2.3.3.4 Transformers **demi complet **

Deep learning applications **pas complet**

Conclusion **pas complet**

**Chpter2**

**2.1Introduction**

The field of **AI**, using the strongest tools available in computer science, works toward imitating intelligence in a human being .These systems can perform a variety of tasks typically attributed to human cognitive abilities, such as decision-making, pattern recognition, and problem-solving. Artificial intelligence has come a long way over the years, fueling innovations such as self-driving cars, intelligent virtual assistants, and highly advanced recommendation systems, revolutionizing industries and everyday life.

In this chapter, basic machine-learning (ML) methodologies are looked into, a major subfield of AI. We will describe the three paradigms of learning: supervised learning, unsupervised learning, and reinforcement learning. Standard algorithms in machine learning will also be addressed followed by a transition into deep learning (DL), which is an enhanced version of ML that exploits multi-layer neural networks. The immediate goal in this instance is to firmly establish some of the fundamental concepts of these methods and their frameworks, in preparation for their application to real-world problems, including cybersecurity and SQL injection detection.

2.2 ma**chine 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.(1)

[1]( Mitchell, T. M, Machine Learning. McGraw-Hill, 1997.)

**Or wla**

Machine learning (ML) is a branch of artificial intelligence focused on developing algorithms and systems that can learn and improve from experience without being explicitly programmed. This field seeks to create computer programs capable of adapting to new data, identifying patterns, and making data-driven decisions. Over the years, machine learning has enabled groundbreaking applications across various domains, such as fraud detection systems that identify suspicious credit card transactions, personalized recommendation engines that adapt to user preferences, and self-driving cars that navigate complex environments. These advancements have been driven by significant progress in the theoretical foundations and algorithmic techniques that underpin machine learning(1) (2) mixed

* Mitchell, T. M. (1997). Machine Learning. McGraw-Hill.9(1)
* Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.(2)

**2.2.2 Machine Learning Types**

Whatever may be the nature and form of learning, machine learning always refers to a very wide-open field wherein computers learn from data to improve performance over time. **In machine learning, this kind of learning process is usually classed into three main parts which are supervised learning, unsupervised learning, and reinforcement learning**. Each type serves a certain distinct purpose and is used with certain types of problems. Besides those three types, hybrid approaches such as commodity inclusion and special techniques have also emerged to maintain and solve more complex issues.

Kbira bzf 9adrin na9sou fiha hadi l intro

**2.2.1.1 Supervised Learning: Learning from Labeled Data**

Supervised learning is a fundamental paradigm in machine learning where labeled data is used to train models. Under this paradigm, the input data have known target outputs, thereby allowing the model to learn the relationship between input and output. By spotting patterns and correlations in data, supervised learning algorithms can then make predictions or determine classifications on fresh, never-before-seen data.

**Main Applications:**

Examples of supervised learning applications are:

**Classification:**

In a classification problem, the intention is to assign a set of data points to defined predefined categories or labels. Examples include classifying an email as spam or not spam with the model trained, or diagnosing different medical conditions based on patient data. The well-known algorithms used for classification are decision trees, support vector machines, and neural networks.

**Regression**:

The regression task is concerned with the prediction of a continuous quantity. This may include estimating house prices, stock market trends, and variations in temperature. Algorithms such as linear regression and polynomial regression are commonly used in this scenario.

**2.2.1.2 Unsupervised Learning: Discovering Hidden Patterns**

[Unsupervised learning](https://www.ibm.com/think/topics/unsupervised-learning), also known as unsupervised machine learning, uses machine learning algorithms to analyze and cluster unlabeled datasets (subsets called clusters). These algorithms discover hidden patterns or data groupings without the need for human intervention.

Unsupervised learning’s ability to discover similarities and differences in information make it ideal for exploratory data analysis, cross-selling strategies, customer segmentation, and image and pattern recognition. It’s also used to reduce the number of features in a model through the process of dimensionality reduction. [Principal component analysis (PCA)](https://www.ibm.com/think/topics/principal-component-analysis) and singular value decomposition (SVD) are two common approaches for this. Other algorithms used in unsupervised learning include neural networks, [k-means clustering](https://www.ibm.com/think/topics/k-means-clustering), and probabilistic clustering methods. (3)

(3)( https://www.ibm.com/think/topics/machine-learning)

**2.2.1.3 Reinforcement Learning: Learning Through Interaction**

Reinforcement learning problems involve learning how to map situations to actions to maximize a numerical reward signal. These problems are inherently closed-loop, as the system’s actions influence its future inputs. Unlike other forms of machine learning, the learner is not explicitly told which actions to take but must discover the best ones through trial and error. In more complex scenarios, actions impact not only immediate rewards but also future states and long-term rewards, making decision-making more challenging (4)

(4)( Reinforcement Learning: An Introduction Second edition, in progress Richard S. Sutton and Andrew G. Barto c 2014, 2015).

