Introduction to Deep Learning

Book · April 2022

CITATIONS
0

READS
614

3 authors, including:

Dr Subba Rao Polamuri
Bonam Venkata Chalamayya Engineering College
23 PUBLICATIONS **114** CITATIONS
SEE PROFILE

Makhan Kumbhkar
Devi Ahilya University, Indore
15 PUBLICATIONS **98** CITATIONS
SEE PROFILE

About the Book

Neural networks and deep learning are one of the most important programming paradigms ever invented. When we programme traditionally, we instruct the computer what to do by dividing large issues down into smaller, more manageable jobs. We, on the other hand, don't instruct the computer how to solve our issue in a neural network. Instead, it takes in information from the world around it and uses it to come up with a solution to the issue at hand.

As a concept, it's exciting to think about. There was a time when neural networks were only capable of outperforming more conventional methods in a few specific cases. The development of methods for learning in so-called deep neural networks in 2006 revolutionized the field. Deep learning is a term for these approaches. Deep neural networks and deep learning have been further improved, and currently they are able to perform very well on a wide range of computer vision, voice recognition, and natural language processing tasks.

Neural networks and deep learning are covered in this book, which aims to teach you about the fundamentals of neural networks in depth. You will be able to created neural network and deep learning code to tackle challenging pattern recognition tasks after going through the book. Using neural networks and deep learning to solve issues of your own design will be much easier now with the help of this book.

AGPH **Books**

Price: 550 INR

9 789394 339217

AGPH Books

FIRST EDITION

INTRODUCTION TO DEEP LEARNING

Subba Rao Polamuri
Makhan kumbhkar
Dr. D. Arul Pon Daniel

AGPH BOOKS

INTRODUCTION TO DEEP LEARNING

by

Subba Rao Polamuri

Makhan kumbhkar

&

Dr. D. Arul Pon Daniel

AG PH Books

2022

INTRODUCTION TO DEEP LEARNING

Subba Rao Polamuri, Makhan kumbhkar and

Dr. D. Arul Pon Daniel

ISBN – 978-93-94339-21-7

Published by:

AGPH Books (Academic Guru Publishing House)

Bhopal, M.P. India

Contact: +91-7089366889

ABOUT AUTHORS

Subba Rao Polamuri now examining Ph. D in Computer Science and Engineering at Jawaharlal Nehru Technological University, Kakinada, East Godavari, Andhra Pradesh. My regions of examination incorporate Bigdata, Data Mining, Artificial Intelligence, Machine learning, Areas of interest Stock computing, Deep Learning techniques Extract quality of results, Image Processing, Analytical of Data Representation and Deep Learning. He was distributed Two SCIE Indexing diary and more Scopus Indexing Journal. He is likewise distributed one patent.

Makhan kumbhkar Working as an Asst. Professor in the Department of Computer Science & Elex. at Christian Eminent College, M.P. Indore & Visiting faculty at school of data science and forecasting Devi Ahilya University, Indore. He graduated in Bachelor of Computer Science at Vikram University, Ujjain, Madhya Pradesh, India. He secured Master of Computer Application in Computer Application at Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, India. He is Pursuing Ph.D. in the field of Data Science at School of Computer Science & Information Technology, DAVV, Indore, Madhya Pradesh, India. He is in teaching profession for more than 12 years. He has presented number of papers in National and

International Journals, Conference and Symposiums. He has done many patents and copyrights. He is editorial board member of different journals. His main area of interest includes Machine learning, Deep learning, Natural language processing and Internet of Things.

Arul Pon Daniel was born at Tuticorin District, Tamilnadu (TN), India, in 1986. He received his Doctor of Philosophy in Computer Science from Periyar University, Salem, TN, India, in 2018. He is currently serving as Vice-principal (Administration) cum Assistant Professor in the Department of Computer Science& Applications, Loyola College, Namakkal, TN, India. He was recognized by Elsevier as Outstanding Reviewer in 2017 and Research Excellence award by Institute of Scholarsin 2020. He has published 23 periodicals and got one patent grant. He has delivered many faculty developments workshops and webinars to other HEI's. He has been a member of various international Technical Associations, Editorial & Review Boards. His research interests in Data Science include Data Preparation, Feature Engineering, Pattern Recognition and Data Visualization.

PREFACE

Just a few years ago, there were no legions of deep learning scientists developing intelligent products and services at major companies and startups. When we entered the field, machine learning did not command headlines in daily newspapers. Our parents had no idea what machine learning was, let alone why we might prefer it to a career in medicine or law. Speech recognition and computer vision were two of the few practical uses of machine learning before it became a mainstream academic field. Many of these applications needed so much subject expertise that they were seen as distinct fields, with machine learning serving just as a minor part of the overall picture. This book focuses on deep learning approaches that are based on neural networks, which were considered antiquated at the time.

It is the goal of this book to simplify and teach the ideas of deep learning in a very complete way with relevant, full-fledged examples of neural network designs, such as recurrent neural networks (RNNs) and Sequence to Sequence (seq2seq) for NLP applications. Bridge the gap between theory and practice with this book.

The book offers a good starting point for people who want to get started in deep learning, with a focus on NLP.

ACKNOWLEDGMENTS

Having an idea and turning it into a book is as hard as it sounds. The experience is both internally challenging and rewarding. Especial thanks the individuals that helped make this happen.

Heartly thank you to the people who encouraged the work and helped in growing. Your advice on both research as well as on career have been invaluable.

Thankful to Government of India, various other resources that have helped in providing data for the completion of this book.

Nobody has been more important in the pursuit of this project than the members of family. Thank you, Mom and Dad. Without your unconditional support, this book never would have come to be. They are the ultimate role models.

Last but not least, thankyou reader, for your interest, time, and trust to work with this book.

TABLE OF CONTENT

CHAPTER 1

THE NEURAL NETWORK

1.1. Building Intelligent Machines

One of the most remarkable parts of the body is the brain. Each of these senses is shaped by our innate predispositions. It allows us to preserve our memories, feel emotions, and even fantasies about the future. You wouldn't know what to do without it. Without it, we'd be nothing but primitive creatures. We are what we are because of our brains.

The baby brain weighs less than a pound, but it is able to solve problems that even the most powerful supercomputers cannot. A few months after birth, children are able to identify their parents' faces, distinguish between different things in their surroundings, and even

distinguish between different sounds. Objects can be tracked even when partly or entirely occluded, and they've already formed a sense for natural physics within a year.

And by the time they're in kindergarten, they have a sophisticated grasp of grammar and a vocabulary of thousands of words.

One of the most amazing parts of the human body is the brain. Everything we see, hear, smell, taste, and feel is filtered via our sense of touch. Memory, emotions, and even dreams may all be stored in our minds because to it. Without it, we'd be little more than primordial creatures, able to perform only the most basic of reactions. We are what we are because of our brains.

In spite of its small size, the baby brain manages to solve issues that our most powerful supercomputers find insurmountable. A few months after birth, children are able to identify their parents' faces, distinguish between different things in their surroundings, and even distinguish between different sounds. Objects can be tracked even when partly or entirely occluded, and they've already formed a sense for natural physics within a year.

And by the time they're in kindergarten, they have a sophisticated grasp of grammar and a vocabulary of thousands of words.

Since the dawn of time, we've fantasized of creating intelligent machines with minds like ours—robots that clean our houses, self-driving automobiles, and microscopes that instantly identify illness. These artificially intelligent devices demand us to tackle some of the most complicated computing problems we have ever encountered; challenges that our brains can already answer in microseconds. To solve these issues, we'll need a whole new approach to computer programming based on methods that have been created in the last decade. Deep learning is a term used to describe an area of artificial computer intelligence that is particularly active.

1.2. Limits of Traditional Computer Programs

Why are certain issues so hard for computers to solve? There are two things that old-school computers excel at: 1) doing quick math, and 2) specifically following an instruction manual. So, if you're looking to undertake a lot of financial analysis, you're in luck. It is possible to use

conventional computer programmes for this purpose. What if our goal is more ambitious, like building a software that can read someone's work without them having to do it themselves?

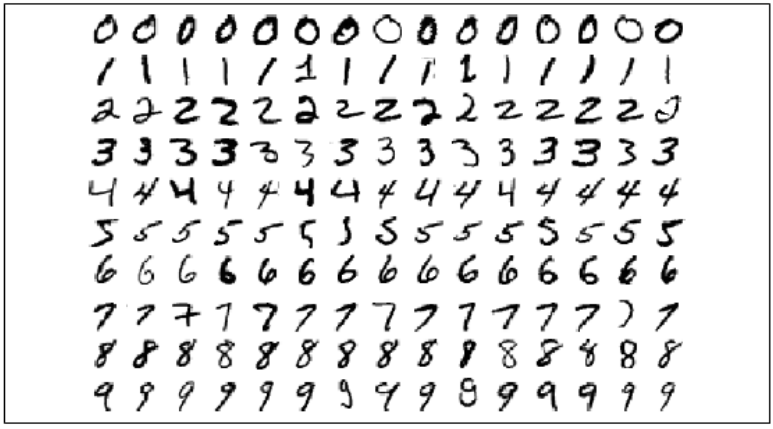


Figure 1.1: Image of handwritten dataset[*]

Source:https://www.researchgate.net/figure/MNIST-dataset-of-handwritten-digits_fig1_324877673

It doesn't matter how many times we look at the digits in Figure 1.1, we can always tell which ones are zeros and which ones are ones since they are all written in a slightly different manner. Try writing a computer programme to

[*]https://upload.wikimedia.org/wikipedia/commons/thumb/2/27/MnistExamples.png/320px-MnistExamples.png

solve this problem. How can we discern one digit from the other?

Let's begin with something easy! For example, if our picture has just one closed loop, we may say that we have a zero. While all of the instances in Figure 1-1 seem to meet this requirement on the surface, this is not a necessary condition. What if someone misses a step in the process? Furthermore, how can you tell a muddy zero from an even worse six?

Figure 1.2: A zero that's algorithmically difficult to distinguish from a six[*]

[*]Source:https://www.oreilly.com/library/view/fundamentals-of-deep/9781491925607/ch01.html

Some form of cutoff for the distance between the beginning and finish of a loop might be established. But where precisely should we draw the line? But this is only the beginning of our problems. What is the difference between a three and a five? Is it between the ages of 4 and 9? Even if we're cautious in our observation and experimentation, it's evident that this isn't going to be an easy endeavour.

For example, object identification, voice understanding, automatic translation, and many more come into this area. Because we don't understand how our brains work, we have no idea what kind of software to develop. We may not know how to accomplish it, even if we knew, since the software may be really difficult.

1.3. The Mechanics of Machine Learning

For these types of difficulties to be resolved effectively, we'll need a method that is fundamentally different. When it comes to traditional computer programmes, they share many characteristics with the teachings we learned in school as youngsters. An instruction set is the method through which we learn how to multiply, solve an

equation, or extract a derivation from an equation. The most natural things we learn at a young age, on the other hand, are learned by observation rather than through formula.

The fact that our parents did not teach us how to recognise a dog by measuring its nose or body proportions began when we were only a few months old. When we made a bad guess, we were reprimanded and given more examples of dogs to help us learn to identify them. To put it another way, our brains already had a model for how we would see the world before we were born. We learned to rely on that model as a child because it used our senses to make educated guesses about what we were feeling and experiencing. Our model would be strengthened if our parents' guesses were correct.

If our parents told us that we were mistaken, we would change our model to reflect this new knowledge. As we learn from more and more instances, our model of the world grows more and more accurate. Though it's obvious that we're all doing this without even recognising, we can still utilise this to our benefit.

In artificial intelligence, deep learning is a subfield of machine learning, which is based on the principle of learning by doing. A computer may be trained to solve problems using a model and a limited set of instructions, rather than a long list of rules. This approach is called machine learning. In the long run, we believe that a well-fitting model will be able to correctly address the issue.

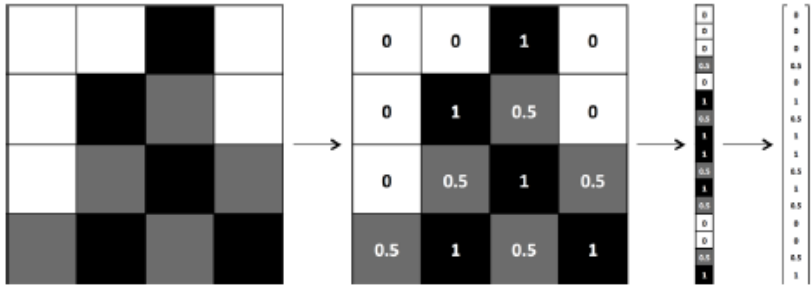


Figure 1.3: The process of vectorizing an image for a machine learning algorithm[*]

For the sake of formalising this concept mathematically, let's be a little more specific about what this entails. Let us express our model as a function h (x,θ) in order to better understand it. x is a vector representation of an example. For example, if x were a grayscale image, the vector's

[*]Source: https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm

components would be pixel intensities at each position, as shown in Figure 1.3.

Let's go through a simple example to have a better grasp on how machine learning models work on an intuitive level. Consider the following scenario: we were trying to figure out how to forecast test results based on the number of hours we get to sleep and the number of hours we study the day before. We collect a lot of data, and for each data point $X = [x_1 x_2]^T$ we record the number of hours of sleep we got (x1), the number of hours we spent studying (x2), and whether or not we outperformed the rest of the class. Our goal, then, might be to learn a model h (x, θ) with parameter vector $\theta = [\theta_0 \theta_1 \theta_2]^T$such that:

$$h(\mathbf{x}, \theta) = \begin{cases} -1 & \text{if } \mathbf{x}^T \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} + \theta_0 < 0 \\ 1 & \text{if } \mathbf{x}^T \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} + \theta_0 \geq 0 \end{cases}$$

To put it another way, we believe that the above-described blueprint represents our model h (x,)(This specific design depicts a linear classifier that splits the coordinate plane into two parts geometrically). Then, given an example x,

we want to figure out a parameter vector that allows our model to correctly forecast (-1 if we perform below average, and 1 otherwise) given an input example x. A linear perceptron is a model that's been around since the 1950s and is known as a linear perceptron.

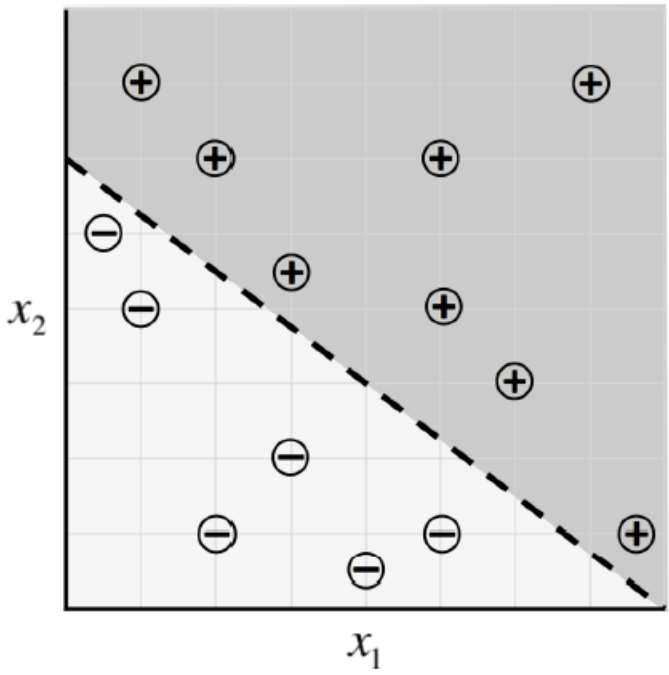


Figure 1.4: Sample data for our exam predictor algorithm and a potential classifier[*]

[*]Source: https://docs.oracle.com/cd/E18283_01/datamine.112/e16808/classify.htm

Then it comes to light, by selecting $\theta = [-24\ 3\ 4]^T$, Every data point is correctly predicted by our machine learning algorithm:

$$h(\mathbf{x}, \theta) = \begin{cases} -1 & \text{if } 3x_1 + 4x_2 - 24 < 0 \\ 1 & \text{if } 3x_1 + 4x_2 - 24 \geq 0 \end{cases}$$

An ideal value for this parameter vector θ, the classifier is positioned in order to maximise the number of accurate predictions. A large number of options (perhaps an infinite number) exist for that are optimum in the vast majority of circumstances. Most of the time these choices are so near to each other that the difference is insignificant. Our choice of might be narrowed down if this is not the choice of θ.

Despite what seems to be a decent set-up, a number of important problems remain. To begin with, how do we determine an ideal value for the parameter vector θ? This difficulty can only be solved via a method known as optimization. Iteratively changing the parameters of a machine learning model until the error is reduced is the goal of an optimizer.

Another thing is clear: this specific model (the linear perceptron) has a very limited ability to learn new associations. In Figure 1.5, for example, a linear perceptron cannot adequately explain the data distributions.

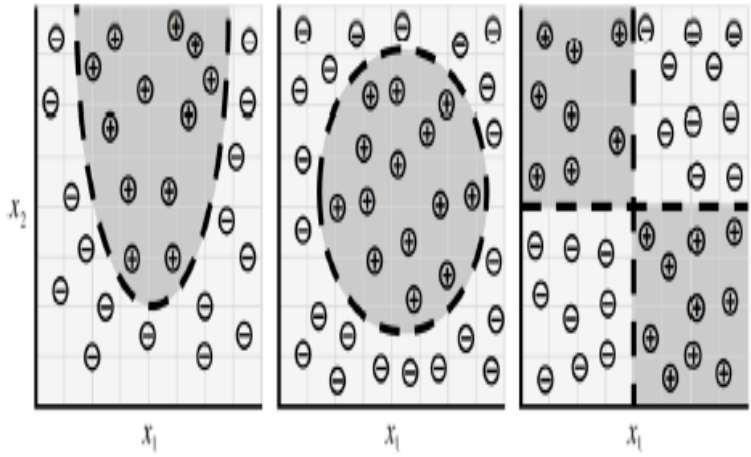


Figure 1.5: As our data takes on more complex forms, we need more complex models to describe them[*]

Nevertheless, this is only the top of a very large iceberg. Object recognition and text analysis are two examples of more complicated issues that need the use of high-dimensional data and nonlinear interactions. Machine learning researchers have sought to construct models that

[*]Source:https://www.softwaretestinghelp.com/data-mining-process/

approximate the structures used by human brains in order to handle this level of complexity. In the field of computer vision and natural language processing, deep learning has shown to be a powerful tool for solving complex issues. Other machine learning algorithms can't compare to these methods in terms of accuracy; in fact, they may even outperform them!

CHAPTER 2

Artificial Neural Network

2.1. Background

Inspired by human central nervous systems, artificial neural networks were created. Neurons (sometimes called "neurode"; processing components; or units) are used in an artificial neural network to create a network that replicates a real neuronal network.

When it comes to artificial neural networks, there is no one definition that is universally agreed upon. When these properties are present in a statistical model, it might be labelled "Neural.":

1. comprises sets of learning-based adaptive weights, i.e. numerical parameters, and
2. in the ability of their inputs to approximate non-linear functions.

Training and prediction activate the adaptive weights, which may be compared to the strength of the connections between neurons.

When it comes to neural networks, there is no apparent demarcation between the subtasks that are assigned to each individual unit, which is similar to the way biological neural networks function. The term "neural networks" refers to models that are often used in statistics, cognitive psychology, and artificial intelligence applications. When it comes to simulating the central nervous system, both theoretical and computational neuroscience use neural network models.

In modern software implementations of artificial neural networks, bio-inspired approaches have been largely abandoned in favors of a more practical approach based on statistics and signal processing, which is more efficient. Multiple of these systems have adaptive and nonadaptive features that exist side by side. Neuronal networks or parts of neural networks (such as artificial neurons) have a role in both of these systems. Traditional artificial intelligence connectionist models have nothing in common with the more general approach used by these systems, which is

more suited for dealing with real-world issues than the traditional models.

Processing and adaptation in non-linear, distributed, parallel, and local environments are common themes among all of them. A shift away from high-level (symbolic) artificial intelligence and toward low-level (sub-symbolic) machine learning was signaled by neuronal network models in the late 1980s, with information encoded in the parameters of a continuously evolving system.

2.2. ANN

ANN (artificial neural networks) is a branch of artificial intelligence whose design is influenced by the structure and function of the human brain. Artificial neural networks are often based on the structure of the neural networks seen in the human brain, which is a widespread practise. Artificial neural networks, which are analogous to actual brains, are composed of neurons that are linked to one another at various layers of the networks. These sorts of neurons are referred to as nodes in the scientific literature.

The lecture on artificial neural networks goes through every aspect of the subject in great detail. The topics covered in this section include anything from artificial neural networks to adaptive resonance theory, Kohonen self-organizing maps, fundamental building blocks, unsupervised learning, and genetic algorithms, among other things.

