People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

IBN KHALDOUN UNIVERSITY OF TIARET

Dissertation

Presented to:

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE
DEPARTEMENT OF COMPUTER SCIENCE

in order to obtain the degree of

MASTER

Specialty: software engineer

Presented by:

BENFERHAT Tinhinane
AZZOUZ Aya

On the theme:

Modeling and simulation of photocatalytic reactors using deep learning methods

Defended publicly on 15/ 06 /2023 in Tiaret in front the jury composed of:

Mr BENDAOUD Mebarek	Prof	Tiaret University	Chairman
Mr AID Lahcene	MCA	Tiaret University	Supervisor
Mr MOSTEFAOUI Kadda	MAA	Tiaret University	Examiner

2022-2023

DEDICATION

This Dissertation is dedicated

To my parents and sisters, whose unwavering love and support have been my anchor throughout this journey. Your encouragement, understanding, and belief in me have inspired and motivated me to push beyond my limits and pursue my academic goals. I am grateful for your sacrifices and for always being there to celebrate my successes and lift me up during challenges. This accomplishment is as much yours as it is mine, and I dedicate this work to each one of you. Thank you for being my rock and for believing in my dreams.

To my partner and friends,

To everyone who has supported me and helped me get to where I am now,

Grace.

Tinhinane.

DEDICATION

I dedicate this work:

To my very dear Father *AZZOUZ Mohamed,*

To my very dear mother *SEGHIER Karima,*

thank you for your unwavering love, support, and belief in me. You have been my constant pillars of strength, providing the guidance and encouragement I needed to overcome every obstacle.

To my sisters *Ferial, Ilhem, Amel* *et* *Batoul.*

To my partner,

To my entire Family,

To mu friends,

thank you for your camaraderie, laughter, and shared experiences.

And to all those who have participated from afar or
up close and have helped us in the development of
this dissertation.

Grace.

Aya.

Acknowledgements

First of all, we thank the Almighty God, for giving us the strength and patience to complete this modest work.

We would like to express my heartfelt gratitude to all those who have contributed to the completion of this project.

First and foremost, I would like to thank my supervisor, **Dr. AID Lahcen**, for their guidance, support, and invaluable insights throughout the entire process. Their expertise and dedication have been instrumental in shaping this work, and I am truly grateful for their mentorship.

We would like to thank the members of the jury **Mr. BENDAOUD Mebarek** and **Mr. MOSTEFAOUI Kada** for their time and expertise in evaluating this project. Their valuable feedback and constructive criticism have helped refine the final outcome.

A special thanks goes to **Mr. DEHBI Abdelkader**, **Mr. Bassaid Salah** and **Miss.BOUAZZA** for their valuable help, they gave us a strong hand in order to complete our project.

Deeply thankful to everyone who has played a part in this project. Your support and collaboration have been invaluable, and I am honored to have had the opportunity to work with such amazing individuals.

Abstract

Deep learning-based artificial intelligence has shown promise in a variety of fields applications and research areas, compared to traditional machine learning algorithms, it can produce extremely high-level data representations making them promising for a wide range of applications.

The primary aim of this study is to present a comprehensive methodology for designing deep learning models specifically tailored for addressing complex non-linear regression challenges. More specifically, the focus is on developing two advanced models: a Convolutional Neural Network (CNN) and a Long Short-Term Memory (LSTM) model. These models will be utilized to tackle practical issues, notably simulating a photocatalytic reactor to accurately predict water purification efficacy and estimate its lifespan. To ensure the utmost credibility and applicability, real-world datasets will be employed in this research endeavor. By harnessing the power of cutting-edge deep learning techniques, this study endeavors to advance the understanding and optimize the intricate processes involved in water purification.

As part of this endeavor, two deep learning models were devised to tackle the assigned challenge. The training and validation stages of the deep learning model proved fruitful for the LSTM model, thus substantiating its efficacy in addressing non-linear regression problems. In contrast, the CNN model encountered obstacles during the validation phase, implying that the LSTM model is better suited to handle such complexities.

Key words: Artificial intelligence, Deep learning, linear regression, nonlinear regression, neural networks, CNN, LSTM, photocatalytic reactor.

Résumé

L'intelligence artificielle basée sur l'apprentissage profond a montré des promesses dans de nombreux domaines d'application et de recherche. Comparée aux algorithmes d'apprentissage machine traditionnels, elle est capable de produire des représentations de données extrêmement avancées, ce qui les rend prometteuses pour une large gamme d'applications.

Cette étude vise à présenter une méthodologie rigoureuse pour concevoir des modèles d'apprentissage profond spécifiquement adaptés à la résolution des défis de la régression non linéaire. Une attention particulière est accordée au développement de deux modèles avancés: un réseau de neurones convolutifs (CNN) et un modèle à mémoire à court terme et long terme (LSTM). Ces modèles sont appliqués à des problèmes concrets, tels que la simulation d'un réacteur photocatalytique pour prédire l'efficacité de la purification de l'eau et estimer sa durée de vie. Afin de garantir la validité et la pertinence des résultats, des ensembles de données réelles sont utilisés dans cette recherche, renforçant ainsi la fiabilité des conclusions obtenues.

Dans le cadre de ce travail, deux modèles d'apprentissage profond ont été élaborés pour relever ce défi spécifique. Les phases d'entraînement et de validation de ces modèles ont abouti à des résultats probants pour le modèle LSTM, démontrant ainsi son efficacité dans la résolution de problèmes de régression non linéaire. En revanche, le modèle CNN a rencontré des difficultés lors de la phase de validation, suggérant que le modèle LSTM est mieux adapté pour traiter ces complexités.

Mots clés : Intelligence artificielle, apprentissage profond, régression linéaire, régression non linéaire, réseaux de neurones, CNN, LSTM, réacteur photocatalytique.

ملخص

تم تحقيق تحول ثوري في مجال الانحدار بفضل الذكاء الاصطناعي القائم على التعلم العميق، حيث يوفر حلاً دقيقًا للمشكلات الخطية وغير الخطية على حد سواء. من خلال استغلال القدرات الهائلة للشبكات العصبية العميقة، يمنحنا التعلم العميق القدرة على التقاط العلاقات المعقدة بين المتغيرات، مما يؤدي إلى تحسين دقة التنبؤ. يتمتع هذا النهج المتعدد الاستخدامات بتطبيقات واسعة النطاق في مختلف المجالات.

يهدف هذا البحث في المقام الأول إلى تقديم منهجية شاملة لتصميم نماذج تعلم عميق مصممة خصيصًا لمعالجة التحديات الكبيرة في الانحدار غير الخطي. يتم التركيز بشكل خاص على تطوير نموذجين متقدمين: نموذج شبكة عصبية مترابطة تحويلية (CNN) ونموذج ذاكرة طويلة الأجل وقصيرة الأجل (LSTM) . سيتم استخدام هذه النماذج لمعالجة قضايا عملية، بما في ذلك محاكاة مفاعل ضوئي لتنقية المياه وتوقع كفاءة التنقية وتقدير عمر المفاعل. سيتم استخدام مجموعات بيانات حقيقية لضمان أقصى درجات المصداقية والتطبيقية في هذا البحث. من خلال استغلال قوة تقنيات التعلم العميق الحديثة، يسعى هذا البحث إلى تعزيز الفهم وتحسين العمليات المعقدة المرتبطة بتنقية المياه.

كجزء من هذا السعي، تم تصميم نموذجين للتعلم العميق لمواجهة التحدي المحدد. أثبتت مراحل التدريب والتحقق من النموذج التعلم العميق نجاحها لنموذجLSTM ، مما يدل على فعاليته في معالجة مشكلات الانحدار غير الخطي. وعلى الجانب المقابل، واجه نموذج CNN عقبات خلال مرحلة التحقق، مما يشير إلى أن نموذج LSTM هو الأكثر تواءمًا للتعامل مع تلك التعقيدات.

الكلمات المفتاحية: الذكاء الاصطناعي، التعلم العميق، الانحدار الخطي، الانحدار الغير خطي, الشبكات العصبية العميقة .LSTM , CNN ..