**2.2.2 Machine learning algorithms**

**2.2.2.1** Logistic regression

Logistic regression is a supervised machine learning algorithm used for classification tasks, predicting the probability that an instance belongs to a specific class. It is a statistical method that analyzes the relationship between independent variables and a categorical outcome.

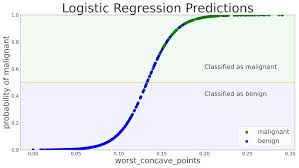
Unlike linear regression, logistic regression applies the sigmoid function to map input values to a probability ranging between 0 and 1. Instead of fitting a regression line, it models an "S"-shaped curve to distinguish between classes.

**Key Points:**

* Logistic regression predicts the output of a categorical dependent variable.
* The outcome is discrete (e.g., Yes/No, 0/1, True/False) but represented as a probability between 0 and 1 (5)

(5)( https://www.geeksforgeeks.org/understanding-logistic-regression/).

**Ttbdl ay flou**

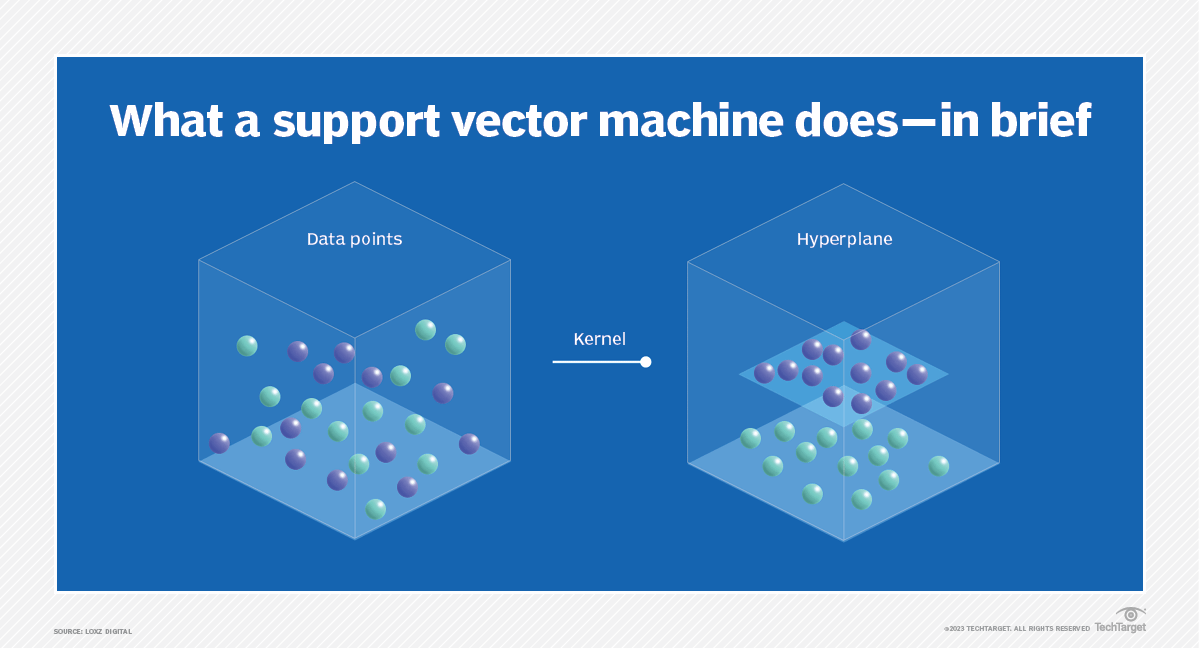


2.2.2.2 SVM

SVM (Support Vector Machine) is one of the supervised learning techniques employed for classification and regression tasks. The classification margin between various classes is maximized by finding the optimal hyperplane. Among the significant concepts are \*\*support vectors\*\* (data points that are critical in defining the margin), \*\*hyperplane\*\*, and \*\*kernel functions\*\* (which map the data into higher dimensions for separating it nonlinearly).

SVM uses three kernels-linear SVM, polynomial SVM, and artillery -to handle \*\*linearly separable\*\* and \*\*non-linearly separable\*\* data. It employs a soft margin to maximize the margin and add a penalty when data points are misclassified. The optimization problem is reduced to minimizing a loss function while ensuring that the classification is being done correctly. The \*\*dual problem\*\* allows efficient computation and non-linear classification by means of \*\*kernel trick\*\*. SVM is widely adopted in text classification, image recognition, and bioinformatics.(4)

(4)( https://www.geeksforgeeks.org/support-vector-machine-algorithm/)



**Meme hadi ida lguina makhir**

Figure(<https://www.techtarget.com/whatis/definition/support-vector-machine-SVM#:~:text=A%20support%20vector%20machine%20(SVM)%20is%20a%20type%20of%20supervised,data%20set%20into%20two%20groups.)>

**2.2.3 Machine learning applications**

Machine learning has a very promising range of application across several industries. It has been successfully applied in relatively recent fields, such as transport, energy, agricultural services, and finance, and several other emerging fields like cybersecurity, marketing, and e-learning.

