Understanding and Analyzing Deep Neural Networks

Deep Neural Networks (DNNs) have gained utmost importance these days due to their ability to solve complex engineering problems and providing critical decisions. Being a part of advanced machine learning techniques, DNNs have recently become the standard tool for solving a variety of computer vision problems. The 'deep' in DNNs refers to the fact that the number of layers can be very large (Tzen S. Toh et al, 2019). Deep Neural Networks (DNNs) have achieved a great performance in classification tasks in a wide range of applications, such as image recognition and natural language processing (Junghoon Chae *et al,* 2017). At its simplest, a neural network with some level of complexity, usually at least two layers, qualifies as a deep neural network (DNN), or deep net for short. Deep nets process data in complex ways by employing sophisticated math modeling (Jonathan Johnson, 2020). Undoubtedly, deep neural networks as a technology have revolutionized the discipline of machine learning. Deep nets allow a model's performance to increase in accuracy. They allow a model to take a set of inputs and give an output. The Deep Neural Networks allows a model to make generalizations on its own and then store those generalizations in a hidden layer, the black box. The black box is hard to investigate. Even if the values in the black box are known, they don't exist within a framework for understanding (John Emmons et al, 2019).

Deep learning is pretty much just a very large neural network, appropriately called a deep neural network. It's called deep learning because the deep neural networks have many hidden layers, much larger than normal neural networks, that can store and work with more information. Deep learning and deep neural networks are a subset of machine learning that relies on artificial neural networks while machine learning relies solely on algorithms (https://www.wgu.edu/blog/neural-networks-deep-learning-explained2003.html) (2020).

To truly understand deep neural networks, however, it's best to see it as an evolution. A few items had to be built before deep nets existed **(Figure 1)**.

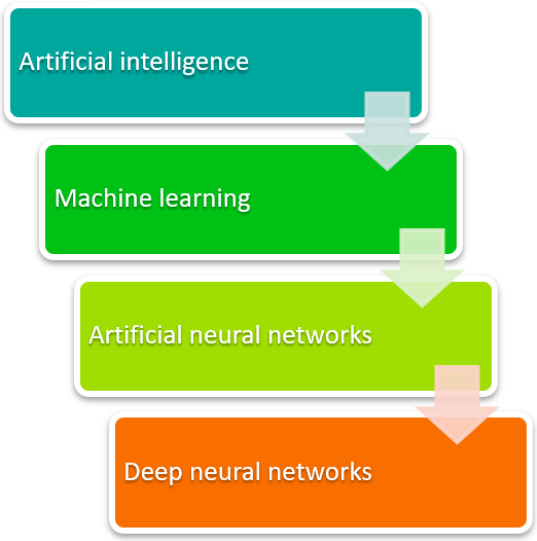


Figure 1 The evolution to Deep Neural Networks (DNN)

First, machine learning had to get developed. ML is a framework to automate (through algorithms) statistical models, like a linear regression model, to get better at making predictions. A model is a single model that makes predictions about something. Those predictions are made with some accuracy. A model that learns—machine learning—takes all its bad predictions and tweaks the weights inside the model to create a model that makes fewer mistakes.

The learning portion of creating models spawned the development of artificial neural networks. ANNs utilize the hidden layer **(Figure 2)** as a place to store and evaluate how significant one of the inputs is to the output. The hidden layer stores information regarding the input's importance, and it also makes associations between the importance of combinations of inputs.

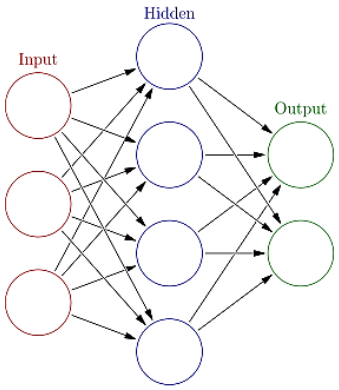


Figure 2 One hidden layer is considered an Artificial Neural Network (ANN)

Deep neural nets, then, capitalize on the ANN component. If that works so well at improving a model—because each node in the hidden layer makes both associations and grades importance of the input to determining the output—then why not stack more and more of these upon each other and benefit even more from the hidden layer?

