# 1. Table of contents

2. Chapter 2 .... 32

2.1 Introduction .... 32

2.2 Machine learning .... 32

2.2.1 Types of machine learning .... 32

2.2.1.1 Supervised learning .... 33

2.2.1.2 Unsupervised learning .... 33

2.2.1.3 Reinforcement .... 33

2.2.2 Machine learning algorithms .... 34

2.2.2.1 Linear Regression .... 34

2.2.2.2 Logistic Regression .... 35

2.2.2.3 Support vector machines .... 35

2.2.2.4 K-Means .... 36

2.2.3 Machine learning applications .... 37

2.3 Deep learning .... 37

2.3.1 Artificial neural networks .... 38

2.3.2 Activation functions .... 40

2.3.3 Deep learning architectures .... 42

2.3.3.1 Recurrent Neural Networks .... 43

2.3.3.2 Long Short-Term Memory Networks .... 44

2.3.3.3 Gated Recurrent Units .... 45

2.3.3.4 Transformers .... 46

2.3.4 Deep learning applications .... 49

2.4 Conclusion ........................................ 49

3. Bibliographie ........................................ 50

## List of figures

Figure 2.1 Graphical representation of linear regression. ........................................ 34

Figure 2.2 Graphical representation of logistic regression. ........................................ 35

Figure 2.3 Graphical representation of Support vector machines. ........................................ 36

Figure 2.4 Graphical representation of k means. ........................................ 36

Figure 2.5 Typical biological-inspired neuron. ........................................ 38

Figure 2.6 Schematic representation of a neural network. ........................................ 39

Figure 2.7 Sigmoid activation function. ........................................ 41

Figure 2.8 ReLU activation function. ........................................ 41

Figure 2.9 Tanh activation function. ........................................ 42

Figure 2.10 Diagram of simple recurrent network. ........................................ 43

Figure 2.11 Long Short-term Memory Neural Network. ........................................ 44

Figure 2.12 Gated Recurrent Unit. ........................................ 45

Figure 2.13 Architecture of transformers [19]. ........................................ 46

# 2. Chapter 2

# Deep Learning

## 2.1 Introduction

Artificial intelligence (AI) is a field of computer science that aims to create intelligent systems capable of performing tasks that typically require human intelligence. From self-driving cars to voice assistants, AI has made remarkable strides, transforming the way we live, work, and interact with technology.

In this chapter, we will explore common methods of machine learning, including supervised, unsupervised, and reinforcement learning. We will also discuss popular algorithmic approaches in machine learning, moving to explore deep learning and its architectures. Our focus will be on understanding the fundamental concepts behind these methods and algorithms.

## 2.2 Machine learning

The field of machine learning is concerned with the question of how to construct computer programs that automatically improve with experience. In recent years many successful machine learning applications have been developed, ranging from data-mining programs that learn to detect fraudulent credit card transactions to information-filtering systems that learn users' reading preferences, to autonomous vehicles that learn to drive on public highways. At the same time, there have been important advances in the theory and algorithms that form the foundations of this field [8].

### 2.2.1 Types of machine learning

Machine learning can be categorized into three main types: supervised learning, unsupervised learning, and reinforcement learning.

#### 2.2.1.1 Supervised learning

Supervised learning involves training machine learning models on labeled data. The model learns to make predictions or classify new data by finding patterns and relationships between the input features and the desired outputs. Examples of supervised learning algorithms include linear regression, decision trees, and support vector machines (SVMs).

Supervised learning addresses two types of problems: classification problems and regression problems.

**Classification** is a type of supervised learning problem where the goal is to predict a discrete class label for each data point. For example, the goal might be to predict whether an image contains a cat or a dog, or whether a patient has cancer or not. Classification problems can be solved using a variety of machine learning algorithms, including decision trees and support vector machines.

**Regression** is a type of supervised learning problem where the goal is to predict a continuous value for each data point. For example, the goal might be to predict the price of a house or the height of a person. Regression problems can be solved using a variety of machine learning algorithms, including linear regression and polynomial regression.

#### 2.2.1.2 Unsupervised learning

Unsupervised learning involves training models on unlabeled data. The algorithms learn to identify patterns, structures, and relationships in the data without explicit guidance. Common tasks in unsupervised learning include clustering, where similar data points are grouped together, and dimensionality reduction, which aims to reduce the complexity of data while preserving important information. Examples of unsupervised learning algorithms include k-means clustering, hierarchical clustering, and principal component analysis (PCA) [10].

#### 2.2.1.3 Reinforcement

Reinforcement learning involves training agents to interact with an environment and learn optimal behaviors through trial and error. The agent receives feedback in the form of rewards or penalties based on its actions, and its objective is to maximize cumulative rewards over time. Reinforcement learning is commonly used in scenarios where an agent needs to make sequential decisions, such as in game playing, robotics, and autonomous vehicle control. Popular reinforcement learning algorithms include Q-learning, policy gradients, and deep Q-networks (DQNs).

It's worth noting that these types of machine learning are not mutually exclusive, and hybrid approaches that combine multiple types can be used to tackle complex problems. Additionally, there are other specialized areas within machine learning, such as semi-supervised learning and transfer learning, which further expand the capabilities of machine learning algorithms [11].

## 2.2.2 Machine learning algorithms

Machine learning algorithms can be thought of as powerful tools that allow us to unlock the potential of data. By applying these algorithms to various problems, we can uncover hidden patterns, detect anomalies, classify data into different categories, and even make accurate predictions about future outcomes.