**Healthcare and medical applications**: include diagnosis of diseases, recommendations for treatments, and tailored patient care.

**Automotive Industry**: Increasingly, autonomous vehicles under construction or improved management of traffic are to be found within this sector.

**Financial Services**: Fraud detection and risk evaluation for the purpose of making investment decisions.

**Energy and Utilities**: Optimize energy usage, outage predictions, and increased reliability of the grid.

Agriculture: Forecasts crop yield, detects plant diseases, and manages irrigation optimally.

**Cybersecurity**: Detection of cyberattacks and suspicious behavior. Improvement of data protection systems and reinforced using AI authentication.

**Marketing and Advertising**: Personalization of advertising campaigns, consumer behavior analysis, and smart-targeting results optimization from marketing strategies.

**E-Learning and Education**: Changing courses according to student performance, suggesting personalized material, and a chatbot powered by AI to support learners.

Apart from these, there are many more industries that are being influenced by machine learning. Today, so much that was previously unimaginable is transforming in people's lives today, and this progress will continue with the advancement of technologies.

****

**2.3 Deep Learnig**

Deep learning is a subfield of machine learning that employs deep neural networks for analyzing and interpreting complex data. Such networks are modeled after the human brain and allow the computer to identify patterns and relationships without human intervention in large amounts of unstructured information. The deep learning model is continuously improving its accuracy by tuning internal parameters with training**.**

Deep learning models can be trained to perform classification tasks and recognize patterns in images, text, audio, and other types of data. This technology also enables automation of tasks that typically require human intelligence, such as image description and audio transcription. Where human brains have millions of interconnected neurons that work together to learn information, deep learning features neural networks constructed from multiple layers of software nodes that work together. (5)

This technique has achieved astonishing outcomes in image recognition, understanding natural language, and processing speech, making it the foundation of contemporary artificial intelligence systems.

(5)( <https://www.techtarget.com/searchenterpriseai/definition/deep-learning-deep-neural-network>)

**2.3.1 Artificial Neural Networks (ANNs)**

Artificial Neural Networks (ANN) are inspired by the way biological neural system works, such as the brain process information. The information processing system is composed of a large number of highly interconnected processing elements (neurons) working together to solve specific problems. ANNs, just like people, learn by example. Similar to learning in biological systems, ANN learning involves adjustments to the synaptic connections that exist between the neurons.



Figur (6)

Here [X1, X2, X3] are the input features to the neural networks represented as X. Whereas the superscript [1] is used to denote the layer. The weights are denoted by [W1, W2, W3] associated with each connection to the neuron from the input of that particular layer. The bias is represented by b associated with the neuron. “z” is the weighted sum of inputs added with the bias which is linear in nature. “a” is the activation function that is applied to z to add non-linearity as complex models can't be represented as a line. (6)

(6)( <https://medium.com/@anushruthikae/basic-notations-and-representation-neural-networks-d46a1be97471>)

The activation function is applied to the weighted sum of inputs to the neuron, including the bias term, and the resulting value becomes the neuron's output, which is then passed to the next layer. Its primary role is to introduce non-linearity into the model, allowing the network to learn complex patterns and approximate any arbitrary function. In this one-layer neural network architecture, the output of the activation function in layer [1] serves as the final output, denoted as y’. This output is used to compute the loss function, L(a, y), which measures the deviation between the predicted and actual output. This deviation is crucial for backpropagation and optimization, which will be discussed in later sections.(6)



An artificial neural network is primarily composed of three layers: the input layer, one or more hidden layers, and the output layer. These layers collectively work to process the information and yield meaningful predictions.

**Input layers** serve as the entry point for data into the neural network. Each neuron in this layer corresponds to either a specific feature of the input dataset or an input vector. For instance, in an image classification problem, all input neurons may specify the intensity of each individual pixel. This layer exists purely for the purpose of passing on the raw input values to the next layer without change.

**Hidden layers** serve to process and transform information passed to them from the input layer. They are called hidden since nothing about their inner workings can be observed directly. In a typical hidden layer, a neuron takes in inputs from the preceding layer, applies a weighted summation, adds a bias, and applies an activation function on the result. Depending on the complexity of the task, a network can have varying numbers of hidden layers with varying numbers of neurons in each layer. DNNs having multiple hidden layers are quite popular in applications such as image recognition, speech processing, and natural language understanding since they can efficiently extract many complex patterns from the data.

**The output layer is** the final layer of the network that provides the prediction made by the model. The structure of this layer is based on the solved problem. In the case of classification networks, each output neuron corresponds to one class; in regression problems, usually only one output neuron provides a continuous value: an activation function is then used in the output layer according to the type of task, e.g., softmax for multi-class classification and any linear function for regression.