Table of contents

CHAPTER 1 Photocatalytic Reactors

CHAPTER II Deep Learning

CHAPTER III Modeling of phototcatalytic reactor

List of figures

List of tables

List Of Abbreviations

- $xt \in R^n$: input vector to the hidden layer,
- $ht \in R^l$: output of hidden layer and denotes hidden state,
- $y\hat{}t \in R^m$: output produced by RNN, $y \in R^m$ desired output,
- $Uhx = [Uji] \in R^{l\times n}$: the input-hidden weight matrix from the ith input node to the jth hidden state,
- $Why = [Wjk] \in R^{l\times m}$: output-hidden weight matrix from the *k*th output node to the jth hidden state,
- $Vyh = [Vkj] \in R^{l\times m}$: hidden-output weight matrix from the jth hidden state to the *k*th output node,
- $bt \in R^l$: bias vectors (called also intercept terms) of hidden layers,
- $ct \in R^m$: bias vectors (called also intercept terms) of output layers,
- $f(hi) \in R$: elementwise activation function of hidden layer,
- $g(yi) \in R$: elementwise activation function of output layer.

INTRODUCTION

Artificial intelligence (AI) exerts a profound and wide-ranging impact on contemporary society, permeating various domains and applications. It encompasses a diverse array of approaches and categories, enabling machines to replicate and even surpass human cognitive processes such as perception, reasoning, and learning. Machine learning, a pivotal facet of AI, endows machines with the ability to acquire knowledge and make predictions based on data, bypassing the need for explicit programming. The efficacy of a machine learning model hinges on the quality of the data and the performance of the underlying learning algorithms. Machine learning encompasses several techniques including supervised, unsupervised, semi-supervised, and reinforcement learning, which facilitate a range of tasks such as classification, clustering, and prediction.

Deep learning, a subfield of machine learning, harnesses the power of multi-layer artificial neural networks to emulate the intricate workings of the human brain. It excels in tackling complex challenges such as image recognition and natural language processing. By capitalizing on deep neural network architectures, deep learning can capture and learn intricate nonlinear relationships between input and output variables. This capability empowers the modeling of complex phenomena and the generation of highly accurate predictions.

Nonlinear regression, on the other hand, represents a statistical method employed to model and forecast nonlinear relationships between variables. Deep learning techniques can be employed to address nonlinear regression tasks by leveraging the hierarchical representations acquired through deep neural networks. This enables more precise predictions and the capacity to model intricate phenomena that conventional linear regression methods may struggle to capture.

For completing our project, we have organized our work as follows.

CHAPTER I: Photocatalytic Reactors.

CHAPTER II: Deep learning.

CHAPTER III: Modeling of photocatalytic reactor.

CHAPTER I

Photocatalytic Reactors

1.1 Introduction

Water scarcity has emerged as a pressing global concern due to the rapid growth of industries, population expansion, and prolonged droughts. The increased demand for safe drinking water, combined with limited supply, presents a significant challenge. Furthermore, the widespread use of synthetic dyes in various sectors, including textiles, printing, and cosmetics, has introduced harmful chemicals that can be detrimental to both human health and the environment if not properly disposed of. One such dye, Blue Methylene, commonly employed in these industries, poses a potential threat to water sources if not adequately treated. Conventional water treatment methods like filtration and chemical treatments often prove inadequate in eliminating organic pollutants such as Blue Methylene. Moreover, these methods can be costly and yield undesirable byproducts. Consequently, there is a need for more effective and environmentally friendly approaches to water treatment. Enter photocatalytic reactors, which harness a combination of catalysts and UV light to generate reactive chemical species that interact with organic pollutants in water, effectively breaking them down into non-toxic compounds. This process facilitates efficient pollutant removal and water purification. Photocatalytic reactors offer a sustainable and eco-conscious solution for treating polluted water. This chapter will delve extensively into this intricate subject matter.

1.2 Photocatalytic

Photocatalysis refers to a chemical reaction in which a catalyst is activated by light to facilitate a chemical reaction. In other words, photocatalysis is the process of using light energy to stimulate a chemical reaction in a material or substance. This technology has found various applications in water purification, air treatment, and energy production [1].

Principle of photocatalysis

The absorption of light radiation (mainly UV) causes excitation of peripheral electrons on the photocatalyst made of a semiconductor material (TiO_2, ZnO, Fe_2O_3, ZnS, CdS...), which move from the valence band (VB) to the conduction band (CB), creating electron-hole pairs capable of reacting with oxygen from the air and/or atmospheric moisture to form radicals and initiate redox reactions. Historically, TiO_2 has been the most industrially utilized photocatalyst, but in recent years, other substances such as ZnO have emerged [2].

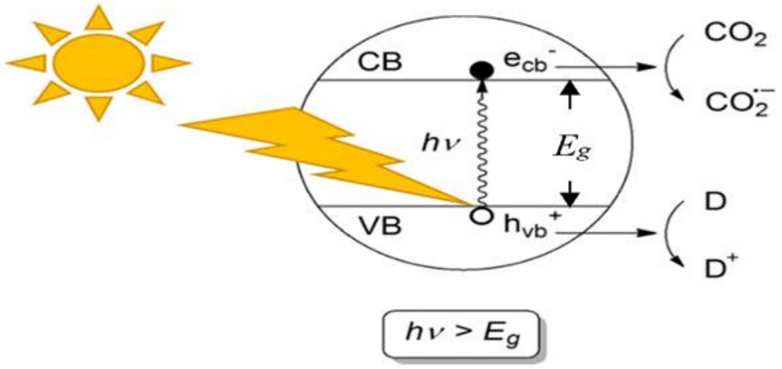


Figure 1: Principle of photocatalysis [3].

In the presence of oxygen and water vapor, ultraviolet radiation activating a semiconductor material (WO_3, TiO_2, ZnO, etc.) allows the molecules that come into contact with the reactive surface to be converted into carbon dioxide and water vapor. This technique, called photocatalysis, eliminates microorganisms or odors, as well as volatile organic and inorganic gas compounds such as nitrogen oxides [3], [4] .

1.3 Reactor

A reactor is a device or vessel in which a chemical reaction takes place. It is designed to contain and control chemical reactions that occur within it, and can be used for a variety of purposes, including chemical synthesis, combustion, and waste treatment. Reactors can come in various shapes and sizes, and can be operated under different conditions such as temperature, pressure, and mixing intensity to achieve specific reaction outcomes [5], [6].

Photocatalytic reaction mechanisms

The photocatalytic degradation of pollutants in a photocatalytic reactor involves several steps, which together make up the reaction mechanism. The following steps provide a general overview:

Light absorption: The photocatalyst absorbs photons of light energy, typically in the UV region, and creates electron-hole pairs [7].

Charge separation: The generated electron and hole are separated due to the potential gradient of the photocatalyst. The electron is typically transferred to an oxygen molecule, creating a superoxide radical, while the hole is transferred to a water molecule, creating a hydroxyl radical [8].

Radical formation: The superoxide and hydroxyl radicals react with water or oxygen to create other reactive species, such as hydrogen peroxide, hydroperoxyl radicals, and hydroxyl ions [9].

Pollutant oxidation: The reactive species attack the pollutant molecules, breaking them down into smaller, less harmful compounds [10].

Final product formation: The end products of the reaction depend on the specific pollutant being degraded, but typically include carbon dioxide, water, and other innocuous compounds.

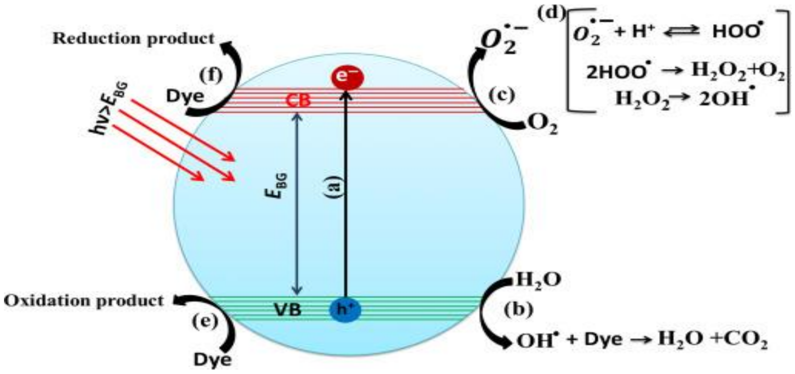

Reduction product
(f)
Dye
$h\nu > E_{BG}$
CB
e^-
E_{BG}
(a)
VB
h^+
Oxidation product
(e)
Dye
$O_2^{\bullet-}$
(d)
$O_2^{\bullet-} + H^+ \rightleftharpoons HOO^\bullet$
$2HOO^\bullet \rightarrow H_2O_2 + O_2$
$H_2O_2 \rightarrow 2OH^\bullet$
(c)
O_2
H_2O
(b)
$OH^\bullet$ + Dye $\rightarrow H_2O + CO_2$

Figure 2: Mechanism of electron-hole pair formation in a TiO_2 particle in the presence of pollutant in water [11].

