

# Chapter 6

## Artificial Intelligence—Machine Learning



### 6.1 What is Machine Learning

Machine learning is a subfield of AI that seeks to replicate the learning capabilities of humans on computers. It has extensive applications in tasks such as object recognition, which involves recognizing and understanding input data to achieve intelligent processing. Some notable applications of machine learning include speech recognition, character recognition, image recognition, anomaly detection, medical diagnosis, financial market prediction, and obstacle sensing in self-driving technology.

Machine learning plays a crucial role in imitating the five human senses—sight, hearing, touch, taste, and smell—by processing perceptual information. For example, computer vision aims to replicate human vision and is often applied in the development of robotic eyes. It goes beyond mere image capture, encompassing tasks like extracting information from images and understanding scenes. Pattern recognition is a process of classifying patterns, such as speech and images, into distinct categories. The next level of pattern recognition is the understanding of patterns and scenes. Natural language processing enables computers to work with human language. It facilitates interaction between machines and humans, including tasks like speech recognition for text input, text-to-speech synthesis for information presentation, and predictive text input on smartphones and personal computers.

The process of machine learning is training a computer by inputting data samples and allowing the computer to derive valuable rules and criteria from these samples. Subsequently, the computer employs the extracted rules to make decisions when presented with new inputs.

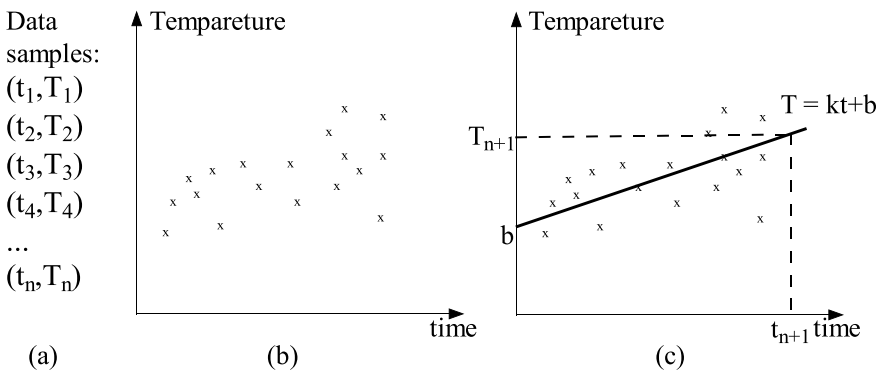
## 6.2 Types of Machine Learning

Machine learning can be broadly categorized into two main types: supervised learning and unsupervised learning. In supervised learning, the supervisor is not a human teacher but the correct output. The computer learns from data samples where each record is labeled the correct output. For example, input many images of cats and dogs labeled with their respective correct answers, i.e. a cat or a dog, into a computer and let the computer learn to recognize whether a newly input picture is a cat or a dog. In contrast, if the computer learns from data samples with no label of the correct output, it is called unsupervised learning.

Supervised learning includes various techniques, and among them, regression and classification are two fundamental techniques. Regression aims to predict numerical values for given inputs [1]. For example, consider Fig. 6.1a, which displays a set of records containing time and temperature data points. Each record, such as  $(t_1, T_1)$ , represents the temperature  $T_1$  at time  $t_1$ . Collectively, these records are referred to as data samples. When we input these data samples into a computer, their distribution is illustrated in Fig. 6.1b. The computer's task is to find a function that accurately represents the distribution or trend of these data. As shown in Fig. 6.1c, the computer may find a function like  $T = kt + b$ . This function can then be used to predict the temperature at a future time, such as temperature  $T_{n+1}$  at time  $t_{n+1}$ . Notably,  $t_{n+1}$  is a new input that was not present in the original data samples, and  $T_{n+1}$  is the predicted output based on the function.

Regression can also be applied to complex data distributions. For example, by inputting historical stock price movement data samples, a function that fits the movement, such as a sine or cosine function, can be discovered and used to predict future price movements.