### 2.2.2.1 Linear Regression

Linear regression is a machine learning algorithm that uses a line to predict a numerical value based on input variables. As shown in Figure 2.1 the line is found by minimizing the difference between the predicted and actual values. Linear regression is simple to understand and interpret, and it is commonly used in a variety of domains [9].

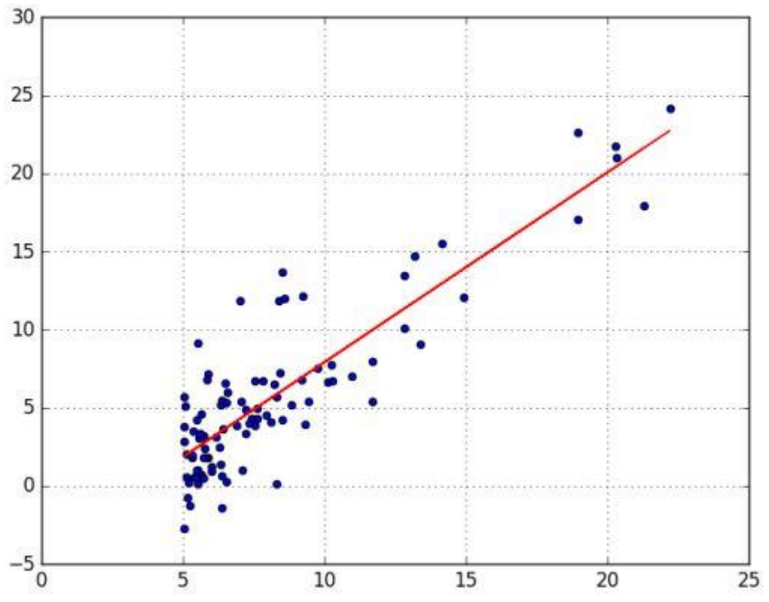

**Figure 2.1** Graphical representation of linear regression.

#### 2.2.2.2 Logistic Regression

Logistic regression is a machine-learning algorithm that predicts the probability of an instance belonging to a specific class. It is a supervised learning method that uses the logistic function to model the relationship between the input variables and the probability of the binary outcome as shown in Figure 2.2. Logistic regression is commonly used in a variety of domains, including customer behavior analysis, fraud detection, and medical diagnosis. Its advantages lie in its simplicity, interpretability, and ability to handle linear and nonlinear relationships between variables [9].

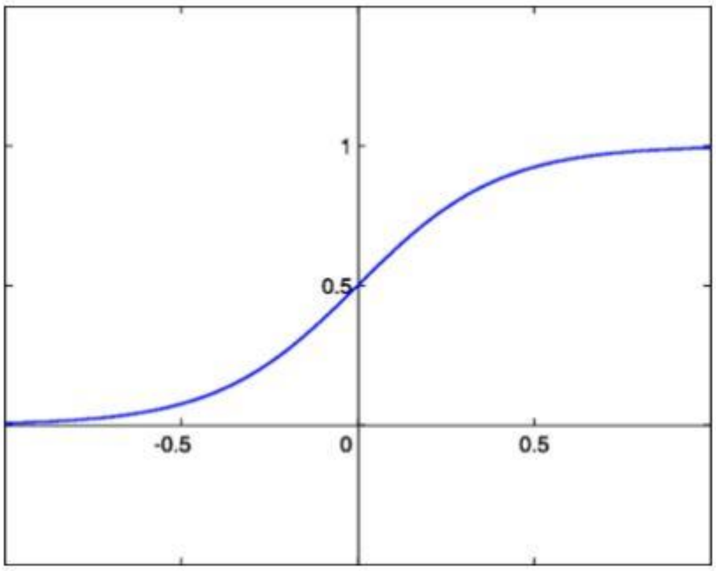

**Figure 2.2** Graphical representation of logistic regression.

#### 2.2.2.3 Support vector machines

Support vector machines (SVMs) are a machine learning algorithm that can be used for both classification and regression tasks. As shown in Figure 2.3 SVMs work by finding the optimal hyperplane that separates data points of different classes with the maximum margin. This means that the hyperplane is as far away as possible from any data points on either side. SVMs can handle both linearly separable and non-linearly separable data by using kernel functions to implicitly map the data into a higher-dimensional space.

Their key strengths lie in their ability to handle high-dimensional data and flexibility in choosing different kernel functions [9].

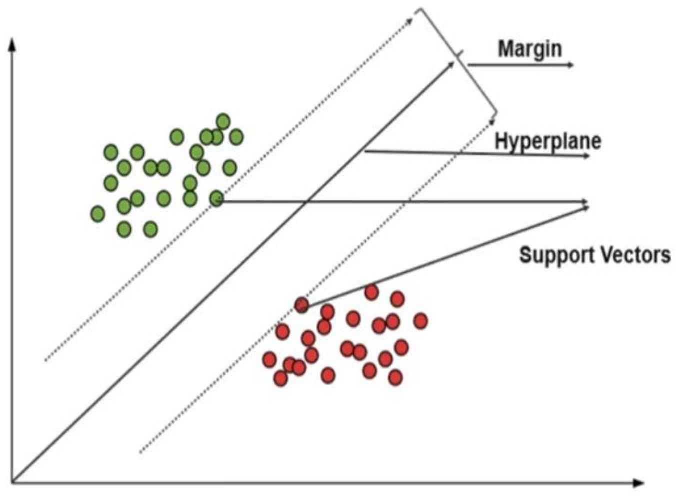

**Figure 2.3** Graphical representation of Support vector machines.

### 2.2.2.4 K-Means

K-means is an unsupervised machine learning algorithm that groups data points into clusters based on their similarities. The algorithm works by iteratively assigning data points to the cluster with the nearest centroid, and then updating the centroids based on the mean of the assigned points as shown in Figure 2.4. The number of clusters, K, is predefined by the user. K-means is a simple and computationally efficient algorithm, making it widely used for clustering tasks.