These layers are interconnected by **weighted links** that determine the importance of each input. The weights are adjusted during training through a system called **backpropagation** whereby errors are carried backward from the output to the hidden layers enabling the network to learn and enhance its performance with time. Persistent adjustments of these weights equate to an increasing ability of the neural network to predict correctly.

or nkhdmou b article li kayn hna (<https://www.v7labs.com/blog/neural-networks-activation-functions>)

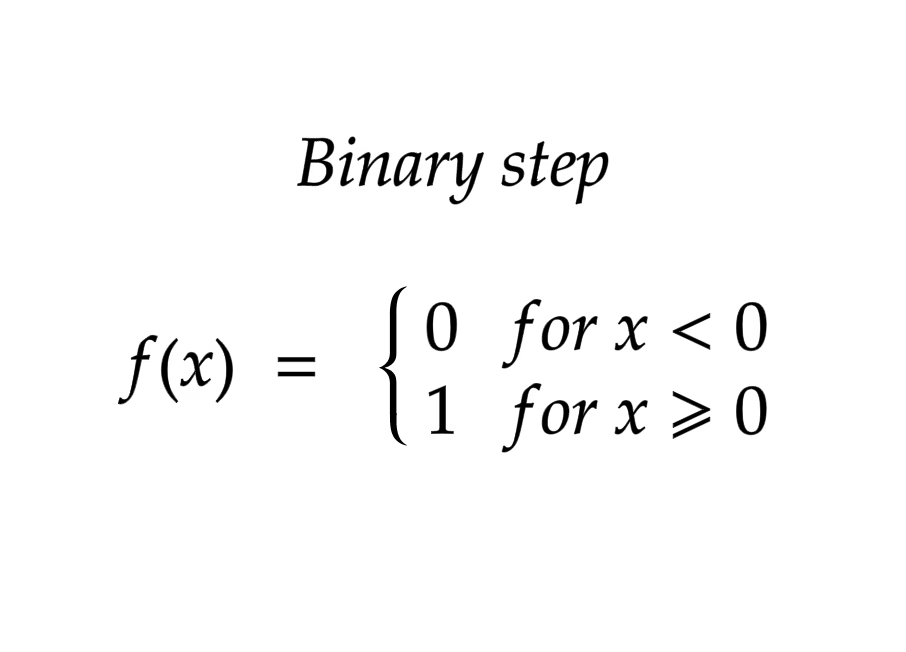
**2.3.2 Activation functions**

Activation functions play a fundamental role in neural networks by determining how neurons process input data and transfer information to subsequent layers. The choice of activation function significantly influences the network's performance and learning capability

**1. Binary Step Function**

The Binary Step Function is a threshold-based activation function that decides whether a neuron is activated or remains inactive. If the input surpasses a predetermined threshold, the neuron is activated; otherwise, it stays dormant, meaning it does not contribute to the next layer.



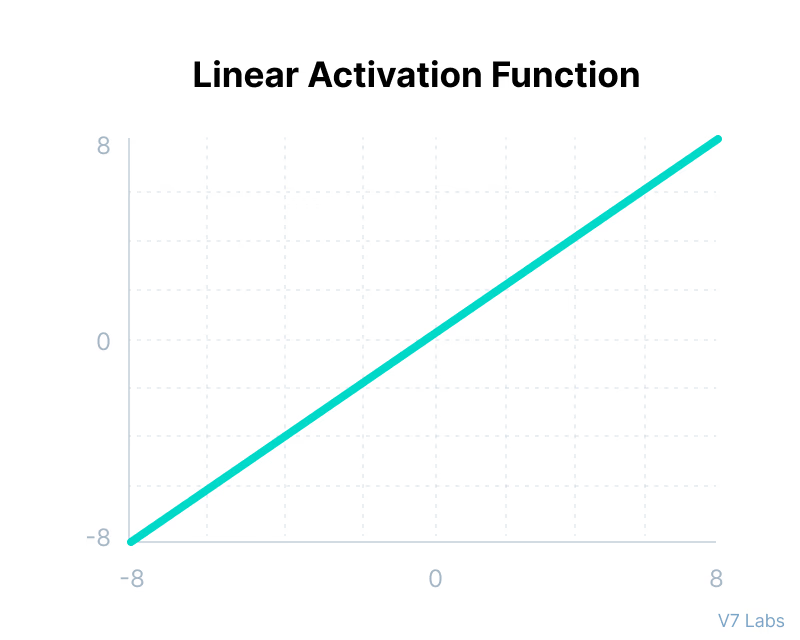


* This function is simple and computationally efficient but is unsuitable for tasks requiring multi-class classification due to its binary output. Additionally, its derivative is zero, preventing the backpropagation process and hindering learning. (7)

(7)( [https://www.v7labs.com/blog/neural-networks-activation-functions#3-types-of-neural-networks-activation-functions](https://www.v7labs.com/blog/neural-networks-activation-functions" \l "3-types-of-neural-networks-activation-functions))

**2. Linear Activation Function**

The Linear Activation Function, also known as the identity function, outputs the input value without modification, allowing direct propagation of signals through the network.