Steps of photocatalytic reactions

The photocatalytic process can be divided into five independent steps [2]:

1. Porosity of the photocatalyst;
2. Adsorption of pollutants on the surface of the catalyst;
3. Chemical reaction in the adsorbed phase;
4. Desorption of the products;
5. Diffusive migration of the products from the surface of the catalyst to the fluid phase.

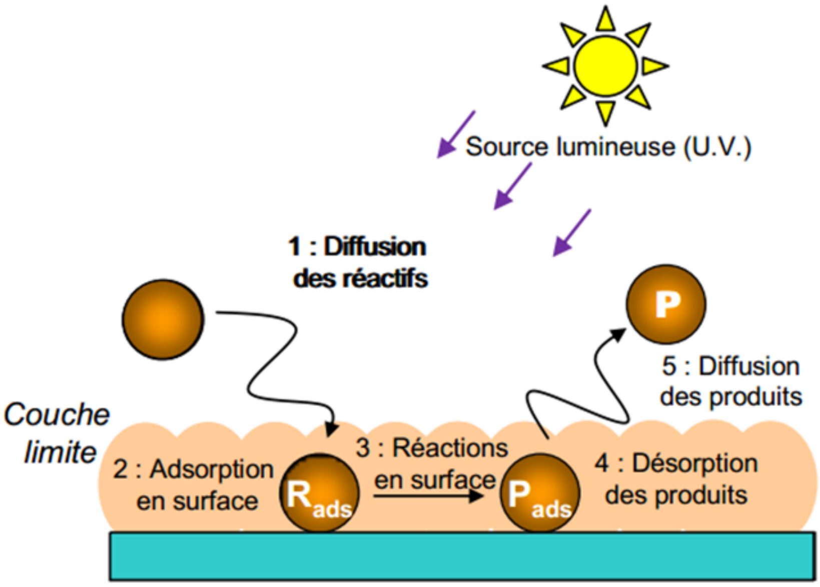


Figure 3: Steps of photocatalytic reactions [12].

Reaction kinetics and mass transfer in photocatalytic reactors

Reaction kinetics and mass transfer are important factors to consider in the design and optimization of photocatalytic reactors.

Reaction Kinetics

Reaction kinetics is the study of the rate at which chemical reactions occur and the factors that influence that rate. In the context of photocatalysis, reaction kinetics can provide insights into the underlying mechanisms of the photocatalytic process and help optimize the design of photocatalytic reactors.

The reaction kinetics of photocatalytic processes can be complex and may involve multiple intermediate steps and reactive species. Therefore, a comprehensive understanding of the reaction kinetics often requires a combination of experimental and theoretical approaches, such as kinetic modeling, spectroscopy, and surface science. [13].

The main reactions that occur

Semiconductor $(\mathrm{TiO_2}) + hv \rightarrow h^+ + e^-$

The holes react with water and organic pollutants adsorbed on the surface of the semiconductor, following the reactions:

$Pollutant + H_2O + hvVB \rightarrow HO^{\cdot} - + H^+$

$Pollutant + h^+VB \rightarrow pollutant^+$

The hydroxyl radicals formed also participate in the degradation of pollutants:

$$Pollutant^{+} + HO^{\cdot}- \rightarrow CO_2 + H_2O$$

The excitation of the semiconductor can be either electrical or photochemical in nature [2].

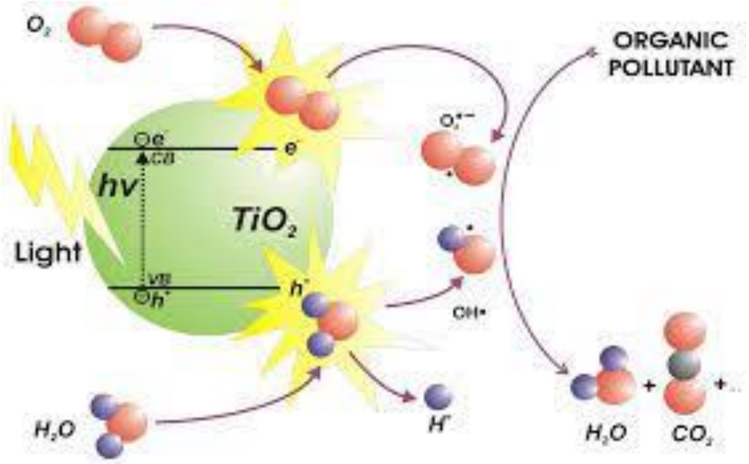


Figure 4: Reaction Kinetics[14].

Photocatalytic reactor types and configurations

Mass Transfer

Mass transfer is the movement of molecules or particles from one location to another in a system. In the context of photocatalytic reactors, mass transfer is a critical factor that affects the efficiency and performance of the photocatalytic process.[15]

The transport of reactants and products to and from the photocatalytic surface is influenced by various factors, including the physical and chemical properties of the catalyst, the properties of the reactants and products (such as molecular weight, size, and solubility), the operating conditions (such as temperature, pressure, and flow rate), and the geometry and design of the reactor. [16].

Photocatalytic reactors are devices used to harness the power of light to catalyze chemical reactions. There are several types and configurations of photocatalytic reactors, each with its own advantages and disadvantages. Here are some examples:

Batch reactors: These are the simplest type of photocatalytic reactors, in which illuminated with UV or visible light. However, they have low efficiency and are not suitable for continuous production [15].

Flow reactors: The flow reactors have higher efficiency than batch reactors and can be easily scaled up for industrial applications. However, they are more expensive and complex to operate [16].

Monolithic reactors: These reactors consist of a single piece of material, such as glass or ceramic. Monolithic reactors are highly efficient and can be used for continuous production, but they are difficult to fabricate and can be prone to clogging [17].

Slurry reactors: Slurry reactors have high efficiency and are easy to operate, but they require frequent replacement of the photocatalyst and can be difficult to scale up for industrial applications [16].

Fixed-bed reactors: Fixed-bed reactors contain a packed bed of photocatalyst particles, through which the reactants flow. Fixed-bed reactors are highly efficient and can be used for continuous production, but they can suffer from catalyst deactivation and fouling [15].

1.4 Applications of photocatalytic reactors

Photocatalytic reactors have a wide range of applications in environmental remediation, energy conversion, and chemical synthesis. Some of the key applications of photocatalytic reactors include:

Water purification: Photocatalytic reactors can be used to remove various pollutants from water, such as organic compounds, heavy metals, and microorganisms. They are particularly effective for the degradation of organic pollutants, such as pesticides, pharmaceuticals, and dyes [17].

And there are other keys applications like: Air purification, Hydrogen production, Energy storage, Chemical synthesis, Self-cleaning surfaces.

Parameters influencing photocatalysis

There are several parameters influencing photocatalysis including [18]:

Effect of oxygen;

Effect of pH;

Effect of temperature;

Effect of pollutant concentration;

Effect of catalyst concentration;

Effect of energy hv;

Photocatalytic degradation of Methylene Blue (BM)

Photocatalysis is a promising method for purifying wastewater. Several studies have been conducted using suspended catalysts, but our study consists in the use of the material TiO_2 /5% curcumin (catalyst) immobilized on a substrate, cellulose paper, to degrade and eliminate a dye, Methylene Blue (BM), by a dynamic regime [20].

A study model of a photocatalytic device has been constructed and optimized within the laboratory of Physical Engineering at the Faculty of SM of Ibn Khaldun University in Tiaret.

Figure 5 presents the decoloration test of the BM solution performed by this implementation.