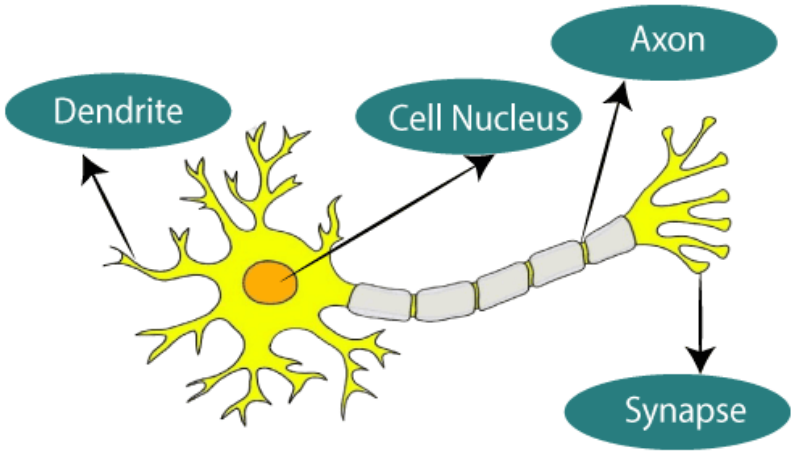


Figure 2.1: Biological Neural Network[*]

[*]Source: https://www.javatpoint.com/artificial-neural-network

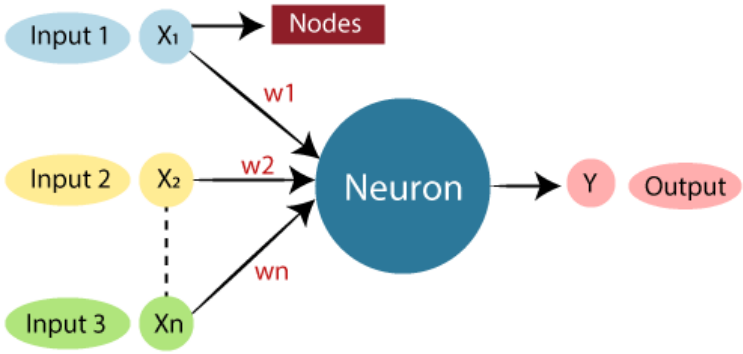


Figure 2.2: typical Artificial Neural Network[*]

An Artificial Neural Network (ANN) is a kind of artificial intelligence (figure 2.2) that attempts to replicate the network of neurons that exists in the human brain as shown in figure 2.1 in order to enable computers to understand and make decisions in the same way that humans do. It is necessary to programme a computer such that it behaves like a network of connected brain cells in order to develop an artificial neural network.

The human brain contains around 1000 billion neurons, which is a large number. Each neuron has an association point that is anywhere between 1,000 and 100,000 in number. The human brain stores information in a

[*]Source: https://www.javatpoint.com/artificial-neural-network

disorganized manner, allowing us to access several bits of information at the same time if necessary. The human brain is a remarkable parallel computer when it comes to its capabilities.

An illustration of a digital logic gate, which receives an input and outputs a result, may be useful in understanding the artificial neural network's operation. A "OR" gate is formed by feeding two inputs onto it. If one or both of the inputs are "On," we obtain the value "On." Similarly, if both inputs are set to "Off," the output will be set to "Off" as well. The output in this case is a function of the input. The functions of the human brain are considerably different from those of the chimpanzee's brain. Our brain's neurons are continually "learning," and as a result, the link between its outputs and its inputs is always evolving.

2.3. The architecture of an artificial neural network:

It is necessary to understand the architecture of a neural network in order to comprehend the concept of an artificial neural network. To develop a neural network made up of many artificial neuronal cells, units are arranged in layers

and connected together. Check out all of the many artificial neural network layers that are available to you.

Three layers comprise the bulk of an artificial neural network:

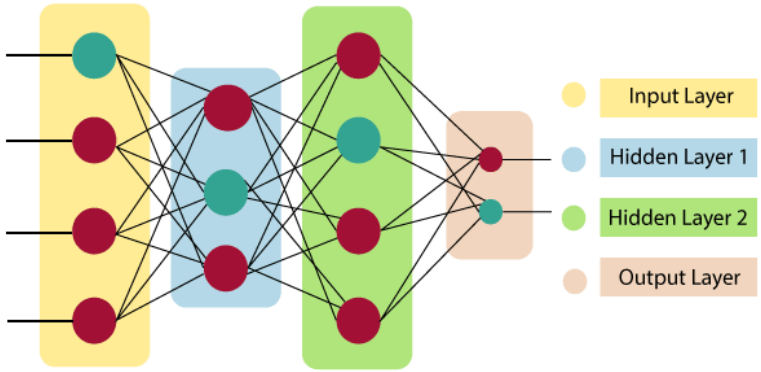


Figure 2.3: Layers of ANN[*]

Input Layer:

Multiple formats are supported by this application, as the name says.

Hidden Layer:

The hidden layer is located in the space between the input and output layers. This programme does all of the

[*]Source: https://www.javatpoint.com/artificial-neural-network

calculations necessary to uncover hidden patterns and traits.

Output Layer:

Using the hidden layer, the input is turned into the output, which is subsequently conveyed with the help of the communication layer.

With the help of an artificial neural network, the input is processed, and the weighted sum of the inputs is calculated, as well as a bias. This computation is expressed mathematically using a transfer function.

$$\sum_{i=1}^{n} Wi * Xi + b$$

As an input to an activation function, it determines the weighted total. It is up to activation functions to decide whether or not a node will fire. The output layer is only accessible to individuals who have been fired. Depending on the kind of work we're doing, we may choose from a variety of activation functions.

2.4. How do artificial neural networks work?

Visualizing an ABN as a weighted directed graph, with the artificial neurons functioning as nodes, is the most straightforward method of understanding it. Neuron outputs and neuron inputs may be represented as directed edges with weights, which can be used to indicate the connection between them. There is an external source that sends a vector-based pattern and picture to the Artificial Neural Network (ANN). Mathematically, inputs are allocated notation x(n) for every n inputs.

Last but not least, the weights assigned to each of the inputs are multiplied by the corresponding values of each input (these weights are the details utilised by the artificial neural networks to solve a specific problem). The connection strength of an artificial neural network is often expressed by the weights allocated to the neurons in the network. During the processing stage, the total weighted inputs are added together.

Bias is applied if the weighted sum equals zero, so that the output does not equal zero or something like. The input for bias is the same, and its weight is 1. It's possible for the

total weighted inputs to have any value between 0 and 1. An activation function is used to maintain the response within the intended range by benchmarking a given maximum value and passing the weighted sum of inputs through the activation function.

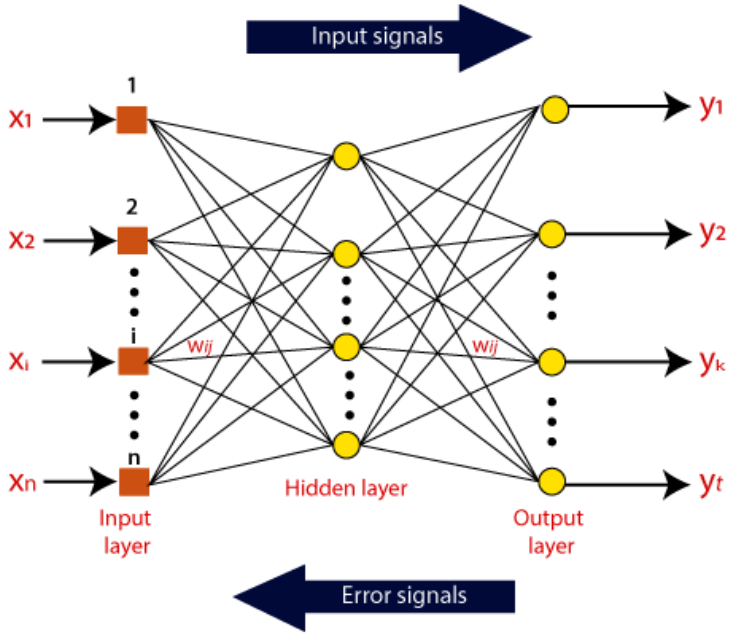


Figure 2.4: Working of ANN[*]

Transfer functions that are employed in the activation function are called activation functions. Non-linear and

[*]Source: https://static.javatpoint.com/tutorial/artificial-neural-network/images/artificial-neural-network6.png

linear activation functions are the most common types of activation functions. Many other sets of activation functions have been developed throughout time, but among of the most often used ones include binary, linear, and hyperbolic sigmoidal Tan. Let us take a look at each of them in details:

Binary:

1 or 0 is the result of using a binary activation function. To achieve this, a threshold value has been established. A value of 1is returned if the activation function's final output is greater than 1, else a value of 0 is returned.

Sigmoidal Hyperbolic:

It's common to think of the Sigmoidal Hyperbola function as a "S" curve. An approximation of the output based on the real net input is achieved here using the tan hyperbolic function. Here's how to define the function:

F(x) = (1/1 + exp(-????x))

2.5. Types of Artificial Neural Network:

There are many different types of artificial neural networks (ANNs), each of which is capable of performing

a certain task in a way that is akin to that of the human brain. It is anticipated that the overwhelming majority of artificial neural networks would exhibit some properties in common with their more intricate biological counterparts and will function magnificently in their intended jobs when properly trained. This may be shown via the use of segmentation and classification.

Feedback ANN:

It is necessary to send the output of this kind of ANN back into the network in order to get the best-evolved result. According to the Center for Atmospheric Research at the University of Massachusetts, Lowell, the state of Massachusetts is experiencing a drought. Feedback networks are excellent for dealing with optimization problems since they feed information back into themselves. To rectify internal system errors, feedback ANNs are used.

Feed-Forward ANN:

There is one layer of input, one layer of output, and at least one layer of neuron in a feed-forward network. By comparing its output to its input, the network's intensity

may be determined based on the collective activity of the linked neurons. With this network, the main benefit is that it can analyse and detect input pattern evaluations.

Convolutional Neural Networks

In the field of image and handwriting recognition, convolutional neural networks are very useful. For example, they are built by first sampling an area of an image, and then utilizing the picture's characteristics to generate a representation of it. This leads to the usage of several layers, as can be seen from the description, making these models the first deep learning models.

Recurrent Neural Networks

A time-varying data pattern necessitates a recurrent neural network. As RNNs mature, they may be expected to unroll. In an RNN, each time step, the same layer is applied to the input (i.e., the state of previous time steps as inputs).

As an input to the next firing, or time index T + 1, RNNs feature feedback loops in which the output from a previous firing or time index T is supplied. A neuron's

output may be sent back to itself as an input in certain instances. In video and translation contexts, where the following word is predicated on the context of the prior text, they are well-suited for applications involving sequences, such as video sequences. RNNs come in a variety of shapes and sizes:

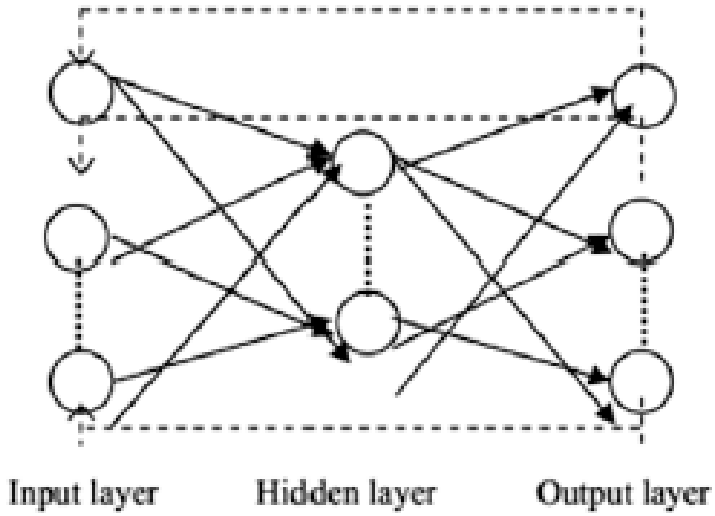
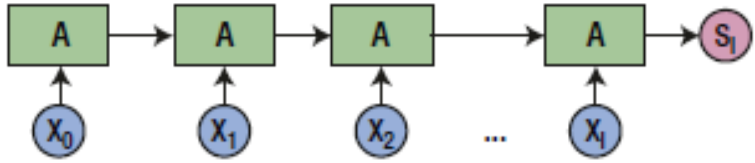


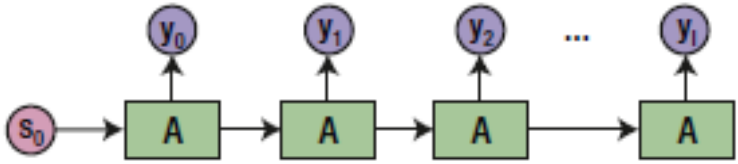
Figure 2.5: Recurrent neural network[*]

This collection of RNNs encodes recurrent neural networks such that the network may accept an input in the form of a sequence

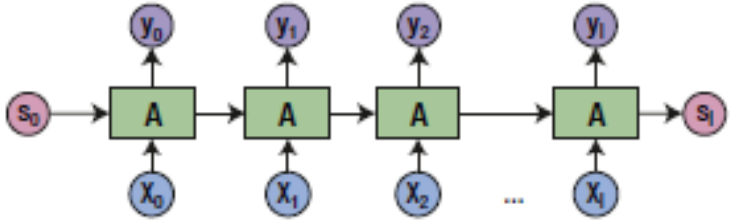
[*]Source:https://towardsdatascience.com/introducing-recurrent-neural-networks-f359653d7020



As with the words and numbers in sentences, these neural networks that generate recurrent neural networks essentially produce a series of numbers or values



Combination RNNs: These networks combine the previous two RNNs into a single RNN model. In NLG (natural language generation) tasks, general RNNs are utilised to produce sequences.



Encoder-Decoder Networks

Networks that employ encoder-decoder networks build internal representations of inputs or "encode" them, which are used as inputs for other networks to produce outputs. A categorization of the input is helpful, but it's important to go beyond that. Text tagging an image based on ideas might be the ultimate output, or it can be the same modality (e.g., language translation).

Recursive Neural Networks

When using a neural network, weights are applied recursively and are utilised largely to find out the data's hierarchy or structure. A recursive neural network is a tree-like structure, while an RNN is a chain. NLP makes extensive use of these networks, for example, to determine the emotional connotation of a statement. Syntactical groupings of words in a phrase may have a significant impact on how the overall emotion is perceived.

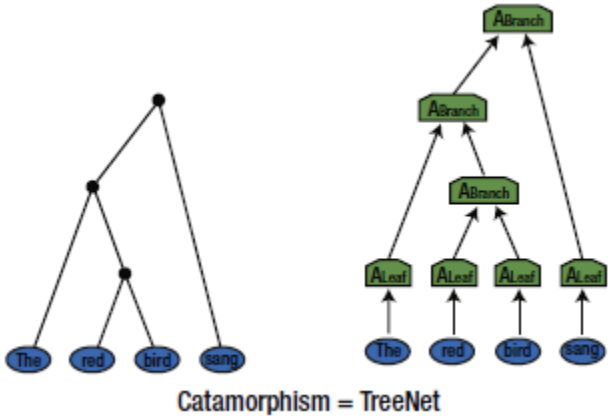


Figure 2.6: Recursive neural network[*]

Because of the wide variety of network types and their varying speeds and quality capabilities, some may be used in many different situations, while others are better suited to specialized uses.

2.6. Learning Paradigm

One learning paradigm corresponds to a concrete goal, the other two relate to abstract ones. Learning may take place in three ways: supervised, unsupervised, and reinforcement-based.

[*]Source: https://www.kdnuggets.com/2016/06/recursive-neural-networks-tensorflow.html

2.6.1. Supervised learning

An example pair is provided to us in the form of a training set in supervised learning (x; y); x Є X; y 2 Y the goal is to discover a functionf : X → Y in the allowed class of functions that matches the examples. For example, we want to infer the data's implicit mapping, and the cost function is connected to the mismatch between our mapping and the data; this previous knowledge is implicit in the issue domain.

The average squared error between the network's output, f(x), and the goal value y, across all the example pairings, is a popular cost to reduce. The well-known backpropagation technique for neural network training may be obtained by minimising this cost using gradient descent for the multilayer perceptrons class of neural networks.

Pattern recognition (also known as classification) and regression are examples of supervised learning tasks (also known as function approximation). It is possible to apply the supervised learning paradigm to sequential data as well (e.g., for speech and gesture recognition). An

"instructor" function, in the form of a function that offers constant feedback on the quality of solutions that have been generated so far, may be seen as this.

2.6.2. Unsupervised learning

In unsupervised learning, data x is supplied and cost is to be minimised, it can be any function of the data x and the network's output, f.

A task's cost function is determined by a number of factors, including our pre-existing assumptions and the nature of the activity being modelled (the implicit properties of our model, its parameters and the observed variables).

2.6.3. Reinforcement learning

In reinforcement learning, generally the data x in not given, it is generated by an agent's interactions with the environment. At every point in time, denoted by t , the agent performs an action yt and the environment generates an observation xt and an instantaneous cost ct , according to some (usually unknown) dynamics. Finding a strategy for making decisions that reduces the predicted long-term

costs, such as the total cost of ownership, is the ultimate goal. Each policy's long-term cost and the dynamics of the environment are unknown, but may be calculated.

ANNs are often used as a component of a reinforcement learning algorithm, which is a widespread practice. Because of their ability to mitigate accuracy losses even when the discretization grid density is reduced for numerically approximating the solution of the original control problems, artificial neural networks (ANNs) have been used in conjunction with dynamic programming to solve multi-dimensional nonlinear problems such as vehicle routing, natural resource management, and medicine.

The reinforcement learning paradigm encompasses a wide range of tasks such as control issues, games, and other sequential decision-making tasks.

2.7. Advantages of Artificial Neural Network (ANN)

Parallel processing capability:

Because of its numerical significance, artificial neural

networks can carry out several tasks at once.

Storing data on the entire network:

Rather than being stored in a database, traditional programming data is preserved on the network as a whole. When a few pieces of data are accidentally lost at one area, the network's capacity to function is not hampered.

Capability to work with incomplete knowledge:

Although the input may be minimal, an ANN is nevertheless capable of producing a result. In this instance, the performance degradation is caused by the significance of the missing data.

Having a memory distribution:

Determine the instances and, by displaying these examples to the network, encourage the network to produce the required output. This will allow the ANN to adapt to the situation. The output of an event may be incorrect if it cannot be viewed by the network in its entirety, and the network's advancement is directly proportional to the number of occurrences that are picked for inclusion.

Having fault tolerance:

In the case of ANN, the extortion of one or more cells does not prevent the network from producing output, and this property lends the network fault-tolerance.

2.8. Disadvantages of Artificial Neural Network:

Assurance of proper network structure:

The structure of artificial neural networks may be determined in a variety of ways, including genetic algorithms. It is possible to attain the optimal network structure via trial and error, experience, and more trial and error.

Unrecognized behavior of the network:

It is ANN's most important problem. When an ANN generates a testing solution, it does not explain why or how it came to that conclusion. It diminishes the trustworthiness of the system.

Hardware dependence:

The construction of artificial neural networks necessitates computers capable of parallel computation. Consequently,

the production of the equipment is contingent on its success.

Difficulty of showing the issue to the network:

Numerical data may be processed by ANNs. Before introducing ANN to a problem, numerical values must be translated. The network's performance will be directly impacted by the presentation technique that is used. It is entirely dependent on the skills and talents of the person playing it.

2.9. Employing artificial neural networks

As a function approximation mechanism, ANNs may be used to learn from observed data, which is perhaps their biggest value. It's not that simple to use them, and a thorough grasp of the underlying theory is required.

Choice of model: Data representation and application are also factors to consider in this decision. Models that are too complicated have a propensity to impair learning.

Learning algorithm: Learning algorithms have a variety of trade-offs. Any method may be used to train on a fixed data set if the relevant hyperparameters are used. A great

deal of trial and error is necessary, however, when it comes to choosing and fine-tuning an algorithm for use in training using hypothetical data.

Robustness: Depending on how well the model, cost function, and learning algorithm are set, the resulting artificial neural network (ANN) may be very durable.

With the proper implementation, artificial neural networks (ANNs) may be used in online learning and large data set applications without difficulty. Because of their straightforward implementation and the existence of a high number of local dependencies in the structure, fast, parallel hardware implementations are achievable in this context.

2.10. Applications

The ability to infer a function from data is at the heart of artificial neural network models' usefulness. A function like this can't be designed by hand due to the sheer complexity of the data or job at hand.

2.10.1. Real-life applications

In general, artificial neural networks are used for the following types of activities:

- a kind of regression analysis that includes time series prediction, fitness approximation, and function modelling, among other things.
- Recognition of patterns and sequences, detection of novelties, and sequential decision-making are all aspects of pattern recognition.
- Filtering, grouping, separating sources blindly, and compressing data are all examples of data processing.
- Manipulators that guide robots, as well as robotic prostheses, are all part of the robotics field.
- Control, especially Computer Numeric Control.

Quantum chemistry, backgammon, chess, poker, pattern recognition (radar systems, face identification, object recognition, and more), sequence recognition (gesture, speech, handwritten text recognition), medical diagnosis, financial applications (e.g. automated trading systems), and data mining (or knowledge discovery in databases, "KDD"), visualization and e-mail spam filtering are some of the areas where these technologies can be put to use.

Several tumors have been identified using artificial neural networks. Lung cancer radiography may be improved by an ANN-based hybrid system called HLND, which increases both the accuracy and speed of diagnosis.