K-means finds applications in various domains, including customer segmentation, image compression, and document clustering.

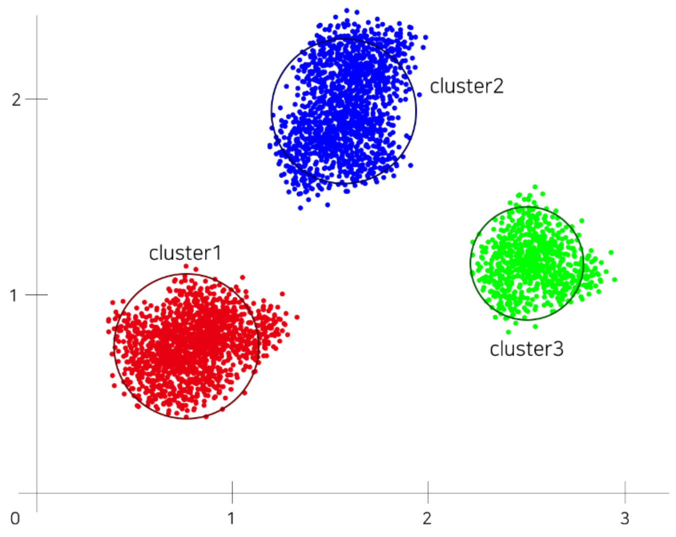

**Figure 2.4** Graphical representation of k means.

### 2.2.3 Machine learning applications

Machine learning is a rapidly evolving field with a wide range of applications. It has been used to improve healthcare, transportation, finance, retail, energy, and agriculture.

**Healthcare and Medicine:** In healthcare, machine learning is used to diagnose diseases, recommend treatments, and personalize care.

**Automotive Industry:** In transportation, machine learning is used to develop autonomous vehicles and improve traffic management.

**Financial Services:** In finance, machine learning is used to detect fraud, assess risk, and make investment decisions.

**Energy and Utilities:** In energy, machine learning is used to optimize energy consumption, predict outages, and improve grid reliability.

**Agriculture:** In agriculture, machine learning is used to predict crop yields, detect pests, and optimize irrigation.

These are just some of the industries that machine learning had been applied to, machine learning is a powerful tool that has the potential to transform many industries.

## 2.3 Deep learning

Deep learning (DL) is a subset of machine learning (ML) that uses multilayer artificial neural networks to model and solve complex problems. These networks are designed to simulate the behavior of neurons in the human brain, enabling them to process and learn from large amounts of unstructured data.

Deep learning algorithms automatically learn to recognize patterns and features in data by analyzing and adjusting the weights and biases of network interconnected nodes. This process, called training, involves optimizing network parameters to minimize errors and improve accuracy.

Deep learning has grown in popularity in recent years due to its ability to process large and diverse datasets, achieve cutting-edge performance in many fields, and achieve breakthroughs in areas such as computer vision, natural language processing, and speech recognition [12].

### 2.3.1 Artificial neural networks

Typical artificial neural networks (ANN) are biologically inspired computer programs inspired by the workings of the human brain. These ANNs are referred to as networks because they are made up of several functions, which collect knowledge by recognizing links and patterns in data using previous experiences referred to as training examples in most literature. The learned patterns in data are adjusted by an appropriate activation function and displayed as the neuron's output.

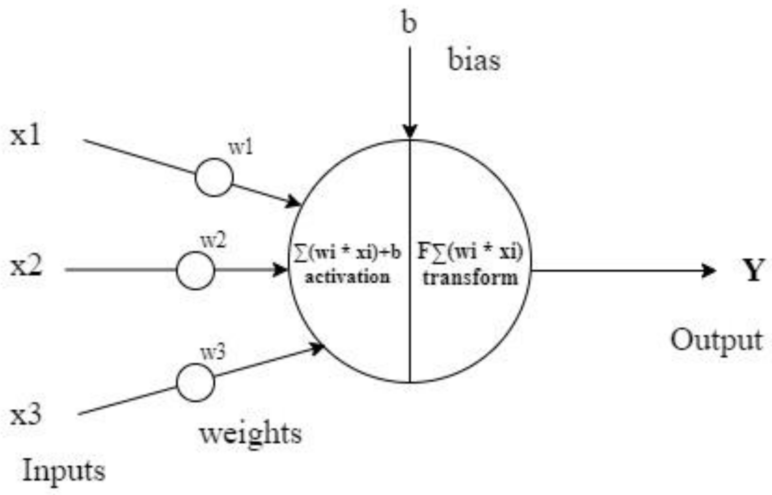

**Figure 2.5** Typical biological-inspired neuron.

The Figure 2.5 Typical biological-inspired neuron. represents an artificial neural network (ANN) with three inputs, denoted as x1, x2, and x3. These inputs are connected to an activation function represented by a circle, which plays a crucial role in determining the output of the neural network. The activation function takes the weighted sum of the inputs and applies a non-linear transformation to produce the final output.