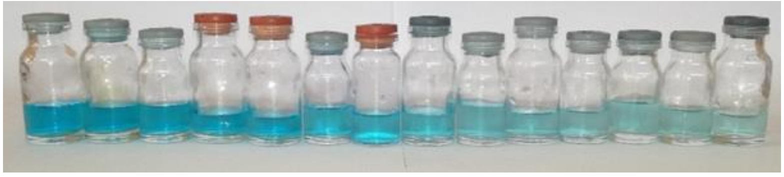


Figure 5: Decoloration test of the BM solution conducted in the presence of the F1 film[19].

And here are some results that represent some photocatalytic effects.

Degradation of the pollutant (BM) on film F1

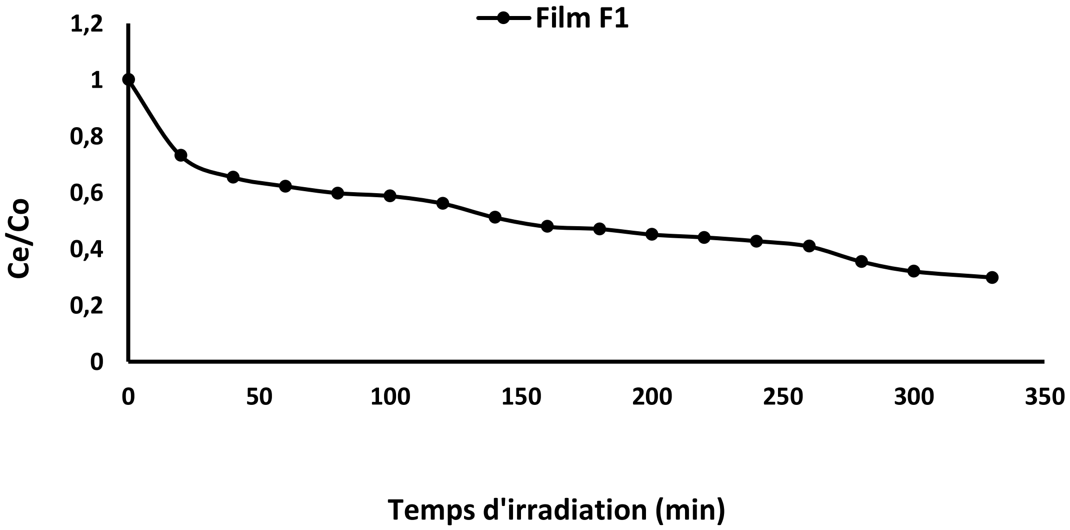


Figure 6: The curve of Ce/C0 as a function of irradiation time for film F1 alone [19].

The degradation process of BM on film F1 occurs due to the photocatalytic effect of the catalyst. When the film is irradiated with light, it generates electron-hole pairs that participate in redox reactions on the surface of the film. These reactions generate reactive oxygen species that are capable of oxidizing the pollutant BM into harmless by products such as carbon dioxide and water. The efficiency of the degradation process is influenced by several factors such as the concentration of the pollutant, the intensity of the light, and the mass of the deposited material on the substrate. Higher pollutant concentrations and lower light intensities can lead to slower degradation rates. However, increasing the mass of the deposited material can lead to an increase in the surface area available for the redox reactions to occur, which can increase the degradation rate [20].

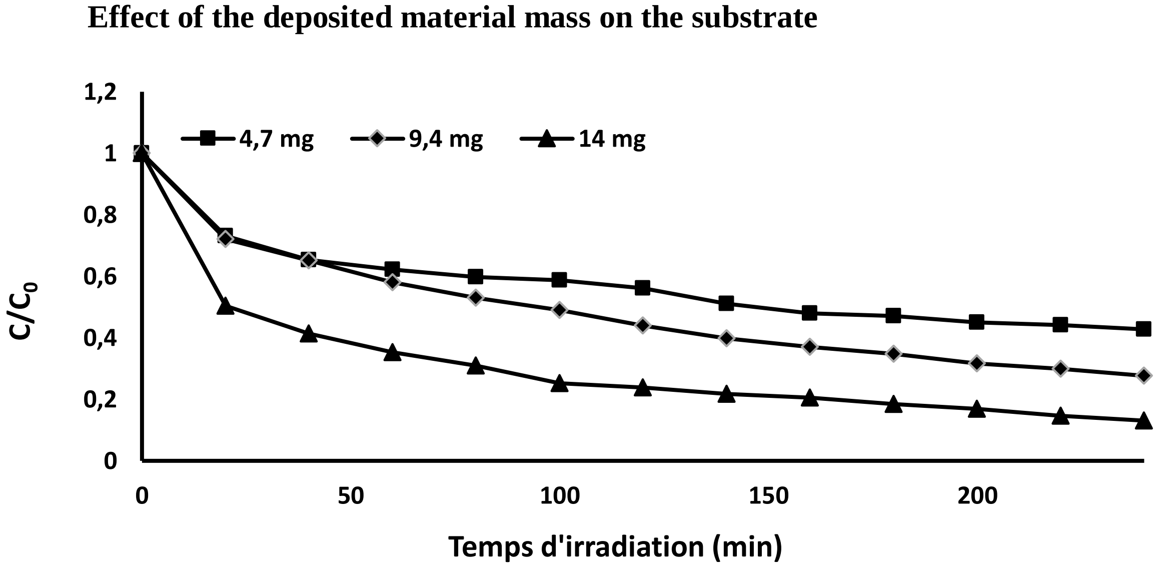


Figure 7: Effect of the deposited material mass on the substrate [19].

Figure 7 depicts the degradation kinetics of BM at different masses of TiO_2/5% curcumin

material deposited on the substrate.

The deposited material mass on the substrate is an important factor that can significantly affect the properties and performance of a material. In the context of photocatalysis, the amount of catalyst material deposited on the substrate can have a significant impact on the degradation rate of pollutants[21].

Increasing the deposited material mass can lead to an increase in the surface area available for photocatalysis to occur. This can result in more active sites for the adsorption and oxidation of pollutants. As a result, the degradation rate of pollutants may increase with an increase in the deposited material mass[22].

Effect of pollutant concentration (BM)

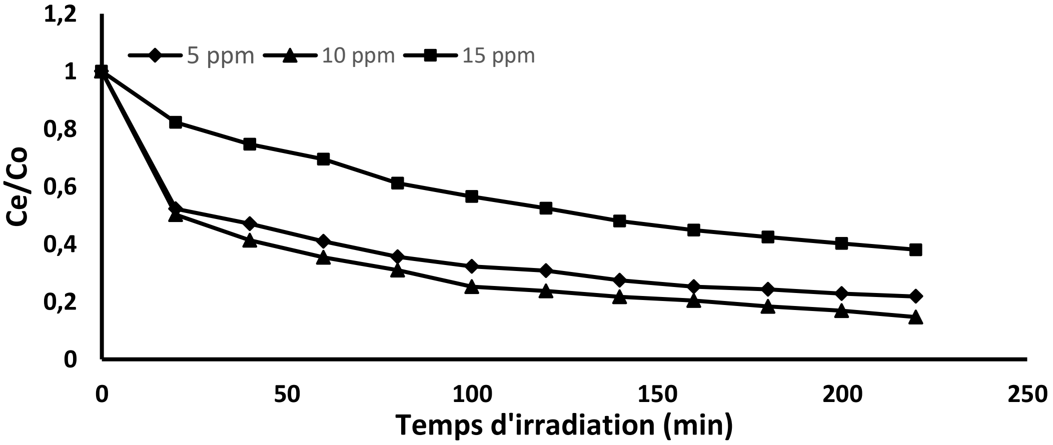


Figure 8: Effect of pollutant concentration (BM) [19].

Figure 7 depicts the effect of varying the initial pollutant concentration (BM) on the type of photocatalytic degradation kinetics is illustrated in Figure 7. Three concentrations were chosen: 5 ppm, 10 ppm, and 15 ppm.

The decrease in photocatalytic efficiency at higher pollutant concentrations can be attributed to several factors. Firstly, at high pollutant concentrations, the photocatalyst may become saturated, leading to a decrease in the rate of pollutant degradation. Secondly, high pollutant concentrations may lead to the formation of reaction intermediates that can inhibit the photocatalytic reaction. Thirdly, high pollutant concentrations may lead to the formation of by-products that can accumulate on the photocatalyst surface, leading to a decrease in its activity.