So, the deep net has multiple hidden layers **(Figure 3)**. 'Deep' refers to a model's layers being multiple layers deep.

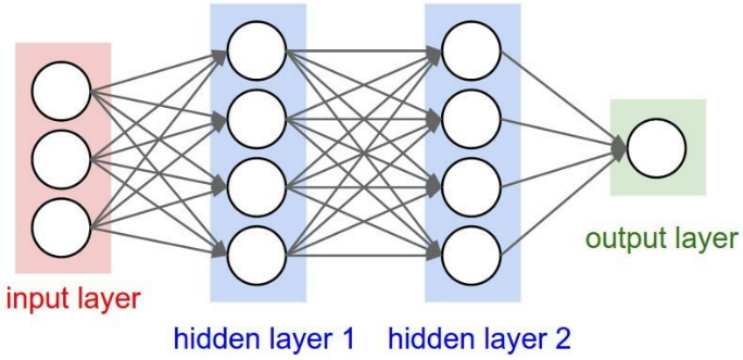


Figure 3 Two or more hidden layers comprise a Deep Neural Network

Improving accuracy: The black box problem

Deep nets allow a model's performance to increase in accuracy. They allow a model to take a set of inputs and give an output. The use of a deep net is as simple as copying and pasting a line of code for each layer. It doesn't matter

which ML platform you use; directing the model to use two or 2,000 nodes in each layer is as simple as typing the characters 2 or 2000.

But using these deep nets creates a problem: How do these models make their decisions? When utilizing these simple tools, a model's explainability is reduced significantly.

The Deep Net allows a model to make generalizations on its own and then store those generalizations in a hidden layer, the black box. The black box is hard to investigate. Even if the values in the black box are known, they don't exist within a framework for understanding.

Understanding Convolutional Neural Networks (CNNs)

CNNs are a class of Deep Neural Networks that can recognize and classify particular features from images and are widely used for analyzing visual images. Their applications range from image and video recognition, image classification, medical image analysis, computer vision and natural language processing.

Basic Architecture

There are two main parts to a CNN architecture **(Figure 4)**

- A **convolution tool** that separates and identifies the various features of the image for analysis in a process called as Feature Extraction.
- A **fully connected layer** that utilizes the output from the convolution process and predicts the class of the image based on the features extracted in previous stages (MK Gurucharan, 2020).

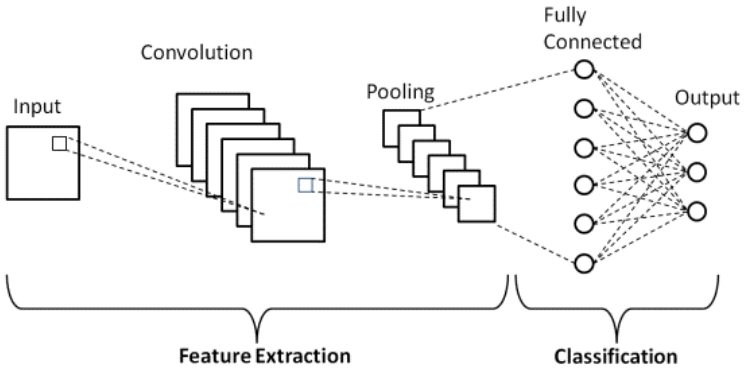


Figure 4 Layers of CNNs

Convolution Layers

There are three types of layers that make up the CNN which are the convolutional layers, pooling layers, and fully-connected (FC) layers. When these layers are stacked, a CNN architecture will be formed. In addition to these three layers, there are two more important parameters which are the dropout layer and the activation function which are defined below.

1. Convolutional Layer

This layer is the first layer that is used to extract the various features from the input images. In this layer, the mathematical operation of convolution is performed between the input image and a filter of a particular size MxM. By sliding the filter over the input image, the dot product is taken between the filter and the parts of the input image with respect to the size of the filter (MxM).

The output is termed as the Feature map which gives us information about the image such as the corners and edges. Later, this feature map is fed to other layers to learn several other features of the input image.

2. Pooling Layer

In most cases, a Convolutional Layer is followed by a Pooling Layer. The primary aim of this layer is to decrease the size of the convolved feature map to reduce the computational costs. This is performed by decreasing the connections between layers and independently operates on each feature map. Depending upon method used, there are several types of Pooling operations.

