 1. Why Deep Learning

Artificial intelligence (AI) attempts to train computers to think and learn as humans do. Deep learning technology drives many AI applications used in everyday products, such as the following:

* Digital assistants
* Voice-activated television remotes
* Fraud detection
* Automatic facial recognition

It is also a critical component of emerging technologies such as self-driving cars, virtual reality, and more.

Deep learning models are computer files that data scientists have trained to perform tasks using an algorithm or a predefined set of steps. Businesses use deep learning models to analyze data and make predictions in various applications.

Deep learning has several use cases in automotive, aerospace, manufacturing, electronics, medical research, and other fields. These are some examples of deep learning:

* Self-driving cars use deep learning models to automatically detect road signs and pedestrians.
* Defense systems use deep learning to automatically flag areas of interest in satellite images.
* Medical image analysis uses deep learning to automatically detect cancer cells for medical diagnosis.
* Factories use deep learning applications to automatically detect when people or objects are within an unsafe distance of machines.

You can group these various use cases of deep learning into four broad categories—computer vision, speech recognition, natural language processing (NLP), and recommendation engines.

**Computer vision**

[Computer vision](https://aws.amazon.com/computer-vision/) is the computer's ability to extract information and insights from images and videos. Computers can use deep learning techniques to comprehend images in the same way that humans do. Computer vision has several applications, such as the following:

* Content moderation to automatically remove unsafe or inappropriate content from image and video archives
* Facial recognition to identify faces and recognize attributes like open eyes, glasses, and facial hair
* Image classification to identify brand logos, clothing, safety gear, and other image details

**Speech recognition**

Deep learning models can analyze human speech despite varying speech patterns, pitch, tone, language, and accent. Virtual assistants such as Amazon Alexa and [automatic transcription software](https://aws.amazon.com/what-is/speech-to-text/) use speech recognition to do the following tasks:

* Assist call center agents and automatically classify calls.
* Convert clinical conversations into documentation in real time.
* Accurately subtitle videos and meeting recordings for a wider content reach.

**Natural language processing**

Computers use deep learning algorithms to gather insights and [meaning from text data](https://aws.amazon.com/comprehend/) and documents. This ability to process natural, human-created text has several use cases, including in these functions:

* Automated virtual agents and chatbots
* Automatic summarization of documents or news articles
* Business intelligence analysis of long-form documents, such as emails and forms
* Indexing of key phrases that indicate sentiment, such as positive and negative comments on social media

**Recommendation engines**

Applications can use deep learning methods to track user activity and develop [personalized recommendations](https://aws.amazon.com/mxnet/). They can analyze the behavior of various users and help them discover new products or services. For example, many [media and entertainment](https://aws.amazon.com/media/) companies, such as Netflix, Fox, and Peacock, use deep learning to give personalized video recommendations.

## **How does deep learning work?**

Deep learning algorithms are neural networks that are modeled after the human brain. For example, a human brain contains millions of interconnected neurons that work together to learn and process information. Similarly, deep learning neural networks, or artificial neural networks, are made of many layers of artificial neurons that work together inside the computer.

Artificial neurons are software modules called nodes, which use mathematical calculations to process data. Artificial neural networks are deep learning algorithms that use these nodes to solve complex problems.

## **What are the components of a deep learning network?**

The components of a deep neural network are the following.

### Input layer

An artificial neural network has several nodes that input data into it. These nodes make up the input layer of the system.

### Hidden layer

The input layer processes and passes the data to layers further in the neural network. These hidden layers process information at different levels, adapting their behavior as they receive new information. Deep learning networks have hundreds of hidden layers that they can use to analyze a problem from several different angles.

For example, if you were given an image of an unknown animal that you had to classify, you would compare it with animals you already know. For example, you would look at the shape of its eyes and ears, its size, the number of legs, and its fur pattern. You would try to identify patterns, such as the following:

* The animal has hooves, so it could be a cow or deer.
* The animal has cat eyes, so it could be some type of wild cat.

The hidden layers in deep neural networks work in the same way. If a deep learning algorithm is trying to classify an animal image, each of its hidden layers processes a different feature of the animal and tries to accurately categorize it.

### Output layer

The output layer consists of the nodes that output the data. Deep learning models that output "yes" or "no" answers have only two nodes in the output layer. On the other hand, those that output a wider range of answers have more nodes.

## https://miro.medium.com/v2/resize:fit:840/1*KYUUg9JC6InYe-VNPMDzAA.png

## **What is deep learning in the context of machine learning?**

Deep learning is a subset of machine learning. Deep learning algorithms emerged in an attempt to make traditional machine learning techniques more efficient. Traditional machine learning methods require significant human effort to train the software. For example, in animal image recognition, you need to do the following:

* Manually label hundreds of thousands of animal images.
* Make the machine learning algorithms process those images.
* Test those algorithms on a set of unknown images.
* Identify why some results are inaccurate.
* Improve the dataset by labeling new images to improve result accuracy.

This process is called supervised learning. In supervised learning, result accuracy improves only when you have a broad and sufficiently varied dataset. For instance, the algorithm might accurately identify black cats but not white cats because the training dataset had more images of black cats. In that case, you would need to label more white cat images and train the machine learning models once again.

## **What are the benefits of deep learning over machine learning?**

A deep learning network has the following benefits over traditional machine learning.