Classification assigns attributes or types to input data. For example, if we categorize human facial expressions into joyful, angry, sad, and neutral, and train a computer



**Fig. 6.1** An example of using regression for prediction

with labeled data samples of numerous facial expressions, it can subsequently classify new facial expressions into one of these types. Classification that can return one of two types is used for tasks like detecting harmful or harmless bacteria, assessing the normalcy or abnormality of a machine, identifying ordinary or spam emails, and determining whether a program is harmless or a virus.

Unsupervised learning does not rely on correct answers and allows a computer to learn automatically from vast datasets by extracting structure, patterns, and rules. One unsupervised learning technique is clustering, which groups input data.

In addition to regression and classification, machine learning encompasses various techniques, including decision tree learning, relational rule learning, reinforcement learning, and neural network. These techniques are not limited to supervised learning or a single type of learning. In this textbook, we will focus on neural network, which has gained significant attention in recent years, especially with the rise of deep learning.

## 6.3 Neural Network

### 6.3.1 What is a Neural Network

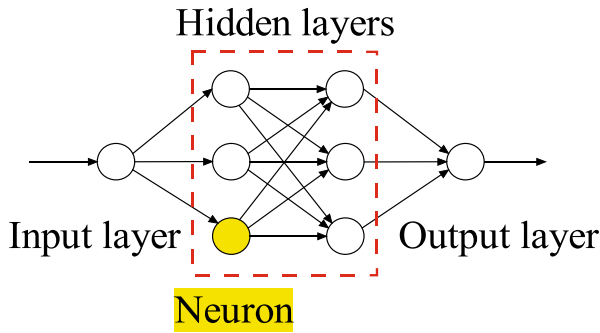
A neural network, also known as an artificial neural network, is a computational model inspired by the neural network that makes up the human brain. It is composed of multiple artificial neurons connected by synapses. By adjusting the strength of synaptic connections through a learning process, a neural network can be employed to solve a wide range of problems.

Figure 6.2 illustrates a simplified neural network, with each circle representing a neuron and the arrows representing synapses [2]. The direction of an arrow represents the flow of information. A neural network consists of three primary layers: the input layer, the output layer, and optional hidden layers. The hidden layers are termed as such because they remain invisible from the user, who can observe only the input and output. Each layer must contain at least one neuron, and each neuron should be connected to all the neurons in the subsequent layer. The strength of connections is known as synaptic strength and may differ among synapses.

A neuron, often referred to as a node or unit, receives information from the input layer or other neurons. The most commonly used model for a neuron is Eq. 6.1 [1].

$$y = f\left(\sum_{i=1}^n w_i x_i - \theta\right) \quad (6.1)$$

In this equation,  $y$  represents the output, and  $x_i$  represents the  $i$ th input. For example, consider the neuron in the output layer illustrated in Fig. 6.2. It has three inputs:  $x_1$ ,  $x_2$ , and  $x_3$ . Each input has a weight  $w_i$ , representing its synaptic strength. In other words,  $w_1$  is the weight of  $x_1$ ,  $w_2$  is the weight of  $x_2$ , and  $w_3$  is the weight of  $x_3$ . The weighted sum of these inputs is denoted as  $\sum_i w_i x_i$ . This sum is compared with



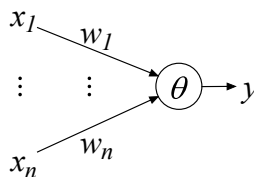
**Fig. 6.2** A simple neural network

a threshold value  $\theta$  of the neuron. If the weighted sum is no less than the threshold, the neuron will become excited, resulting in an increase in the output value  $y$ . In the following, we will provide detailed explanations using the threshold logic unit as an illustrative example.