In Max Pooling, the largest element is taken from feature map. Average Pooling calculates the average of the elements in a predefined sized Image section. The total sum of the elements in the predefined section is computed in Sum Pooling. The Pooling Layer usually serves as a bridge between the Convolutional Layer and the FC Layer.

3. Fully Connected Layer

The Fully Connected (FC) layer consists of the weights and biases along with the neurons and is used to connect the neurons between two different layers. These layers are usually placed before the output layer and form the last few layers of a CNN Architecture.

In this, the input image from the previous layers are flattened and fed to the FC layer. The flattened vector then undergoes few more FC layers where the mathematical functions operations usually take place. In this stage, the classification process begins to take place.

One of the approaches of visualizing the FC layer is by dimensionality reduction. The goal here is to keep the structure of high-dimensional data in low-dimensional space intact as much as possible. Principal Component Analysis is one of the techniques in dimensionality reduction. It reduces the feature vectors' dimensions to two or three dimensions. It becomes harder to visualize the data

points, when the high-dimensional data is too close to the low-dimensional data using linear projections. For, PCA to work, non-linear methods work best in case of high-dimensional data. To solve this problem, another dimensionality reduction technique called the t-Distributed Stochastic Neighbor Embedding (t-SNE) is used for visualizing high-dimensional data. It works by modelling similar data points by small pairwise distance and vice versa. Hence, it captures local data structure while tracing global structure.

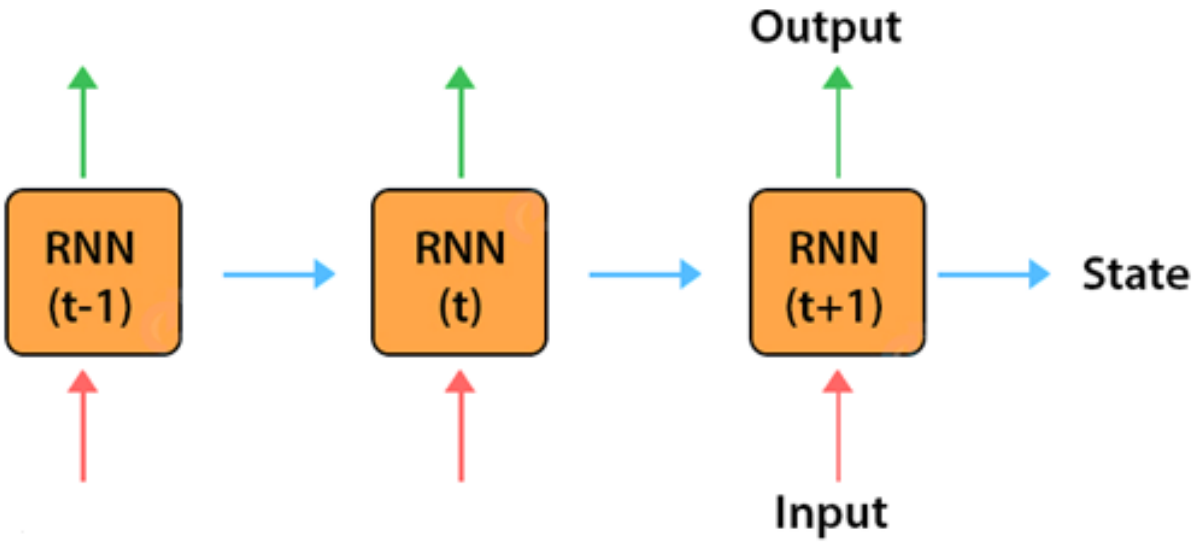
Figure 5 t-SNE visualization of the FC layer features of a pre-trained ImageNet classifier.

After applying t-SNE to the acquired feature vectors of images from the last layer of the trained network, the high-dimensional feature vector is compressed to two-dimensional i.e., each image in the dataset is displayed at its corresponding position **(Figure 5).**

*Understanding Recurrent Neural Networks (*RNNs)

RNNs are considered as exceptionally successful category of neural network models specially in applications with sequential data such as text, video and speech (Atefeh Shahroudnejad, 2021). They are a class of neural networks that are helpful in modeling sequence data. Derived from feedforward networks, RNNs exhibit similar behavior to how human brains function. Simply put, recurrent neural networks produce predictive results in sequential data that other algorithms can't. Common RNN architectures include, VanillaRNN, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU).