The weights w1, w2, and w3 represent the respective strengths or importance assigned to each input. These weights are multiplied with their corresponding inputs and then summed up. The purpose of the weights is to control the impact of each input on the output of the network. By adjusting the values of these weights during the learning process, the neural network can learn to make accurate predictions or classifications based on the given inputs. a bias is an additional parameter that is added to the weighted sum of inputs before passing through the activation function. Bias allows the neural network to make adjustments to the output even when all the input values are zero.

The activation function serves as a decision-making element. It takes the weighted sum of the inputs and applies a specific mathematical function to determine the output of the neural network.

Finally, the output of the activation function represents the result or prediction of the artificial neural network. It can be used for various tasks, such as classification, regression, or pattern recognition, depending on the problem being addressed. The goal of training the neural network is to adjust the weights and biases in such a way that it produces accurate outputs for a given set of inputs.

ANNs are generally composed of many layers, including an input layer, one or more hidden layers, and an output layer. ANNs can catch complicated patterns and generate accurate predictions because of their layer-wise structuring.

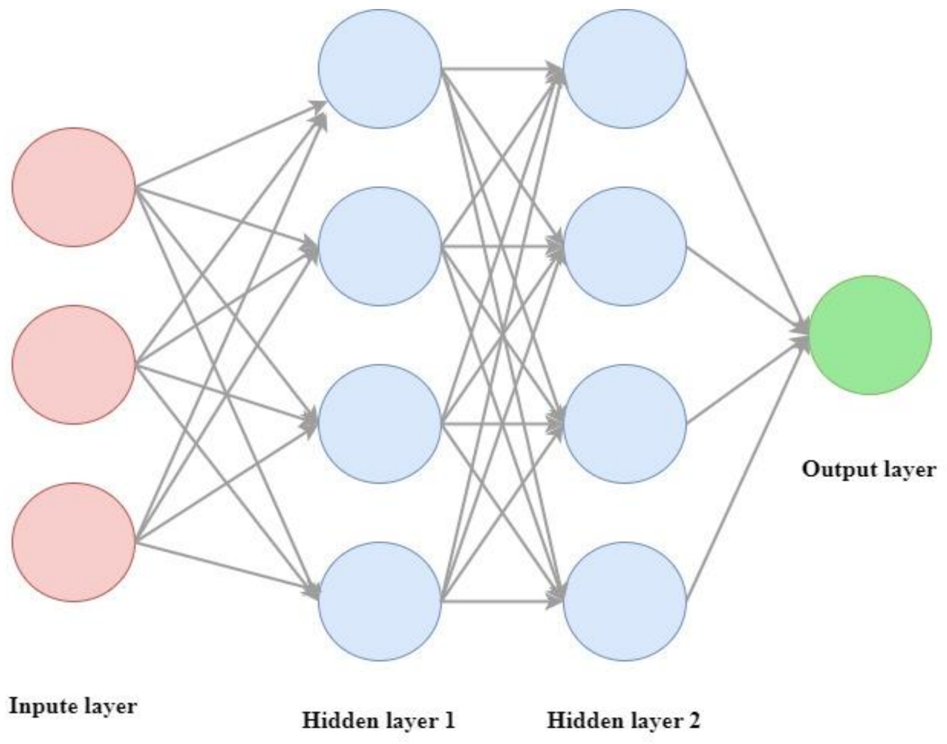

**Figure 2.6** Schematic representation of a neural network.

The **input layer** serves as the entry point for the neural network, receiving the initial input values. Each input node represents a feature or attribute of the problem being solved. For example, in an image recognition task, each input node could represent a pixel value. The input layer simply transmits these values to the next layer.

**Hidden layers**, as the name suggests, are the intermediate layers between the input and output layers, where the processing and transformation of information occur. These layers are called "hidden" because their computations are not directly observable from the outside. Each hidden layer consists of multiple artificial neurons (also called nodes or units) that receive inputs from the previous layer, perform calculations, and pass the results to the next layer. The number of hidden layers and the number of neurons in each hidden layer can vary depending on the complexity of the problem and the desired network architecture. Additional hidden layers and neurons allow the network to learn more intricate representations and potentially improve its performance. Deep neural networks, which have multiple hidden layers, have been successful in solving tasks such as image recognition, natural language processing, and speech recognition.

Finally, we have the **output layer**, which produces the final result or prediction of the neural network. The number of nodes in the output layer corresponds to the number of possible outputs or classes in the problem. For instance, in a binary classification task, there would be two output nodes representing the two possible classes. In a regression task, the output layer might consist of a single node that produces a continuous value.

The connections between the layers, represented by **weights**, determine the strength and influence of information flowing through the network. During the training phase, these weights are adjusted through a process called backpropagation, which involves propagating the error from the output layer back to the hidden layers and adjusting the weights accordingly. This iterative learning process allows the neural network to gradually optimize its performance and improve its ability to make accurate predictions.