Prostate cancer has been detected with the use of these networks. Diagnoses may be used to create specialised models based on data from a large number of patients rather than just one. It is not necessary to assume correlations between variables in the models.

The neural networks may potentially be able to forecast colorectal cancer in the future. Using neural networks, it is possible to predict the presence of colorectal cancer in a patient more accurately than with traditional clinical procedures. Once the networks have been trained, they may be able to forecast patients from a range of different institutions.

2.10.2. Neural networks and neuroscience

In the discipline of theoretical and computational neuroscience, biological brain systems are analysed theoretically and simulated numerically. The topic is tightly linked to cognitive and behavioural modelling

since neurological systems are so directly linked to cognitive processes and behaviour.

The field's goal is to develop models of biological brain systems in order to better understand how these systems operate. Neuroscientists try to connect observable biological processes (data), biologically plausible mechanisms for brain processing and learning (biological neural network models), and theory in order to obtain this insight (statistical learning theory and information theory).

Types of models

There are several degrees of abstraction that may be used to characterise and depict various properties of neural networks in the actual world, and they are discussed in detail below. Therefore, they range from simple neural models to more abstract neuronal modules that represent complete subsystems, depending on their length of time. Models of long- and short-term plasticity of brain systems, as well as their link to learning and memory, are covered at all levels of analysis, from the level of the individual neuron to the level of the whole system.

Memory networks

It has been possible to include distributed representations and self-organizing maps into neural networks for a long time, dating back to the beginning of artificial neural networks. Take, for example, sparse distributed memory, where neural networks encode patterns. In this case, the "neurons" operate as address encoders and decoders for content-addressable memories, and the "neurons" encode and decode the patterns.

Deep learning was shown to be advantageous in semantic hashing when a significant number of texts were analysed. A network model of word-count vectors generated from the analysed data set was employed, and deep learning was found to be beneficial.

Memory addresses for documents are mapped such that semantically comparable documents are stored next to one other. All addresses that vary by a few bits from the query document's address may be accessed to find documents that are comparable to the query document.

Turing Machines, which are connected to deep neural networks and external memory resources, allow Google

Deep-Neural Mind's Turing Machines to improve the capabilities of deep neural networks. The combined system is comparable to a Turing machine in that it is differentiable from beginning to finish, but it may be taught via gradient descent since it is differentiable end to end. Initial studies indicate that NTMs may infer fundamental algorithms from input and output samples such as copying, sorting, and associative recall depending on input and output samples.

Facebook researchers have created Memory Networks, a new kind of neural network that incorporates long-term storage as well as working memory. It is possible to read and write to the long-term memory such that it may be used for prediction.

It has been shown that these models may be used to generate textual responses when used in conjunction with long-term memory as a (dynamic) knowledge base for question answering (QA).

2.11. Theoretical properties

Computational power

The universal approximation theorem establishes that the

MLP is a universal function approximator. When it comes to identifying the number of neurons or the weights to be employed, the evidence, on the other hand, is not useful.

It was shown by HavaSiegelmann and Eduardo D. Sontag that a certain recurrent architecture with rational-valued weights (as opposed to real number-valued weights) has the complete capacity of a Universal Turing Machine when just a limited number of neurons with normal linear connections are used. It is possible to construct a machine with super-Turing power by using weights with irrational values, as well as other methods.

Capacity

The term "capacity" refers to a quality of artificial neural network models that is commonly defined as their ability to imitate any certain function. It has everything to do with the amount of data that can be kept in a network, as well as the idea of complexity in general.

Convergence

It is impossible to generalise about convergence since it is influenced by a variety of variables. To begin with, there

may be multiple local minima. The cost function and the model determine this. Second, even though the optimization technique is guaranteed to converge around a local minimum, it may not do so when farther away. Third, certain procedures become unworkable when dealing with a big set of data or inputs.

When it comes to actual applications of theory, it has been shown that theoretical promises of convergence are unreliable.

Generalization and statistics

Overtraining has emerged as a concern in applications where the objective is to produce a system that can generalise effectively in new situations. This occurs when the network's capacity surpasses the required free parameters in complicated or overspecified systems. To prevent this issue, there are two schools of thought:

Overtraining may be checked for by using cross-validation and other similar approaches, and hyperparameters can be selected to reduce the generalisation error. Secondly, regularisation may be used. When working in a probabilistic (Bayesian) framework, regularisation can be

accomplished by selecting a higher prior probability over simpler models; however, in statistical learning theory, the goal is to minimise two quantities: the "empirical risk" and the "structural risk," which roughly correspond to the error over the training set and the predicted error in unseen data as a result of overfitting, respectively.

Using formal statistical methodologies such as the mean squared error (MSE) cost function, it is possible to quantify the confidence in a trained neural network. An estimation of variance may be made using the mean squared error of a validation set.

Assuming that the network's output follows a normal distribution, this value may be used to construct the confidence interval for that output. This method is acceptable as long as the output probability distribution stays fixed and the network is not changed.

CHAPTER 3

Deep Learning

It is a branch of machine learning that is based on algorithms that try to represent high-level abstractions in data using model architectures, sophisticated structures, or other methods that are constructed from a number of non-linear transformations, such as neural networks. Deep learning is the term used to describe this field of machine learning.

It is part of a larger family of machine learning techniques that use data representations as a basis for training models. To represent an observation (such as an image), you may use many different representations such as a collection of edges, areas with certain shapes, etc., or even a vector of intensity values per pixel. Representations that aid in the learning of tasks (such as recognising faces or facial

expressions) from examples are available. There are claims that deep learning will replace manual features with effective techniques for unsupervised or semi-supervised learning of features and hierarchical extraction of features.

It has been described in a variety of ways, but deep learning is one of the most popular. A subset of machine learning techniques known as "deep learning":

- Nonlinear processing units are used to extract and manipulate features in a cascade of several layers. The output of the previous layer serves as the input for the next layer. Pattern analysis (unsupervised) and classification (supervised) are examples of uses for the methods (supervised).
- Based on learning many layers of features or representations from the input (unsupervised), a hierarchical representation is created by deriving higher level characteristics from lower level ones.
- Learning representations of data is part of the wider area of machine learning

- A hierarchy of ideas may be formed by learning various representations at different degrees of abstraction.

These definitions of non-linear processing all have in common the use of many non-linear processing units and the learning of feature representations from low-level to high-level features. A nonlinear processing unit layer's composition is determined by the job at hand while using deep learning. Artificial neural networks and sophisticated propositional equations have been used in the creation of deep learning. It is possible to organise latent variables in deep generative models, such as the nodes in deep belief networks and deep Boltzmann Machines, layer-wise.

As the signal goes from the input layer to the output layer in a deep learning approach, a parameterized transformation is a processing unit containing trainable parameters, such as weights and thresholds. A credit assignment route is a sequence of transformations from input to output. The length of a causal analysis paragraph (CAP) is up to the author.

CAP depth is equal to the number of hidden layers multiplied by one for feedforward neural networks (the output layer is also parameterized).

The CAP may go on indefinitely in recurrent neural networks, where a signal may pass through a layer more than once. Many researchers in the area believe that deep learning comprises more than two nonlinear layers (CAP > 2), and Schmidhuber considers CAP > 10 to be extremely deep learning.

3.1.1. Fundamental concepts

Distributed representations are used in deep learning techniques. distributed representations are built on the notion that observable data is formed by interactions between many distinct components at various levels. These components are arranged into numerous tiers, which correlate to different degrees of abstraction or composition, according to deep learning's premise. The number of layers and the size of the layers may be varied to obtain different degrees of abstraction.

Deep learning algorithms benefit greatly from the use of layered explanatory factors. Concepts that are more

abstract are learned from more concrete ones. These designs are often built using a greedy layer-by-layer method. Dissecting these abstractions and identifying which traits are valuable for learning is made easier by deep learning.

A radically different strategy is encouraged in the case of supervised learning problems, when label information is readily available during training, by deep learning. Instead of focusing on feature engineering, deep learning techniques focus on the end-to-end learning from raw features rather than feature engineering that may be labor-intensive and vary depending on the task.

However, feature engineering is completely avoided in deep learning. For end-to-end optimization, layered structures are often necessary, starting with raw features and ending with labels. Deep learning is therefore logically linked to raw-feature-based, end-to-end learning from this perspective, which necessitates the use of layered structures. Deep learning may be used in a wide range of fields, many of which need students to work under the guidance of an instructor (e.g., supervised speech and image recognition).

Unsupervised learning challenges frame many deep learning methods. As a result, these algorithms are able to make use of unlabeled data that supervised algorithms are not able to do. These algorithms profit from the fact that unlabeled data is frequently more plentiful than labelled data, making this a key advantage. Unsupervised training is possible for complex structures like the deep belief network.

3.2. Deep Learning in ANN

The use of deep learning algorithms to implicitly draw meaningful inferences from input data is one of the subsets of machine learning known as "deep learning."

Supervised or semi-supervised deep learning is the norm. Representation learning is the foundation of deep learning. Learning from typical instances is preferred to employing task-specific algorithms. A database of cat photos is needed in order to construct a model that can identify cats based on their species.

The most common deep learning architectures are:

- Convolutional neural networks

- Recurrent neural networks
- Generative adversarial networks
- Recursive neural networks

3.3. Deep Learning Working

There are many similarities between the process of learning to recognize a dog with an infant and that of using deep learning in computer systems. A statistical model is the end result of each algorithm in the hierarchy, which performs nonlinear transformations on its inputs. Once the output reaches an acceptable degree of precision, the process repeats again. The term "deep" comes from the sheer volume of data that must be processed before it can be deemed usable.

A dog may be detected in a picture using typical machine learning, but this method requires the programmer to be quite particular when instructing the computer on what it should be searching for. In order to extract features from a dog, a programmer must be able to accurately define the dog's feature set. The advantage of deep learning is that it can develop its own set of capabilities on its own, without the involvement of a human designer. Unsupervised

learning is both quicker and more accurate than supervised learning.

Dog or not a dog metatags may be used to teach computer programmes in the early stages by providing a set of training images. The programme creates a feature set for the dog that is used to build predictive models. As long as there are four legs and a tail in an image, the computer may assume that everything in it should be labelled dog. In spite of this, the computer does not recognise the designations four legs or tail. It will do nothing more than scan digital data for pixel patterns. With each iteration, the prediction model becomes more complex while also improving in accuracy.

Deep learning systems may now be developed with the training data and processing power that programmers require thanks to the emergence of big data and cloud computing technologies. Using unlabeled, unstructured data, deep learning programming may provide precise predictions because of its ability to develop complex statistical models straight from iterative output. Data that people and machines gather is becoming more

unstructured and unlabeled as the Internet of Things (IoT) grows more widespread.

3.4. Methods of Deep Learning

A wide range of approaches may be used to develop deep learning models. There are several instances of these strategies, including learning rate decay, transfer learning, training from scratch, and dropout.

Learning rate decay: The learning rate is a hyperparameter -- element that sets the system's parameters before the learning process begins --Every time the model weights are changed, it determines how big of a shift the model suffers in reaction to the estimated inaccuracy. High rates of learning might lead to unstable training processes or the acquisition of a poor set of weights. When learning rates are too low, a training programme runs the risk of being bogged down and becoming ineffective over time.

The learning rate decay method -- adaptable learning rates or learning rate annealing -- is the practice of adjusting the learning rate in order to improve performance and shorten training periods. During training, the simplest and most

popular methods of adjusting the learning rate include approaches that gradually decrease the learning rate.

Transfer learning: This is a method that needs access to the internal workings of a previously trained model. To begin, fresh data including previously unknown categories is fed into the current network by users. New jobs may be completed with more precise categorization skills when network improvements are made. Due to the fact that it requires far less data than other approaches, this one may be computed in only minutes or hours instead of days or weeks or months.

Training from scratch: Developing a network architecture that can learn the features and models requires a developer to collect a large tagged data set. This approach is very useful for new applications and those with a large variety of output kinds. There are a few instances when this method is used because of the time and resources it consumes during training.

Dropout: To avoid overfitting in networks with a large number of parameters, it is common practice to remove units and connections from neural networks at random

during training. The dropout strategy has been demonstrated to improve neural network performance in supervised learning applications such as speech recognition, document categorization, and computational biology.

3.5. Advantages of deep learning

Now that you know the difference between DL and ML, let's look at some of the advantages of DL.

In 2015, a team of Google engineers studied the categorization processes used by NN. They also discovered, by coincidence, that neural networks are capable of hallucinating and creating visually stunning artwork.

Deep learning's capacity to recognise patterns and abnormalities in massive amounts of raw data allows it to efficiently offer accurate and trustworthy analytical findings to experts. Amazon, for example, has more than 560 million goods for sale and more than 300 million registered users. That many transactions can only be tracked by an AI programme; even an entire army of accountants would struggle to keep track.

In contrast to typical machine learning, deep learning does not depend as much on human knowledge. Even if the developers don't know exactly what they're looking for, deep learning helps us to make discoveries in data. Customer retention is one example of an algorithmic goal you may have, but you are unsure of the attributes of your customers that will allow the system to achieve this goal.

3.6. Problems of deep learning

A large volume of high-quality data requires a lot of time and effort. With 14 million photos and over 20,000 categories, ImageNet has been the most comprehensive and well-prepared collection of samples for a long period of time. While the company was established in 2012, a more comprehensive and flexible database was first launched in 2017.

Another problem with deep learning is that it doesn't explain why it makes the conclusions it does. A lack of knowledge about what an output should be makes it harder to evaluate the model's performance. You won't be able to test the algorithm to see why, for example, your

system determined that the photo is of a cat and not a dog as in conventional machine learning.

The development of algorithms for deep learning is quite expensive. It's difficult to do without the expertise of educated mathematicians. Deep learning also requires a lot of resources. To train the models, you'll need enough of RAM and strong GPUs. Data, weight parameters, and activation functions are stored in a large amount of memory as an input propagates through the network. In certain cases, researchers choose to utilise alternative algorithms instead of deep learning algorithms since they are less power demanding, even if this results in worse prediction accuracy.

3.7. Application of DL in real-life problems

Deep learning is being used in a wide range of businesses and for a diverse range of purposes:

Speech recognition: Deep learning is at the heart of almost every commercially available voice recognition system, including Microsoft Cortana, Amazon Alexa, Google Assistant, and Apple Siri.

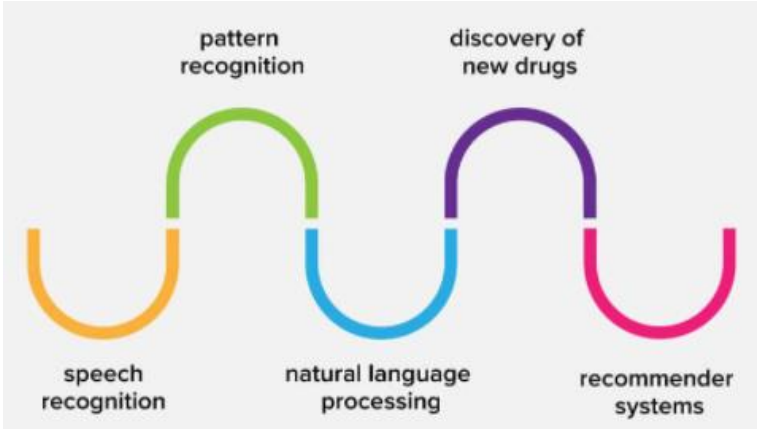


Figure 3.1: Applications of Deep Learning [*]

Pattern recognition Automated medical diagnostics are already more accurate than the human eye, according to new research.

Natural language processing: Since the early 2000s, neural networks have been utilised to develop language models. Because of the development of LSTM, machine translation and language modelling have both improved.

Discovery of new drugs: new biomolecules that may be utilised to treat illnesses like Ebola and multiple sclerosis have been predicted using the Atom Net neural network.

[*]Source: https://serokell.io/blog/deep-learning-and-neural-network-guide

Recommender systems: User preferences across a wide range of industries may be studied using deep learning. Netflix is one of the best instances of this kind of business model out there.

3.8. Deep learning neural networks

Artificial neural networks, a more sophisticated machine learning technology, form the foundation for the majority of deep learning models. Deep learning has been dubbed "deep neural networking" or "deep neural learning" as a euphemism for it.

Some applications benefit more from neural networks using recurrent neural networks (RNNs), while others benefit more from convolutional NNNs (CNNs). But they all function in the same way: by providing the model with data, it determines for itself whether or not it has made the proper interpretation or judgement about a single data point.

Because neural networks are trained by a process of trial and error, a large amount of data is required. In tandem with the increased use of big data analytics, neural networks have risen to prominence. Because the model's

first few iterations include educated guesses, the training data must be labelled so that the model can see whether its prediction about a picture or a portion of speech was true. Even though many firms are using big data, unstructured data has limited value. However, deep learning models cannot be trained on unstructured data, therefore they cannot analyse material that hasn't been trained to an acceptable level of accuracy.

3.9. Deep learning examples

It is possible to apply deep learning models to a broad variety of tasks because they process information in a way comparable to that of the human brain. Today, deep learning is used in the majority of image recognition, NLP, and voice recognition programmes. Two examples of how these technologies are being put to use are self-driving cars and language translation services.

Deep learning is being applied in a wide range of applications, including NLP, language translation, medical diagnosis, stock market trading signals, network security, and picture identification.

These are some of the domains where deep learning is presently being employed:

Customer experience (CX): As of now, chatbots are already using deep learning algorithms. The use of deep learning to enhance CX and boost customer happiness is projected to expand as the technology develops.

Text generation: It is now possible for computers to learn the grammar and style of a piece of writing and then use this model to automatically generate a new piece of text that matches these characteristics.

Aerospace and military: In order to identify regions of interest and safe or risky zones for soldiers, satellites are using deep learning to recognise things.

Industrial automation: Services that automatically recognise when a person or item is too near to a machine are enhancing worker safety in places such as factories and warehouses.

Adding color: Deep learning models may be used to colourize black-and-white images and movies. When this was done manually in the past, it took a long time.

Medical research: Deep learning has been used by cancer researchers as a means of automatically detecting cancer cells.

Computer vision: Deep learning has considerably improved computer vision, allowing computers to recognise and classify objects and images with extraordinary precision, as well as to restore and segment images that have been damaged.

3.10. Limitations and challenges

Because deep learning models learn by observing, this is their largest restriction. In other words, they can only draw conclusions about the data they used to train. Data from a single source that does not cover the whole functional area cannot be used to build a generalizable model.

There is also a concern with biases in deep learning models. Data that has biases will be reflected in the predictions of a model trained on that data. This has been a problem for deep learning programmers since models learn to differentiate based on the slightest of variations in input. When it comes to designing a system, programmers are frequently kept in the dark regarding which elements

are most important. Using parameters such as race or gender in a face recognition model, for example, might lead to assumptions about people's attributes being made without the programmer's consent.

Deep learning models may be tested to their limits by the pace at which they learn new information. Increasing the rate of convergence will result in a less-than-optimal solution, if the model converges too rapidly. As a result, it will be more difficult to find a solution if the rate is too low.

For deep learning models, there are hardware constraints. GPUs and other comparable processing devices with several cores and high performance are essential if you want to maximise productivity while minimising workload. However, these devices are pricey and use a lot of energy. Random Access Memory (RAM) and a Hard Disc Drive (HDD) or RAM-based Solid-State Drive (SSD) are also required.

Another set of limits and obstacles include the following:

- Deep learning demands an abundance of data to be effective. It's also likely that the more

powerful and accurate models will demand more parameters and data.

- Deep learning models are rigid and incapable of multitasking after they have been trained. Only one issue can be solved efficiently and accurately by them. Even if a comparable issue could be solved, the system would need to be retrained.
- The scientific method, long-term planning, or algorithmic data manipulation cannot be handled by deep learning, even with enormous data sets.

3.11. Deep learning vs. machine learning

As a subset of machine learning, deep learning handles issues differently than other methods. Machine learning need the expertise of a specialist in the field. Deep learning, on the other hand, learns about characteristics piecemeal, eliminating the requirement for specialised knowledge. In contrast to machine-learning algorithms, which only need a few seconds to a few hours of training, deep learning algorithms take significantly longer to train. During testing, on the other hand, the situation is the exact

opposite. As the amount of the data rises, the time it takes for machine learning algorithms to execute a test increases as well.

Unlike the high-end, high-performance GPUs needed for deep learning, machine learning does not need these same high-end hardware requirements.

Data scientists prefer classical machine learning since it's easier for them to understand the results, rather than using deep learning. For tiny datasets, it is advised to use machine learning algorithms.

Using deep learning when there is a lot of data, a lack of domain knowledge for feature introspection, or when dealing with complex challenges like speech recognition and NLP is the best solution.

CHAPTER 4

Deep Learning Libraries

TensorFlow and Keras, two of the most popular deep learning libraries, will be discussed in this section, along with some basic explanations on how to use them.

4.1. Theano

A Yoshua Bengio-supervised open-source project, Theano was principally created at the Université de Montréal. As with NumPy, it is a Python numerical calculation library. With multidimensional arrays, it is possible to compute complicated mathematical expressions in a short period of time. For neural networks, this makes it a great candidate.