Effect of light intensity

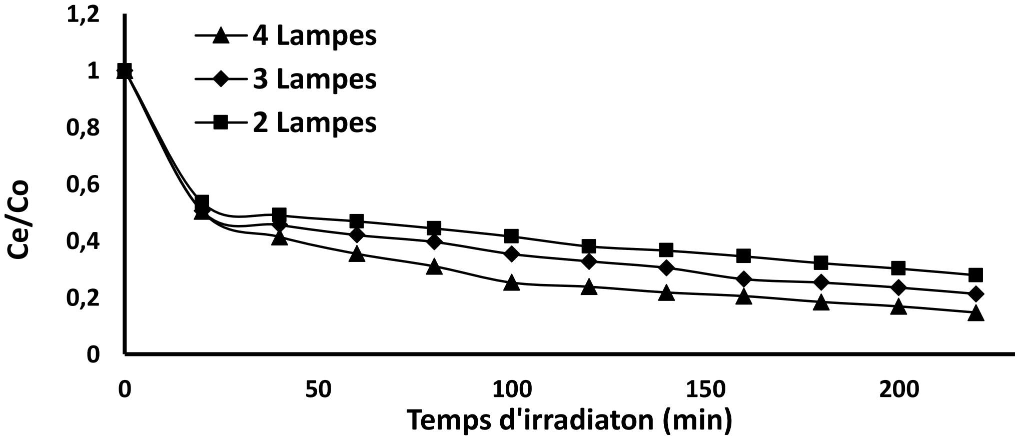


Figure 9: Effect of light intensity [19].

Figure 9 represents the degradation of BM as a function of UV light intensity, where the mass of TiO_2 /5% curcumin material deposited on the substrate was fixed at 14 mg, and the initial pollutant concentration (BM) was set at 10 ppm. Three tests were conducted using 4 UV lamps, 3 UV lamps, and 2 UV lamps, respectively.

The increase in photocatalytic efficiency with increasing light intensity can be attributed to several factors. Firstly, higher light intensities can increase the formation of electron-hole pairs, which can react with pollutants more efficiently. Secondly, higher light intensities can improve the penetration of light into the reactor, leading to more uniform illumination of the photocatalyst. Thirdly, higher light intensities can reduce the recombination of electron-hole pairs, leading to longer lifetimes and increased reaction rates [23].

Conclusion

This chapter has provided a comprehensive overview of the vital importance of photocatalytic reactors in addressing the global water scarcity challenge. We have delved into intricate details regarding their operational mechanisms and impressive capability to degrade organic pollutants, establishing their critical role in tackling the worldwide water crisis. With their ability to efficiently break down organic pollutants and purify water, photocatalytic reactors have emerged as a valuable and promising technological solution. However, in order to fully harness their potential, continuous scientific and technological advancements are necessary. This entails ongoing research, collaborative efforts among various stakeholders, and dedicated initiatives to optimize reactor efficiency, manufacturing costs, and sustainability. By pushing the boundaries of this innovative technology, we can effectively secure a sustainable water supply for future generations while preserving this invaluable resource.

CHAPTER II

Deep Learning

2.1 Introduction

Artificial intelligence occupies a central position in the field of digitalization, generating significant interest in the digital world. Its primary resource is data, which it uses to learn and improve its performance. The more diverse, rich, and high-quality the data, the better the AI can develop accurate and high-performing models. Data collection can occur through various means, including sensors, connected devices, databases, or even public online sources.

Among the different techniques of AI, machine learning plays a crucial role. In recent years, machine learning has experienced exponential growth in data analysis, enabling applications to make intelligent decisions using algorithms that analyze and learn from data. Additionally, deep learning, considered a pillar of AI, machine learning, and data science, has garnered significant interest in the computing domain due to its ability to learn from complex data.

This chapter will provide an exciting exploration of the world of AI, machine learning, and deep learning, serving as an essential foundation for our study of implementation and design in the final chapter.

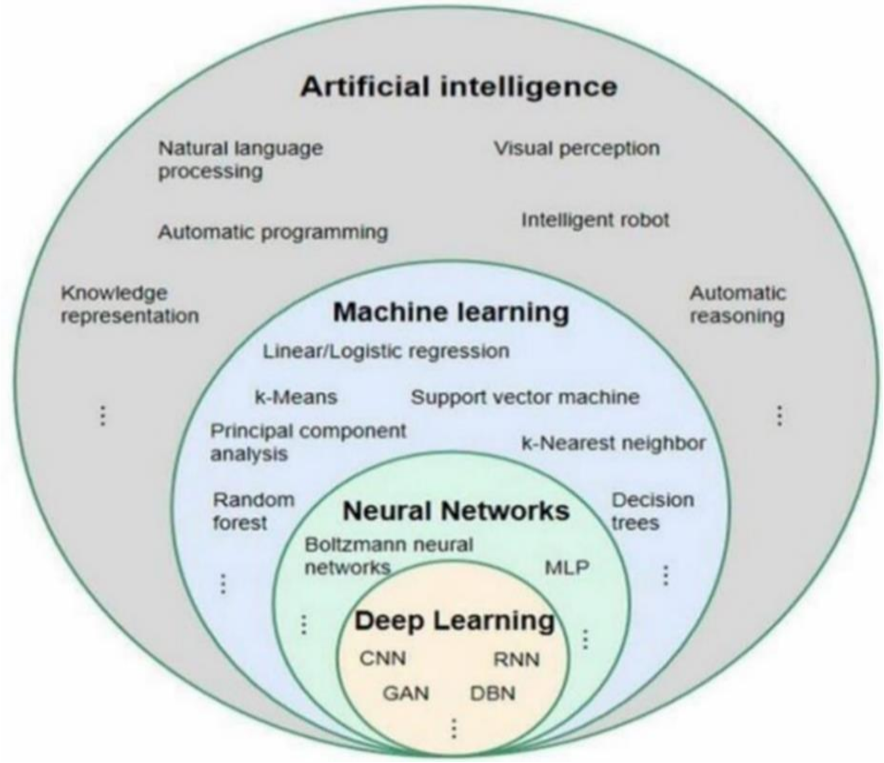


Figure 10: An illustration of the position of deep learning (DL), machine learning (ML) and artificial intelligence (AI) [24].

History of Neural Network and Deep Learning Models

Neural networks and deep learning models based on the years and developments mentioned in the figure 11:

1984: Boltzmann machines are introduced for unsupervised learning and feature extraction.

1985: Boltzmann machines are trained using Gibbs sampling by Hinton and Sejnowski.

1986: Backpropagation algorithm for training feedforward neural networks is introduced by Rumelhart, Hinton, and Williams.

1991: Mixture of Experts is introduced as a way to combine multiple neural networks with different architectures.

2006: Deep learning with autoencoders and neural networks is used to improve speech recognition accuracy.

2008: t-SNE (t-Distributed Stochastic Neighbor Embedding) is introduced as a technique for visualizing high-dimensional data.

2009: Deep Belief Networks (DBNs) are introduced as a way to train deep neural networks using unsupervised pre-training.

2010: Rectified Linear Units (ReLU) are introduced as a way to improve the training of Restricted Boltzmann Machines (RBMs).

2011: Recurrent Neural Networks (RNNs) are used for Natural Language Processing (NLP) tasks such as language modeling and machine translation.

2012: RMSProp is introduced as a way to improve stochastic gradient descent, and AlexNet achieves state-of-the-art performance on the ImageNet challenge, using a deep neural network architecture.

2013: RNNs are used for speech recognition tasks, achieving state-of-the-art performance.

2014: The CIFAR-10 dataset is introduced as a benchmark for image classification tasks.

2015: Distillation Networks are introduced as a way to transfer knowledge from large, complex models to smaller, more efficient models.**2016:** Layer normalization is introduced as a way to improve the training of deep neural networks.