*Working of Recurrent Neural Networks (*RNNs)

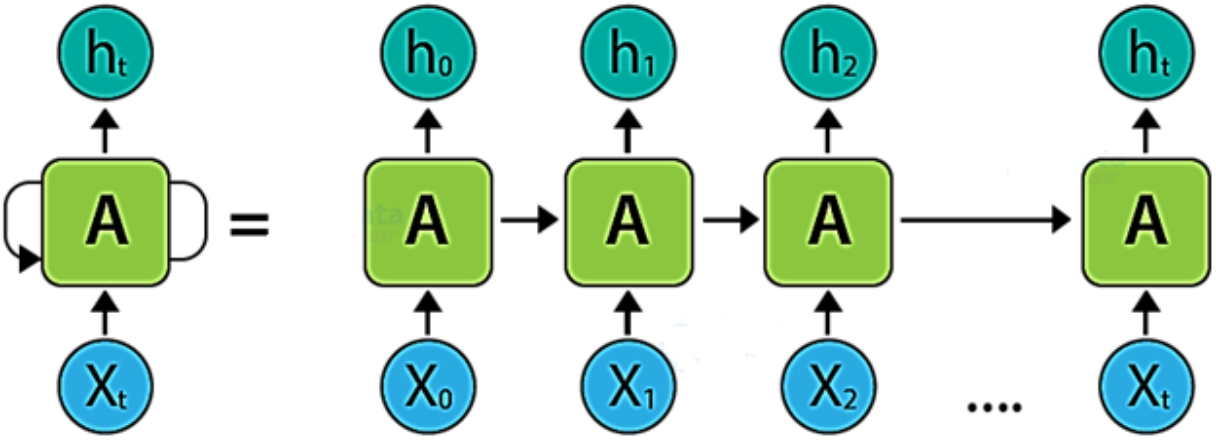


Artificial recurrent neural networks (RNNs) are widely used to solve tasks involving temporal data, e.g., speech and handwriting recognition, audio classification or time series forecasting. RNNs are characterized by the presence of feedback connections in a hidden layer, which allows generating a state–space representation that equips the network with short-term memory capability. RNNs are universal approximators of dynamical systems, meaning that, given enough neurons in the hidden layer, it is possible to fine-tune the weights to achieve any desired level of accuracy. Nevertheless, training via back-propagation through time is difficult due to the vanishing/exploding gradient

problem (Zachary C. Lipton et al, 2015). This has led to the development of new and faster techniques for training RNNs, including a different paradigm known as reservoir computing. Echo state networks (ESNs) constitute an important example of reservoir computing, where a recurrent layer (called a reservoir) is composed of a large number of neurons with randomly initialized connections that are not fine-tuned via gradient-based optimization mechanisms. The main idea behind ESNs is to exploit the rich dynamics generated by the reservoir with an output layer, the read-out that is optimized to solve a specific task (Dishashree Gupta, 2017).

Why RNNs?

Traditional neural networks lack the ability to address future inputs based on the ones in the past. For example, a traditional neural network cannot predict the next word in the sequence based on the previous sequences. However, a recurrent neural network (RNN) most definitely can. Recurrent Neural networks, as the name suggests are recurring. Therefore, they execute in loops allowing the information to persist.



In the above diagram, a neural network takes the input 'X_t' and gives use the output 'h_t'. Therefore, the information is passed from one step to the successive

step. This recurrent neural network, when unfolded can be considered to be copies of the same network that passes information to the next state.

DNNs Behavioral Analysis

The main goal of a DNN's explanation is analyzing its behavior and understanding its overall and internal behaviors. Functional Analysis attempts to explain the overall behavior by examining the relation between inputs and outputs, whereas Decision Analysis provides information on internal behavior through probing internal components rolls.