Create machine learning models that can be utilised across many datasets using Theano, a mathematics framework.

On top of Theano, a wide range of applications have been developed. Principally, it includes:

- Blocks
- Keras
- Lasagne
- PyLearn2

4.2. TensorFlow

Developed by Google for large-scale machine learning applications, TensorFlow is referred to as TensorFlow. Actually, TensorFlow is Google's DistBelief software framework, which allows huge models to be trained on computer clusters with tens of hundreds of thousands of machines.

They came up with the idea for TensorFlow as part of the Google Group (now Alphabet), which is largely focused on the usage of deep learning in various industries.

A detailed explanation of how data flow diagrams are used in numerical calculation can be found below. In order to handle calculations on CPUs or GPU systems across a

single desktop, server, or mobile device using a single API, it was built.

As a result of TensorFlow, computationally expensive activities may be moved from CPUs to heterogeneous GPU systems with just the tiniest of code modifications.

An Android-enabled mobile device, for example, may finalize the execution of a model learned on one system.

DeepDream, an automatic image-captioning programme, and RankBrain, a tool that helps Google analyse search results and give more relevant search results to consumers, are all built on TensorFlow.

4.3. Data Flow Graphs

TensorFlow uses data flow graphs to illustrate the mathematical calculations it does. It makes use of nodes and edges in directed graphs. Data may be entered into or produced from the nodes and permanent variables can be read/written to them. The relationships between nodes are handled by the edges. Nodes are connected by tensors, which are multidimensional data arrays that may change in size as they move through the network. TensorFlow is a

word used to describe the flow of these tensor units across the whole network. The nodes in a graph run asynchronously and in parallel as soon as they get all of their corresponding tensors from the incoming edges.

A data flow graph depicts the overall design and flow of calculations in a session, which are subsequently conducted on the appropriate machines.

TensorFlow, with the Python, C, and C+ APIs offered, relies on C++ for optimized computations.

Machine learning requires huge parallelism and scalability, and TensorFlow is the best option for these requirements.

- **Deep flexibility:** TensorFlow is open source, so anybody may develop their own extensions. TensorFlow takes care of the rest, all you have to do is generate a graph representing the calculation.
- **True portability:** Programmers may use the extensibility provided by TensorFlow for the training of GPU-accelerated models that can then be used in the final product or distributed on docker as a cloud service.

- **Automatic differentiation:** The automated differentiation function of TensorFlow handles the calculation of derivatives for gradient-based machine learning algorithms. In order to comprehend the expanded value graph, it is helpful to compute the derivatives of values.
- **Language options:** TensorFlow has Python and C++ APIs for building and executing computation graphs.
- **Performance maximization:** To ensure optimal speed, TensorFlow makes use of a variety of threading, queuing, and asynchronous processing techniques. TensorFlow may be used on a variety of devices.

4.4. Keras

Using Keras on top of Theano or TensorFlow, you may build a wide variety of neural networks. Keras is one of the few libraries that can operate on both the GPU and the CPU.

It's common knowledge that a model is nothing more than a collection of interconnected, freely programmable

components. Neural layers, cost functions and optimizers, initialization techniques, activation functions, and regularisation schemes are all separate modules that may be combined to form new models.

4.4.1. Keras Principles

A model is one of Keras' primary data structures. Customizable models are made up of multiple layers, cost functions and activation and regularisation strategies. Convolution, dropout, pooling, locally connected, recurrent, noise, and normalising are a few of the pre-built layers available in Keras for use with neural networks. An input item for the next layer is a layer of a network.

Additionally, code snippets in Keras will be supplied in the following chapters, in addition to their appropriate neural network implementations.

4.5. PyTorch

PyTorch is a machine learning library for Python developed by Facebook. If you're more interested in C++ than Python, PyTorch includes a C++ interface that you may use. As a prominent competitor in the battle to be the

finest machine learning and deep learning framework, PyTorch is challenged by TensorFlow.

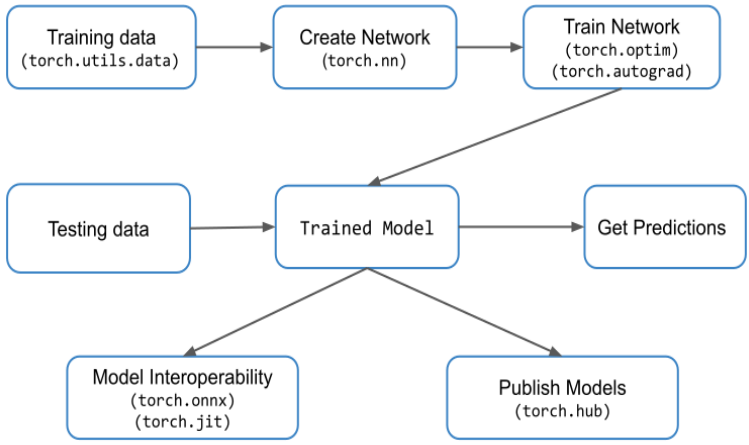


Figure 4.1: Basic PyTorch Workflow[*]

PyTorch differs from TensorFlow in many key ways, including the following:

- GPU-accelerated Tensor computation
- Python is a great environment for learning, using, and integrating with.
- Support for tape-based auto-diff neural networks.

[*]Source: https://towardsdatascience.com/best-python-libraries-for-machine-learning-and-deep-learning-b0bd40c7e8c

4.6. Scikit-learn

Another popular Python machine learning package is Scikit-learn. NumPy and Pandas are only two of the many ML programming libraries that can be easily integrated with this software. Many algorithms are supported by Scikit-learn, such as R and Python:

- Classification
- Regression
- Clustering
- Dimensionality Reduction
- Model Selection
- Preprocessing

For the sake of simplicity and flexibility, Scikit-learn focuses on data modelling rather than additional activities like loading, manipulating, and visualising the information it contains. It may be utilised as a whole machine learning system, from research through implementation.

4.7. Pandas

As the name implies, Pandas is a Python package that specialises in working with large datasets. Before the

dataset is prepared for training, it comes into play. For machine-learning programmers, Pandas make it simple to deal with time series and multidimensional data. When it comes to data management, some of Pandas' best features include:

- Reorganization and reorientation of datasets
- Data merging and combining
- In the case of incomplete data, as well as data alignment.
- Options for several types of indexing: hierarchical axis indexing and fancy indexing.
- Filtering options for your data

In order to provide programmers with a two-dimensional representation of data, Pandas provides them with Data Frame objects, a technical word.

4.8. NLTK

A Python package for natural language processing, NLTK stands for Natural Language Toolkit and is known as NLTK. One of the most widely used libraries for working with human language data is this one. NLTK provides a broad range of lexical resources to programmers, including

FrameNet, WordNet, Word2Vec, and many more. NLTK's most notable features are the following:

- Searching keywords in documents
- Tokenization and classification of texts
- Recognition on voice and handwriting
- Lemmatizing and Stemming of words

Students, engineers, academics, linguists, and businesses that deal with language believe NLTK and its collection of packages to be a sound investment.

4.9. Spark MLlib

Spark MLlib is a machine learning package developed by Apache that makes it simple to scale your calculations. Simple to use, fast to set up, and seamless connection with other tools are all hallmarks of this software. For creating machine learning algorithms and applications, Spark MLlib was an excellent choice.

The tools that Spark MLlib brings to the table are:

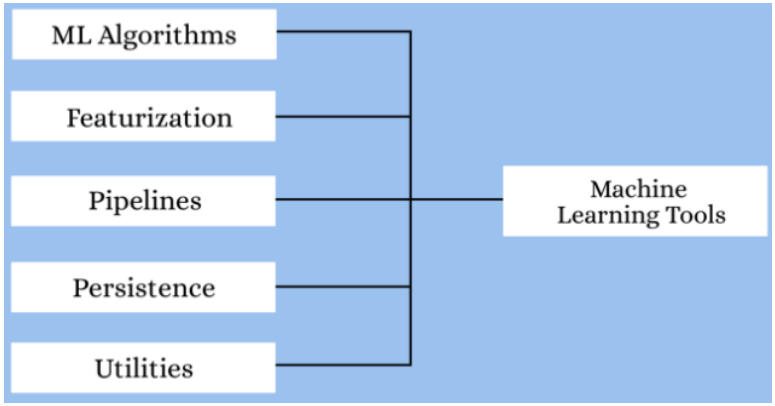


Figure 4.2: Spark Mllib Machine Learning Tools[*]

4.10. Numpy

Mathematical functions and large multi-dimensional data sets are the focus of the NumPy package for Python. Fast calculation and execution of complex functions on arrays may be achieved with NumPy. NumPy's advantages include the following:

- Operational aids for math and logic
- The ability to change the shape of an object.
- Capabilities for sorting and selecting
- Transformations of the Fourier series discretely
- Algebra and statistics for beginners

[*]Source: https://towardsdatascience.com/best-python-libraries-for-machine-learning-and-deep-learning-b0bd40c7e8c

- Simulations based on chance
- Arrays of n dimensions are supported.

The scientific community loves NumPy because of its object-oriented approach and tools for integrating C, C++, and Fortran programming.

CHAPTER 5

Deep Learning Architectures

Deep architectures come in a plethora of flavors, yet the most of them descend from a single ancestor. In many cases, it is impossible to evaluate the performance of several architectures on the same data set. New architectures, variations, and algorithms may be introduced in the area of deep learning every few weeks.

5.1. Deep neural networks

DNNs are artificial neural networks having numerous hidden layers of units between the input and output. DNNs, like shallow ANNs, are capable of modelling complicated non-linear connections. Compositional models, such as those generated by DNN architectures for object identification and parsing, represent objects as

layered compositions of picture primitives. Additional layers permit the construction of lower-layer features, allowing the possibility to represent complicated data with fewer units than a shallow network that performs similarly.

All the time, the neural network must learn how to do tasks more effectively, or perhaps find new ways to get better results. It learns how to respond to a new circumstance when it receives fresh information in the system.

Learning deepens as the difficulty of the activities increases. If a system employs multiple layers of nodes to extract high-level functions from incoming information, this is known as a deep neural network. The data is transformed into a more abstract and creative component by doing this.

Let's assume an image of a typical guy to better comprehend the results of deep learning. Even if you've never seen this image before, you'll instantly recognise the subject as a human being and set them apart from other critters. An illustration of the deep neural network in

action. In order to identify the item, creative and analytical elements of information are examined and organised. The ML system needs to adapt and derive these components since they are not introduced into the system directly.

5.2. Deep belief networks

There are numerous layers of hidden units that make up a deep belief network (DBN). Each layer is made up of a collection of smaller learning modules.

Furthermore, an unsupervised machine learning model is used to generate outcomes in a deep belief network (DBN), a more advanced sort of generative neural network. This kind of network exemplifies some of the current work in unsupervised model construction utilising relatively unlabeled data.

Pre-training a DNN may be done using the learnt weights as the beginning weights for a DBN. These weights may then be fine-tuned using algorithms such as backpropagation or discrimination. When there is a limited amount of training data, this is especially useful since incorrectly initialised weights may have a major influence on the model's performance. There is a good

chance that these weights will be in an area of the weight space that is close to the ideal ones (as compared to just random initialization). In addition, this provides for better modelling capabilities and quicker fine-tuning.

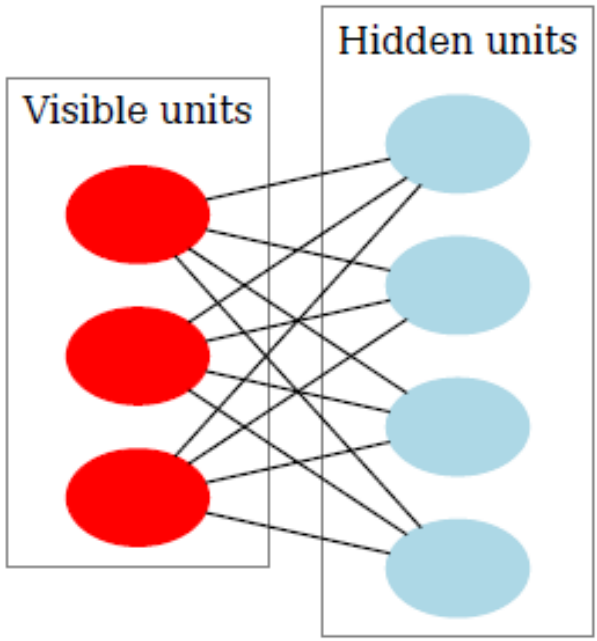


Figure 5.1: A restricted Boltzmann machine (RBM) with fully connected visible and hidden units. Note there are no hidden-hidden or visible connections[*]

[*]Source:https://www.kdnuggets.com/2016/06/recursive-neural-networks-tensorflow.html

A DBN may be trained in an unsupervised, layer-by-layer fashion using limited Boltzmann machines, which are generally used in the layers (RBM). RBMs may be used to train a DBN, as detailed below. With a single hidden layer, RBMs are generative energy-based models that are undirected. There are no visible-visible or hidden-hidden links between the visible units of the input layer and the hidden units of the hidden layer as shown in figure 5.1.

5.2.1. Evolution of DBN

Perceptron's were employed in the first generation of neural networks to identify a certain item or anything else based on its "weight" or pre-fed qualities. This technique is limited in its applicability to high-tech applications because of the Perceptrons. Second Generation Neural Networks introduced Back propagation, in which the received output is compared to the intended outcome and error values are decreased to zero to tackle these problems. By referring to previously supplied test cases, Support Vector Machines were able to generate and comprehend additional test cases. The development of directed cyclic graphs, also known as belief networks, aided in the solution of inference and learning-related issues. Deep

Belief Networks were used after that to generate values for leaf nodes that were free of bias.

5.2.2. Restricted Boltzmann Machines

Persistent Confidence RBMs are an example of an unsupervised network in a larger network. In this case, the invisible layer of each subnetwork is the visible layer of the next. It is impossible to link the hidden or unseen layers to each other. Joint configuration networks' probabilities across both visible and hidden layers are determined by their energy relative to that of all other joint networks.

5.2.3. Training a Deep Belief Network

The initial step is to develop a set of attributes that can directly access the pixels' input signals. It's now time to learn the features of the previously discovered features in another hidden layer by using this layer's values as pixels. The lower limit on the log likelihood of the training data set will improve as more characteristics or features are added to the belief network.

5.3. Convolutional neural networks

When given a picture, a Convolutional Neural Network (ConvNet/CNN) can determine the relevance of various characteristics or objects in the image, then use that information to distinguish between them. When compared to other classification methods, the amount of pre-processing needed by a ConvNet is much reduced. Filters are created by hand in rudimentary approaches, but with enough training, ConvNets may learn to create these filters and attributes themselves.

Architecture of ConvNet is similar to that of human brain connection patterns and was inspired by visual cortex arrangement. The Receptive Field is the area of the visual field in which individual neurons react to stimuli. The whole visual field is covered by a group of such fields that overlap one another.

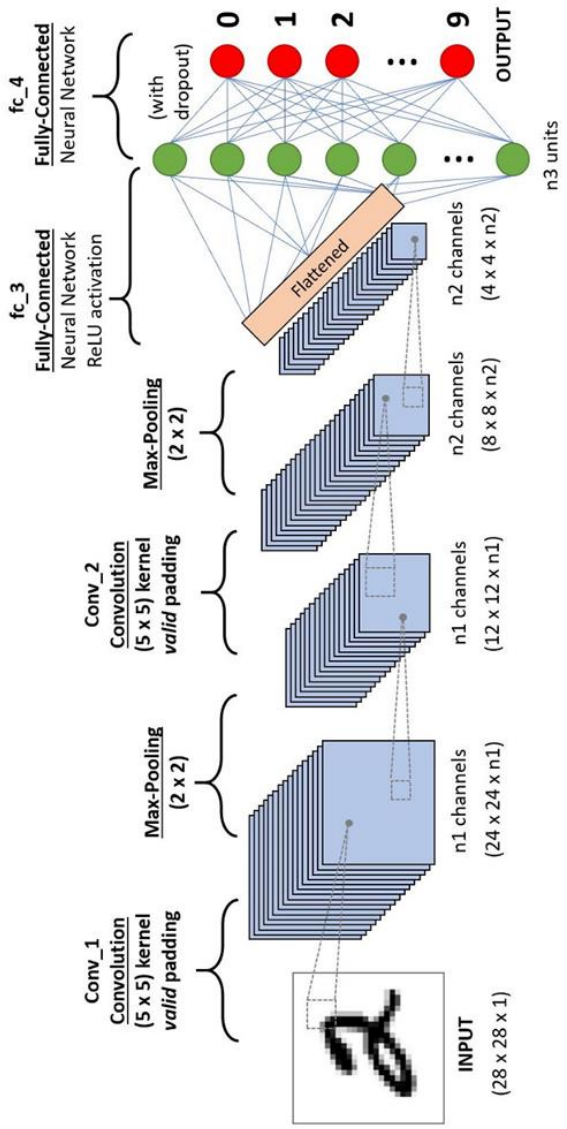


Figure 5.2: CNN[*]*

[*]Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

5.3.1. Working of CNN

With images, voice, and audio signals, convolutional neural networks are superior than conventional neural networks. In general, they contain three primary layers, each of which is distinct:

- Convolutional layer
- Pooling layer
- Fully-connected (FC) layer

The first layer of a convolutional network is called the convolutional layer. Despite the fact that further convolutional or pooling layers might follow, the fully-connected layer is the last layer. The complexity of the CNN rises with each layer, allowing it to recognize more of the picture. Colors and borders are the primary emphasis of the first design stages. With each successive layer of the CNN, more and more details of the picture are picked up, until the final layer recognizes the target item.

5.3.2. Convolutional Layer

CNNs are built around the convolutional layer, which is where most of their processing takes place. Many

components are needed, including input data, filtering, and feature mapping. Assume we'll be dealing with a picture that's composed of a 3D matrix of pixels. There will be three dimensions: height, breadth and depth, each of which corresponds to an image's three primary colours (red, green, and blue). To check for the presence of a particular feature, we have an image feature detector (also known as a kernel or a filter), which moves through the picture's receiving fields. Convolution is the term used to describe this process.

A two-dimensional (2-D) array of weights serves as the feature detector and represents a portion of the picture. The receptive field's size is determined by the filter's 3x3 matrix, which may vary in size. Dot product between input and filter pixels is determined after the filter is applied to the picture. The output array is then given this dot product. When the kernel has covered the whole picture, the filter makes another step forward and repeats the process. the sequence of dot products from the input and filter are known as a feature map, activation map, or a convolved feature, and the final output.

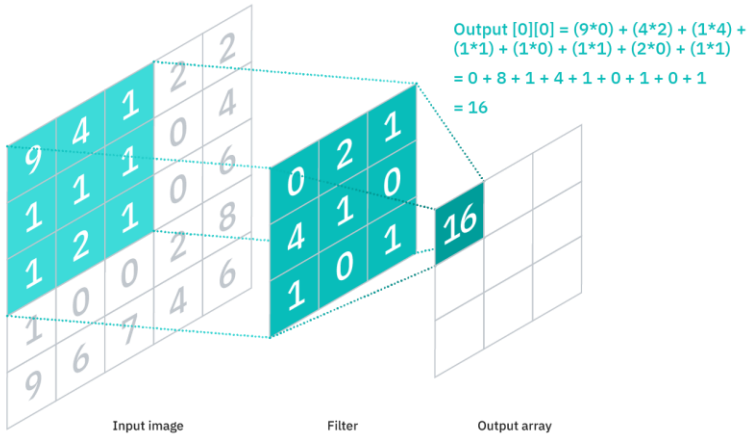


Figure 5.3: Feature map[*]

The output values in the feature map do not have to correspond to the pixel values in the input picture, as seen in the figure above. If it's in the receptive field, it merely needs to connect to the filter. "Partially connected" refers to the fact that the output array does not need to be physically linked to each input value. Local connection, on the other hand, is another way of describing this trait.

The weights of the feature detector stay constant as it traverses over the picture, which is also known as parameter sharing. Backpropagation and gradient descent

[*]Source: https://www.ibm.com/cloud/learn/convolutional-neural-networks

are used to change certain parameters, such as weight values, during training. The output volume size is affected by three hyperparameters, which must be specified before neural network training starts. These include:

The depth of the output depends on the number of filters. With three independent filters, you might have three different feature maps, each with its own depth.

The amount of pixels the kernel travels over a given input matrix in one stride. Although stride values of two or more are uncommon, a longer stride results in a lesser output.

When the filters don't fit the picture, zero-padding is the most common solution. The output will be greater or equal in size since all entries outside of the input matrix will be set to zero. There are three types of padding:

Valid padding: No padding is another term for this situation. If the dimensions do not match, the final convolution is discarded.

Same padding: Adding this padding guarantees that the output layer is the same size as the input layer.

Full padding: Zeros are added to the input border in order to enhance the size of the output using this sort of padding.

ReLU transforms the feature map after each convolution operation in a CNN, which introduces nonlinearity into its model.