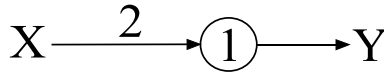
### 6.3.2 Threshold Logic Unit (TLU)

The threshold logic unit (TLU), depicted in Fig. 6.3, is a single neuron having  $n$  inputs  $x_1$  to  $x_n$  and one output  $y$ . If the weighted sum of inputs  $\sum_{i=1}^n w_i x_i$  is less than the threshold, the output will be 0. If the weighted sum of inputs  $\sum_{i=1}^n w_i x_i$  is greater than or equal to the threshold, the output will be 1 (Eq. 6.2) [3].

$$y = \begin{cases} 0, & \text{if } \sum_{i=1}^n w_i x_i < \theta, \\ 1, & \text{if } \sum_{i=1}^n w_i x_i \geq \theta. \end{cases} \quad (6.2)$$



**Fig. 6.3** Threshold logic unit



**Fig. 6.4** A TLU with a single input

**Table 6.1** Truth table of the TLU in Fig. 6.4

Input	Output
$X$	$Y$
0	0
1	1

Figure 6.4 shows a TLU with a threshold of 1 and a single input  $X$  whose weight is 2. Table 6.1 presents the truth table of this TLU. Let us explain why the output values  $Y$  are 0 for  $X = 0$  and 1 for  $X = 1$ , respectively.

When  $X = 0$ , the weighted sum of inputs is  $wX = 2 \times 0 = 0$ . Compared with the threshold 1, the weighted sum is less. According to the model function Eq. 6.2, when the weighted sum is less than the threshold value, the output  $Y$  will be 0.

When  $X = 1$ , the weighted sum becomes  $wX = 2 \times 1 = 2$ . Compared with the threshold 1, the weighted sum is greater. According to the model function, when the weighted sum is greater than or equal to the threshold value, the output  $Y$  will be 1.

When the weight  $w$  and threshold  $\theta$  are modified to 1 and 2, respectively, as illustrated in Fig. 6.5, the truth table transforms into Table 6.2. When the input is 0, the weighted sum is  $1 \times 0 = 0$ . Compared with the threshold 2, the weighted sum is less. According to the model function, if the weighted sum is less than the threshold, the output  $Y$  will be 0. When the input is 1, the weighted sum is  $1 \times 1 = 1$ , still less than the threshold 2. According to the model function, the output  $Y$  will be 0.

Comparing the neurons in Figs. 6.4 and 6.5, it can be observed that only the weight and threshold values differ. In summary, altering the synaptic strength (weights and threshold) of a neuron leads to a change in the output for the same input.

**Table 6.2** Truth table of the TLU in Fig. 6.5

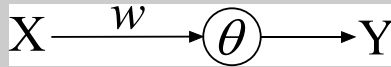
Input	Output
$X$	$Y$
0	0
1	0



**Fig. 6.5** Another TLU with a single input

### • Questions

1. Can a TLU with a single input (Fig. 6.6) produce an output of 1 for both  $X = 0$  and  $X = 1$ ? In other words, can the truth table of a TLU be Table 6.3? If possible, please give an example of the weight  $w$  and threshold  $\theta$  values that would achieve this.



**Fig. 6.6** TLU for questions 1 and 2

**Table 6.3** Truth table for question 1

Input	Output
$X$	$Y$
0	1
1	1

Hint: when  $X = 0$ , the weighted sum is  $w \times 0 = 0$ . According to the model function Eq. 6.2, if  $0 \geq \theta$ , the output  $Y$  will be 1. When  $X = 1$ , the weighted sum is  $w \times 1 = w$ , and if  $w \geq \theta$ , the output  $Y$  will be 1. Therefore, the question can be rephrased as “Are there values of  $w$  and  $\theta$  that can satisfy both  $0 \geq \theta$  and  $w \geq \theta$ ?” If so, please give an example of the values for  $w$  and  $\theta$ .

2. Can a TLU with a single input (Fig. 6.6) produce an output of 1 for  $X = 0$  and an output of 0 for  $X = 1$ ? In other words, can the truth table of a TLU be Table 6.4? If possible, please give an example of the weight  $w$  and threshold  $\theta$  values that would achieve this.