Backpropagation is ineffective with this function since its derivative is constant and does not depend on input values. Additionally, using a linear activation function across multiple layers results in a network that is functionally equivalent to a single-layer model, limiting its ability to learn cmplex patterns.. (7)

### 3. Non-Linear Activation Functions (7)

A network using only a linear activation function is essentially equivalent to a simple linear regression model, limiting its ability to capture complex patterns in data. Non-linear activation functions enable deep networks to model intricate relationships between inputs and outputs.

They allow backpropagation by ensuring derivatives depend on input values, facilitating effective weight adjustments. They also enable the creation of deep networks, where transformed outputs from one layer pass non-linearly to the next, improving the model's ability to learn complex representations.

**4. Sigmoid (Logistic) Activation Function(7)**

This function takes any real value as input and outputs values in the range of 0 to 1.making it useful for probabilistic models and binary classification tasks.

 The larger the input (more positive), the closer the output value will be to 1.0, whereas the smaller the input (more negative), the closer the output will be to 0.0, as shown below.

****

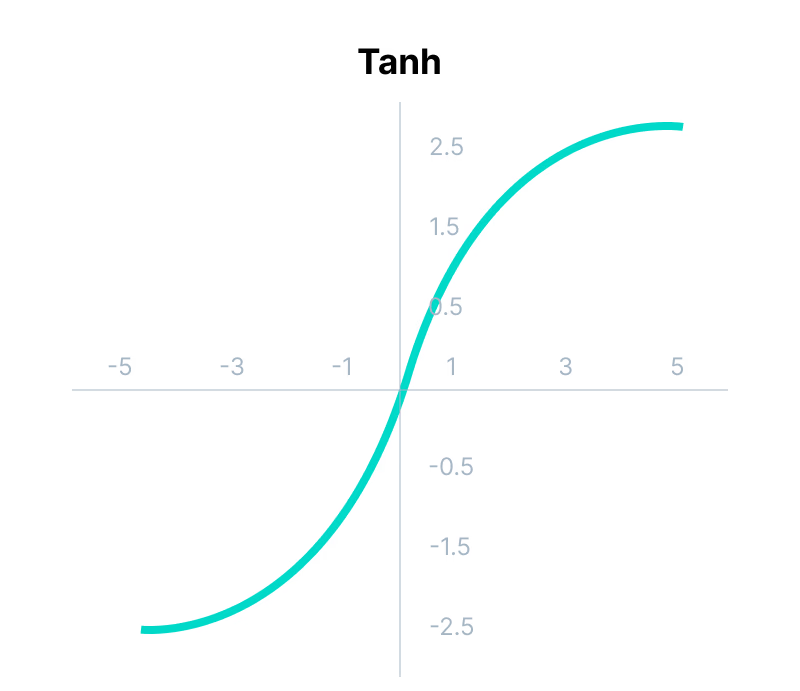


It is ideal for probability-based applications due to its constrained output range and is differentiable, ensuring smooth gradient updates during optimization. However, it suffers from the vanishing gradient problem, as extreme values lead to near-zero derivatives, hindering learning. Additionally, it is not zero-centered, which can slow down the training process.

**5. Tanh (Hyperbolic Tangent) Function(7)**

The Tanh function is similar to the sigmoid function but maps input values to a range between -1 and 1, providing stronger non-linearity.

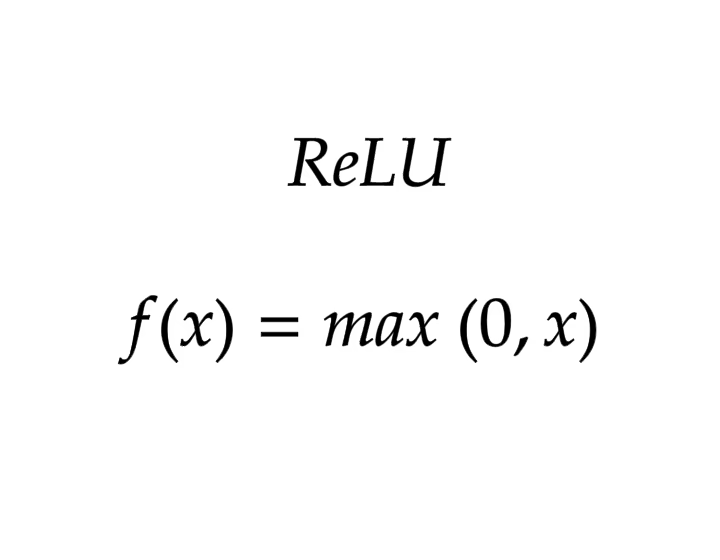
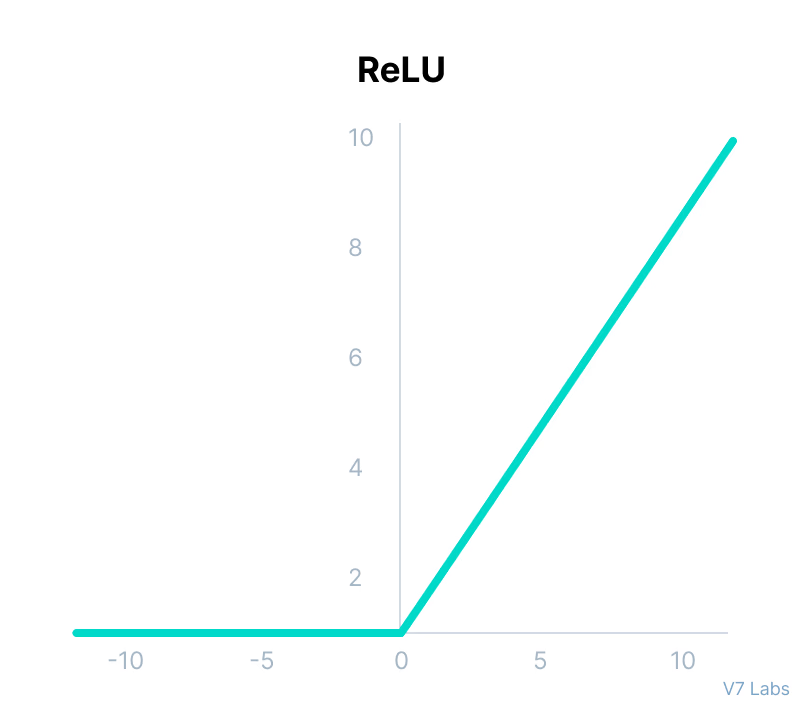
****