2022: The forward-forward algorithm is introduced as a new way to perform probabilistic inference in neural networks, allowing for more accurate predictions in uncertain environments.[25]

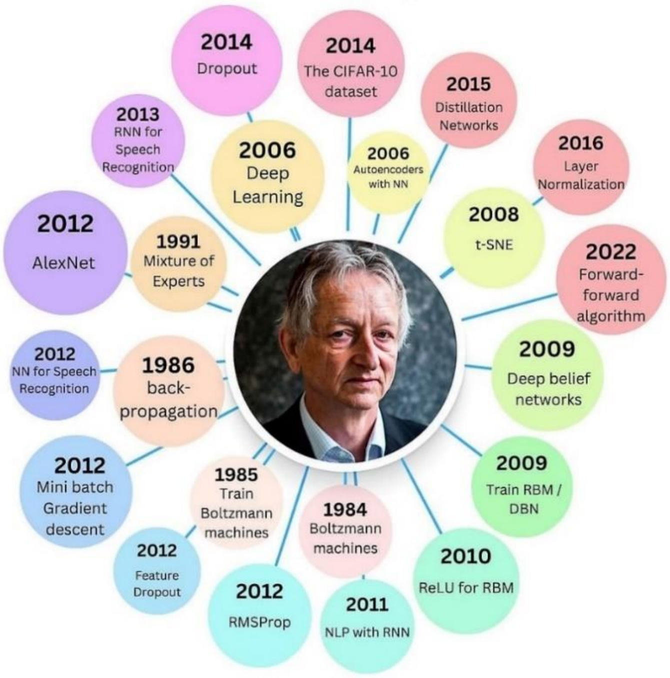

2014 Dropout
2014 The CIFAR-10 dataset
2015 Distillation Networks
2013 RNN for Speech Recognition
2006 Deep Learning
2006 Autoencoders with NN
2016 Layer Normalization
2008 t-SNE
2012 AlexNet
1991 Mixture of Experts
2022 Forward-forward algorithm
2012 NN for Speech Recognition
1986 back-propagation
2009 Deep belief networks
2012 Mini batch Gradient descent
1985 Train Boltzmann machines
1984 Boltzmann machines
2009 Train RBM / DBN
2012 Feature Dropout
2012 RMSProp
2011 NLP with RNN
2010 ReLU for RBM

Figure 11: History of Neural Network and Deep Learning since 1884.

2.2 Artificial Intelligence

Artificial intelligence (AI) is a broad field of computer science and engineering focus on modeling a wide range of topics and functions in the domain of human intelligence. It is mostly about understanding and performing intelligent tasks such as acquire new skills, think and adapt to new situations and challenges. We explore different types of AI, including analytical, functional, interactive, textual, and visual.

2.3 Machine Learning

Machine learning is a subfield of artificial intelligence, which is broadly defined as the capability of a machine to imitate intelligent human behavior. Artificial intelligence systems are used to perform complex tasks in a way that is similar to how humans solve problems.

Machine learning is one way to use AI. It was defined in the 1950s by AI pioneer Arthur Samuel as “the field of study that gives computers the ability to learn without explicitly being programmed”[26].

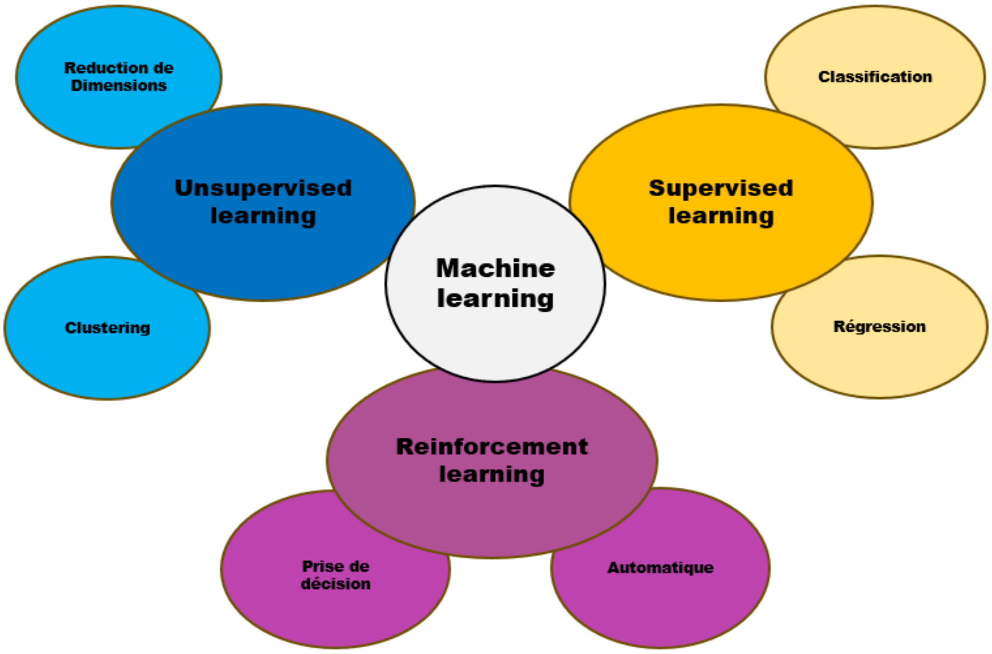


Figure 12: Machine Learning [27].

2.3.1,Methods of Machine Learning

Supervised learning

A problem in which a model is used to learn a representation between input examples and a target variable is represented by supervised learning[28]. Supervised learning problems are known as systems where the training data contains examples of input vectors and the target vectors that correspond to them. There are two major types of problems with supervised learning: classification involving detection of regression and a class mark involving detection of a significant value [29]. Classification is represented as a supervised problem of learning which requires the prediction of a class label. Regression is a problem of supervised learning involving predicting a numerical label [30]. Algorithms are related to as supervised because, when an input data is given, they learn by making predictions, and those models are controlled and improved by an approach that can help determine the outcome [31]. Some methods can be perfectly suited for classification (e.g., logistic regression) or regression (e.g., linear regression), while some are employed for both types of problems with minor modifications (such as artificial neural networks)[31].

Unsupervised learning

Unsupervised learning identifies some difficulties involving in the use of data relationship model that describes or removes data relationships. Unsupervised learning works in comparison with supervised learning, only the input data is used, with no outputs or target variables [31]. As such, unsupervised learning close to supervised learning doesn't have an instructor to correct the model. There are several ways of unsupervised learning, but they have two key issues which a practitioner frequently encounters: clustering involves grouping the data and estimating range, which entails a summary of data distribution. Clustering is represented as an unsupervised problem of learning which requires finding data for classes [32].

Reinforcement learning

Reinforcement learning is a set of challenges in which an individual must learn to use feedback to work in a given context. It is identical to supervised learning, even though feedback maybe delayed, and since the model is systematically noisy, it has some responses from which to learn, which finds it challenging for the entity and model to link causality. Deep reinforcement learning, Q-learning, and temporal-difference learning are some common examples of reinforcement learning algorithms [33].

Semi-supervised learning

It can be defined as a hybridization of the supervised and unsupervised methods, as it operates on both labeled and unlabeled data. Thus, it falls between learning without supervision and learning with supervision .in the real world, labeled data could be rare in several contexts, and unlabeled data are numerous, where semi-supervised is useful. The ultimate goal of semi-supervised learning model is to deliver a better prediction result than that obtained from the model's labeled data alone. Some application areas where semi supervised learning is used include machine translation, labeling data, fraud detection and text classification [34].

2.4 Neural networks

A neural network is a type of machine learning that is modeled after the human brain. It consists of layers of interconnected nodes that process information and learn to recognize patterns in data[35].

Artificial Neural Network

Artificial neural network is a subset of machine learning and is the core of deep Learning algorithms. A typical neural network mainly consists of many simple, the connected processing elements or processors are called neurons, and each neuron produces a sequence of real-valued activations for the target outcome [24].

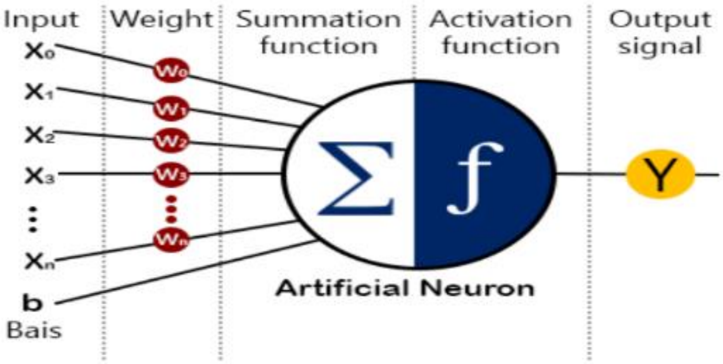


Figure 13: Representation of an artificial neuron's mathematical model[24].