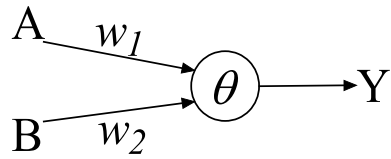
**Table 6.4** Truth table for question 2

Input	Output
$X$	$Y$
0	1
1	0

Hint: Similar to question 1, if there are values of  $w$  and  $\theta$  that satisfy both conditions  $0 \geq \theta$  and  $w < \theta$ , it will be possible to output 1 and 0 for inputs 1 and 0, respectively. Consider if such values exist.

As shown in Figs. 6.5 and 6.6 and the above questions, the output of a neuron can vary for the same input if the weights and threshold values change. This ability

**Fig. 6.7** A TLU with the same number of inputs as logic gate OR



**Table 6.5** Truth table of the logic OR operation

Input		Output
A	B	T
0	0	0
0	1	1
1	0	1
1	1	1

to produce different outputs for the same input is a fundamental feature of neural network. This feature can be used to implement various functions.

Next, we provide an example of using a TLU to imitate the logic gate OR, one of the basic logic gates introduced in Sect. 1.7. This means we will demonstrate how to configure the weights and threshold of a TLU to produce the same results as the logic OR operation.

First, create a TLU with two inputs as shown in Fig. 6.7, because the logic gate OR has two inputs.

Second, set the weights  $w_1$ ,  $w_2$ , and threshold  $\theta$  of the TLU to match the output of logic gate OR. The value-setting process includes three steps.

Step 1: create the truth table for the inputs and the correct outputs of the logic OR operation, as presented in Table 6.5. In this table, the inputs are A and B, and the output is T. When at least one input is one, the output will be one. Only when both inputs are zero will the output be zero.

Step 2: create a constraint for each of the four input patterns. For the input pattern  $A = 0$  and  $B = 0$ , the weighted sum of inputs is  $w_1 \times 0 + w_2 \times 0$ . According to the model function Eq. 6.2, if this sum is less than  $\theta$ , the output will be 0, matching the correct output (T column in Table 6.5). Therefore, the constraint is  $w_1 \times 0 + w_2 \times 0 < \theta$ . In other words, we need to set the values of  $w_1$ ,  $w_2$ , and  $\theta$  to satisfy  $w_1 \times 0 + w_2 \times 1 < \theta$  so that the TLU's output aligns with the logic OR output for this input pattern.

Similarly, for the input pattern  $A = 0$  and  $B = 1$ , the weighted sum of inputs is  $w_1 \times 0 + w_2 \times 1$ . According to the model function, if this sum is greater than or equal to  $\theta$ , the output will be 1, matching the correct output. Therefore, the constraint is  $w_1 \times 0 + w_2 \times 1 \geq \theta$ . In other words, we need to set the values of  $w_1$ ,  $w_2$ , and  $\theta$  to satisfy  $w_1 \times 0 + w_2 \times 1 \geq \theta$  so that the TLU's output aligns with the logic OR output for this input pattern.

In summary, when the correct output is 0, the weighted sum should be less than  $\theta$ , and when the correct output is 1, the weighted sum should be greater than or equal

to  $\theta$ . The resulting four constraints are shown in Eqs. 6.3–6.6. Equations 6.7–6.10 are the calculation results, such as  $w_1 \times 0 + w_2 \times 0 = 0$  and  $w_1 \times 0 + w_2 \times 1 = w_2$ .

$$w_1 \times 0 + w_2 \times 0 < \theta \quad (6.3)$$

$$w_1 \times 0 + w_2 \times 1 \geq \theta \quad (6.4)$$

$$w_1 \times 1 + w_2 \times 0 \geq \theta \quad (6.5)$$

$$w_1 \times 1 + w_2 \times 1 \geq \theta \quad (6.6)$$

$$0 < \theta \quad (6.7)$$

$$w_2 \geq \theta \quad (6.8)$$

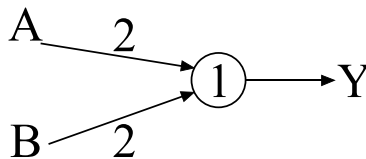
$$w_1 \geq \theta \quad (6.9)$$

$$w_1 + w_2 \geq \theta \quad (6.10)$$

Step 3: determine the values for  $w_i$  and  $\theta$  that satisfy all the constraints. Since both  $w_i$  and  $\theta$  are real numbers, numerous combinations can satisfy the four constraints. For example,  $w_1 = 2, w_2 = 2, \theta = 1$ ;  $w_1 = 2.4, w_2 = 1.7, \theta = 0.6$ ; and  $w_1 = 15.7, w_2 = 159, \theta = 0.02$ .