### 2.3.2 Activation functions

Activation functions are an essential component of neural networks as they introduce non-linearity, allowing the network to learn and approximate complex patterns in the data. Here are some commonly used activation functions:

**Sigmoid:** The sigmoid function is a smooth S-shaped curve that maps the input to a value between 0 and 1. It is often used in binary classification problems or as an output activation function in models that require probability-like outputs. Mathematically sigmoid is represented as:

$$\mathrm{f(t)} = \frac{1}{(1 + \mathrm{e}^{-\mathrm{t}})}$$

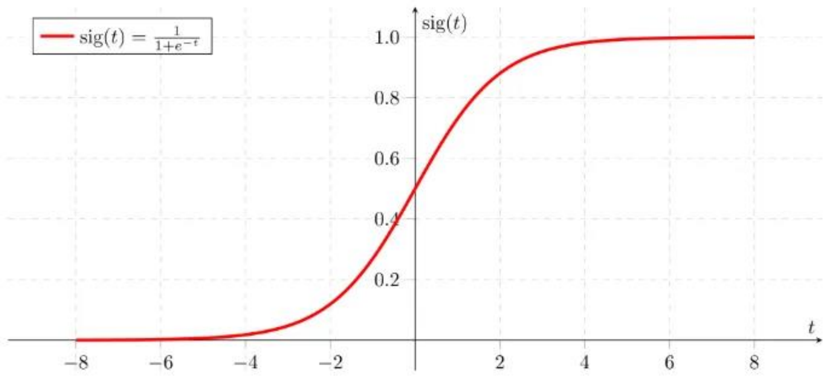

**Figure 2.7** Sigmoid activation function.

**Softmax:** The softmax is a more generalized form of the sigmoid, it is a specialized activation function used in multi-class classification problems. It takes a vector of real numbers as input and normalizes them into a probability distribution over multiple classes, where the sum of the probabilities is 1. for each element zi of the input vector z, mathematically Softmax is represented as:

$$\text{f(x)} = \frac{\exp\ (xi)}{\Sigma \exp\ (\text{xj})}$$

**ReLU (Rectified Linear Unit):** ReLU is a piecewise linear function that returns the input as if it is positive, and 0 otherwise. It is widely used in hidden layers of deep neural networks due to its simplicity and computational efficiency.  Mathematically ReLU is represented as:

$$f(x)\ =\ max(0, x)$$

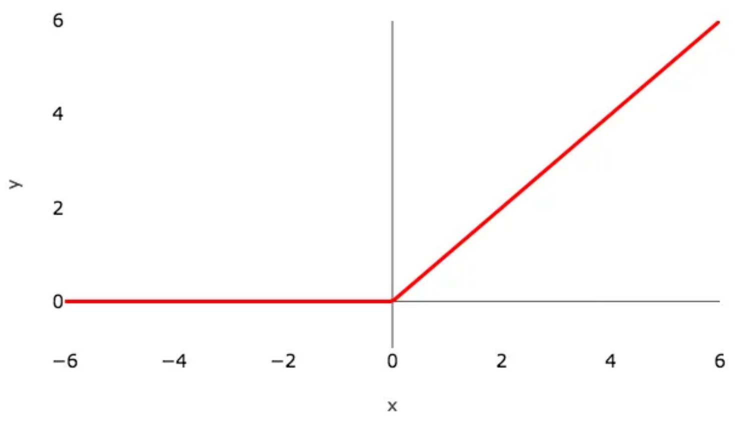

**Figure 2.8** ReLU activation function.

**Tanh (Hyperbolic Tangent):** The hyperbolic tangent function is similar to the sigmoid function but maps the input to a value between -1 and 1. It provides stronger non-linearity than the sigmoid function and is often used in recurrent neural networks (RNNs) and convolutional neural networks (CNNs). Mathematically Tanh is represented as:

$$f(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$$

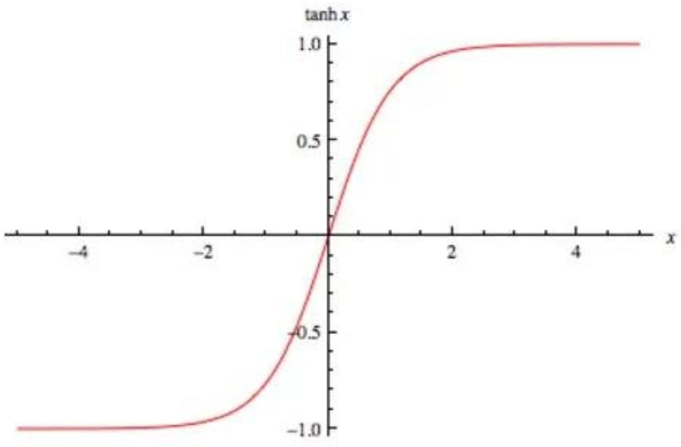

**Figure 2.9** Tanh activation function.

These activation functions serve different purposes and may be more suitable for specific tasks or network architectures. Choosing the appropriate activation function depends on the nature of the problem and the desired behavior of the neural network.

### 2.3.3 Deep learning architectures

Deep Learning is a growing field with applications that span across several use cases. In each use case, a different architecture is predominant and gives the best efficiency.

There are many different types of deep learning architectures, many of which are derived from original architectures. Some of the most popular ones are Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memory Networks (LSTMs).

For the purposes of this discussion, we will talk about Recurrent Neural Networks, Long Short-Term Memory Networks, Gated Recurrent Units (GRU) and focus on one type of architecture known as transformers, which have gained popularity in recent years for their ability to process sequential data with parallelization and attention mechanisms.

#### 2.3.3.1 Recurrent Neural Networks

Recurrent neural networks (RNNs) are a type of neural network that is particularly effective in processing sequential data. Unlike traditional neural networks, which process input data independently, RNNs have a feedback mechanism that allows them to retain and utilize information from previous steps in the sequence. This makes them well-suited for tasks such as natural language processing, speech recognition and machine translation [12].

RNNs work by processing each step in the sequence one at a time. At each step, the RNN takes as input the current input data and the output from the previous step. The RNN then uses this information to calculate a new output. This output is then used as input for the next step, and so on [14].