### Efficient processing of unstructured data

Machine learning methods find unstructured data, such as text documents, challenging to process because the training dataset can have infinite variations. On the other hand, deep learning models can comprehend unstructured data and make general observations without manual feature extraction. For instance, a neural network can recognize that these two different input sentences have the same meaning:

* Can you tell me how to make the payment?
* How do I transfer money?

### Hidden relationships and pattern discovery

A deep learning application can analyze large amounts of data more deeply and reveal new insights for which it might not have been trained. For example, consider a deep learning model that is trained to analyze consumer purchases. The model has data only for the items you have already purchased. However, the artificial neural network can suggest new items that you haven't bought by comparing your buying patterns to those of other similar customers.

### Unsupervised learning

Deep learning models can learn and improve over time based on user behavior. They do not require large variations of labeled datasets. For example, consider a neural network that automatically corrects or suggests words by analyzing your typing behavior. Let's assume it was trained in the English language and can spell-check English words. However, if you frequently type non-English words, such as danke, the neural network automatically learns and autocorrects these words too.

### Volatile data processing

Volatile datasets have large variations. One example is loan repayment amounts in a bank. A deep learning neural network can categorize and sort that data as well, such as by analyzing financial transactions and flagging some of them for fraud detection.

## **What are the challenges of deep learning?**

As deep learning is a relatively new technology, certain challenges come with its practical implementation.

### Large quantities of high-quality data

Deep learning algorithms give better results when you train them on large amounts of high-quality data. Outliers or mistakes in your input dataset can significantly affect the deep learning process. For instance, in our animal image example, the deep learning model might classify an airplane as a turtle if non-animal images were accidentally introduced in the dataset.

To avoid such inaccuracies, you must clean and process large amounts of data before you can train deep learning models. The input data preprocessing requires large amounts of data storage capacity.

### Large processing power

Deep learning algorithms are compute-intensive and require infrastructure with sufficient compute capacity to properly function. Otherwise, they take a long time to process results.

The conventional methods used for image classifying are part and piece of the field of artificial intelligence (AI) formally called machine learning. Machine learning consists of a feature extraction module that extracts important features such as edges, textures, etc., and a classification module that classifies based on the features extracted. The main limitation of machine learning is, that while separating, it can only extract a certain set of features on images and is unable to extract differentiating features from the training set of data. This disadvantage is rectified by using deep learning [2]. Deep learning (DL) is a subfield of machine learning, capable of learning through its method of computing. A deep learning model is introduced to persistently break down information with a homogeneous structure like how a human would make determinations. To accomplish this, deep learning utilizes a layered structure of several algorithms expressed as an artificial neural system (ANN). The architecture of an ANN is simulated with the help of the biological neural network of the human brain. This makes deep learning more capable than the standard machine learning models [3, 4].In deep learning, we consider the neural networks that identify the image based on its features. This is accomplished for the building of a complete feature extraction model that is capable of solving the difficulties faced due to conventional methods.

The extractor of the integrated model should be able to extract the distinguishing features from the training set of images accurately.

* 1. Deep Learning vs ML

1. Deep learning vs. machine learning
2. If deep learning is a subset of machine learning, how do they differ? Deep learning distinguishes itself from classical machine learning by the type of data that it works with and the methods in which it learns.
3. Machine learning algorithms leverage structured, labeled data to make predictions—meaning that specific features are defined from the input data for the model and organized into tables. This doesn’t necessarily mean that it doesn’t use unstructured data; it just means that if it does, it generally goes through some pre-processing to organize it into a structured format.
4. Deep learning eliminates some of data pre-processing that is typically involved with machine learning. These algorithms can ingest and process unstructured data, like text and images, and it automates feature extraction, removing some of the dependency on human experts. For example, let’s say that we had a set of photos of different pets, and we wanted to categorize by “cat”, “dog”, “hamster”, et cetera. Deep learning algorithms can determine which features (e.g. ears) are most important to distinguish each animal from another. In machine learning, this hierarchy of features is established manually by a human expert.
5. Then, through the processes of gradient descent and backpropagation, the deep learning algorithm adjusts and fits itself for accuracy, allowing it to make predictions about a new photo of an animal with increased precision.
6. Machine learning and deep learning models are capable of different types of learning as well, which are usually categorized as supervised learning, unsupervised learning, and reinforcement learning. Supervised learning utilizes labeled datasets to categorize or make predictions; this requires some kind of human intervention to label input data correctly. In contrast, unsupervised learning doesn’t require labeled datasets, and instead, it detects patterns in the data, clustering them by any distinguishing characteristics. Reinforcement learning is a process in which a model learns to become more accurate for performing an action in an environment based on feedback in order to maximize the reward.

1.4 Hardware Requirements

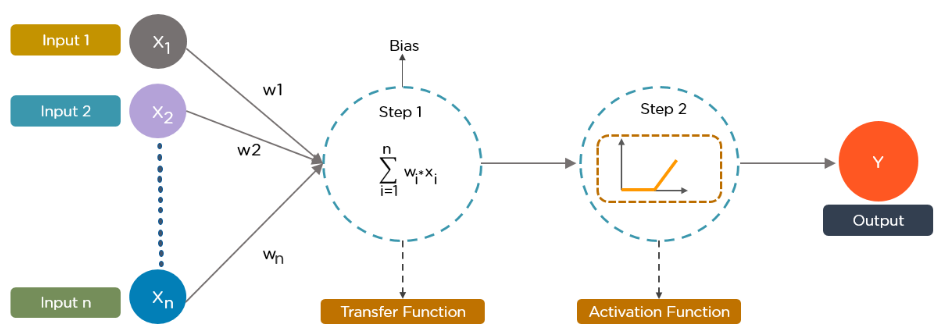
Deep learning hardware requirements

Deep learning requires a tremendous amount of computing power. High performance [*graphical processing units (GPUs)*](https://www.ibm.com/blog/video-what-is-a-gpu/) are ideal because they can handle a large volume of calculations in multiple cores with copious memory available. However, managing multiple GPUs on-premises can create a large demand on internal resources and be incredibly costly to scale.