The first convolution layer may be followed by a second convolution layer, as we discussed before. Because the subsequent layers are able to observe the pixels in earlier layers' receiving fields, the CNN's structure may become hierarchical as a result. Let's pretend we're attempting to figure out whether a picture has a bicycle as an example. Parts of the bicycle may be seen as an entity in and of itself. Frame, handlebar, wheel, and pedal components are all included. In the CNN, each section of the bicycle is a lower-level pattern, and the combination of its parts is a higher-level pattern, producing a feature hierarchy.

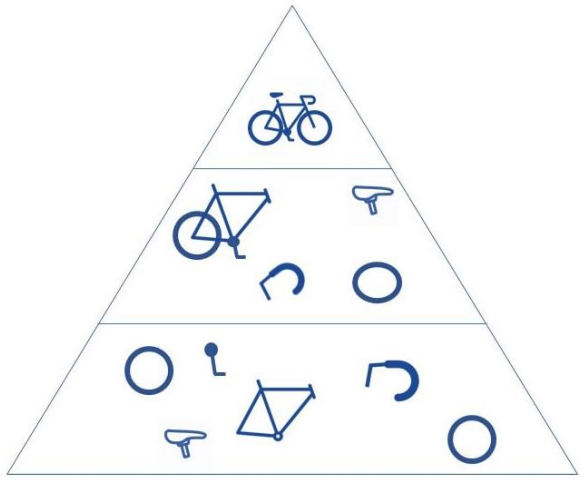


Figure 5.4: Determining the image of bicycle[*]

Image data are transformed into numerical values by the convolutional layer of the neural network in order to extract significant patterns.

5.3.3. Pooling Layer

This process, also known as down sampling, reduces the amount of factors that need to be taken into account. The pooling operation sweeps a filter across the whole input, similar to the convolutional layer, however this filter lacks weights. As a result, the kernel uses an aggregation

[*]Source: https://www.ibm.com/cloud/learn/convolutional-neural-networks

function to fill the output array with the receptive field's values. There are two main types of pooling:

Max pooling: Whenever the filter passes over the input, it looks for the pixel with the highest value and sends that information to the output array. As a side note, this method is more often used than typical pooling.

Average pooling: The average value inside the receptive field is sent to the output array as the filter passes across the input.

While the pooling layer loses a lot of information, it provides some advantages for the CNN. Using these tools reduces the danger of overfitting and improves efficiency.

5.3.4. Fully-Connected Layer

The full-connected layer's name accurately reflects its function. In partly linked layers, the pixel values from the input picture are not connected directly to the output layer, as previously explained. As a result, each node in the outgoing layer links to a node in the preceding layer directly.

Based on the characteristics retrieved from the preceding layers and their varied filters, this layer conducts the duty of categorization. For FC layers, a softmax activation function is often used to categorise inputs, generating a probability range of 0 to 1 for each input.

5.3.5. Convolutional neural networks and computer vision

Image identification and computer vision are powered by convolutional neural networks. If you've ever looked at a photograph or video, you've likely noticed that it's impossible for a computer to decipher meaning from it. That's where computer vision comes in. Images can be recognised, but the ability to provide suggestions separates it from other image recognition jobs. Today, computer vision is used in a variety of ways, including in the fields of medicine and law enforcement:

Marketing: To make tagging friends in picture albums on social media platforms simpler, these platforms present ideas about who could be in a photo that has been put on a profile.

Healthcare: It is now possible to detect malignant tumours in healthy tissues using computer vision, which has been integrated into radiological equipment.

Retail: In certain e-commerce platforms, marketers may propose things that go well with a customer's current wardrobe using visual search.

Automotive: It's still early days for autonomous vehicles; but the underlying technology has begun to find its way into autos, enhancing driver and passenger safety with features like identification of lanes.

5.3.6. Limitations

In spite of their enormous processing capacity and resource requirements, CNNs are able to provide detailed results. Recognizing patterns and nuances that are so minute and undetectable to the naked eye is all it comes down to. However, when it comes to deciphering an image's substance, it falls short.

Take a look at the following illustration figure 5.5. Using the picture below, a CNN can identify a person in their 30s and a kid, perhaps under the age of ten, when presented

with it. However, when faced with the identical sight, we are compelled to conceive of a plethora of possible outcomes. They may be out on a day trip with Dad, having a picnic, or going camping with him and his kid. There is a possibility that the boy has just scored a goal in the schoolyard and his father is lifting him up in his joy.

Figure 5.5: Example for CNN detection[*]

It's clear when it comes to actual implementations that these limits are in place. When it came to social media posts, CNN's were often utilised. Even though they were

[*]Source: https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/

trained on a massive library of photos and videos, the system is unable to entirely filter and delete improper material. Apparently, a 30,000-year-old statue's nudity was noticed on Facebook.

ImageNet-trained neural networks have been demonstrated to be ineffective in detecting objects when they are seen from new perspectives and in varied lighting conditions.

Is this to say that CNNs are no longer useful? Convolutional neural networks, with all its flaws, have ushered in a new era of artificial intelligence despite their shortcomings. Facial recognition software, picture search and manipulation, and augmented reality are just a few of the computer vision applications that now employ CNNs. Our progress in convolutional neural networks shows that although our accomplishments are impressive and valuable, we are still a long way from recreating the most important aspects of human intellect.

5.3.7. Real-world applications of Convolutional neural network (CNN)

In computer vision, convolutional neural networks (CNNs) perform very well. Here are some examples of how CNN may be used in the real world:

Computer vision systems have been used to identify faces in photographs. Using an image as input, the network creates a collection of values that represent various aspects of a person's face or features on a person's face. According to different study studies, CNN has showed a 97 percent success rate in detecting faces. Additionally, CNN may be used to lessen the degree of distortion in a person's face. It is possible for CNNs to recognise characteristics such as the eyes, nose, and mouth with excellent precision, eliminating distortions from factors like angles or shadows on the faces.

When it comes to distinguishing between distinct facial expressions, CNNs have been employed to aid in the process. CNNs may be tweaked to work effectively in a wide range of illumination and facial-angle situations.

When it comes to object detection, CNN has been used to identify things based on their appearance in an image. From ordinary things like food and celebrities to more exotic ones like dollar notes and weaponry, CNN models have been developed that can recognize a broad variety of objects. Semantic or instance segmentation are two strategies used in object detection. Drones and self-driving vehicles have been using CNNs to recognize and locate items in photos and to produce multiple perspectives of those objects.

Self-driving or autonomous cars: For example, CNN has been employed in the context of driverless cars to help them identify impediments or read street signs. Reinforcement learning, a form of machine learning that focuses on positive and negative feedback from the environment, has been used in combination with CNNs to enhance how CNN models react to particular scenarios.

Auto translation: Automatic translation across language pairings such as English and French makes use of CNN in the context of deep learning. Translation across languages like Chinese and English may now be done without the

need for human translators or translators who are fluent in both languages.

Next word prediction in sentence: If you know what you're talking about, CNNs can anticipate what the next word will be. "I am from India," followed by "I speak Hindi," is an example of a common sequence of words in a using CNN model to analyse the phrase.

Handwritten character recognition: CNNs may be used to identify characters written by hand. Character images are broken down into smaller portions by CNNs, which then detect points that potentially link or overlap with other points in the bigger character. Many diverse languages, including Chinese, Arabic, and Russian, may be recognized by CNN models despite the fact that they are written differently.

In medical imaging, CNNs have been used to detect cancers or other anomalies in X-ray pictures. A picture of a human body component may be analysed by CNN networks to identify possible tumour locations based on earlier photos of the same body part processed by CNN networks. X-ray pictures may be analysed using CNN

models to look for anomalies. CNNs have been used to locate tumours and other anomalies in X-ray images, such as broken bones.

Cancer detection: Medical imaging like as mammograms and CT scans have been utilised to diagnose cancer with the help of CNNs. A CNN model compares a patient's picture to a database of images with comparable features, recognising signals of malignancy or cell damage caused by both heredity and environmental variables such as smoking behaviours. Pathologists were only able to detect malignant cells with an accuracy of 85% to 90%, whereas CNNs have been able to identify cancerous cells with an accuracy of 95%.

Visual question answering: When asked a natural language query about an image, CNN can provide an accurate response based on the picture. Any query concerning a picture may be answered via a natural language response from a CNN.

Image captioning: New photographs that are supplied into CNN networks are also being utilised to generate brief descriptions of what's included in the images, as well as

combining many photos from social networking sites, such as Instagram. For each set of input photos, CNN models may produce one or more sentences describing what's in those images.

Biometric authentication: The physical qualities of a person's face may be linked to their identification using CNN for biometric authentication. It is possible to train CNN models on photos or videos of individuals in order to get a facial feature vector, such as the distance between eyes, nose shape, lip curvature, or any other characteristic of a person's facial characteristics. Facial photos and videos of humans have also been used to train CNN models on various emotional expressions, such as happiness or sadness. If a person is blinking in a picture, CNNs can assess the general shape of facial images including numerous frames and identify critical areas on each frame that are helpful for training CNN models to grasp what's included in those images.

Document classification: Using CNNs, papers have been sorted into several subcategories. CNN models may classify a document based on its content, such as an article on sports or politics, for example. While searching for

information, CNN may utilise both text and pictures to find important terms or phrases that pertain to the subject matter of a given piece of content. Additionally, CNN models have been used to summarise papers by analysing their content, finding and explaining the most important elements in each document.

3D medical image segmentation: Medical imaging scans, including as MRI pictures, have been segmented using CNN. Based on previous comparable pictures processed by CNN networks, CNN models can analyse a slice of a three-dimensional scan and identify where distinct tissue types are located.

5.3.8. Implementation of CNN: Tensor flow; Keras

Tensor Flow

Open-source software package TensorFlow was created by Google using data flow graphs to do numerical computations.

Data arrays (tensors) link nodes in the graph, which indicate mathematical processes. Nodes in the graph are represented by the graph's nodes. Using several CPUs or

GPUs on a server, desktop, or mobile device does not need rewriting any code.

Figure 5.6: Tensor flow[*]

Tensor Board provides data visualisation features in addition to TensorFlow.

CNN with TensorFlow

You may now build a convolutional neural network using TensorFlow. It will be used for CNN image classification using MNIST data.

Using CNN, you'll go through the following stages to classify images:

[*]Source: https://www.analyticsvidhya.com/blog/2016/10/an-introduction-to-implementing-neural-networks-using-tensorflow/

Step 1: Upload Dataset

Step 2: Input layer

Step 3: Convolutional layer

Step 4: Pooling layer

Step 5: Second Convolutional Layer and Pooling Layer

Step 6: Dense layer

Step 7: Logit Layer

Step 1: Upload Dataset

Use scikit-learn with the MNIST dataset by visiting this URL. Thank you for downloading this file. Fetch mldata("MNIST original") when instructed to do so.

- Create a train/test set

Using train test split, you must divide the dataset

- Scale the features

As a last option, you may use MinMaxScaler to scale the feature, as seen in the image classification example below, which uses TensorFlow CNNs.

Define the CNN: An advantage of CNNs over more traditional neural nets is their ability to extract more detail from raw pixel input. The first step in creating a CNN is defining what a CNN is:

A convolutional layer: n filters may be applied to the feature map. After the convolution stage, you must use a Relu activation function to make the network more non-linear.

Pooling layer: This down sampling occurs after the feature max has been convolutional. If you want to prevent overfitting, you may reduce the number of dimensions in the feature map. The classic approach of max pooling is used to partition most feature maps into 2x2 pieces, but only the highest values are kept.

Fully connected layers: All neurons in the previous layers are connected to the neurons in the next layer. Each label will be classified by the CNN based on attributes gleaned from convolutional layers and further reduced by the pooling layer.

CNN architecture

Convolutional Layer: Uses 14 5x5 filters with ReLU activation (extracting 5x5-pixel subregions)

Pooling Layer: Its maximum pooling capacity is achieved with a 2x2 filter and a stride of two (which specifies that pooled regions do not overlap).

Convolutional Layer: 36 5x5 filters are activated by the ReLU function.

Pooling Layer #2: With a 2x2 filter and a stride length of 2, it performs at its maximum capacity as previously.

There are 1,764 neurons with a dropout regularisation rate of 0.4. (probability of 0.4 that any given element will be dropped during training)

Dense Layer (Logits Layer): Each of the nine-digit classes (0–9) has a single neuron, for a total of 10.

To build a CNN, there are three key components:

- conv2d().The number of filters, filter kernel size, padding, and activation function are all used as

inputs to form a two-dimensional convolutional layer.

- max_pooling2d().A two-dimensional data pooling layer is created using the max-pooling approach.
- dense().For a thick layer, hidden layers and units are used

You must first develop a function before you can proceed with building the CNN. Let's take a deeper look at how each component is put together before putting everything together in a function.

Step 2: Input layer

The form of your tensor must be determined by the data. That may be done using the tf.reshape module. This module requires both the tensor to be reshaped and the tensor's shape to be supplied. The first argument to a function is a description of the data's properties.

Step 3: Convolutional layer

The 14 filters in the first convolution layer are housed in paddocks of 5x5. Paddled in the same manner as its input,

it must be padded in the same manner as its output. Tensorflow would add zeros to each row and column to ensure the same number of rows and columns.

Step 4: Pooling layer

After convolution, comes the calculation for pooling. The pooling calculation would lower the number of dimensions in the data. This module may be used with a 2x2 stride and a max pooling2d size of 2. This layer receives its input from the previous layer [batch size, 14, 14, 14].

Step 5: Second Convolutional Layer and Pooling Layer

The output size of the filters in the second convolution layer is [batch size], with a total of 32. As before, the output and pooling layers have the same size ([batch size, 14, 14, 18]).

Step 6: Dense layer

The next stage is to construct a layer that is completely interconnected, since this is a must. The feature map must first be flattened before it can be connected to the thick

layer. A 7*7*36-sized module may be reshaped using the module reshape.

Neurons in the thick layer will have 1764 interconnections between them. Activation of Relu is included. Using the dropout regularisation term, 30 percent of the weights would be adjusted to 0 in this situation." A student may only depart for training purposes.

Step 7: Logit Layer

TensorFlow image classification may employ the model's prediction as the last layer in its architecture. In this batch, there are ten photographs, thus the output shape is equivalent to ten times the batch size.

Pooling and Flattening

In the course of this presentation, we've focused on the convolution layer. Extra layers separate from convolution include pooling and flattening as shown in figure 5.7. Let's begin with the most basic concept. A pool is a collection of resources that are brought together. As a consequence, it's mostly used for data compression.

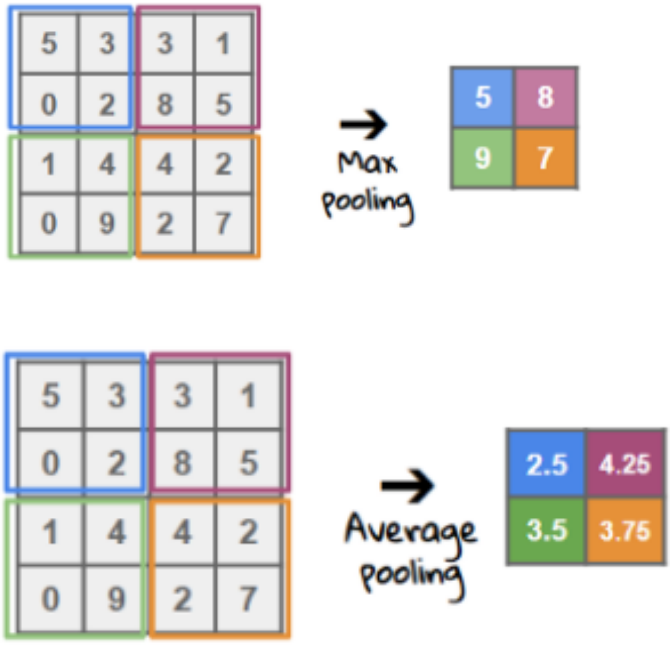


*Figure 5.7: Pooling and Flattening**

Sliding windows are more valuable on the left side since we receive all of them. 'Max pooling' is the word used to describe this. The average values are shown on the right. "Average pooling" is being used here. We have the same control over the stride as we have over the convolution layer.

*Source: https://towardsdatascience.com/the-most-intuitive-and-easiest-guide-for-convolutional-neural-network-3607be47480

Isn't this just a waste of time and money? Is there a good reason for reducing the size? It may seem that data is being lost, but in fact more "useful" data is being collected. Reducing overfitting as well as speeding up computations may be achieved by eliminating some noise and maintaining just the relevant data.

Image features are reduced to a feature vector, which is supplied into a multi-layer perceptron for the production of probabilities. The process of flattening is shown in the following graphic:

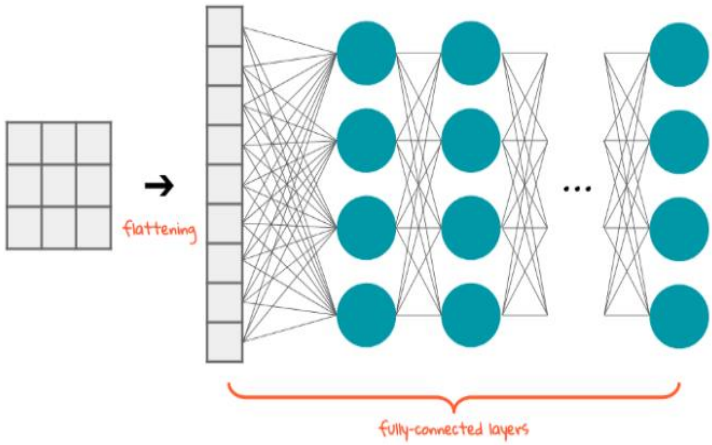


*Figure 5.8: Flattening process**

*Source: https://towardsdatascience.com/the-most-intuitive-and-easiest-guide-for-convolutional-neural-network-3607be47480

Paging in convolutional neural networks refers to the number of pixels added to an image as it is processed by the kernel of the network. Padding and new pixels would both have the same value, for example. If the padding is set to one, it is possible to apply a one-pixel border to an image with no padding at all.

Padding

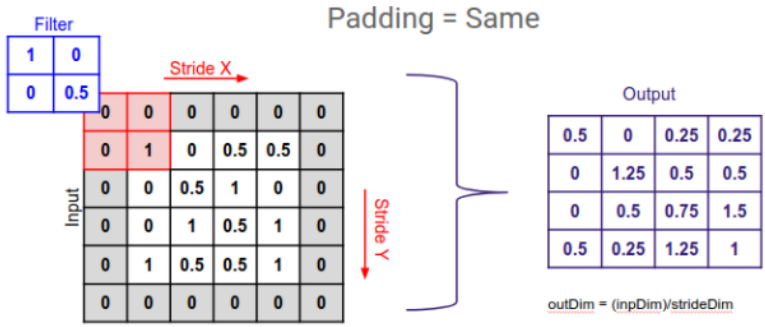


Figure 5.9 Padding[*]

Padding is a technique used by convolutional neural networks to expand the processing region of an image as shown in figure 5.9. A neural networks filter is used to transform each pixel into a smaller or larger format when the image is scanned. Kernel performance is improved by

[*]Source: https://analyticsindiamag.com/guide-to-different-padding-methods-for-cnn-models/

adding padding to an image's frame. When CNN images are padded, they are more accurate.

ow does Padding work?

With padding, the processing area of an image may be expanded in convolutional neural networks. Using neural networks, every pixel in the image is reduced or increased in size as it is scanned. Padding an image's frame allows the kernel to process it more quickly and efficiently. Paddled images are more accurate when processed by a CNN.

Types of Padding

There are three types of padding:

- Same padding
- Causal padding
- Valid padding

Same Padding: It is possible to utilize the filter we are applying to cover the matrix as well as to do the inference with the padding layers that add zero values to the outer frame of images or data.

Valid Padding: We use every point/pixel value when learning this model to ensure adequate padding; it doesn't work with input size, it works with pixel value validation.

TensorFlow will reject VALID cells on the right and bottom if your filter and stride don't cover the whole input image (i.e., no padding mode).

Causal Padding: In combination with this special padding technique, one-dimensional convolutional layers are used. A common use is time series analysis. A time series employs sequential data, which adds zeros at the beginning, to help estimate the values of the early time steps.

Stride

As part of convolutional neural networks, which are used to minimise the size of both still and moving images, Stride is a crucial component. Stride is a metric used by the neural network filter to determine how quickly the image or video travels. If the neural network's stride is set to 1, then the filter will only move one pixel in the image (or one unit). Because the filter size impacts the encoded

output volume, stride is often set to a full integer rather than a fraction or a decimal.

Convolutional neural networks are used in this case to decipher the meaning of an image that is sent into them through a computer. It is possible to decrease a 3x3 pixel filter to a single pixel on the output layer by applying a 3x3 pixel filter. The output reduces as the stride or movement lengthens. Padding, a function that adds blank or empty pixels to the image frame, allows you to minimize the output layer's size while still maintaining high quality results. Stride reduces the image's size; therefore, this is a technique to compensate for it. A convolutional neural network is incomplete without padding and stride.