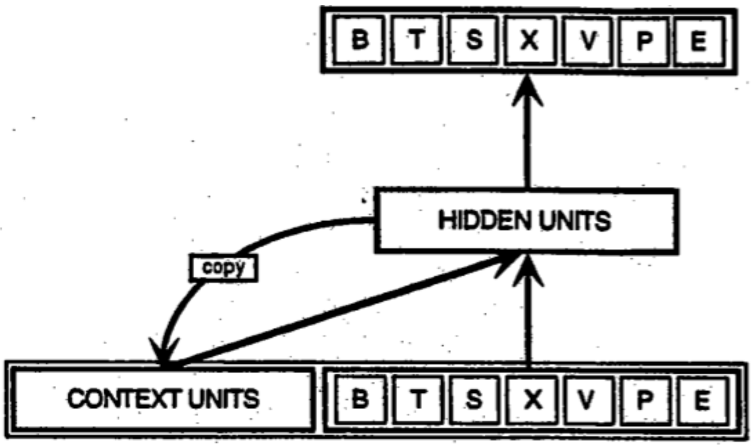

**Figure 2.10** Diagram of simple recurrent network.

The diagram shown in Figure 2.10 Diagram of simple recurrent network. of an early, simple recurrent network, where the BTSXVPE at the bottom of the drawing represents the input example in the current moment, and CONTEXT UNIT represents the output of the previous moment.

Some popular variants of RNNs include LSTMs (Long Short-Term Memory) and GRUs (Gated Recurrent Units). LSTMs and GRUs are designed to address the vanishing gradient problem that can occur in traditional RNNs, which hinders their ability to capture long-term dependencies. LSTMs incorporate memory cells and gating mechanisms that enable them to selectively retain and update information over time. GRUs have gating mechanisms that control the flow of information within the network, allowing them to capture long-term dependencies efficiently [15].

### 2.3.3.2 Long Short-Term Memory Networks

Traditional RNNs suffer from a problem known as the vanishing gradient problem, which can hinder their performance when dealing with long-term dependencies.

The vanishing gradient problem arises during the training of RNNs when the gradients used for learning become extremely small as they propagate backward through time. This occurs because the gradient calculation involves multiplying a series of weight matrices, and the gradient values tend to diminish exponentially with each multiplication. As a result, the network struggles to capture and propagate information from earlier time steps, limiting its ability to model long-term dependencies in the data [16].

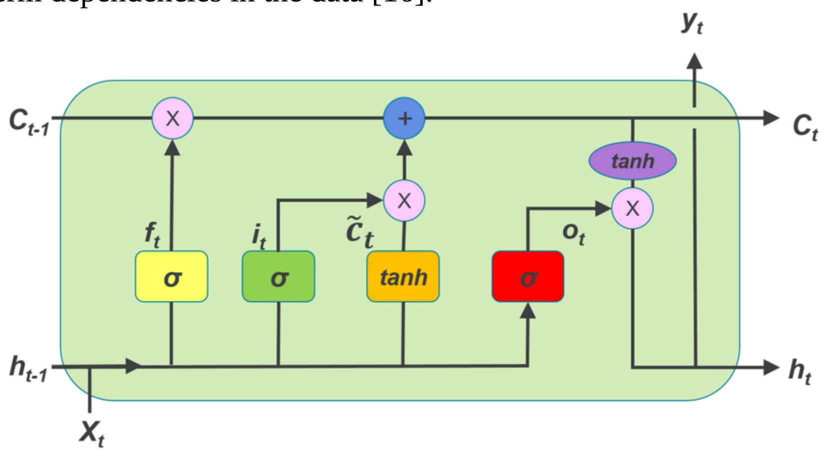

**Figure 2.11** Long Short-term Memory Neural Network.

To address the vanishing gradient problem, the Long Short-Term Memory (LSTM) architecture was introduced. LSTM networks incorporate specialized memory cells that allow for better information retention and controlled information flow.

The key innovation of LSTM is the inclusion of memory cells, which serve as a way to store and access information over long time scales. These memory cells are equipped with three main components: an input gate, a forget gate, and an output gate. These gates regulate the flow of information into and out of the memory cells, allowing them to selectively retain and forget information as needed.

The input gate determines how much new information is stored in the memory cells, while the forget gate controls which information is discarded. The output gate determines the information to be passed on to the next time step. By carefully controlling the flow of information, LSTM networks can effectively address the vanishing gradient problem and capture long-term dependencies in sequential data [17].

The LSTM architecture has been widely successful in various applications, including natural language processing, speech recognition, machine translation, and video analysis. Its ability to model long-term dependencies makes it a crucial component in many state-of-the-art deep learning models.

#### 2.3.3.3 Gated Recurrent Units

The Gated Recurrent Unit (GRU) is a type of recurrent neural network (RNN) architecture that addresses the vanishing gradient problem of traditional RNNs. GRUs were introduced in 2014 by Cho et al. to capture long-term dependencies in sequential data while mitigating the issue of information loss over time.