1.5 History

The first step toward neural networks was taken in 1943, when Warren McCulloch, a neurophysiologist, and Walter Pitts, a young mathematician, published a paper on how neurons may work. They proposed an electrical circuit-based neural network. Donald Hebb proposed in1949 that brain connections became stronger with each usage [12]. In the 1950s, IBM researcher Nathanial Rochester used IBM 704 computers to mimic abstract neural networks [13]. In 1956, four scientists collaborated on the Dartmouth Summer Research Project on Artificial Intelligence, which took place during the summer. John McCarthy, Marvin L. Minsky, Nathaniel Rochester, and Claude E. Shannon were the four scientists. They made a significant contribution to AI research [14]. Following the Dartmouth study in 1957, John Von Neumann claimed that telegraph relays or vacuum tubes may be used to mimic the function of a single neuron. Frank Rosenblatt, a Cornell neurobiologist, began working on the Perceptron in 1958. He was enthralled by the activity of a fly's eye. In a fly's eye, a large part of the preparation that instructs it to flee is done. The Perceptron, which was developed as a result of this research, is the most well-known and widely used neural network today. A single layer perceptron was shown to be useful for classifying a single-valued collection of inputs into one of two categories. The perceptron calculates a weighted sum of the data sources, subtracts a limit, and outputs one of two possible qualities. Bernard Widrow and Marcian Hoff of Stanford developed the ADALINE and MADALINE 1 models in 1959. Multiple ADAptive LINear Elements were used in these models, which gave them their moniker. MADALINE was the first neural network to be used to solve a problem in the real world. It's an adaptive channel for removing echoes from telephone lines. This neuronal structure is still used in the workplace. Surprisingly, these previous victories led people to exaggerate the capabilities of neural networks, especially given the hardware limitations at the time. The excessive excitement that emanated from the academic and technical disciplines poisoned the writing of the day. As promises were unfulfilled, disillusionment crept in. Similarly, as essayists considered the impact of "figuring machines" on a man, a sense of dread developed. Asimov's arrangement on robots revealed the implications for man's ethics and attributes when machines were capable of performing all of humanity's tasks. Interest in the field was reignited in 1982. Caltech's John Hopfield presented a paper to the National Academy of Sciences 2. His strategy was to use bidirectional wires to create more valuable devices. Previously, there was just one route for neurons to connect. A combined US-Japan Conference on Cooperative/Competitive Neural Networks was also held in 1982. Japan announced a new Fifth-Generation effort on neural networks, while US journals raised concerns that the US would be left behind in the sector (Fifth-Generation processing incorporates computerized reasoning). The first era used switches and wires, the second era used transistors, the third era used strong state technology such as integrated circuits and higher-level programming dialects, and the fourth era used code generators.) As a result, there was increased subsidizing and, as a result, more field exploration. The American Institute of Physics began a yearly conference called Neural Networks for Computing in 1985. The first International Conference on Neural Networks, held by the Institute of Electrical and Electronics Engineers (IEEE) in 1987, gathered over 1,800 people. Schmidhuber and Hochreiter proposed the Long Short-Term Memory (LSTM) recurrent neural network structure in 1997. In the realm of deep learning, long momentary memory (LSTM) is an artificial recurrent neural network (RNN) architecture [1]. LSTM has feedback connections, unlike normal feedforward neural networks. It not only cycles single information items (such as pictures), but also the entire stream of data (for example, speech or video). Yann LeCun released Gradient-Based Learning Applied to Document Recognition in 1998, which was a significant step forward in data learning [15].

DEEP LEARNING ALGORITHMS



Deep learning has exploded in prominence in scientific computing, with its techniques being utilized by a wide range of sectors to solve complicated issues. To perform certain tasks, all deep learning algorithms employ various forms of neural networks

While deep learning algorithms use self-learning representations, they rely on artificial neural networks (ANNs) that mimic how the brain processes information. Algorithms leverage unknown elements in the input distribution to extract features, organize objects, and uncover important data patterns throughout the training phase. This happens at various levels, employing the algorithms to develop the models, much like training machines for self-learning. Several algorithms are used in deep learning models. While no network is flawless, certain algorithms are better suited to specific jobs than others. To select the best, it's necessary to have a thorough understanding of all primary algorithms.