Activation functions

In artificial neural networks, the activation function defines how the weighted sum of the input layer is changed from the input layer to the output layer through the hidden layers in between. If the input provided is large enough, the corresponding neuron is fired and passed to the next network layer. The goal of this activation function is to introduce non-linearity in the network [36].

Types of Activation Function

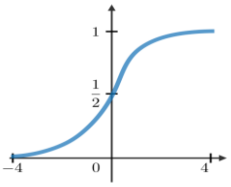
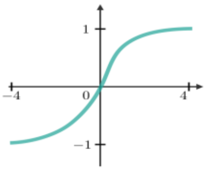
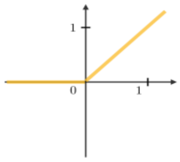
1. Linear Activation Function
2. Binary step function
3. Non-linear Activation Function.

Activation functions are used at the end of a hidden unit to introduce non-linear complexities to the model. Here is a schematic representation of an artificial neuron where:

$$\Sigma = wj, x + bj.$$

Here are the most common one

Table 1: Activation functions[37].

Sigmoid functions	Hyperbolique Tangent function **(Tanh)**	Rectifier functions **ReLU**
$g(z) = \frac{1}{1+e^{-z}}$	$g(z) \;= \frac{e^{z}-e^{-z}}{e^{z}-e^{-z}}$	$g(z) = max(0, z)$
		

Architecture of neural network

According to the connection topology of the neurons, they can be classified into two main categories: non-looped networks (static or feedforward) and looped networks (dynamic, feed back or recurrent) [38].

Multi-layer Perceptron (MLP)

The Perceptron consists of an input layer and an output layer which are fully connected. MLPs have the same input and output layers but may have multiple hidden layers in between the aforementioned layers, as seen below[38].

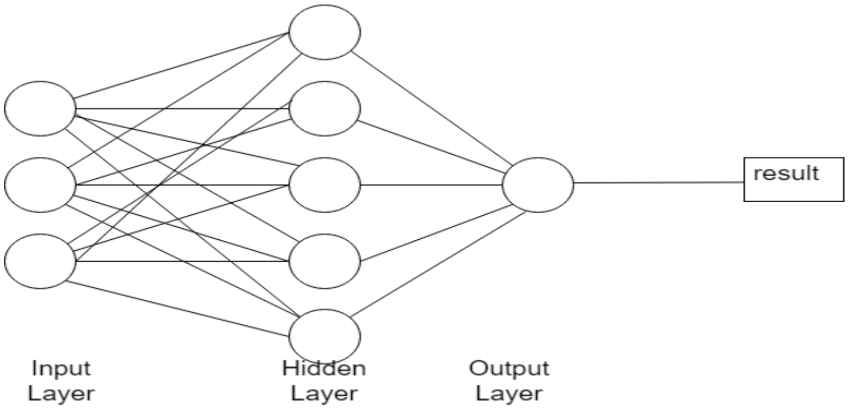


Figure 14 : Multilayer Perceptron[38].

Estimation of the parameters

Once the architecture of the network has been chosen, the parameters (the weights w_j and biases b_j) have to be estimated from a learning sample. As usual, the estimation is obtained by minimizing a loss function with a gradient descent algorithm. We first have to choose the loss function[39].

Loss Function

It is classical to estimate the parameters by maximizing the likelihood (or equivalently the logarithm of the likelihood). This corresponds to the minimization of the loss function which is the opposite of the log likelihood. Denoting θ the vector of parameters to estimate, we consider the expected loss function [39].

$$L(\theta) = -E(X,Y) \sim P\ (log(p\theta(Y/X)).$$

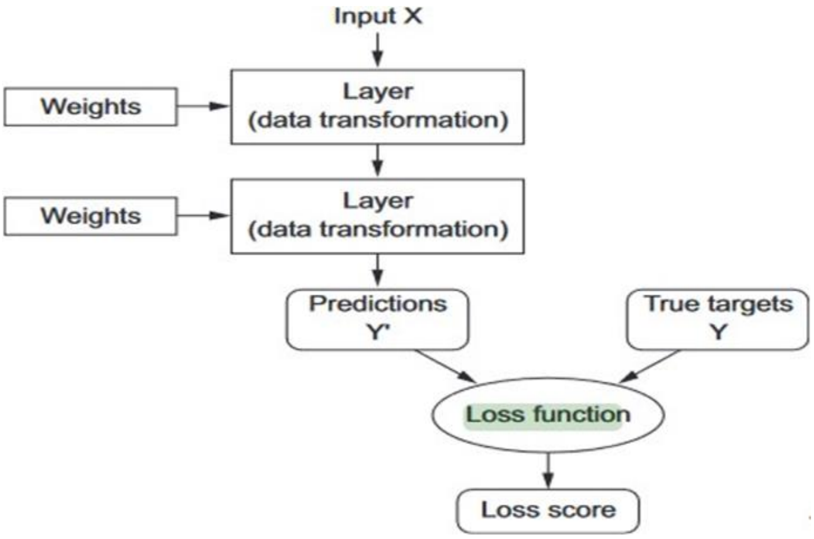


Figure 15: The loss function measures the quality of the output of the neural network[40].

Penalized empirical risk

The expected loss can be written as $L(\theta) = E(X,Y) \sim P\ [`(f(X,\theta),Y)]$ and it is associated to a loss function[39].

Backpropagation

Backpropagation is a method to update the weights in the neural network by taking into account the actual output and the desired output. The derivative with respect to weight (w) is computed using chain rule[39].

Updating weights

In a neural network, weights are updated as follows:

- Step 1: Take a batch of training data,
- Step 2: Perform forward propagation to obtain the corresponding loss,
- Step 3: Backpropagate the loss to get the gradients,
- Step 4: Use the gradients to update the weights of the network[39].

Initialization

The input data have to be normalized to have approximately the same range. The biases can be initialized to 0. The weights cannot be initialized to 0 since for the tanh activation function, the derivative at 0 is 0, this is a saddle point. They also cannot be initialized with the same values, otherwise, all the neurons of a hidden layer would have the same behaviour.

We generally initialize the weights at random: the values $W\ (k)\ i,j$ is $i.i.d.$ Uniform on $[-c,c]$ with possibly $c\ =\frac{\surd\,\sigma}{N_k+N_k-1}$

where N_k is the size of the hidden layer k. We also sometimes initialize the weights with a normal distribution N (0, 0.01) [39].

Optimization algorithms

Many algorithms can be used to minimize the loss function, all of them have hyperparameters, that have to be calibrated, and have an important impact on the convergence of the algorithms. The elementary tool of all these algorithms is the Stochastic Gradient Descent (SGD) algorithm[39].

Stochastic Gradient Descent algorithm (SGD)

Gradient descent is an iterative algorithm, that starts from a random point on a function and travels down its slope in steps until it reaches the lowest point of that function[39].

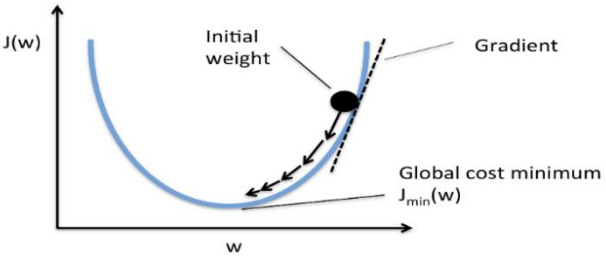


Figure 16: Stochastic Gradient Descent algorithm (SGD)[41].

Momentum

Momentum is a technique used in the training of neural networks to speed up the learning process and help avoid getting stuck in local minima. In the context of neural networks, the gradient descent algorithm is used to update the weights and biases of the network to minimize the cost function [39].