After setting values for the weights and threshold, it is recommended to verify the TLU's actual output. For example, for a TLU with  $w_1 = 2, w_2 = 2$ , and  $\theta = 1$  (as shown in Fig. 6.8), you can calculate its actual output using Eqs. 6.11–6.14. When  $A = 0$  and  $B = 0$ , the weighted sum of inputs is  $0 \times 2 + 0 \times 2 = 0$ , which is less than the threshold 1, resulting in an output of 0. If all four actual outputs match the values in the  $T$  (target) column in Table 6.5, it is confirmed that the weights and threshold have been set correctly.

$$A = 0, B = 0 \rightarrow 0 \times 2 + 0 \times 2 = 0 < 1 \rightarrow Y = 0 \quad (6.11)$$



**Fig. 6.8** A TLU with  $w_1 = 2, w_2 = 2$ , and  $\theta = 1$



$$A = 0, B = 1 \rightarrow 0 \times 2 + 1 \times 2 = 2 \geq 1 \rightarrow Y = 1 \quad (6.12)$$

$$A = 1, B = 0 \rightarrow 1 \times 2 + 0 \times 2 = 2 \geq 1 \rightarrow Y = 1 \quad (6.13)$$

$$A = 1, B = 1 \rightarrow 1 \times 2 + 1 \times 2 = 4 \geq 1 \rightarrow Y = 1 \quad (6.14)$$

Besides the logic OR operation, a single TLU can be employed to implement various logic operations, including the logic AND and combinations of logic circuits. Several examples will be given in the exercises.

### 6.3.3 Learning of TLU

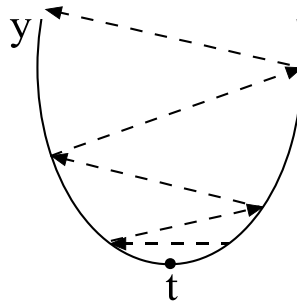
The purpose of neural network learning is to enable a computer to automatically set the weights and thresholds for each neuron. For a TLU, the values to be set are  $w_1$  to  $w_n$  and  $\theta$ . During TLU learning, the input vector is  $(x_1, x_2, \dots, x_n)$ , the correct output is  $t$  (target), and the actual output is  $y$ . If  $y \neq t$ , the weight vector  $(w_1, w_2, \dots, w_n)$  and  $\theta$  will be updated as follows [4].

$$\forall i \in \{1, \dots, n\} : \begin{cases} \theta^{(new)} = \theta^{(old)} + \Delta\theta, & \Delta\theta = -\eta(t - y) \\ w_i^{(new)} = w_i^{(old)} + \Delta w_i, & \Delta w_i = \eta(t - y)x_i \end{cases} \quad (6.15)$$

If the actual output  $y$  differs from the correct output  $t$ ,  $\Delta\theta$  will be calculated as the difference between  $y$  and  $t$  multiplied by a learning coefficient  $\eta$ . The new  $\theta$  value is obtained by adding  $\Delta\theta$  to the old  $\theta$  value. Similarly, let  $\Delta w_i = \eta(t - y)x_i$ , the new  $w_i$  value is obtained by adding  $\Delta w_i$  to the old  $w_i$  value. These updates to  $\theta$  and  $w_i$  will continue until  $y = t$ .

The learning coefficient  $\eta$  is in the range of  $(0, 1)$ . When  $\eta$  approaches 0, the updates have smaller magnitudes, resulting in slower learning. Conversely, as  $\eta$  approaches 1, the updates become larger, potentially leading to faster learning. However, it is essential to note that increasing  $\eta$  indiscriminately does not necessarily accelerate learning. This is because overly large updates may hinder convergence of  $y$  toward the target  $t$ , especially when dealing with complex functions.