Deep learning algorithms can handle practically any type of data and require a lot of processing power and data to solve complex problems. Let's take a look at the top ten deep learning algorithms. The following is a list of the top ten most widely used deep learning algorithms:

1 Convolutional Neural Networks (CNNs)

2 Long Short-Term Memory Networks (LSTMs)

3 Recurrent Neural Networks (RNNs)

4 Generative Adversarial Networks (GANs)

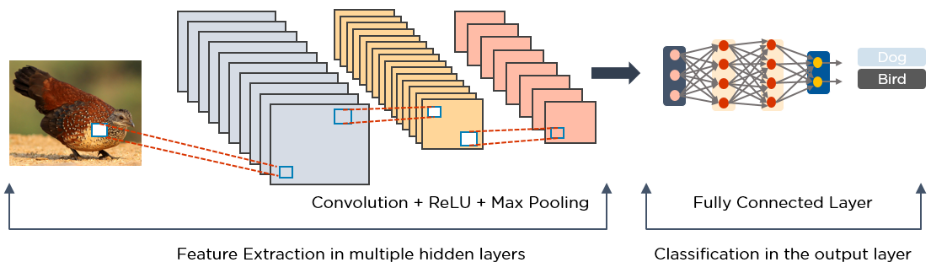
5 Radial Basis Function Networks (RBFNs)

6 Multilayer Perceptrons (MLPs)

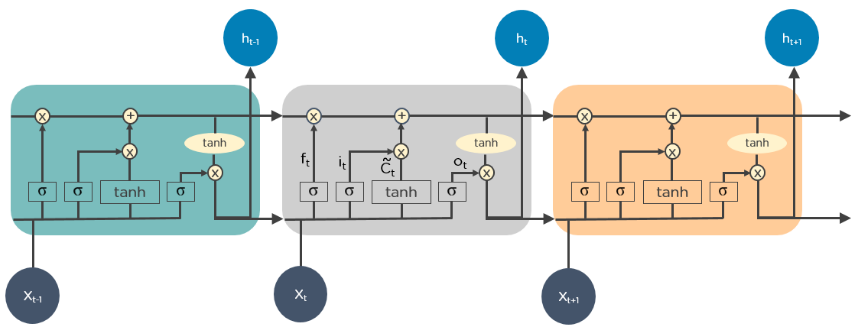
7 Self-Organizing Maps (SOMs)

1. Deep Belief Networks (DBNs)
2. 9 Restricted Boltzmann Machines (RBMs)
3. 10 Autoencoders

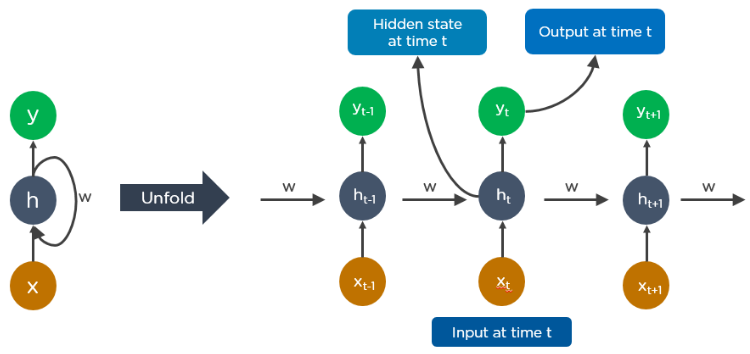
6.1. Convolutional Neural Networks (CNNs) CNNs [16], also known as ConvNets, are multilayer neural networks that are primarily used for image processing and object detection. In 1988, Yann LeCun created the first CNN, which he called LeNet. It could recognize characters such as ZIP codes and numerals. CNNs are commonly used to detect abnormalities, identify satellite photos, interpret medical imaging, forecast time series, and identify anomalies. Convolutional Neural Networks (CNN) are mostly employed in image processing. It assigns weights and biases to different items in the image and distinguishes them. In comparison to other classification methods, it requires less preparation. In order to capture the spatial and temporal dependencies in a picture, CNN employs relevant filters [17, 18]. LeNet, AlexNet, VG-GNet, GoogleNet, ResNet, and ZFNet are some of the different CNN architectures. Object detection, semantic segmentation, and captioning are just a few of the applications that CNNs are utilized for. Multiple layers process and extract features from data in CNNs: CNN features a convolution layer that consists of many filters that perform the convolution operation. CNNs have a Rectified 6 Linear Unit (ReLU) layer that performs operations on elements. A rectified feature map is the result. The rectified feature map is fed into a pooling layer after that. Pooling is a down sampling procedure that decreases the feature map's dimensionality. By flattening the two-dimensional arrays from the pooled feature map, the pooling layer turns them into a single, long, continuous, linear vector. When the flattened matrix from the pooling layer is given as an input, a fully connected layer arises, which classifies and labels the images. Figure 2 is an example of a CNNprocessed image. Figure 2 Example of Convolutional Neural Networks (CNNs)



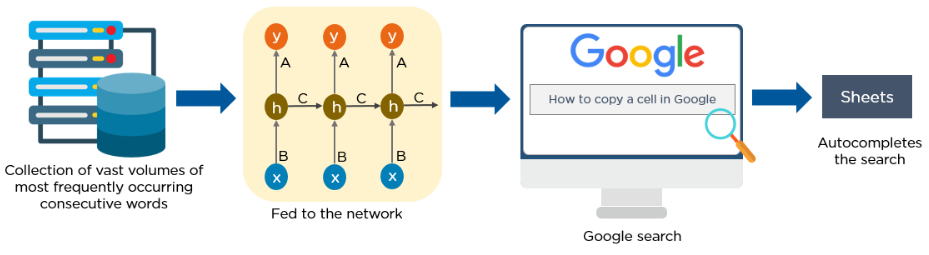
6.2. Long Short-Term Memory Networks (LSTMs) Long-term dependencies can be learned and remembered using LSTMs [19], which are a form of Recurrent Neural Network (RNN). The default behavior is to recall past information over long periods of time. LSTMs keep track of data throughout time. Because they remember past inputs, they are valuable in time-series prediction. Four interacting layers communicate in a unique way in LSTMs, which have a chain-like structure. LSTMs are commonly employed for voice recognition, music creation, and pharmaceutical research, in addition to time-series predictions. First, they forget about the portions of the previous state that aren't significant. They then update the cell-state values selectively. Finally, the state of some portions of the cell's output. Figure 3 is a diagram illustrating how LSTMs work. 7 Figure 3 Long Short-Term Memory Networks (LSTMs)