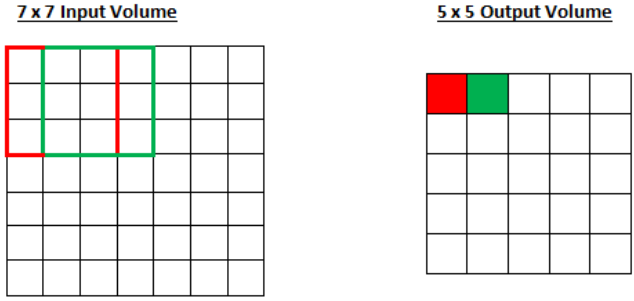


*Figure 5.10: Five stride**

*Source: https://deepai.org/machine-learning-glossary-and-terms/stride

CHAPTER 6

Natural language Processing

6.1. Introduction to Natural Language Processing

Computers can now comprehend human language thanks to Natural Language Processing (NLP). New Natural Language Processing (NLP) employs algorithms to extract meaning from utterances and provide outcomes behind the scenes. To put it another way, it understands human language so that it can carry out a wide range of jobs.

Virtual assistants like Google Assist, Siri, and Alexa are perhaps the most well-known instances of NLP in action. Using natural language processing (NLP), a question like "Hey Siri, where's the closest gas station?" may be

translated into numerical values that computers can comprehend and use.

Chatbots are another well-known use of NLP. They aid customer service representatives in resolving difficulties by automatically translating queries made in a variety of languages.

If you've ever used an app that uses Natural Language Processing (NLP), you may not have even noticed it. When drafting an email, offering to translate a Facebook post written in a foreign language, or filtering undesired commercial emails into your spam box, text suggestions are provided.

NLP aims to make human language — which is complicated, ambiguous, and immensely varied — easier for robots to grasp by making it more understandable.

6.1.1. Evolution

From the mid-20th century forward, computer science and computational linguistics discoveries have been used in the development of NLP. The following significant events marked its progression:

The Turing Test, created by Alan Turing in the 1950s to evaluate whether or not a computer is actually intelligent, had its origins in this decade. As a measure of a person's cognitive abilities, the exam utilizescomputerized interpretation and the production of natural language.

Early in the history of natural language processing (NLP), rules were used to specify how computers would interpret language.

1990s: As computer power increased, a statistical approach to NLP development began to take precedence over a top-down, language-first strategy for NLP development. It was possible to establish rules based on linguistic data without the need for an actual linguist to create them. During this decade, data-driven natural language processing gained popularity. When it comes to natural language processing, the focus has changed away from linguistics and toward engineering, which draws on a broader range of scientific fields.

The phrase "natural language processing" exploded in prominence between 2000 and 2020. Advances in computer power have led to a wide range of practical

applications for natural language processing. Traditional linguistics and statistical techniques to NLP are used in modern approaches to NLP.

When it comes to technology and how we use it, natural language processing is a critical component. Chatbots, cybersecurity, search engines, and big data analytics are just some of the real-world uses of this technology. NLP is likely to remain a significant element of business and daily life, despite its problems.

6.1.2. NLP Techniques

In order to assist computers, comprehend text, Natural Language Processing (NLP) employs two methods: A syntactic and semantic investigation.

Syntactic Analysis

Syntactic analysis, often known as parsing, is the process of examining text using fundamental grammatical principles to determine the structure of sentences, the arrangement of words, and the relationships between different words.

A few of its most important sub-tasks are as follows:

- Text is simplified through tokenization, which divides it into smaller units called tokens (which might be phrases or words).
- Tagging tokens as verbs, adverbs, adjectives, nouns, etc. is known as PoS tagging. There are several ways to employ the word "book" in a sentence, but the most common is to use it as a noun.
- Inflected words may be reduced to their simplest form via lemmatization and stemming.
- In order to eliminate words like I, they, have, like, yours, etc., stop-word elimination eliminates frequently occurring ones that don't have any further semantic meaning.

Semantic Analysis

Textual meaning is the topic of semantic analysis. It begins by delving into the nuances of each individual word's context (lexical semantics). Then, the combination of words and what they signify in context are examined. Semantic analysis has the following primary sub-tasks:

- Disambiguation of word sense is the process of determining the meaning a word is intended to convey in a certain situation.
- The goal of relationship extraction is to figure out the connections between different types of things (such as locations, people, and organisations) in a piece of writing.

6.1.3. Importance of NLP

Businesses must be able to analyse enormous amounts of unstructured, text-heavy data quickly. When it comes to analysing data, organisations were previously unable to do so since so much of it was made up of human language and housed in databases. Natural language processing comes in handy in this situation.

When evaluating the following two sentences, it is clear that natural language processing has a significant advantage: Every service-level agreement should include cloud computing insurance, and "a solid SLA assures an easier night's sleep—even in the cloud." In natural language processing, the software will detect that cloud computing is an entity, the abbreviation cloud is used, and

SLA is an industry abbreviation for service-level agreement.

These are the kinds of ambiguous features that emerge regularly in human language and that machine learning algorithms have hitherto been unable to comprehend. Algorithms now have the ability to understand them better because to advancements in deep learning and machine learning. These changes allow for a wider range of data to be evaluated.

Large volumes of textual data

Using natural language processing, computers can converse with people in their own language, as well as speed up other language-related processes. Computers can now read text, hear voice, and analyse it using NLP. They can also gauge how people feel about what they hear and use that information to make decisions.

Automated systems now are able to process more language-based data than people do without weariness or prejudice. Considering how much unstructured data is created every day, from medical records to social media,

automation will be important for the effective analysis of text and voice data entirely.

Structuring a highly unstructured data source

The sheer breadth and depth of the English language is mind-boggling. Both vocally and written, we may communicate ourselves in an unlimited number of ways. Each language has its own set of grammar and syntactic rules, terminology, and slang, which are all unique to that language. Our writing is littered with typos and omissions of punctuation. It's common for us to talk with regional accents, stammer, stutter, and use words borrowed from other languages when we speak.

It's becoming more common to employ machine learning techniques like supervised and unsupervised learning and deep learning to model human language, but there is a need for additional syntactic and semantic knowledge as well as domain expertise.

For many downstream applications, such as voice recognition or text analytics, NLP is critical because it helps resolve ambiguity in language and provides meaningful quantitative structure to the data.

6.1.4. NLP in Business

When it comes to customer feedback, NLP techniques are helping firms get a better understanding of how their consumers regard them across all channels of communication.

Automating tedious and time-consuming processes using artificial intelligence (AI) may free up employees to concentrate on more interesting and rewarding projects. AI can help organisations better understand their consumers' online discussions and how they speak about them.

Here are some of the most common uses of NLP in the workplace:

Sentiment Analysis

Emotions in text may be detected via sentiment analysis, and the results can be categorised as positive, negative, or neutral. Pasting text into this free sentiment analysis tool will show you how it works.

Companies can learn a lot about their consumers by monitoring social media postings, product reviews, and online surveys. You might, for example, monitor tweets

referencing your business in real time and respond to complaints from dissatisfied consumers as soon as they arise.

Sending out a survey to find out how your consumers feel about the service you provide could be a good idea. It is possible to identify which areas of your customer service earn favorable or negative feedback by evaluating open-ended replies to NPS surveys.

Language Translation

Facebook's translations in 2019 achieved superhuman performance because to advances in machine translation technology made in the last several years.

Using translation software, firms may expand their worldwide reach or enter new markets by communicating in other languages.

It is possible to teach translation machines to recognise certain terms in a specific field, such as finance or medical. As a result, you won't have to stress about using generic translation tools and getting poor quality results.

Text Extraction

Text extraction is a technique that allows you to extract specific data from a body of text. This application helps you identify and extract significant keywords and attributes (such as product codes, colours, and specifications), as well as named entities, from enormous volumes of data (like names of people, locations, company names, emails, etc).

In addition to finding essential phrases in legal documents, identifying key words in customer service requests, and extracting product specifications from a paragraph of text, companies may utilise text extraction for a variety of additional purposes. What do you think? You may use the following keyword extraction tool to get started:

Chatbots

Artificially intelligent (AI) systems, such as chatbots, may converse with people through text or voice.

More and more businesses are turning to chatbots to help with customer service because of their ability to respond

quickly, manage many inquiries at once, and free up human agents from answering the same questions over.

You can depend on chatbots to complete basic and repetitive tasks because they learn from each contact and get better at interpreting user intent. If they encounter a client inquiry that they are unable to address, they will forward it to a human representative for further assistance.

Topic Classification

Unstructured text may benefit from topic categorization since it allows you to group similar pieces of text together. It's a terrific technique for organisations to obtain insights from customers' comments.

Consider the possibility that you'd want to examine the open-ended replies to a slew of NPS questionnaires. Your customer service is mentioned in how many responses? What proportion of consumers bring up the subject of "Pricing" while speaking about their shopping experience? You'll be able to quickly tag all of your NPS feedback data using this subject classifier.

Topic categorization may also be used to automate the process of categorizing and routing inbound support requests to the appropriate individual.

6.1.5. NLP tools and approaches

Python and the Natural Language Toolkit (NLTK)

NLP problems may be tackled using a broad variety of tools and modules available in Python. Natural Language Toolkit (NLTK), an open-source collection of libraries, tools, and educational materials for constructing NLP programmes, contains several of them.

For example, NLTK offers libraries for NLP tasks including sentence parsing, word segmentation, stemming and lemmatization techniques of cutting words down to their roots, tokenization, and tokenization subtasks (for breaking phrases, sentences, paragraphs and passages into tokens that help the computer better understand the text). Semantic reasoning, the capacity to draw logical inferences from data collected from text, is also included in the libraries.

Statistical NLP, machine learning, and deep learning

Hand-coded, rule-based NLP systems were developed in the early days of NLP, but they couldn't keep up with the ever-growing amounts of text and speech data that were being generated.

Automated extraction and classification of text and speech data using statistical NLP is now available; thanks to computer algorithms that integrate machine learning and deep learning models to give a statistical probability to each probable interpretation of those components. It is now possible for NLP systems to 'learn' as they operate, extracting ever more accurate meaning from massive quantities of raw, unstructured, and unlabeled text and speech data sets thanks to convolutional neural networks (CNNs) and recurrent neural networks (RNNs).

6.1.6. Benefits of natural language processing

Humans and machines can now interact more effectively thanks to NLP. Code, the computer's native language, is the most direct means of controlling a computer. Human-computer interaction may be made considerably more

natural by allowing computers to comprehend human language.

Other benefits include:

- enhanced documentation quality and efficiency; capacity to automatically provide a comprehensible summary of a bigger, more complicated source material; beneficial for personal assistants like Alexa by allowing spoken word understanding;
- uses chatbots in the customer service department of a firm
- Sentiment analysis is much more straightforward, and complex analytics are now possible despite the high data volume.

6.1.7. Challenges of natural language processing

Natural language processing has a variety of difficulties, most of which stem from the fact that natural language is always developing and ambiguous. They include:

Precision: There are two ways to communicate with computers: using a precise, unambiguous and highly

organised programming language or by using a restricted number of well enunciated voice instructions. Although human speech is frequently confusing, the language structure may be influenced by numerous complicated elements, such as regional dialects and social context.

Tone of voice and inflection: The art of understanding natural language is still a work in progress. Semantic analysis, for example, is still a hurdle. Another problem is that programmes have a hard time understanding language used in an abstract way. Natural language processing, for example, has a hard time picking up sarcasm. For the most part, you'll have to pay attention to the words people say and the context in which they're being said. Another illustration: the emphasis placed on a word or syllable may alter the connotation of a statement. Speech recognition algorithms may overlook tiny but significant tone changes in a person's voice when using NLP. Different dialects have varying tones and inflections, making it difficult for an algorithm to comprehend their speech.

Evolving use of language: Language and the way people use it are always changing, which makes it difficult to

digest naturally. Rules of language do exist, but they are not fixed and may change throughout time. As the properties of real-world language evolve, the hard computational principles that work today may become outdated.

CHAPTER 7

Memory Augmented Neural Networks

7.1. Basics of MANN

To put it another way, Meta Learning, or "Learning to Learn," is a rapidly expanding area of study in the science of AI, especially Reinforcement Learning as shown in figure 7.1. DNN, CNN, and RNN are examples of traditional Deep Learning architectures that are designed or created to perform well in a certain application or problem domain. They're needed to get the settings(weights and biases) just right, in other words, given that a training data set for the particular task.

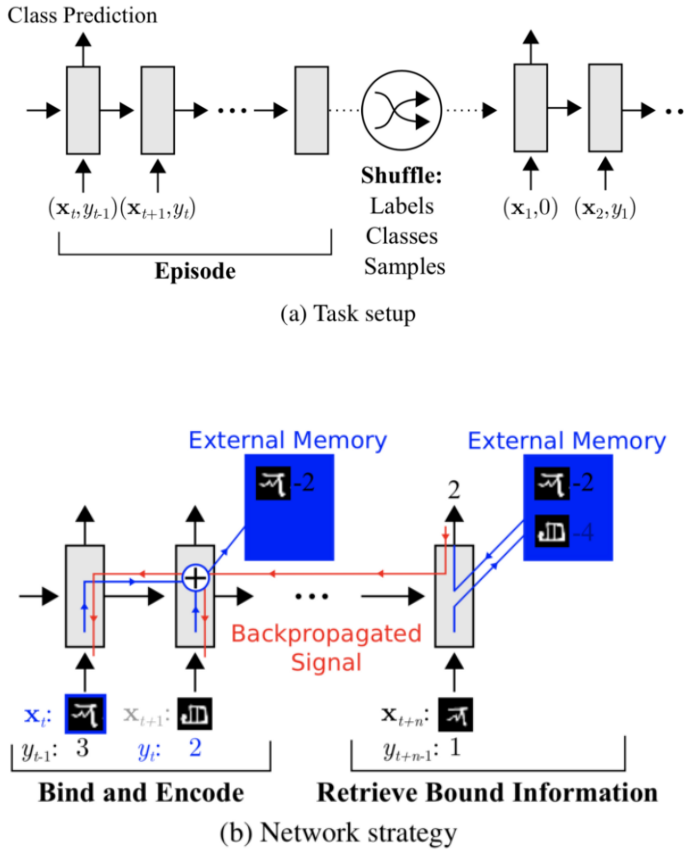


Figure 7.1: Memory Augmented Neural Network Architecture[*]

Meta Learning is the concept of allowing a neural network to gain knowledge from past tasks and use that knowledge to complete an entirely new assignment. Meta learning has

[*]Source: https://medium.com/the-ai-team/memory-augmented-neural-network-for-meta-learning-case-study-56af9cc81ae2

been the subject of several investigations, with innovative architectures having been presented in the process. MANN is one of the neural networks that was inspired by Neural Turing Machine's external memory.

For one-shot learning tasks, the most often used MANN is a variation of the Neural Turing Machine. One-shot operations need MANN's location-independent addressing; it cannot be used with NTM's. A new addressing scheme, dubbed least recently used access, has been implemented by MANN. Behind the scenes, content-based addressing is employed to identify the least recently used memory location during a read operation. So, we employ content-based addressing for reading and writing to the place that hasn't been used in the last several minutes.

7.2. Neural Turing Machines

Although theoretically conceivable, achieving universality in reality is incredibly tough! Because we're looking at an enormous search space of alternative RNN wirings and parameter values, gradient descent is unable to identify an

effective solution for any arbitrary issue because the search space is so wide.

You can think about this easy reading comprehension question:

```
Mary travelled to the hallway. She grabbed the milk glass there.
Then she travelled to the office, where she found an apple
and grabbed it.
```

The answer is so trivial: It's two! How did our minds come up with the solution so quickly and easily, though? If we were to design a computer programme to answer the comprehension question, we might try something like this:

```
1. allocate a memory location for a counter
2. initialize counter to 0
3. for each word in passage
    3.1. if word is 'grabbed'
        3.1.1. increment counter
4. return counter value
```

In the same manner that a computer programme solves a basic problem, our brains do the same thing. As soon as we begin reading, our brains begin allocating memory and storing the bits of information we're receiving. We begin by saving Mary's current position, which is the hallway following the first statement. The items Mary is carrying

are listed in the second phrase, and at this point it's only a glass of milk.

For example, if we view the third phrase, we'll remember that the office is where the first memory place was. By the conclusion of the fourth phrase, both the milk and the apple have been added to the second memory location. To get the answer, we go to the second place in our memories and count what's there, which comes out to be two! A working memory is a mechanism for temporarily storing and manipulating information in neuroscience and cognitive psychology, and it's a major motivation for the studies we'll be covering in the remainder of this chapter.

7.3. Attention-Based Memory Access

It is necessary that the whole architecture be differentiable in order to calculate the gradient of some output loss with regard to the model parameters that process input before being able to train an NTM using a gradient-based search strategy. As the inputs and outputs may be differentiated, this quality is referred described as "end-to-end-differentiable."

For example, if we tried to access the NTM's memory in the same way a digital computer accesses its RAM, using discrete address values, we would lose our ability to train the model using a gradient-based technique because of the discontinuities in gradients that would be introduced. When we want to access our memories, we need a technique to "focus" our attention. Concentration strategies may help you acquire this level of focused attention!

The number of memory locations is the same as the number of normalised SoftMax attention vectors we let each head create instead of a discrete memory address. All of our memory locations will be accessible at the same time through this attention vector, with each value of the vector representing how much attention we'll pay to a certain place or location's likelihood of being accessible.

For example, at the moment t, we want to read the vectors out of our N x W Attention vectors or weighting vectors of size N are generated using NTM's memory matrix (Mt) (the number of locations and W is the location's size), and the product may be used to create our read vector:

$$\mathbf{r}_t = M_t^\top w_t$$

Where T denotes the matrix transpose operation.

7.4. NTM Memory Addressing Mechanisms

7.4.1. Neural Turning Machine

A neural network model with working memory is known as a Neural Turing Machine. As the name implies, it connects a neural network with external memory resources. With gradient descent, the whole building may be differentiated from top to bottom. Copying, sorting, and associative memory are all activities that may be predicted by the models.

A neural network controller and a memory bank make up the fundamental architecture of an NTM. The NTM architecture may be seen at a high level in the figure. Input and output vectors are used to communicate between the controller and the outside environment. In addition to interacting with a memory matrix, it does selective read and write operations, which it does not do in a typical network. The network outputs that parameterize these processes are referred regarded as "heads" by analogy to the Turing machine.

All of the architecture's components may be rearranged and reordered. In order to do this, we define “blurry” read and write operations that interact with all of the components in memory to varying degrees (rather than addressing a single element, as in a normal Turing machine or digital computer).

Each read and write operation interacts with a tiny piece of memory while disregarding the rest due to an attentional "focus" mechanism that constrains the blurriness to that region. In order to avoid interference, the NTM prefers to store data in a non-interfering manner. The heads' specific outputs select which memory region is brought into focus. Using these outputs, the memory matrix (also known as memory "locations") may be weighted in a normalised manner.

There are two weightings for each read or write head: one for each location and one for each head. In this way, a person's brain may pay close attention to a single memory or disperse its attention among a large number of memories.

In the most basic sense, an NTM is made up of a controller and a 2D matrix known as a memory bank, matrix, or just

memory. The neural network gets input from the outside world and delivers output to the outside world at each step in the time computation. In addition to reading and writing to specific memory locations, the network has the capacity to read from and write to specific memory locations.

7.5. Differentiable Neural Computers

Even while NTMs are very powerful, there are a couple drawbacks to how they store information. The first of these drawbacks is that NTMs do not have the ability to verify that written data does not conflict or overlap with each other. As a result of this "differentiable" writing process, we write new data everywhere in memory to a certain amount that we can control. A generally interference-free behaviour of the NTM may be achieved when the attention mechanisms learn to concentrate the write weightings on just one memory region, although this is not always the case.

If the NTM ever achieves interference-free behaviour, a memory location that has already been written to will never be usable again, even if the data stored there is no

longer relevant. second drawback of NTM is its inability to release and reuse memory areas.

Any temporal information about data being written is only possible with contiguous writing: successive data is written to the same spot at the same time. The third constraint of NTMs is the inability of a read head to restore the temporal connection between data written before and after a write head jump in the memory.

7.6. Interference-Free Writing in DNCs

NTMs' initial drawback was that they couldn't guarantee interference-free writing behavior. Designing the architecture around a single free memory location instead of waiting for NTM to learn to do so is an easy solution to this problem. We need a new data structure that can carry this kind of information to keep track of which spots are free and which are busy. Henceforth known as the utilization vector.

There are elements in a usage vector ut with the size of N, each of which carries an integer value between 0 and 1, indicating how much the associated memory address is being utilised.

The initial value of the use vector, with 0 representing a completely free location and 1 representing a used.

Initially the usage vector contains 0s (u0=0) and is updated with info across steps. In light of this information, it's obvious that the area with the lowest utility value should get the greatest attention from the weights. The list of location indices is referred to as a free list, and it is denoted by t. Sorting the use vector first yields the list of indices in ascending order of usage. In order to identify where fresh data should be stored, we may use an intermediate weighting based on that free list called the allocation weighting. We calculate at using:

$$a_t[\phi_t[j]] = (1 - u_t[\phi_t[j]])\Pi_{i=1}^{j-1} u_t[\phi_t[i]] \quad \text{where} \quad j \in 1, \cdots, N$$

This at first sight, an equation may seem to be unintelligible.