To illustrate this, consider the case where  $y$  is represented by a curve depicted in Fig. 6.9, and  $t$  is its lowest point. If the learning coefficient is set too high, the model may skip over many values during each iteration, making it challenging to reach  $t$  accurately. Therefore, selecting an appropriate value for  $\eta$  is crucial and should strike a balance between rapid learning and convergence to the desired output.



**Fig. 6.9** An example of radical updates of  $y$

### • Questions

3. Which type of learning does the learning of a TLU belong to?

- A. Supervised learning
- B. Unsupervised learning

Hint: consider if the learning of a TLU needs the correct output or not.

The learning process of a TLU will update the weights ( $w_1, w_2, \dots, w_n$ ) and threshold  $\theta$  until the actual output  $y$  becomes the same as the correct output  $t$ . It is important to note that not all problems can be effectively solved using a single TLU. For example, in cases where there are no combinations of weights  $w_i$  and threshold  $\theta$  values that can satisfy all the constraints, such as those illustrated in Eqs. 6.16–6.19, the TLU's learning process may never converge to a solution.

$$0 < \theta \tag{6.16}$$

$$w_2 \geq \theta \tag{6.17}$$

$$w_1 \geq \theta \tag{6.18}$$

$$w_1 + w_2 < \theta \tag{6.19}$$

To address such problems, it becomes necessary to use a neural network with multiple neurons, and even hidden layers. In the upcoming section, we will provide a brief introduction to deep learning, which is a type of neural network characterized by its multilayered architecture.

### 6.3.4 Deep Learning

Deep learning is a problem-solving approach that uses large-scale hierarchical neural networks with multiple hidden layers.

Figure 6.10 illustrates the architecture of a deep learning neural network. A distinguishing characteristic of deep learning is that it is performed on neural networks with multiple (more than one) hidden layers. In other words, neural networks with zero (e.g. a TLU) or one hidden layer are commonly referred to as shallow networks rather than deep learning networks. Deep learning neural networks, as opposed to those with fewer hidden layers, exhibit superior capabilities in achieving higher accuracy in tasks like image and pattern recognition.

Deep learning offers several advantages, primarily stemming from its ability to handle complex data processing tasks when a sufficient amount of training data is available. With an ample dataset for training, a well-trained neural network can tackle tasks that are often too intricate for traditional machine learning algorithms. Furthermore, the accuracy of the trained neural network tends to increase proportionally with the quantity of training data.

On the downside, deep learning has its limitations. When the amount of training data is insufficient or when access to high-performance computing resources is lacking, deep learning may struggle to achieve high performance or become infeasible.

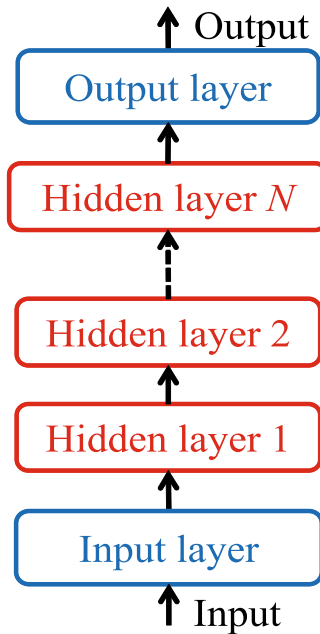


Fig. 6.10 Structure of a deep learning neural network

The most commonly used deep learning neural networks are Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).

CNNs are forward-propagating neural networks that excel at extracting local information and exhibiting location universality. Two-dimensional CNNs share structural similarities with neurons in the human visual cortex, which makes them particularly adept at pattern recognition. Figure 6.11 illustrates the architecture of a CNN used for image recognition.