Its outputs are zero-centered, which improves convergence speed in deep networks, and it is often used in recurrent neural networks (RNNs) and convolutional neural networks (CNNs). However, it still suffers from the vanishing gradient problem, albeit less than the sigmoid function.

**6. ReLU (Rectified Linear Unit) Function(7)**

ReLU is one of the most commonly used activation functions in deep learning. It introduces non-linearity by outputting zero for negative inputs while retaining positive values unchanged.



It is computationally efficient, as only a subset of neurons activate at a time, and helps accelerate gradient descent convergence due to its non-saturating nature. However, it suffers from the Dying ReLU problem, where neurons can become permanently inactive for negative inputs, preventing further updates..

**Kaml meme ressource**

**Bzf wa9ila**

**2.3.3 Deep learning architectures**

Deep learning has enjoyed tremendous advancement in the last few years, serving as the major pillar for innovation in many different fields. Each architecture is designed for particular problems, with the performance in each case optimized for the specific needs of the task at hand.

Over the years, many deep learning models have been developed, often extending some fundamental designs. Among these, convolutional neural networks (CNNs), recurrent neural networks (RNNs), and long short-term memory networks (LSTMs) are the most commonly known. In their respective areas, these architectures have been very efficient, thereby enabling progress in image recognition, sequence modeling, and time series analysis.

**2.3.3.1 Recurrent Neural Networks**

Recurrent Neural Networks (RNNs) are neural networks designed to recognize patterns in sequences of data. They’re used for identifying patterns such as text, genomes, handwriting, or numerical time series data from stock markets, sensors, and more.

Unlike traditional [feedforward neural networks](https://www.analyticsvidhya.com/blog/2022/03/basic-introduction-to-feed-forward-network-in-deep-learning/" \t "_blank), where inputs are processed only once in a forward direction, RNNs possess a unique feature: They have loops in them, allowing information to persist.

This looping mechanism enables RNNs to remember previous information and use it to influence the processing of current inputs. This is like having a memory that captures information about what has been calculated so far, making RNNs particularly suited for tasks where the context or the sequence is crucial for making predictions or decisions.(8)

(8)( <https://shelf.io/blog/recurrent-neural-networks/>)



Figur (9)+ expliquation (9)

RNNs are made of neurons: data-processing nodes that work together to perform complex tasks. The neurons are organized as input, output, and hidden layers. The input layer receives the information to process, and the output layer provides the result. Data processing, analysis, and prediction take place in the hidden layer.

### **Hidden layer**

RNNs work by passing the sequential data that they receive to the hidden layers one step at a time. However, they also have a self-looping or recurrent workflow: the hidden layer can remember and use previous inputs for future predictions in a short-term memory component. It uses the current input and the stored memory to predict the next sequence.

For example, consider the sequence: Apple is red. You want the RNN to predict red when it receives the input sequence Apple is. When the hidden layer processes the word Apple, it stores a copy in its memory. Next, when it sees the word is, it recalls Apple from its memory and understands the full sequence: Apple is for context. It can then predict red for improved accuracy. This makes RNNs useful in speech recognition, machine translation, and other language modeling tasks.

### **Training**

Machine learning (ML) engineers train deep neural networks like RNNs by feeding the model with training data and refining its performance. In ML, the neuron's weights are signals to determine how influential the information learned during training is when predicting the output. Each layer in an RNN shares the same weight.

ML engineers adjust weights to improve prediction accuracy. They use a technique called backpropagation through time (BPTT) to calculate model error and adjust its weight accordingly. BPTT rolls back the output to the previous time step and recalculates the error rate. This way, it can identify which hidden state in the sequence is causing a significant error and readjust the weight to reduce the error margin.