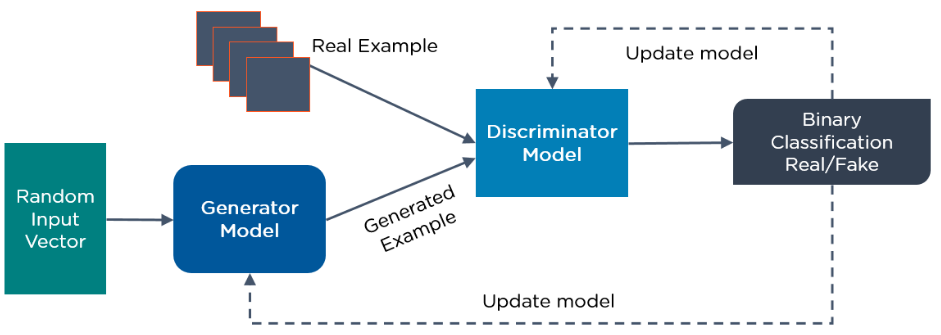
6.3. Recurrent Neural Networks (RNNs) The outputs from previous states are given as input to the present state in recurrent neural networks (RNN) [20]. RNN's hidden layers have the ability to remember information. The output created in the previous state is used to update the concealed state. RNN may be used to predict time series since it has Long Short-Term Memory [19], which allows it to remember prior inputs. The outputs from the LSTM can be given as inputs to the current phase since RNNs contain connections that create directed cycles. The LSTM's output becomes an input to the current phase, and its internal memory allows it to remember prior inputs. Image captioning, time-series analysis, natural-language processing, handwriting identification, and machine translation are all common uses for RNNs. Figure 4 shows how an RNN looks like after it's fully unfolded. Figure 4 Recurrent Neural Networks (RNNs)



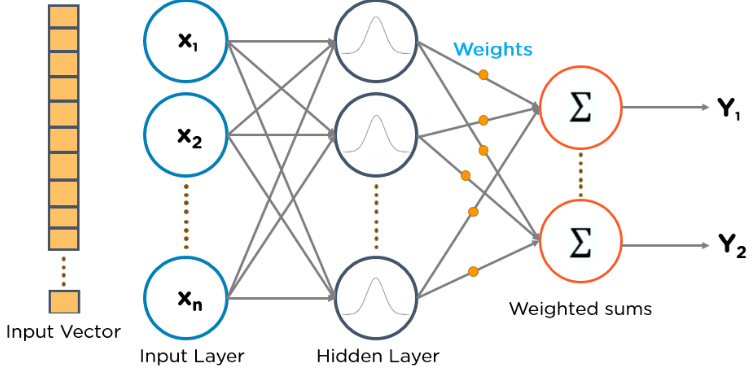
8 At time t-1, the output feeds into the input at time t. The output at time t feeds into the input at time t+1 in the same way. RNNs can handle any length of the input. The computation takes into consideration historical data, and the model size does not grow in proportion to the input size. An example of how Google's autocompleting feature works is illustrated in Figure 5. Figure 5 Recurrent Neural Networks (RNNs) for Google



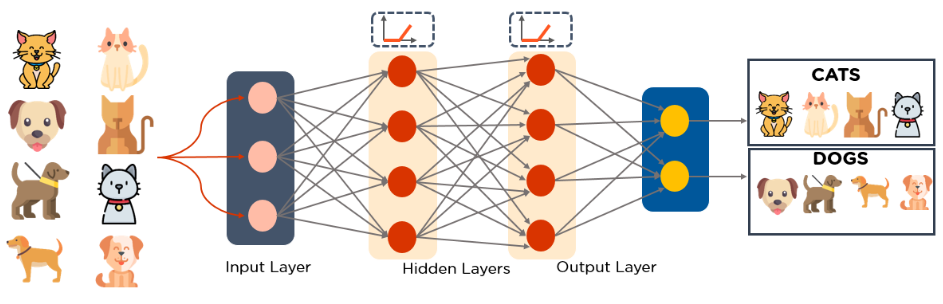
6.4. Generative Adversarial Networks (GANs) Ian Goodfellow spoke on Generative Adversarial Networks (GAN). It is made up of two networks: a Generator network and a Discriminator network. The generator creates the content, while the discriminator checks it for accuracy. The generator makes natural-looking images, and the discriminator determines whether or not they are natural. The GAN algorithm is a two-player minimax algorithm. Convolutional and feed-forward Neural Nets are used in GANs [21]. GANs are deep learning generative algorithms that generate new data instances that are similar to the training data. GAN is made up of two parts: a generator that learns to generate fake data and a discriminator that learns from that data. GANs have become increasingly popular over time. They can be used to improve astronomy photographs as well as to imitate gravitational lensing for dark matter investigations. GANs are used by video game producers to upscale lowresolution, 2D graphics in older games by using image training to recreate them in 4K or greater resolutions. GANs aid in the creation of realistic images and cartoon characters, as well as the creation of photographs of human faces and the rendering of 3D objects. The discriminator learns to tell the difference between the bogus data generated by the generator and the genuine sample data. The generator generates fraudulent data during early training, and the discriminator quickly learns to recognize it as such. To update the model, the GAN delivers the results to the generator and discriminator. Figure 6 is a diagram illustrating how GANs work. 9 Figure 6 Generative Adversarial Networks (GANs)