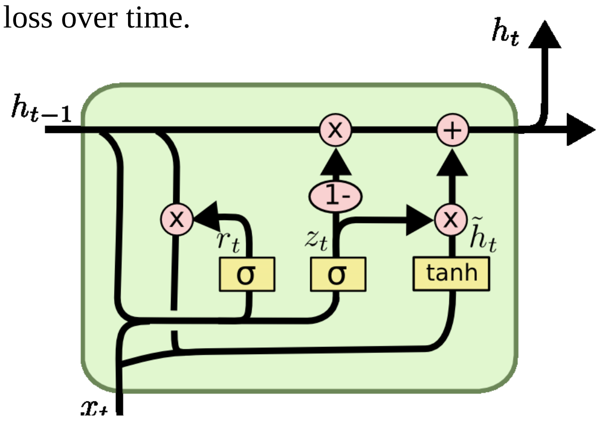

**Figure 2.12** Gated Recurrent Unit.

GRUs consist of two main gates: the reset gate and the update gate. The reset gate determines how much of the previous hidden state should be forgotten, while the update gate determines how much of the new candidate state should be incorporated into the current hidden state. These gates allow GRUs to selectively retain and update information over time, facilitating the learning of complex dependencies in sequential data.

GRUs have several advantages over traditional RNNs and even other gated architectures like LSTMs. They have fewer parameters, making them computationally more efficient and easier to train. Additionally, GRUs have been found to perform competitively or even outperform LSTMs on certain tasks, while requiring less training time and data [18].

Overall, GRUs are a powerful and versatile RNN architecture that offers several advantages over traditional RNNs and LSTMs.

#### 2.3.3.4 Transformers

As we already mentioned that RNNs have several limitations, including difficulty in parallelization and difficulty in capturing long-term dependencies. These limitations have led to the development of alternative models, such as transformers.

The Transformer is a neural network architecture that was proposed by Vaswani in 2017. It is a powerful model that has revolutionized the field of natural language processing (NLP). Transformer models introduced a new approach to sequence modeling without recurrent connections, which makes them more efficient and easier to train than traditional RNN-based models. Transformer models have been widely adopted and have become the state-of-the-art models in various NLP tasks, including machine translation, question answering, and text generation [19].

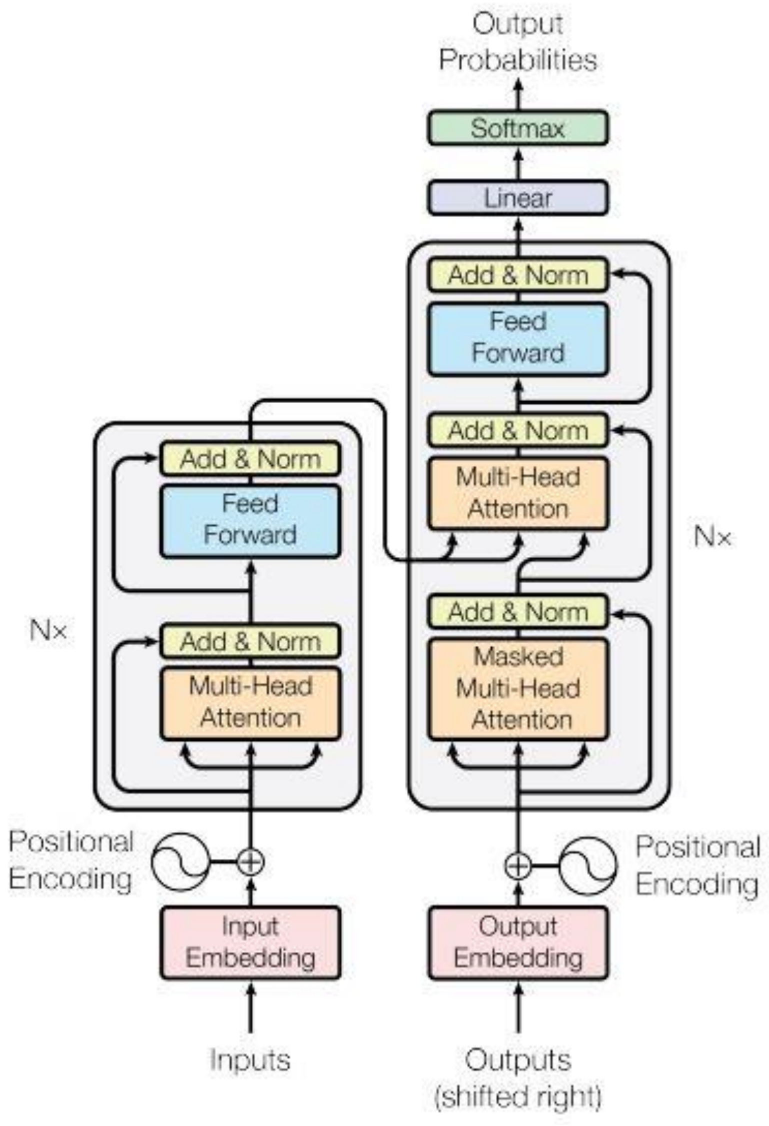

**Figure 2.13** Architecture of transformers [19].

The Transformer architecture adopts an encoder-decoder structure. As shown in **Erreur ! Source du renvoi introuvable.**, the encoder maps an input sequence of symbol representations (x1, ..., xn) to a sequence of continuous representations (z1, ..., zn). The decoder, using the encoded representation z, generates an output sequence (y1, ..., ym) by producing one symbol at a time. This process is auto-regressive, as the model incorporates previously generated symbols as additional input for generating the next symbol. The Transformer architecture follows this overall design, employing stacked self-attention and point-wise, fully connected layers for both the encoder and decoder [19].

**Encoder**

The encoder is made up of a stack of N = 6 identical layers. Each layer has two sub-layers. The first sub-layer is a multi-head self-attention mechanism, and the second is a simple feed-forward network. Each of these two layers is followed by a normalization layer. To facilitate the connections in the model, all sub-layers produce outputs of dimention dmodel = 512.