The left part of a CNN, known as the feature extraction component, takes input images of various sizes and extracts features from them. This is performed using filters (also known as kernels) that are smaller than the input image. The filters act like windows, examining the image part by part as they traverse over the image (as shown in Fig. 6.12). On the right side, the classification component uses the extracted features to make predictions or determine the content of the image, outputting the final result. If you are interested in delving deeper into image recognition using CNNs, please refer to the reference [5].

RNNs are bidirectional neural networks that incorporate a recursive structure in its middle layers, where some of the outputs are used as inputs, as illustrated in Fig. 6.13. RNNs are well-suited for managing variable-length data, including audio

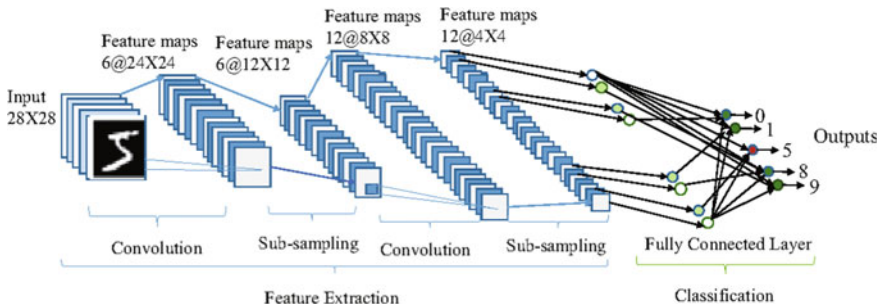


Fig. 6.11 Structure of a CNN used for image recognition

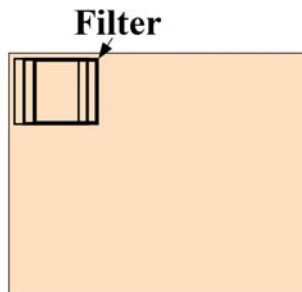


Fig. 6.12 A filter moving over an image

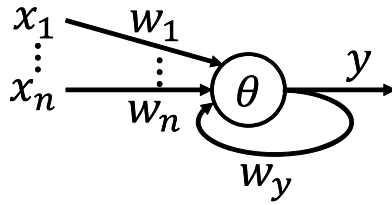


Fig. 6.13 Bidirectional propagation in an RNN

and video, and they excel in tasks such as speech recognition, video recognition, and natural language processing.

**Exercises**

1. Use a TLU to implement the logic calculation  $X_1 \text{ AND } X_2 \text{ AND } X_3$ .

**• Guidance**

Figure 6.14 illustrates the TLU for this problem. You need to specify the values for  $w_1$ ,  $w_2$ ,  $w_3$ , and  $\theta$ .

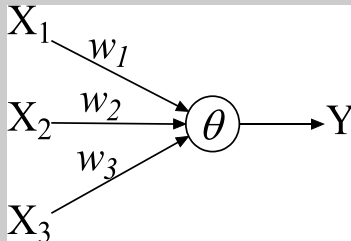


Fig. 6.14 A TLU with three inputs

First, create the truth table for the logic operation  $X_1 \text{ AND } X_2 \text{ AND } X_3$ . The truth table for the logic AND operation was previously provided and explained in Chap. 1. In case you need a refresher, please review it.

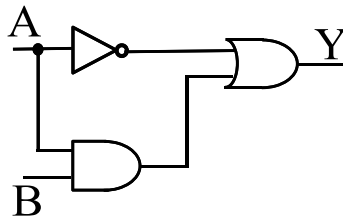
Table 6.6 is an example of a truth table with three inputs,  $X_1$ ,  $X_2$ , and  $X_3$ . Each input has a value of either 0 or 1, resulting in a total of  $2^3 = 8$  possible combinations. Please fill in the empty cells with the appropriate values.

**Table 6.6** Truth table to complete for exercise problem 1

Input			Output
$X_1$	$X_2$	$X_3$	T

Once you have completed the truth table, proceed to derive a constraint based on the information in each row, following the examples provided in Sect. 6.3.2. Finally, set the values of  $w_1$ ,  $w_2$ ,  $w_3$ , and  $\theta$  to ensure that all the eight constraints are satisfied.