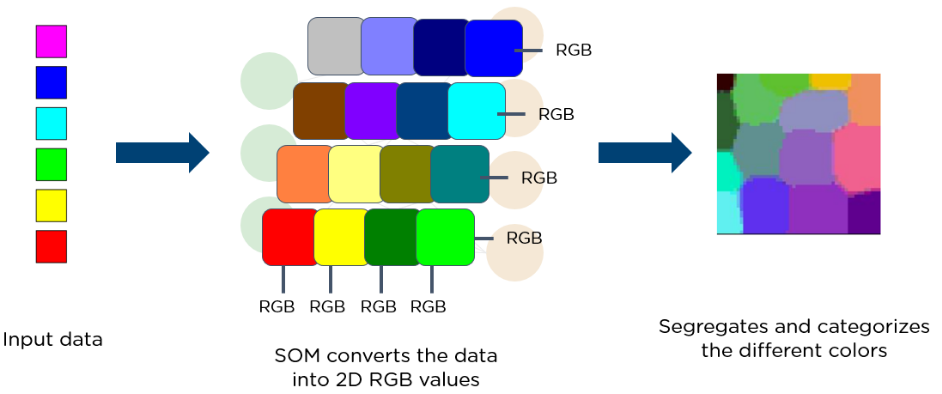
* 1. Radial Basis Function Networks (RBFNs) Radial basis functions are used as activation functions in RBFNs [22], which are a sort of feedforward neural network. They are used for classification, regression, and time-series prediction and have an input layer, a hidden layer, and an output layer. The similarity of the input to examples from the training set is used by RBFNs to do classification. The input layer of RBFNs is fed via an input vector. They have an RBF neuron layer. The output layer has one node per category or class of data, and the function finds the weighted total of the inputs. The Gaussian transfer functions, which have outputs that are inversely proportional to the distance from the neuron's center, are found in the neurons in the hidden layer. The output of the network is a linear combination of the radial-basis functions of the input and the parameters of the neuron. Consider the RBFN shown in Figure 7. Figure 7 Radial Basis Function Networks (RBFNs)



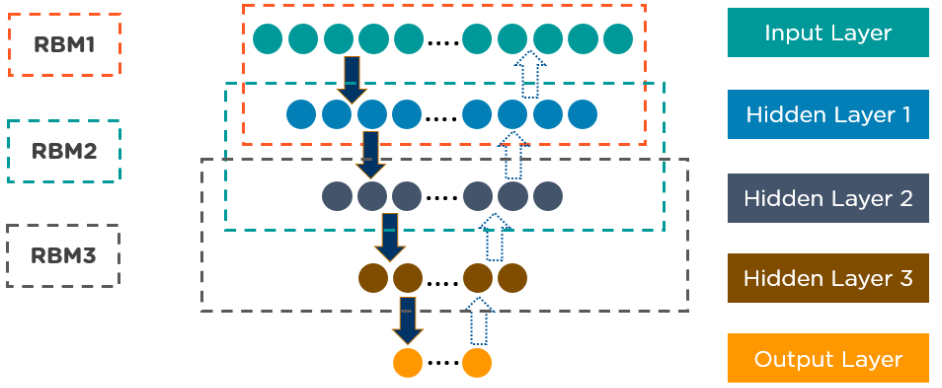
* 1. Multilayer Perceptrons (MLPs) MLPs [23] are a great starting point to learn more about deep learning. MLPs are a type of feedforward neural network that includes multiple layers of perceptron with activation functions. MLPs are made up of two fully connected layers: an input layer and an output layer. They have 10 the same set of input and output layers, but they can have several hidden layers, and they can be used to create speech recognition, image recognition, and machine translation software. The data is fed into the network's input layer using MLPs. The signal flows in one way because the layers of neurons are connected in a graph. MLPs use the weights that exist between the input layer and the hidden layers to compute the input. To decide which nodes to fire, MLPs use activation functions. ReLUs, sigmoid functions, and tanh are all activation functions. From a training data set, MLPs train the model to grasp the correlation and learn the dependencies between the independent and target variables. An MLP is shown in Figure 8 as an example. To classify photos of cats and dogs, the diagram computes weights and bias and applies appropriate activation functions. Figure 8 Multilayer Perceptrons (MLPs)