## types of recurrent neural networks

RNNs are often characterized by one-to-one architecture: one input sequence is associated with one output. However, you can flexibly adjust them into various configurations for specific purposes. The following are several common RNN types.

### **One-to-many**

This RNN type channels one input to several outputs. It enables linguistic applications like image captioning by generating a sentence from a single keyword.

### **Many-to-many**

The model uses multiple inputs to predict multiple outputs. For example, you can create a language translator with an RNN, which analyzes a sentence and correctly structures the words in a different language.

### **Many-to-one**

Several inputs are mapped to an output. This is helpful in applications like sentiment analysis, where the model predicts customers’ sentiments like positive, negative, and neutral from input testimonials.

(9)( https://aws.amazon.com/what-is/recurrent-neural-network/)

**2.3.3.2 Long Short-Term Memory Networks (10) klch f lstm**

Traditional RNNs struggle with long-term dependencies due to the vanishing and exploding gradient problem. To address this, **Long Short-Term Memory (LSTM) networks** were introduced.

A long short-term memory (LSTM) network is a type of recurrent neural network (RNN). LSTMs are predominantly used to learn, process, and classify sequential data because they can learn long-term dependencies between time steps of data.

LSTM layers use additional gates to control what information in the hidden state is exported as output and to the next hidden state. These additional gates overcome the common issue with RNNs in learning long-term dependencies. In addition to the hidden state in traditional RNNs, the architecture for an LSTM block typically has a memory cell, input gate, output gate, and forget gate. The additional gates enable the network to learn long-term relationships in the data more effectively. Lower sensitivity to the time gap makes LSTM networks better for analyzing sequential data than simple RNNs. In the figure below, you can see the LSTM architecture and data flow at time step t.



Data flow at time step t for an LSTM unit. The forget gate and memory cell prevent the vanishing and exploding gradient problems.

The weights and biases to the input gate control the extent to which a new value flows into the LSTM unit. Similarly, the weights and biases to the forget gate and output gate control the extent to which a value remains in the unit and the extent to which the value in the unit is used to compute the output activation of the LSTM block, respectively.

The following diagram illustrates the data flow through an LSTM layer with multiple time steps. The number of channels in the output matches the number of hidden units in the LSTM layer



Data flow for an LSTM with multiple time steps. Each LSTM operation receives the hidden state and cell state from the previous operation and passes an updated state and cell state to the next operation.

LSTMs work well with sequence and time-series data for classification and regression tasks. LSTMs also work well on videos because videos are essentially a sequence of images. Similar to working with signals, it helps to perform feature extraction before feeding the sequence of images into the LSTM layer. Leverage convolutional neural networks (CNNs) (e.g., GoogLeNet) for feature extraction on each frame. The following figure shows how to design an LSTM network for different tasks.



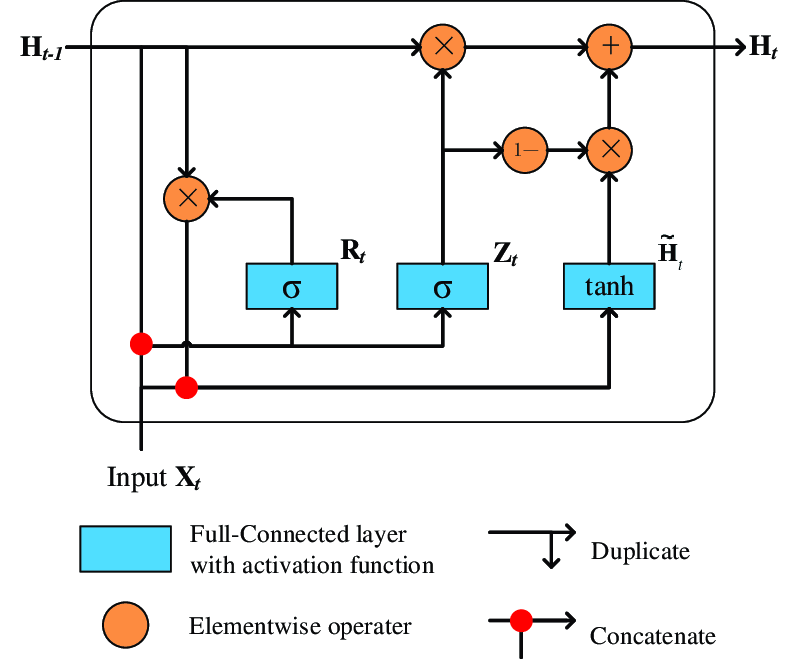
Figure LSTM network architecture for classification, regression, and video classification tasks.

(10) (https://www.mathworks.com/discovery/lstm.html)

**2.3.3.3 Gated Recurrent Units (GRU)**

The Gated Recurrent Unit (GRU) is a variant of the Recurrent Neural Network (RNN) designed to overcome the vanishing and exploding gradient problems encountered during the training 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."