**Decoder**

The decoder is also made up of a stack of N = 6 identical layers. Each layer has three sub-layers. In addition to the two sub-layers in each encoder the decoder inserts a third sub-layer which is a multi-head attention over the output of the encoder. Similar to the encoder each of the three sub-layers is followed by a normalization layer. In the decoder stack, the self-attention sub-layer is modified to make sure positions don't pay attention to the positions that come after them, using a technique called masking. This, along with the fact that the output embeddings are shifted by one position, ensures that the predictions for a particular position only rely on the already known outputs at positions before it.

**Attention**

An attention function can be defined as a mapping between a query and a set of key-value pairs, producing an output. In this mapping, the query, keys, values, and output are all vectors. The output is computed by calculating a weighted sum of the values, where the weight assigned to each value is determined by a compatibility function between the query and the corresponding key.

### Scaled Dot-Product Attention

To compute the scaled dot-product attention, the query and key vectors are multiplied together to obtain a similarity score for each pair. The dot product represents how well the query aligns with each key. These scores are then scaled by the square root of the dimension of the key vectors to avoid overly large values.

Next, the scaled scores are passed through a softmax function, which normalizes the scores and assigns a weight to each value vector. The softmax function ensures that the weights sum up to 1, making the attention mechanism a proper distribution.

Finally, the values are multiplied by their corresponding weights and summed up to produce the output of the attention mechanism. This output captures the relevant information from the value vectors based on the similarity between the query and key vectors.

### Multi-Head Attention

Instead of performing a single attention function with keys, values, and queries of dmodel dimensions, it has been found beneficial to linearly project the queries, keys, and values h times. Each projection uses different learned linear transformations to convert them into dk, dk, and dv dimensions, respectively. Subsequently, the attention function is applied in parallel to these projected versions of queries, keys, and values, resulting in dv-dimensional output values. These values are then concatenated and projected again, producing the final values.

### Feed-Forward Networks

In addition to the attention sub-layers, each layer in the encoder and decoder includes a fully connected feed-forward network. This network operates on each position independently and uniformly. It comprises two linear transformations with a ReLU activation function in between [13].

### Embeddings and Softmax

Similar to other sequence transduction models, learned embeddings are used to convert input tokens and output tokens into dmodel-dimensional vectors. The decoder output is transformed into predicted probabilities for the next token using a learned linear transformation and softmax function.

**Positional Encoding**

Positional encoding adds position-related information to the input data, allowing the model to understand the order of tokens. It involves incorporating fixed-length vectors into the input embeddings to capture sequential dependencies. This enables transformers to effectively process sequential data for tasks like machine translation and language understanding.

### 2.3.4 Deep learning applications

Deep learning (DL) is a powerful tool that has revolutionized many fields by delivering unprecedented accuracy and efficiency in processing complex data. One of the main strengths of DL is its ability to automatically extract relevant features and patterns from large and diverse datasets without the need for manual feature extraction. This makes deep learning particularly suitable for applications such as image and video processing, natural language understanding, speech recognition, and autonomous decision-making. Some of deep learning applications are:

**Image recognition:** Deep learning models can be used to identify objects in images. This technology is used in a variety of applications, such as facial recognition, self-driving cars, and medical image analysis.

**Natural language processing:** Deep learning models can be used to understand and process human language. This technology is used in a variety of applications, such as speech recognition, machine translation, and chatbots.

**Machine translation:** Deep learning models can be used to translate text from one language to another. This technology is used in a variety of applications, such as online translation services and multilingual software.

**Robotics:** Deep learning models can be used to control robots. This technology is used in a variety of applications, such as self-driving cars and industrial robots.

## 2.4 Conclusion

In conclusion, this chapter has provided an overview of both machine learning and deep learning. We have discussed their fundamental concepts, algorithms, and architectures, shedding light on these powerful techniques in the field of artificial intelligence.

## 3. Bibliographie

[8] Mitchell, T. M, Machine Learning. McGraw-Hill, 1997.

[9] Hastie, T., Tibshirani, R., & Friedman, J, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition, Springer, 2009

[10] Bishop, C. M, Pattern recognition and machine learning, Springer, 2006.

[11] Sutton, R. S., & Barto, A. G, Reinforcement learning: an introduction, 2nd Edition, MIT Press, 2018.

[12] Goodfellow, I., Bengio, Y., & Courville, A, Deep learning, MIT Press, 2016

[13] Nielsen, M. A, Neural Networks and Deep Learning, Determination Press, 2015.

[14] Graves, A. 2013. Generating sequences with recurrent neural networks. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2013arXiv1308.0850G/abstract, [Accessed 24 Mai 2023]

[15] Chung, J., Gulcehre, C., Cho, K., & Bengio, Y, 2014, Empirical evaluation of gated recurrent neural networks on sequence modeling, [Online]. Available: https://arxiv.org/abs/1412.3555

[16] Sepp Hochreiter, & Jürgen Schmidhuber, Long short-term memory. Neural computation, MIT, 1997, [Online]. Available: https://direct.mit.edu/neco/article-abstract/9/8/1735/6109/Long-Short-Term-Memory

[17] Graves Alex, Supervised sequence labelling with recurrent neural networks, Studies in computational intelligence, Springer 2012 .

[18] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y., Learning phrase representations using RNN encoder-decoder for statistical machine translation, 2014, [Online]. Available: https://arxiv.org/abs/1406.1078

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, Attention is all you need, version 5, 2017. [Online]. Available : https://arxiv.org/abs/1706.03762