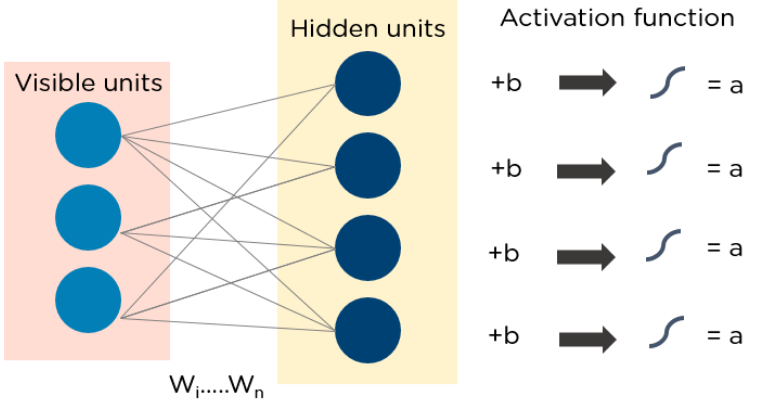
6.7. Self-Organizing Maps (SOMs) Professor Teuvo Kohonen created SOMs [24], which enable data visualization by using selforganizing artificial neural networks to reduce the dimensions of data. The problem of humans being unable to visualize high-dimensional data is addressed through data visualization. SOMs are designed to assist people in comprehending this multi-dimensional data. SOMs use a vector at random from the training data to initialize weights for each node. SOMs look at each node to see which weights are most likely to be the input vector. The Best Matching Unit is the winning node (BMU). The BMU's neighborhood is discovered through SOMs, and the number of neighbors decreases with time. The sample vector is given a winning weight using SOMs. The weight of a node changes as it gets closer to a BMU. The farther away a neighbor is from the BMU, the less it learns from it. For N iterations, SOMs repeat step two. A diagram of an input vector with various colors is shown in Figure 9. This information is fed into a SOM, which converts it to 2D RGB values. Finally, it categorizes and divides the various colors. 11 Figure 9 Self-Organizing Maps (SOMs)



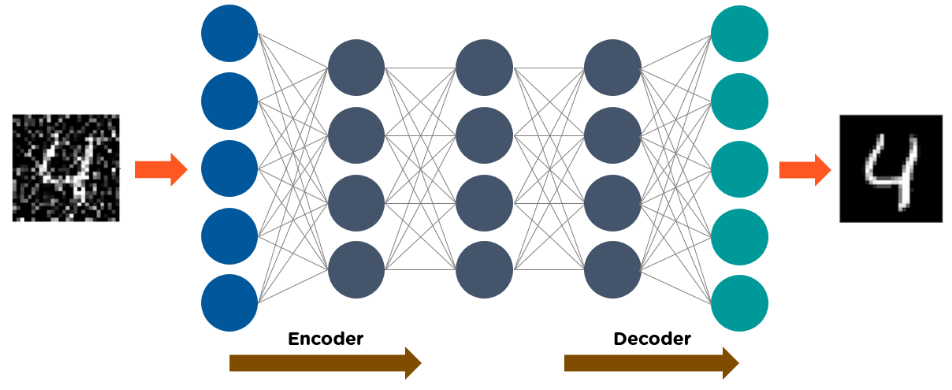
6.8. Deep Belief Networks (DBNs) The first step for training the deep belief network is to learn features using the first layer. Then use the activation of trained features in the next layer. Continue this until the final layer. Restricted Boltzmann Machines (RBM) is used to train layers of the Deep Belief Networks (DBNs), and the feed-forward network is used for fine-tuning. DBN learns hidden pattern globally, unlike other deep nets where each layer learns complex patterns progressively [25]. DBNs are generative models that consist of multiple layers of stochastic, latent variables. The latent variables have binary values and are often called hidden units. DBNs are a stack of Boltzmann Machines with connections between the layers, and each RBM layer communicates with both the previous and subsequent layers. Deep Belief Networks (DBNs) are used for imagerecognition, video-recognition, and motion-capture data. Greedy learning algorithms train DBNs. For learning the top-down, generative weights, the greedy learning method employs a layer-bylayer approach. On the top two buried layers, DBNs do Gibbs sampling steps. The RBM defined by the top two hidden layers is sampled in this stage. DBNs use a single pass of ancestral sampling through the rest of the model to generate a sample from the visible units. DBNs learn that a single bottom-up pass can infer the values of the latent variables in each layer. An example of DBN architecture is shown in Figure10: Figure 10 Example of Deep Belief Networks (DBNs)



6.9. Restricted Boltzmann Machines (RBMs) 12 RBMs [26] are randomized neural networks developed by Geoffrey Hinton that can learn from a probability distribution across a collection of inputs. For dimensionality reduction, classification, regression, collaborative filtering, feature learning, and topic modelling, this deep learning algorithm is utilized. RBMs are the fundamental components of DBNs. RBMs are divided into two layers: visible and hidden units. Every visible unit is linked to every hidden unit. RBMs have no output nodes and have a bias unit that is coupled to all of the visible and hidden units. RBMs have two phases: forward pass and backward pass. RBMs accept the inputs and translate them into a set of numbers that encodes the inputs in the forward pass. RBMs combine every input with individual weight and one overall bias. The algorithm passes the output to the hidden layer. In the backward pass, RBMs take that set of numbers and translate them to form the reconstructed inputs. RBMs combine each activation with individual weight and overall bias and pass the output to the visible layer for reconstruction. At the visible layer, the RBM compares the reconstruction with the original input to analyze the quality of the result. Figure 11 illustrates how RBMs function: Figure 11 Restricted Boltzmann Machines (RBMs)