7.7. DNC Memory Reuse

The worst-case scenario is that we determine the weighting for allocation and all locations are utilised, or in other words ut = 1? No additional data may be allocated to memory as a result of this change in the allocation

weightings. As a result, having the option to release and reuse memory becomes more important.

Knowing which regions can be liberated vs others that can't, we construct a retention vector ψt of size N that specifies how much of each location should be retained and not get freed. Each member of this vector has a value between 0 and 1, with 0 indicating that the relevant place may be released and 1 suggesting that it should be preserved. The formula used to come up with this vec is:

$$\psi_t = \Pi_{i=1}^{R} \left(\mathbf{1} - f_t^i \mathbf{w}_{t-1}^{r,i} \right)$$

This equation essentially says that the amount of free memory a read head has read from a certain place is proportional to how much it has read from that location in the most recent time steps.

Prior to computing the allocation, we may make room for additional data by updating our consumption. Furthermore, we're able to make effective use of and recycle a finite quantity of memory, which is a drawback of NTMs.

7.8. Temporal Linking of DNC Writes

No positional association will be established between the memory location sought for allocation by DNCs and the preceding write's location, because of the dynamic memory management procedures utilised by DNCs. NTM's method of keeping temporal connection with contiguousness is not appropriate for this sort of memory access. In order to maintain track of the sequence of the written data, we'll need to keep a detailed record.

In DNCs, two extra data structures are used in addition to the memory matrix and the use vector to accomplish this explicit recording. One of the first things you'll notice about pt is that it's an N-sized vector that's regarded a probability distribution across the memory locations, with each value indicating how probable the position is to have been the last one to have been written in. Initially, the precedence is set to zero p0 = 0 and is brought up to date in the subsequent phases via:

$$p_t = \left(1 - \Sigma_{i=1}^{N} w_t^w[i]\right) p_{t-1} + w_t^w$$

When updating, the previous values of the precedence are reset using a reset factor that is commensurate to the amount of writing that was just done to the memory. A location with a big write weighting (which is the most recent place written to) receives a large value in the precedence vector as a result of the write weighting being added to the reset value.

7.9. Visualizing the DNC in Action

DNC's functioning may be seen in action by training it on a basic job, which allows us to examine the weightings and the parameters' values and visualise them in an interpretable manner. We'll utilise a slightly modified version of the copy issue we experienced with NTMs for this easy work.

Instead of copying a single binary vector sequence, our objective here is to replicate a succession of such sequences.

Figure (a) shows the single sequence input. A single sequence input and output would complete the DNC's programme and its memory would be reset in a manner that would prevent us from seeing how it manages its

memory dynamically. Instead we'll treat a series of such sequences, shown in Figure (b), as a single input.

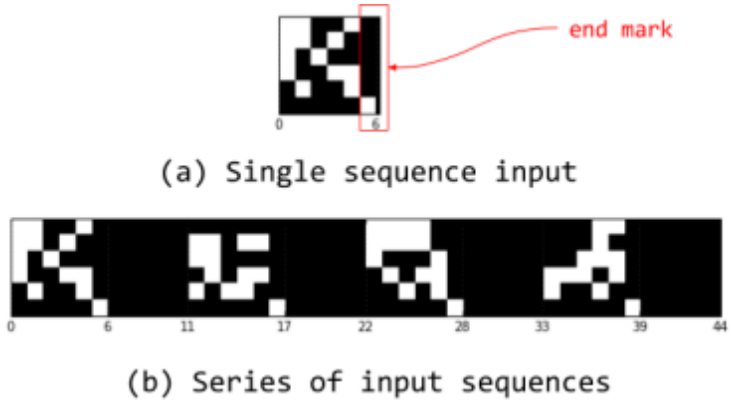


Figure 7.2: Single sequence input versus a series of input sequences[*]

We can see how the DNC is writing each of the five vectors sequentially into a separate memory region in this graphic. Read heads begin reading from these positions just as they did before the end mark is observed.

[*]Source: https://www.researchgate.net/figure/An-example-structure-of-the-input-data-set-to-LSTM-model-Here-the-light-colored-time_fig1_348426928

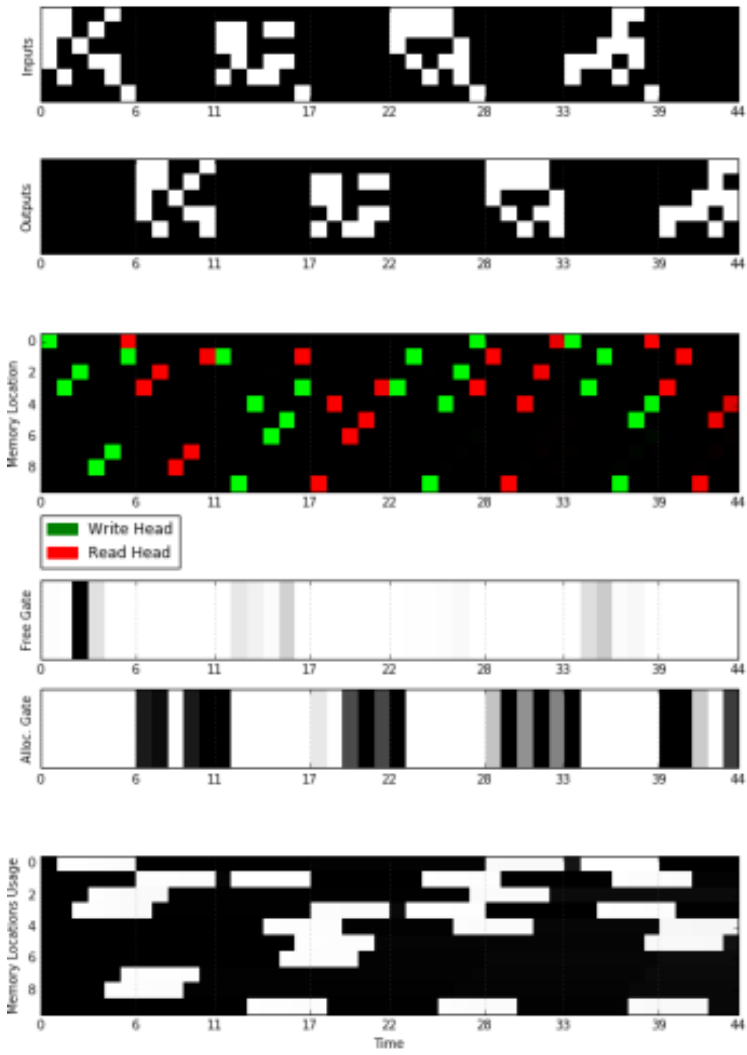


*Figure 7.3: Visualization of the DNC operation on the copy problem**

*Source:https://www.researchgate.net/figure/A-summary-of-the-five-major-steps-involved-in-the-development-of-machine-learning-based_fig1_349198111

During the writing and reading stages of each sequence in the series, we can observe how the allocation and free gates alternately activate. After writing to a memory address, we can see from the bot's vector chart that its utilisation goes from 0 to 1 precisely, and that it goes back to 0 after reading from that place, showing that it was released and may be reused yet again.

Mostafa Samir's open-source version of the DNC design includes this graphic. For reading comprehension, we'll use a simplified version of DNC, which we'll go through in the next part.

CHAPTER 8

Deep Reinforcement Learning

8.1. Introduction

With the aid of artificial neural networks, software agents may learn how to achieve their objectives via reinforcement learning. Thus, it combines function approximation with goal optimization, mapping states and actions to their resulting rewards.

A lot of these phrases may be unfamiliar to you; they will be described in more detail and plainer language below, drawing from your own particular experiences as a person moving through the world.

Recent developments in artificial intelligence (AI) have been made possible by neural networks and reinforcement learning algorithms working together, as in Deepmind's

AlphaGo, which defeated the world champions of the Go board game. That is the reason why you should be concerned about real life in the real world.

Algorithms that learn to maximise along a given axis over a large number of iterations are known as reinforcement learning algorithms. They may, for example, learn to maximise the number of points earned in a game over many iterations. In the correct circumstances, reinforcement learning systems may begin with a blank slate and acquire superhuman capability. Reinforcement is the process of punishing algorithms for bad judgments and rewarding them for good ones, much as a pet would be.

Many Atari video games, like Starcraft II and Dota-2, may be beaten using reinforcement algorithms that use deep neural networks. Although it may seem simple to non-gamers, this is a huge leap forward for reinforcement learning, and the field is moving quickly.

When it comes to connecting instant behaviours with long-term effects, reinforcement learning is the answer. Decisions made by a reinforcement learning system may

take some time before they bear fruit. While it might be difficult to discern which activity leads to which consequence across many time steps, they work in a delayed-return environment.

Slowly but surely, reinforcement learning algorithms are becoming more effective in real-world situations where they may choose from an infinite number of potential behaviours, rather than the restricted alternatives in a video game. To put it another way, they're making progress in the actual world. Deep RL may be able to assist you meet your KPIs if you have specific ones in mind. As recently as May 2021, DeepMind asserted that reinforcement learning will be sufficient to produce artificial general intelligence (AGI).

Deep reinforcement learning is starting to be used in industry to solve issues.

8.1.1. Reinforcement Learning Definitions

Agents, environments, states, actions, and rewards all play a role in reinforcement learning, which we'll go over in more detail below. A is a collection of all conceivable actions, whereas an is a single action that is part of that set.

Agent: A person or machine that performs a task, such as a drone delivering packages or Super Mario navigating his way through a video game, is an example of an agent. The algorithm acts as the protagonist in this theory. Consider the fact that you are the agency in your own life.

Action (A): All conceivable actions taken by the agent are included in A. However, it should be remembered that agents often choose from a list of distinct, viable actions. Running, leaping, crouching, and standing motionless are all options in video games. You may purchase, sell, or hold any number of different assets and their derivatives on the stock market. If you're using aerial drones, you have a wide range of options for speed and acceleration in 3D.

Discount factor: By multiplying the discount factor by the agent's known future benefits, the influence of those rewards on the agent's choice of action is dampened. Why? Because of this, the agent is forced into a state of short-term hedonism, where future benefits are seen less valuable than those received now. The Greek letter gamma, written in lower case, is often used to represent this: γ. An incentive of 10 points after three stages is worth 0.8^3x 10 in current value if γ is8. Future benefits

would be valued the same as immediate ones if they were discounted by a factor of 1. Those of us in this room are engaged in a battle against delayed gratification.

Environment: The environment in which the agent operates and which is responsive to the agent's actions. When the environment receives input from the agent, it delivers the agent's reward and the next state it will be in. As an agent, your acts are processed by the laws of physics and the norms of society, which decide the outcomes of your actions.

State (S): Every time an agent is in a state, it is in reference to other important things like tools, barriers, opponents, or rewards that it is positioned in respect to other significant things like a certain location and time. It might be the present circumstance, or any future scenario that the environment returns. If so, have you ever found yourself in an awkward situation? That's a state.

Reward (R): A reward is the feedback we use to determine whether or not an agent's activities were successful or unsuccessful in a particular situation. In a video game, for example, Mario gets points if he touches a coin. An agent

transmits actions to the environment from any given state, and the environment responds with the new state of the agent (which was the outcome of acting on the previous state) and any rewards. Rewarding behavior might be rewarded immediately or in the future. They are able to accurately assess the actions of the agent.

Policy (π): When deciding what to do next, an agent utilizes a technique known as a policy. It identifies the most rewarding acts based on a person's current condition.

Trajectory: In other words, a series of states and acts that impact those states. "Tossing across" is a Latin expression. The life of an agent is like that of a contemporary person, who is thrown through space and time like a ball with no anchor.

Key distinctions: Value is the total of all the benefits you may expect from a certain condition, while reward is the immediate signal you get when you're in that state. Reward is a short-term pleasure, while value is a long-term expectation. While spinach salad is a nutritious choice for supper in the hopes of living an extended and healthy life, the pleasure of eating cocaine for dinner is

well worth the trade-off. They have different time frames. There are situations where value and reward diverge: you may obtain a low, immediate reward (spinach) while moving to a position with tremendous long-term worth; or you may receive a high instant reward (cocaine) that leads to declining possibilities over time. Reinforcement learning aims to forecast and regulate the value function, not immediate rewards.

8.2. Neural Networks and Deep Reinforcement Learning

Function approximators like neural networks are especially beneficial when the state or action space is too big to be fully understood.

A value function or a policy function may be approximated using a neural network. When it comes to mapping states - actions into Q values, neural networks can do so. In reinforcement learning, we may utilize neural networks trained on samples from the state or action space to learn to forecast how valuable those samples are in relation to our objective rather than rely on a lookup table

to store, index, and update all potential states and their values.

Neural networks employ coefficients to approximate the function of input-to-output, and their learning consists of iteratively tweaking the weight of each coefficient along gradients that promise to provide the least amount of error.

Convolutional networks may be used in reinforcement learning to detect an agent's state when the input is visual; for example, Mario's screen or the landscape in front of a drone are good examples. In other words, they're doing what they do best: recognising images.

Reinforcement learning, on the other hand, uses convolutional networks to interpret pictures in a different way than supervised learning. In supervised learning, a label is applied to an image by the network, which then matches pixels with their respective names.

In reality, the labels that are most likely to match the picture will be given the highest probability rankings. A horse, a donkey, or a dog may be identified as 80% likely

to be a donkey, 50% likely to be a horse, and 30% likely to be a dog when shown with an image of one.

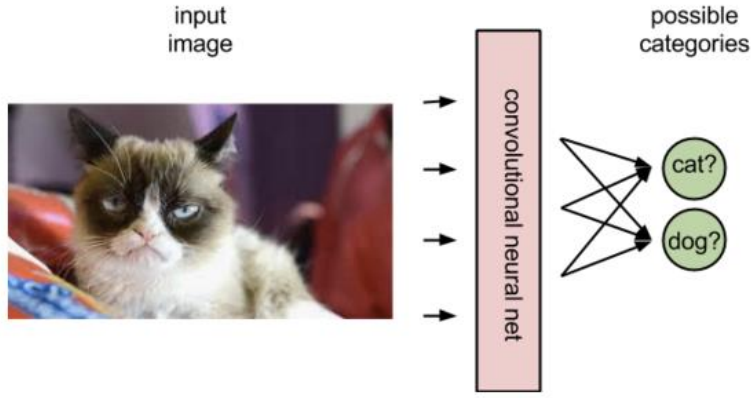


Figure 8.1: Convolution classifier[*]

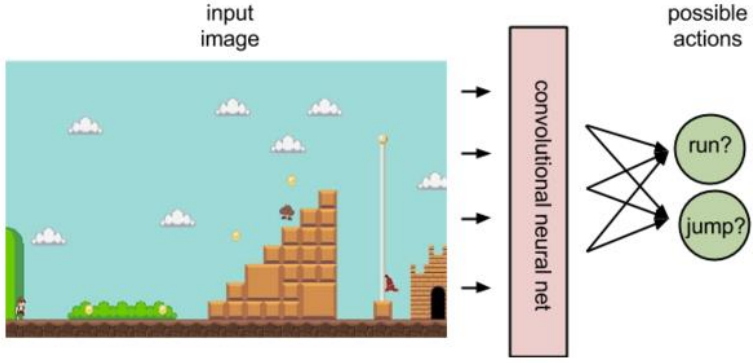


Figure 8.2Convolution Agent[†]

For example, it may anticipate that running right would earn five points, leaping will earn seven, and running left

[*]Source: https://wiki.pathmind.com/deep-reinforcement-learning
[†]Source: https://wiki.pathmind.com/deep-reinforcement-learning

will earn none using reinforcement learning and a convolutional net.

An example of the work of a policy agent may be seen in the graphic above, which depicts a state and the most effective course of action.

$$a = \pi(s)$$

A policy maps a state to an action.

If you remember, this is different from Q, which maps state action pairs to rewards.

Precisely, Q -maps State-action pairings are mapped to the maximum possible combination of immediate reward and all future benefits that may be gathered by further actions in their trajectory. Here is the equation for Q, from Wikipedia:

$$Q(s_t, a_t) \leftarrow (1-\alpha) \cdot \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right)}^{\text{learned value}}$$

The Q function simply picks the state-action combination with the greatest so-called Q value after assigning values to the predicted rewards.

The neural network coefficients may be initialised stochastically or randomly at the beginning of reinforcement learning. Feedback from the environment allows a neural network to alter its weights based on how much it expects to be rewarded vs how much it actually receives.

Backpropagation of errors in supervised learning is akin to this feedback loop. Supervised learning, on the other hand, starts with an understanding of the real-world labels that the neural network is attempting to predict. For this project, the aim is to build an image-to-name model.

For reinforcement learning to work, it needs a scalar number from the environment for every new action. Noise may be introduced into the feedback loop by varying, delaying, or influencing the environment's rewards.

While immediate incentives are important, they are just part of the equation when it comes to the Q function, which takes into consideration the long-term benefits of a particular activity.

8.3. Safely and Security of DRL

DRL agents may have to deal with potentially dangerous real-world environments, such as robots or autos. It's a big problem, and there's an important topic called safe RL that aims to address it, for example, learning a policy that optimises rewards while working within preset safety restrictions.

In addition, like with any other software system, DRL agents are vulnerable to an attack. Because we are working with systems that are far more difficult to comprehend and describe, DRL introduces a few additional attack avenues beyond typical machine learning systems.

The scope of this post's introduction does not extend to discussing DRL systems' safety and security. If a DRL system is ever implemented, bear in mind that you should address this issue in more detail for the benefit of the reader.

8.4. Successful applications of deep reinforcement learning

To demonstrate how Deep Mind's Alpha Zero uses deep reinforcement learning, chess experts have used the system and declared it the winner on several occasions. Alpha Zero basically taught itself how to play and master the game from scratch.

Chess engines like Stockfish and IBM's Deep Blue rely on hundreds of rules and situations created by expert human players to organise their strategy. This helps them anticipate every conceivable outcome. Instead of relying on a set of human rules, Alpha Zero uses deep neural networks and algorithms to train itself for each game from a position of random play, without any knowledge of the game's basic rules. It then uses deep reinforcement learning to find a solution that will position itself as the strongest player in history for that game.

As time passes and wins, losses, and draws are recorded, the neural network is fine-tuned to become more effective. As a result, it starts to make better decisions as it

progresses. DeepMind claims that AlphaZero learned the game of chess in under nine hours.

Oil and gas giant Royal Dutch Shell is putting resources into artificial intelligence research and development in an attempt to meet its goals of producing cleaner electricity, enhancing service station safety, and staying on top of the rapidly changing energy market landscape. For example, it already uses reinforcement learning for exploration and drilling to reduce the high cost of gas extraction and enhance every stage of the supply chain.

In order to guide the gas drills into a subsurface, Shell is using deep-learning algorithms that are trained on past drilling data and simulations. Data from the drill bit, such as pressure and temperature, is also included in the DRL technology, as are results from subsurface seismic surveys. Due to the improved awareness of the drilling environment, human operators of the drilling machine are able to achieve faster results and less wear and tear on costly drilling gear.

Many different industries, including health, robotics, smart grids, and finance, might benefit from deep reinforcement

learning's capacity to handle complicated problems that were previously unsolvable by machines. We have yet to witness the full effect that artificial neural networking and reinforcement learning will have on business and science as a whole, given the technology's capacity to handle unstructured data and learn like a human brain.

8.5. Some other important applications

Automotive

The car business includes a wide range of products. Deep reinforcement learning is also overpowered by a massive dataset. The technology is already in use in self-driving cars like Uber or Tesla. Transformation of industries, maintenance of automobiles will benefit from it. In addition, the sector is moving toward full-scale automation. Quality, pricing, and security are the driving forces behind the sector. With the help of data from customers and dealers, DRL will open up new possibilities. To improve the product's safety record while simultaneously lowering costs and increasing quality.

Resource Management

As a result, businesses worldwide are always on the go. Constantly looking for better ways to allocate scarce resources. As a whole, it is a procedure that requires a thorough comprehension of the subject matter. The systems that control it are also continually changing.

Reinforcement learning may be used by corporations to build deep neural networks. In this way, they are able to learn and allocate computer resources to any upcoming projects. If you're going to slow down the economy, you're going to have to make sure that you're allocating resources properly. RL intends to slash it. Divides resources in a way that maximises the organization's outcomes.

AI toolkits

OpenAI Gym, Psychlab, and DeepMind Lab are just a few of the top AI toolkits. They provide a setting for training. Deep reinforcement learning algorithms need large-scale innovation. DRL agents may be trained using these open-source technologies. The more businesses employ deep RL to solve their own specific business problems, the more successful they will be. This means that we'll have the

opportunity to see a significant growth in practical applications.

Bidding and Advertising

It's still in the early phases, but it's promising. RL, on the other hand, has shown to be a force that might fundamentally alter the worldwide ad-bidding process. Marketing and business professionals from all across the world may participate in bidding platforms. The structure of the process is an issue. In a method that meets the needs of all parties without causing a conflict of interest. Observing and documenting each of their interconnected acts as they go.

In addition to learning about bidding patterns and methods, you'll also learn about interaction. It is also important to consider the influence that interconnected activities might have on individual bidding behaviour under such high-pressure conditions.