Unlike traditional RNNs, GRUs incorporate two specialized gates:

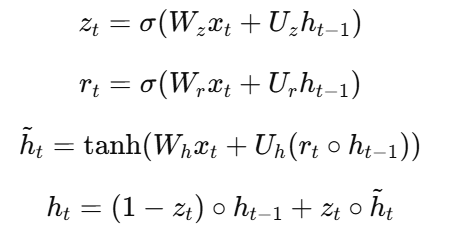
* The update gate, which determines how much of the past information should be carried forward.
* The reset gate, which decides how much of the previous hidden state should be forgotten.
* 
* Figure (11)

(11)(https://medium.com/@muhabd51/comparison-and-architecture-of-lstm-gru-and-rnn-1c9afe11b09f)

* Li be rouge baalak tetna7a

1. These mechanisms "allow GRUs to selectively retain and update information over time, facilitating the learning of complex dependencies in sequential data." The ability to dynamically control information flow makes GRUs highly effective for processing sequential data, such as time series, speech recognition, and natural language processing

The GRU processes input sequences using the following set of equations:



where:

 xt​ and hth\_tht​ represent the **input and output** of the GRU at the current time step ttt, respectively.

 ztz\_tzt​ and rtr\_trt​ correspond to the **outputs of the update and reset gates**, respectively.

 hth\_tht​ is the **output of the hidden state**.

 σ(⋅)\sigma(\cdot)σ(⋅) denotes an **activation function** (typically the sigmoid function).

 ∘\circ∘ represents the **Hadamard (element-wise) product**.

 WWW and UUU are **trainable weight matrices** of the network.

 ztz\_tzt​, rtr\_trt​, and h~t\tilde{h}\_th~t​ are the **update gate, reset gate, and candidate hidden state**, respectively. (12)

(12) (Gated recurrent unit neural network (GRU) based on quantile regression (QR) predicts reservoir parameters through well logging data) ARticle

**2.3.3.4 Transformers**

Recurrent neural networks (RNNs), including LSTMs and GRUs, are state-of-the-art for sequence modeling and transduction tasks like machine translation. However, their sequential computation limits parallelization, especially for long sequences. Attention mechanisms improve dependency modeling but are typically used with RNNs. The Transformer eliminates recurrence, relying entirely on attention for global dependencies, enabling greater parallelization and state-of-the-art results. Unlike convolutional models (e.g., ByteNet, ConvS2S), the Transformer reduces operations for distant dependencies to a constant. It is the first model to use only self-attention, avoiding RNNs or convolutions. (13)



Figure : The Transformer- model architecture. (13)

**Encoder**: The encoder is composed of a stack of N = 6 identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, positionwise fully connected feed-forward network. We employ a residual connection [11] around each of the two sub-layers, followed by layer normalization [1]. That is, the output of each sub-layer is LayerNorm(x + Sublayer(x)), where Sublayer(x) is the function implemented by the sub-layer itself. To facilitate these residual connections, all sub-layers in the model, as well as the embedding layers, produce outputs of dimension dmodel = 512. (13)

**Decoder**: The decoder is also composed of a stack of N = 6 identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. Similar to the encoder, we employ residual connections around each of the sub-layers, followed by layer normalization. We also modify the self-attention sub-layer in the decoder stack to prevent positions from attending to subsequent positions. This masking, combined with fact that the output embeddings are offset by one position, ensures that the predictions for position i can depend only on the known outputs at positions less than i. (13)

**Attention**: An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key (13)

(13)( 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>)

**Scaled Dot-Product Attention**

The scaled dot-product attention mechanism is an important part of Transformer architecture and works with queries, keys, and values as the input. First, now, the model computes a dot product over the similarity of each query with all the keys. To avoid any training instability due to very large similarity scores, the values of these similarity scores are scaled down by dividing them by the square root of the key dimension. These scaled scores are subsequently put through a softmax, which returns the normalized weights. The weights tell the model how much importance to attach to each value whilst combining them into a final output. This mechanism enables an efficient focus on only the relevant parts of the input. Scaled dot-product attention, being much faster and memory-efficient compared to additive attention that involves computing similarities with a small neural network, perfectly fits the bill in case of large-scale applications.

**Multi-Head Attention**

**Multi-head attention enhances the standard attention mechanism by allowing the model to focus on different parts of the input simultaneously. Instead of using a single attention function, it projects the queries, keys, and values multiple times using separate learned transformations. These projections are processed in parallel, and their results are combined and projected again to produce the final output. This approach enables the model to capture diverse information from different representation subspaces, improving its ability to handle complex patterns. For example, Transformers often use eight attention heads, each working on smaller dimensions to maintain computational efficiency**

**Feed-Forward Networks**

**Embeddings and Softmax**

**Positional Encoding**

**Transformers Pas complet**

**2.3.3.5 Autoencoders balak**

**Conclusion**

In summary, this chapter explored the essential principles of machine learning and deep learning, covering their fundamental concepts, key algorithms, and architectural frameworks. By studying these advanced techniques, the reader gains a clearer understanding of their significance in the broader field of artificial intelligence. These methods continue to drive innovation, paving the way for future AI applications across various domains