6.10. Autoencoders Autoencoders [27] are a kind of feedforward neural network where the input and output are both the same. In the 1980s, Geoffrey Hinton invented autoencoders to overcome unsupervised learning difficulties. They're neural networks that have been trained to repeat data from the input layer to the output layer. Autoencoders are utilized in a variety of applications, including drug discovery, popularity prediction, and image processing. The encoder, the code, and the decoder are the three essential components of an autoencoder. Autoencoders are designed to take in information and turn it into a different form. Then they try to recreate the original input as closely as possible. When a digit's image isn't clear, it's sent into an autoencoder neural network. Autoencoders encode the image first, then compress the data into a smaller form. Finally, the image is decoded by the autoencoder, which produces the reconstructed image. Figure 12 shows how autoencoders work: 13 Figure 12 Autoencoders Autoencoders are used to reduce the dimension of data, as well as to solve problems like novelty detection and anomaly detection. The first layer in an autoencoder is produced as an encoding layer and then transposed as a decoder. Then, using the unsupervised method, teach it to duplicate the input. Fix the weights of that layer after training. Then go to the next layer until all of the deep net's layers have been pre-trained. Then go back to the original issue (Classification/Regression) that we want to solve with deep learning and optimize it using stochastic gradient descent, starting with the weights learned during pre-training. Autoencoder network consists of two parts [28]. The input is translated to a latent space representation by the encoder, which can be denoted in (1): ℎ = 𝑓(𝑥) (1) The input is reconstructed from the latent space representation by the decoder, which can be denoted in (2): 𝑟 = 𝑔(ℎ) (2) In essence, autoencoders can be described in (3). r is the decoded output which will be similar to input x: 𝑔(𝑓(𝑥)) = 𝑟 (3)



My Work :

Key Components:

Input Layer:

Represents a 28x28 grid of pixels, accommodating a single grayscale clothing image.

Flatten Layer:

Transforms the 2D image data into a 1D array of 784 values (28x28 pixels).

Dense Layers (128 neurons, ReLU activation):

Internal layers that extract features and patterns from the flattened image data.

Dropout Layer (20%):

Randomly drops connections during training to prevent overfitting.

Output Layer (10 neurons, softmax activation):

Generates a probability distribution over the 10 clothing classes.

Flow of Data:

Image Input: A 28x28 grayscale image is fed into the input layer.

Flattening: The image is flattened into a 784-element vector.

Processing through Dense Layers: The vector passes through the dense layers, where neurons activate based on learned patterns.

Dropout: During training, 20% of connections are randomly dropped to prevent overfitting.

Output: The final dense layer outputs 10 probabilities, one for each clothing class.

Interpretation: The class with the highest probability is considered the model's prediction.

Additional Points:

Training Process: The model learns by iteratively adjusting its weights and biases based on feedback from labeled training data.

Evaluation: The model's performance is assessed using a separate test set to determine its accuracy in classifying unseen images.

Visualizations: The code includes visualizations to display sample images, predicted labels, and model confidence scores, though these aren't directly part of the core model functionality.

Creating a Visual Design:

Tools: Use a diagramming tool or software like Draw.io, Lucidchart, or even PowerPoint to create a visual representation.

Components: Visually represent the input layer, flatten layer, dense layers, dropout layer, and output layer as distinct blocks.

Connections: Use arrows to depict the flow of data between layers.

Labels: Clearly label each component and connection to enhance clarity.

Additional Information: Optionally include boxes for data loading, model compilation, training, and evaluation steps to provide a more comprehensive overview.

import gzip

import numpy as np

from keras.utils import to\_categorical

from keras import layers, models

def load\_data(filepath, num\_samples):

with gzip.open(filepath, 'rb') as f:

data = np.frombuffer(f.read(), dtype=np.uint8, offset=16)

data = data.reshape((num\_samples, 28, 28, 1)).astype(np.float32) / 255.0

return data

def load\_labels(filepath, num\_samples):

with gzip.open(filepath, 'rb') as f:

labels = np.frombuffer(f.read(), dtype=np.uint8, offset=8)

return to\_categorical(labels, num\_classes=10)

# File paths

train\_images\_path = 'train-images-idx3-ubyte.gz'

train\_labels\_path = 'train-labels-idx1-ubyte.gz'

test\_images\_path = 't10k-images-idx3-ubyte.gz'

test\_labels\_path = 't10k-labels-idx1-ubyte.gz'import tensorflow as tf

mnist = tf.keras.datasets.mnist

(x\_train, y\_train),(x\_test, y\_test) = mnist.load\_data()

x\_train, x\_test = x\_train / 255.0, x\_test / 255.0

model = tf.keras.models.Sequential([

tf.keras.layers.Flatten(input\_shape=(28, 28)),

tf.keras.layers.Dense(128, activation='relu'),

tf.keras.layers.Dropout(0.2),

tf.keras.layers.Dense(10, activation='softmax')

])

model.compile(optimizer='adam',

loss='sparse\_categorical\_crossentropy',

metrics=['accuracy'])

model.fit(x\_train, y\_train, epochs=5)

model.evaluate(x\_test, y\_test)class\_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat',

'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']plt.figure()

plt.imshow(train\_images[0])

plt.colorbar()

plt.grid(False)

plt.show()plt.figure(figsize=(10,10))

for i in range(25):

plt.subplot(5,5,i+1)

plt.xticks([])

plt.yticks([])

plt.grid(False)

plt.imshow(train\_images[i], cmap=plt.cm.binary)

plt.xlabel(class\_names[train\_labels[i]])

plt.show()model = tf.keras.Sequential([

tf.keras.layers.Flatten(input\_shape=(28, 28)),

tf.keras.layers.Dense(128, activation='relu'),

tf.keras.layers.Dense(10)

])

Conclusion

The model has an accuracy of 85 % in determining the class of the image.

References