Manufacturing

In warehouses and fulfilment centres, intelligent robots are becoming increasingly widespread. To sort and distribute

a plethora of items to the appropriate recipients. Any time a robot picks up a piece of equipment and places it into a container of any kind. Deep RL, on the other hand, aids it in maturing and making better use of its newfound wisdom.

Healthcare

Whether or whether the best treatment strategies are involved. Or the creation of new drugs and the automated use of them. Deep reinforcement learning has a lot of promise for improving healthcare. One of the primary uses of deep RL in healthcare right now is illness diagnosis. Pharmaceutical production and clinical trial and research are also included.

Traffic Control

Congestion in the streets of the city as a whole is a concern. All around the world, it has been a problem. There have been several attempts to address this developing problem by architects and developers. Few have succeeded at deciphering it.

With its multi-agent system, RL enters the fray. It contributes to the development of a traffic signal control

network. Learning is accomplished via the use of its recommendation system. Real-time learning can help construct traffic systems that not only give insight into already-existing data. To assist academics and city planners better understand the behaviour of their community, you may also help build patterns. At the same time, the amount of time spent stuck in traffic is reduced.

Bots

The UI paradigm of discussion. Deep reinforcement learning (RL) is key to the development of AI bots. The bots are becoming better at understanding the subtleties of language in a variety of contexts. In order to comprehend both spoken language and computerised speech.

Video Games

Deep RL is being used to create interactive video games that are very complicated. When the RL agent plays the game, it uses what it's learned to adjust its behaviour in order to maximise its score. 'Chess' and 'Atari games' are examples of PC games that utilize it.

As a result, opponents alter their strategy and tactics dependent on the player's performance.

In a world where the game business is becoming more competitive. It's no wonder that so many developers are hopping on the RL bandwagon with the advancement of technology. By making their games more interactive and establishing a dedicated following. Because these technologies can "learn" about human behaviour, they may respond accordingly. These developers are able to draw in a new audience like no other. Bring complexity down to the level of the average human being. Both of which contribute to a more enjoyable gaming experience.

CHAPTER 9

Advanced Topics in Deep Learning

9.1. Introduction

1. Attention models: All of the information in the environment is not being used by humans at any one moment. As a result, they concentrate on just those parts of the data that are important to their current project. The term "attention" refers to this biological concept. Artificial intelligence applications may also benefit from this kind of thinking.

In order to concentrate on just the parts of the data that are important to the job at hand, models with attention employ reinforcement learning (or other approaches). In recent years, several techniques have been used to increase performance.

2. Models with selective access to internal memory: This class of models is closely similar to models that concentrate on certain areas of the stored data, but there is one key distinction. Humans' ability to do certain activities by accessing information stored in their memory serves as a useful illustration. The brain's memory cells store a massive amount of information for humans. A little portion of it is used for the work at hand, yet it is always accessible. Modern computers contain a lot of memory, but they are meant to be used selectively and controlled via the use of variables, indirect addressing techniques. Hidden states are a common feature of all neural networks. It's difficult to separate data access from calculations since they're so intertwined. It is possible to manipulate the internal memory of a neural network more carefully and directly introduce the concept of addressing methods, which results in calculations that more closely mimic the human programming style. Out-of-sample forecasts are often better predicted by such networks than by more typical neural networks. Selective memory access may be seen as a type of internal attention to a neural network's

memory. A neural Turing machine or a memory network is the name given to the resultant design.

3. Generative adversarial networks: It is the goal of Generative adversarial networks to construct generative models of data from small samples of input data. The use of two antagonistic networks in these systems allows them to provide data samples with realistic appearances. Two networks, one of which is a generator and the other a discriminator, are used to classify a combination of actual and produced data. At some point, the discriminator can no longer tell the difference between genuine and fraudulent samples in an adversarial game. As a result, it is also feasible to use certain contexts (e.g., picture captions) to direct the development of desired samples.

Choosing which bits of a dataset to focus on might be a difficult task for attention processes. To put it another way, this decision might be seen as akin to those faced by a machine learning system. However, there are ways that do not substantially rely on reinforcement learning for developing attention-based models.

A family of systems known as memory networks is closely connected to neural Turing machines. Developing computers to answer questions has recently showed some promise, although the results are still rather basic. Building a neural Turing machine may open the door to a wide range of AI capabilities that have yet to be completely realised. With greater data and computing power, these promises will come to fruition as they have in the past with neural networks.

9.2. Attention Mechanisms

People seldom employ all of their senses to attain a single goal. As an example, imagine trying to locate a given home number on a particular street. As a result, identifying a house's mailbox or door number is an essential part of the operation. Retinal images of a larger picture are commonly there, although they are seldom focused on by the observer. The macula, or centre fovea, of the retina, has an unusually high resolution compared to the rest of the eye, which is called the retina. Most of the eye's non-central regions have poor resolution and a majority of color-insensitive rods, while this area contains a large

concentration of color-sensitive cones. Figure depicts the many parts of the eye.

A picture of the number is projected onto an area of retinal tissue that correlates to macula while reading street numbers (and especially the fovea). However, even while one is aware of other items beyond this centre field of vision, it is almost hard to employ pictures in the peripheral area to conduct detail-oriented activities. For example, seeing letters projected on the retina's periphery is very challenging. The diameter of the foveal area is 1.5 mm, which is a minuscule percentage of the whole retina. When the picture falls on the whole retina, the eye only communicates a high-resolution rendition of less than 0.5 percent of the image's surface area. This method is physiologically favorable since just a small portion of the picture is communicated in high resolution, which decreases the amount of internal processing necessary for the particular job at hand. To comprehend how selective attention works, the eye's anatomy makes it easier to grasp, although this selectivity is not limited to visual characteristics. In most cases, the human's other senses, such as hearing or smell, are focused on a specific scenario.

After discussing the concept of attention in relation to computer vision, we'll go on to explore other areas, such as text.

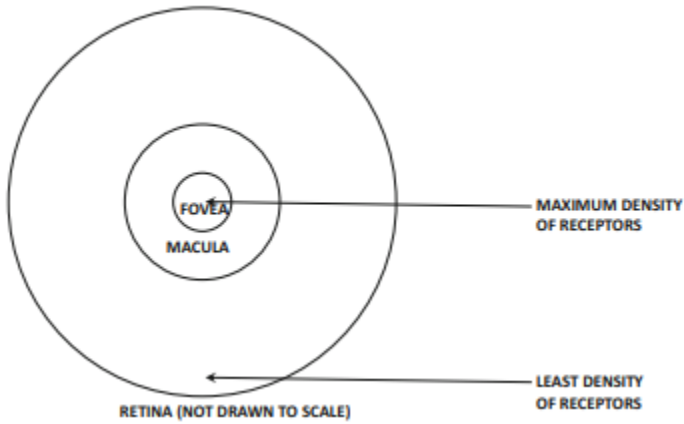


Figure 9.1: Resolutions in different regions of the eye. Most of what we focus on is captured by the macula[*]

Here, the fundamental problem is that there is no method of recognizing the relevant section of the picture based on information that is already accessible. As a result, searching for particular areas of the picture requires an iterative technique that builds on prior iterations' expertise. Biological species may serve as a beneficial

[*]Source:https://www.researchgate.net/figure/Illustration-of-the-major-landmarks-in-the-retina-The-left-image-illustrates-the_fig1_328211285

source of inspiration in this case. To acquire what they desire, living things use visual signals from whatever they're concentrating on to determine where they should look next. As an example, if we happen to see the doorknob by coincidence, then our trained neurons tell us that we should glance to the top left or right of it for the street number. Reinforcement learning techniques, which were addressed in the previous chapter, use an iterative process in which prior stages provide clues to help the learner understand how to receive rewards for their actions (i.e., accomplish a task like finding the street number). As we'll see later, attention and reinforcement learning are often used together.

Recurring neural networks are frequently unable to concentrate on the relevant parts of the source text while translating it to the target language because to this problem. The target sentence should be aligned with relevant parts of the source sentence in such circumstances. To isolate the key elements of the source text, attention processes might be beneficial in producing a particular component of the goal. If you think about attention processes, they don't necessarily need to be

framed in terms of reinforcement learning. Rather than using reinforcement learning, most attention algorithms in natural language models employ attention to softly weight certain input components.

9.2.1. Reinforcement Learning

A robot might employ this method instead of image recognition or classification since it is based on the principles of reinforcement learning rather than simple image recognition or classification tasks. Unsupervised learning, on the other hand, is a straightforward exception to this general rule. Using a SoftMax prediction, the user may choose the class label in the following way. Assuming rt is right after time stamps of t, the reward may be 1 or 0, depending on whether the categorization is valid at that point in time. The total of all reduced prizes across subsequent time stamps yields the overall reward Rt at the t_{th} time stamp. However, the application at hand may alter this procedure. You might think of this as an activity that corresponds to selecting the next word in a caption for a picture.

The following equation describes the gradient of the predicted reward at time stamp:

$$\nabla E[R_t] = R_t \nabla \log(\pi(a_t))$$

This gradient and policy rollouts are used in the neural network for backpropagation. As a result, there will be many rollouts and each one will include various activities. As a result, the ultimate ascent direction will have to be calculated by adding the gradients of all these activities (or a subset of these acts) together. In order to limit the variation, a baseline is deducted from the incentives, as is usual in policy gradient approaches. A class label is generated at each time stamp, thus the accuracy improves with each further peek. The method works effectively with a variety of data formats and between six and eight peeks.

9.3. Neural Networks with External Memory

In recent years, a number of analogous architectures have been developed that supplement neural networks with permanent memories, whereby one can regulate how computations access and change specific memory areas.

The LSTM may be regarded to have a permanent memory, despite the fact that the memory and computations are not clearly separated. This is because the values in the hidden states, which are used to store the intermediate outputs of the computations, are strongly intertwined with the calculations in a neural network.

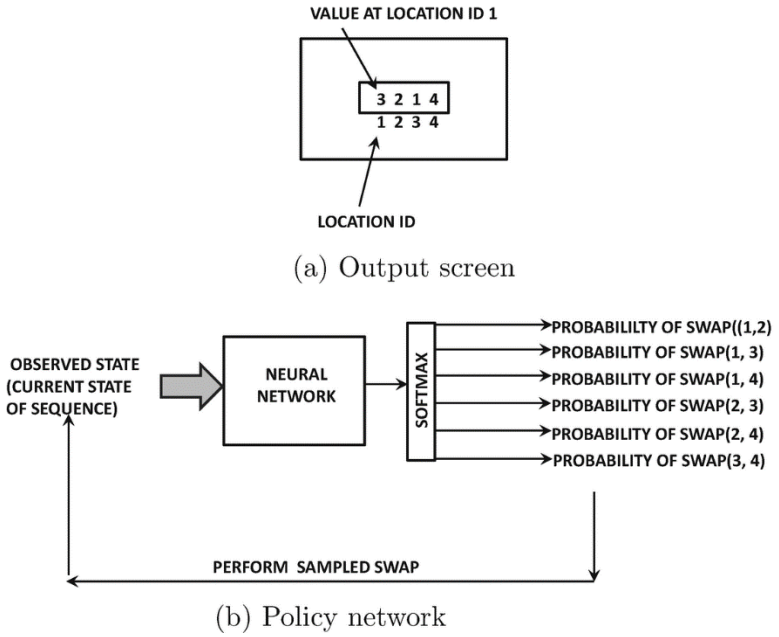


*Figure 9.2: The output screen and policy network for learning the fantasy game of sorting**

*Source: https://link.springer.com/chapter/10.1007/978-3-319-94463-0_10

Neuronal Turing machines have external memory. The controller of the computation is the basic neural network, which is able to read and write to external memory. Most neural networks lack the ability to store long-term memories, except for LSTMs. Traditional neural networks, on the other hand, do not explicitly segregate computation from memory (including LSTMs).

Persistent memory, coupled with a clear separation of memory from computations, allows a computer to mimic algorithms using instances of input and output as shown in figure 9.2. The idea of neural Turing machines, differentiable neural computers, and memory networks are all based on this notion.

9.3.1. Architecture of Controller

The controller's neural architecture is a critical design decision that must be taken into consideration. Using a recurrent neural network, which already has a concept of temporal states, is a logical option.

An LSTM, on the other hand, adds an extra internal storage capacity to the exterior storage capacity of the neural Turing machine. There are states in the neural

network that are like CPU registers that are utilised for internal calculation, but they are not durable (unlike the external memory).

It's worth noting that a recurrent network isn't required after we've established the idea of external memory. For this reason, reading and writing from the same regions at repeated timestamps results in temporal statefulness, as in a recurrent neural network, the memory is able to store the concept of states.

A feed-forward neural network may also be used as a controller, which is more transparent than the controller's hidden states. the number of read and write heads in the feed-forward architecture limits the number of operations per time-stamp.

9.3.2. Differentiable Neural Computer: A Brief Overview

Using extra structures to control memory allocation and monitor temporal sequences of writes, the differentiable neural computer improves on neural Turing computers.

Neuronal Turing machines have two major flaws that are addressed by these improvements:

1. To prevent writing on overlapping blocks while using shift-based procedures to address continuous blocks of locations on the neural Turing computer, which is capable of both content-based and location-based addressing. This problem is overcome in current computers by ensuring that the correct amount of memory is allocated at all times. Memory allocation techniques are included into the design of the differentiable neural computer.

2. The sequence in which memory regions are written is irrelevant to the neural Turing computer. A series of instructions, for example, may benefit by keeping track of the order in which memory locations are written.

Memory allocation in a differentiable neural computer is based on the ideas that (i) newly written but unread locations are likely to be beneficial and (ii) newly read locations lose their usefulness after being read. Memory allocation maintains track of how much space a certain place has been used. When a place is written to, its consumption is automatically raised, and it may be

lowered. Prior to writing, controllers emit a set of free gates from each read head to decide if the most recently read places should be released. They're then employed to bring the prior time-use stamp's vector into the present day.

In order to maintain track of the sequential ordering of the memory locations at which writes are executed, the differentiable neural computer solves a second problem. In this case, it is vital to note that the memory writes are soft, therefore it is impossible to establish a tight sequence.

Many of the concepts of neural Turing machines, memory networks, and attention processes are closely connected. There were at least two people that came up with the first two concepts at the same time. These subjects were examined in a variety of ways in the early articles.

For example, Simple copying and sorting activities were used to test the neural Turing machine, while more complex tasks such as answering questions were used to test the memory network.

However, in following tests using the differentiable neural computer, the line between the two was much more

blurred. There is still a long way to go before these applications can be employed economically, although they are now in their infancy.

9.4. Generative Adversarial Networks (GANs)

First, we'll cover the concepts of generative and discriminative models, which are both employed in the creation of generative adversarial networks. The following are two distinct kinds of educational models:

1. Discriminative models:Using the feature values in $\bar{X}$, discriminative models directly estimate the conditional probability P(y|$\bar{X}$) of the label y. An example of a discriminative model is logistic regression.

2. Generative models: Jointprobability P($\bar{X}$,y) is estimated using generative models, which is a generative probability of a data instance. Joint probability can be used to estimate the conditional probability of y given $\bar{X}$ by using the Bayes rule as follows:

$$P(y|\overline{X}) = \frac{P(\overline{X}, y)}{P(\overline{X})} = \frac{P(\overline{X}, y)}{\sum_z P(\overline{X}, z)}$$

Naive Bayes classification is an example of a model that generates data.

Generic models may be utilised in supervised as well as unsupervised contexts, but discriminative models can only be employed in supervised circumstances. It is possible to develop a generative model for just one of the classes in a multiclass environment by first establishing an acceptable prior distribution for that class and then sampling from the prior distribution to produce instances of that class.

In generative adversarial networks, two neural network models are used at the same time. Synthetic instances that are comparable to genuine cases may be generated using the first model. To achieve this, the objective is to build synthetic things that are so lifelike that a trained observer will be unable to tell the difference between a synthetic item and a genuine on.

The generative network modifies its weights such that the discriminative network has a tougher difficulty identifying samples created from synthetic objects when the discriminative network properly flags them as fakes. New samples are created and the procedure is repeated once the

generator network's weights have been modified. Generic networks become better and better at manufacturing counterfeit goods over time. In the end, the discriminator is unable to tell the difference between actual and artificially created items. When it comes to the Nash equilibrium of a game like MiniMax, the distribution of points generated by the generator may be formally proved to be identical to that of data samples. Having a high-capacity model and access to a large amount of data are critical for the approach's success.

Many machine learning algorithms may benefit from vast volumes of synthetic data supplied by the manufactured objects. It is feasible to utilize this strategy to generate objects with diverse characteristics by adding context. Example: "spotted cat with collar" might be the caption input and the result would be a fantasy picture matching the caption. It is not uncommon for the created things to be used in creative activities as well.

Image-to-image translation is another recent use of these techniques. Images may be accurately translated from one form to another via image-to-image translation. Before we

get into the applications, we'll go through how to train a generative adversarial network.

9.5. Competitive Learning

The majority of the learning strategies covered here focus on adjusting the weights in the neural network in order to correct for errors made during the training process. In terms of learning, there are substantial contrasts between competitive and noncompetitive learning, with the latter representing a fundamentally different paradigm. The fight amongst neurons for the right to respond in a certain manner to an assortment of similar input data is more like a sport than anything else. Consequently, the learning process differs significantly from the backpropagation approach used by neural networks in terms of complexity.

The following is a general notion in training: An output neuron is more likely to be activated if its weight vector is more comparable to the input. The weight vector of the neuron is expected to have the same dimension as the input. The activation may be computed using the Euclidian distance between the input and the weight vector. Smaller distances result in bigger activations. For

purposes of moving, it closer to an input, the output unit with the greatest activation is deemed the winner.

Only the winning neuron (i.e., the neurons with the greatest activity) is updated in the winner-take-all approach, leaving the rest of the neurons untouched. Other competitive learning paradigm versions enable other neurons to participate in the update based on pre-defined neighborhood interactions. They are called cooperative learning paradigms. Neurons may also block one another via various means. Using these processes, one may learn representations with a pre-defined structure, which is beneficial in applications such as two-dimensional visualisations.

9.6. Limitations of Neural Networks

Many tasks, including picture classification, have been surpassed by deep learning in recent years, and it has even outperformed humans.

Reinforcement learning, on the other hand, has shown remarkable effectiveness in games that demand sequential planning. As a result, it is tempting to speculate that artificial intelligence may one day be able to match or even

surpass human talents. A number of basic technological challenges must be overcome before we can construct computers that can learn and think like humans.

Because of the enormous quantity of data needed to train neural networks, they are unable to provide results of the same quality as humans. Neural networks, on the other hand, spend significantly more energy than humans do while doing the same job.

9.6.1. An Aspirational Goal: One-Shot Learning

It's no secret that deep learning has gained popularity in recent years, because to its superior performance on massive data sets, but this also highlights a critical flaw in current deep learning technologies. There are instances when deep learning has outperformed humans, but it has done it in a sampling-inefficient manner.

There are over a million photos in the ImageNet database and a neural network would typically need thousands of samples to correctly categorise each one. In order to identify a vehicle as such, we don't need to look at tens of thousands of photographs of it. When a youngster sees a vehicle for the first time, he or she is more likely to identify

another truck of the same type, shape, and color. This shows that people are better able to adapt to novel situations than artificial neural networks. One-shot learning is a broad idea that refers to the ability to learn from just one or a small number of instances.

Generalizing with fewer instances is not unexpected since the human brain's connection is sparse and has been built by nature to be efficient. This architecture has been handed down through the generations for millions of years. In a roundabout way, the structure of human brain connections already encodes "knowledge" gleaned through millions of years of "evolutionary experience."

In addition, humans accumulate a range of skills and information throughout the course of their lives, which allows them to pick up new skills more quickly. A person's lifetime of experience and the encoded information they are born with allow them to acquire certain skills (like how to recognize a vehicle). To put it another way, human beings have mastered the art of transferring their knowledge across generations.

Another kind of transfer learning is based on the idea of learning across different activities. Training work that has previously been completed in one activity may be reused to help the learner with another assignment. Learning-to-learn is a term used to describe this kind of learning.

9.6.2. An Aspirational Goal: Energy-Efficient Learning

Sample efficiency and energy efficiency are intertwined concepts. High-performance hardware-based deep learning systems are wasteful consumers of electricity. It is possible to use more than one kilowatt of electricity by using many GPU units in parallel to perform a compute-intensive operation.

To put it another way: A light bulb, on the other hand, takes a lot more power than an average human brain does to operate. Another thing to keep in mind is that the human brain does not always conduct accurate calculations, instead relying on educated guesses. Many educational contexts may be satisfied with this level of generalization, and it may even enhance it. That designs that place an emphasis on generality rather than precision

may be more energy efficient is a reasonable conclusion to draw from this research.

Many algorithms have lately been devised that compromise off computation accuracy for increased computational efficiency. Noise effects from low-precision calculations boost generalization in several of these strategies.

When a neural network is kept small and unnecessary connections are trimmed, energy efficiency is frequently attained. Getting rid of unnecessary connections might also assist with regularisation.

In a second route, neural network-specific hardware may be developed. Even if the divide between software and hardware is useful for computer maintenance, it is also a cause of inefficiency for the human brain, which does not share this distinction. Recent advancements in neuromorphic computing have been made.