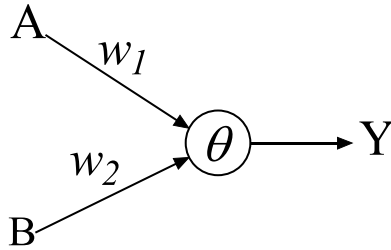
- 2. Use a TLU to produce the output of the circuit shown in Fig. 6.15.



**Fig. 6.15** Circuit for exercise problem 2

**• Guidance**

Similar to problem 1, start by creating the truth table for the circuit. This table should have two inputs, A and B, and one output. Since both A and B can be either 0 or 1, the table will consist of  $2^2 = 4$  rows.



**Fig. 6.16** A TLU for exercise problem 2

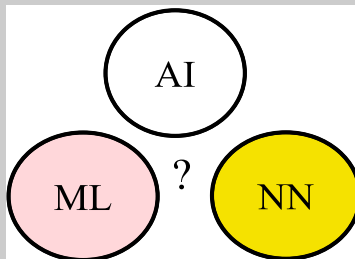
After completing the truth table, create a constraint based on the information in each row. Finally, set the values of the weights and threshold in such a way that all the four constraints are satisfied.

If you need a visual representation of the TLU for this circuit, refer to Fig. 6.16. It depicts a TLU with inputs A and B, and only one neuron is enough. You should specify the values of  $w_1$ ,  $w_2$ , and  $\theta$ . Since a single neuron can produce the output for this circuit, there is no need to create additional neurons.

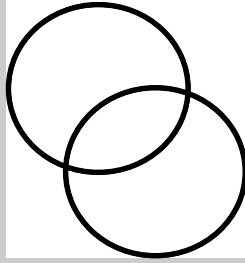
3. According to the content of this textbook, what is the relation between the three fields: artificial intelligence (AI), machine learning (ML), and neural network (NN)?

• **Guidance**

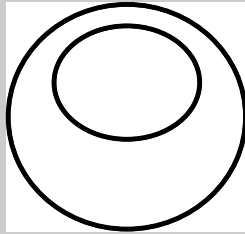
If we use three circles to represent the three fields, as shown in Fig. 6.17, then Fig. 6.18 indicates partial overlap between two fields, and Fig. 6.19 indicates one field entirely encompassing another. You can illustrate the relation between the three fields using such visual expressions.



**Fig. 6.17** Diagram for exercise problem 3



**Fig. 6.18** Expression of partial overlap between two fields



**Fig. 6.19** Expression of one field encompassing another

- \*4. Use a TLU to implement the logic operation  $X_1$  AND ( $X_2$  OR  $X_3$ ).
- \*5. Please explain why a TLU cannot produce the output shown in Table 6.7.

**Table 6.7** Truth table for exercise problem 5

Input			Output
$X_1$	$X_2$	$X_3$	T
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0



## References

1. Kruse R, Borgelt C, Braune C, Mostaghim S (2016) Introduction to neural networks. In: Computational intelligence. Texts in computer science. Springer, London. [https://doi.org/10.1007/978-1-4471-7296-3\\_2](https://doi.org/10.1007/978-1-4471-7296-3_2)
2. Orr G, Schraudolph N, Cummins F (1999) Lecture notes of neural networks. <https://willamette.edu/~gorr/classes/cs449/intro.html>
3. Blais A, Mertz D (2001) An introduction to neural networks—pattern learning with back propagation algorithm. Gnosis Software, Inc. <https://www.sci.brooklyn.cuny.edu/~sklar/teaching/s06/ai/papers/nn-intro.pdf>
4. Stergiou C, Siganos D (2011) Neural networks. [https://wiki.eecs.yorku.ca/course\\_archive/2011-12/F/4403/\\_media/report.pdf](https://wiki.eecs.yorku.ca/course_archive/2011-12/F/4403/_media/report.pdf)
5. Ujjwalkarn (2016) An intuitive explanation of convolutional neural networks. <https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/>