Functional Analysis explains the whole neural network as a non-opening black-box and trying to find the most relevant pixels for a specific decision regarding the input image. In other words, network's operation is interpreted through discovering the relation between inputs and their corresponding outputs. One of the approaches is by applying sensitivity analysis where we inspect the impact of each feature has on the model's prediction. For calculating feature sensitivity, we change the input image and observe the consequent changes in the results. Another approach, named as Local Interpretable Model-Agnostic Explanations (LIME), as the name suggests works on the assumptions that all models are locally linear. i.e., it approximates any black box ML model with a local, interpretable model to explain each and every individual prediction. One of the recently proposed approach for functional analysis is the Contrastive Explanations method (CEM). It works by finding the minimum features in the input which are enough to produce the same prediction, accompanied with the minimum features which should be absent in the input, to prevent the final prediction change.

Decision Analysis analyzes internal component of neural networks for extended transparency of the learned decisions overcoming the shortcomings of functional

methods such as not showing which neurons play more important role in making a decision.

Conclusion: DNNs are powerful algorithms based loosely on the human brain. Each node (neuron) in the neural network can only perform simple calculations, but DNNs work by connecting the nodes to form a number of layers, where each layer performs a calculation based on the output of the previous layer. In this way, the DNN can perform tasks that are much more complex than what a single node could achieve. In some cases, DNNs outperform other ML algorithms due to their capability of finding patterns in vast amounts of unstructured data. As the number of nodes and layers implies a large number of parameters that need to be learned, DNNs often require large amounts of data to achieve top performance. A type of DNN, called a deep recurrent neural network (RNN), consists of an input layer, numerous hidden layers corresponding to each feature of the input data, and an output layer. RNNs are particularly useful for modeling a sequence of data (e.g., designing chemical compounds) because the feedback loops allow the network to effectively retain a memory of inputs it has seen previously. Inspired by the visual cortex, convolutional neural networks (CNNs) allow for more effective processing of the complexity of a given input (e.g., raw images); this is done by transforming them into simpler forms, without the loss of important features for a prediction.

References

Junghoon Chae, Shang Gao, Arvind Ramanthan, Chad Steed, Georgia D. Tourassi, "Visualization for Classification in Deep Neural Networks", ***Oak Ridge National Laboratory, U.S. Department of Energy*** (2017).

Grégoire Montavon, Wojciech Samek, Klaus-Robert Müller, "Methods for Interpreting and Understanding Deep Neural Networks", ***Digital Signal Processing,*** Vol.73, February, P. 1-15 (2018).

Jonathan Johnson, July 27, Machine Learning & Big Data Blog, ***https://www.bmc.com/blogs/deep-neural-network/*** (2020).

Neural networks and deep learning explained, ***https://www.wgu.edu/blog/neural-networks-deep-learning-explained2003.html*** (2003).

MK Gurucharan, "Basic CNN Architecture: Explaining 5 Layers of Convolutional Neural Network", ***https://www.upgrad.com/blog/basic-cnn-architecture/***, Dec 7, (2020).

Atefeh Shahroudnejad, ***arXiv:2102.01792v1*** ***[cs.LG]*** 2 Feb, "A Survey on Understanding, Visualizations, and Explanation of Deep Neural Networks (2021).

Dishashree Gupta, "Fundamentals of Deep Learning – Introduction to Recurrent Neural Networks", ***https://www.analyticsvidhya.com/blog/2017/12/introduction-to-recurrent-neural-networks/***, December 7 (2017).

Zachary C. Lipton, John Berkowitz, Charles Elkan "A Critical Review of Recurrent Neural Networks for Sequence Learning", ***arXiv:1506.00019v4*** ***[cs.LG]*** 17 Oct (2015).

John Emmons, Sadjad Fouladi, Ganesh Ananthanarayanan, Shivaram Venkataraman, Silvio Savarese and Keith Winstein "Cracking open the DNN black-box: Video Analytics with DNNs across the Camera-Cloud Boundary" ***Proceedings of HotEdgeVideo '19, October 21, Los Cabos, Mexico*** (2019).

Tzen S. Toh a , Frank Dondelinger b , Dennis Wang c, "Looking beyond the hype: Applied AI and machine learning in translational medicine", ***EBioMedicine***, Volume 47, September, Pages 607-615 (2019).