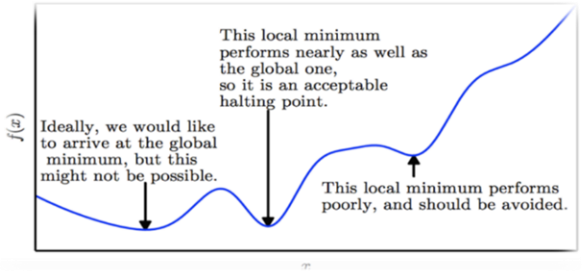
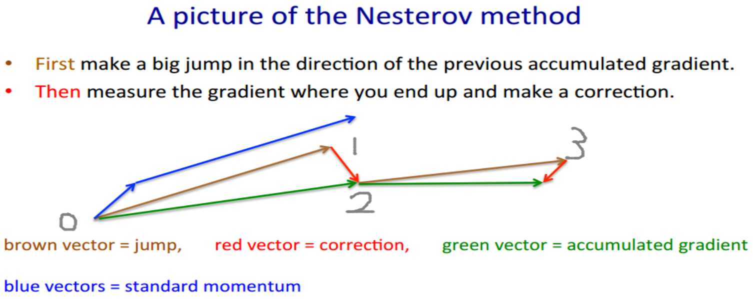


Figure 17: Momentum.

Nesterov Accelerated Gradient (NAG)

One of the most famous is the RMSProp algorithm or Adam (for Adaptive Moments) algorithm.



A picture of the Nesterov method

- First make a big jump in the direction of the previous accumulated gradient.
- Then measure the gradient where you end up and make a correction.

brown vector = jump, red vector = correction, green vector = accumulated gradient

blue vectors = standard momentum

Figure 18: Nesterov method [42].

Dropout

Dropout is a technique meant to prevent overfitting the training data by dropping out units in a neural network [37].

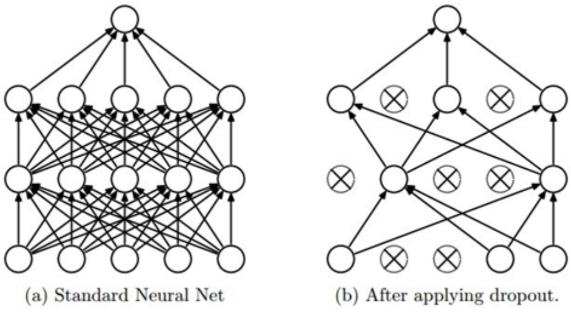


Figure 19: Neural net with and without dropout[43].

Data preparation

The preparation of data is indeed an integral part of the parameter estimation process in machine learning and statistical modeling. When estimating model parameters, it is crucial to ensure that the data used for estimation is suitable and appropriately prepared. Here's how data preparation relates to parameter estimation:

Data Cleaning: The parameter estimation process requires that the dataset be free of any inconsistencies. This step is crucial to maintaining data integrity and reducing the likelihood of bias.

Data Transformation: Modifying variables to match particular distributions, applying logarithmic or power transformations, or scaling and normalizing features are some of the transformations that may be necessary.

Outlier Detection and Handling: Part of the data preparation process is identifying and appropriately handling outliers, as they can greatly impact parameter estimation and model performance.

Feature Engineering: Creating new features or selecting relevant ones from the existing dataset is what feature engineering is all about.

Data Splitting: Dataset is divided into three sets: training, validation, and testing, prior to estimating model parameters. The estimation of model parameters is done based on the training set while model selection or tuning is done based on the validation set. To evaluate the final performance of the model, the testing set is used.

By appropriately preparing the data, researchers aim to improve the accuracy, reliability, and generalizability of the estimated parameters.

2.5 Deep Learning

Deep learning is a field of research in machine learning that is based on a particular type of learning mechanism. It is characterized by the effort to create a multi-level learning model in which the deeper levels take into account the outcomes of the previous levels, transforming them and abstracting them further. This hierarchical learning approach is inspired by how the brain processes information and learns by reacting to external stimuli. Each level of learning is assumed to correspond to one of the different areas that make up the cerebral cortex [44].

Deep Learning Properties and Dependencies

A Deep Learning model typically follows the same processing stages as machine learning modeling. In Figure 2, we have shown a deep learning workflow to solve real-world problems, which consists of three processing steps, such as data understanding and preprocessing, DL model building, and training, and validation and interpretation[44].

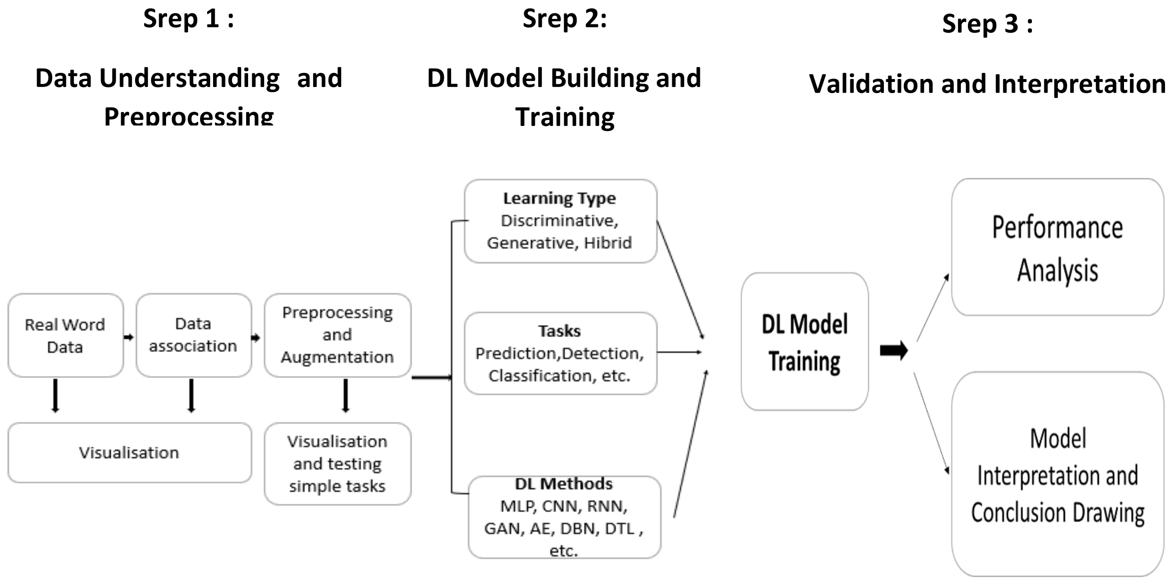

Srep 1 :
Data Understanding and Preprocessing
Srep 2:
DL Model Building and Training
Srep 3 :
Validation and Interpretation
Real Word Data
Data association
Preprocessing and Augmentation
Visualisation
Visualisation and testing simple tasks
Learning Type
Discriminative, Generative, Hibrid
Tasks
Prediction,Detection, Classification, etc.
DL Methods
MLP, CNN, RNN, GAN, AE, DBN, DTL , etc.
DL Model Training
Performance Analysis
Model Interpretation and Conclusion Drawing

Figure 20*:* A typical DL workfow to solve real-world problems[24].

which consists of three sequential stages (i) data understanding and preprocessing (ii) DL model building and training (iii) validation and interpretation.

In the following, we discuss the key properties and dependencies of DL techniques, that are needed to take into account before started working on DL modeling for real-world application.

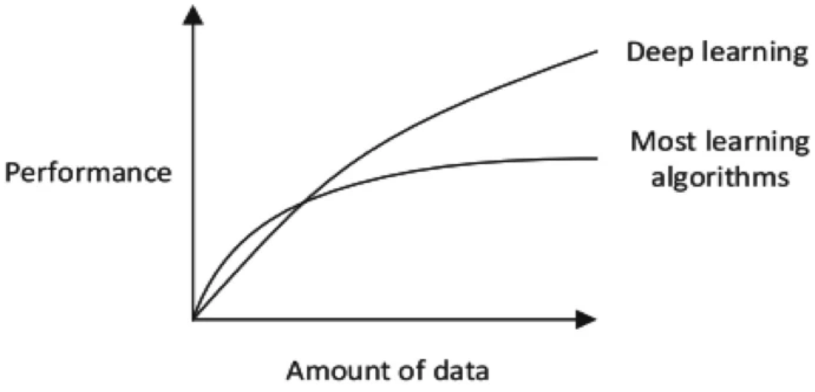


Figure 21: An illustration of the performance comparison between deep learning (DL) and other machine learning (ML) algorithms, where DL modeling from large amounts of data can increase the performance[24].

Deep Learning Techniques

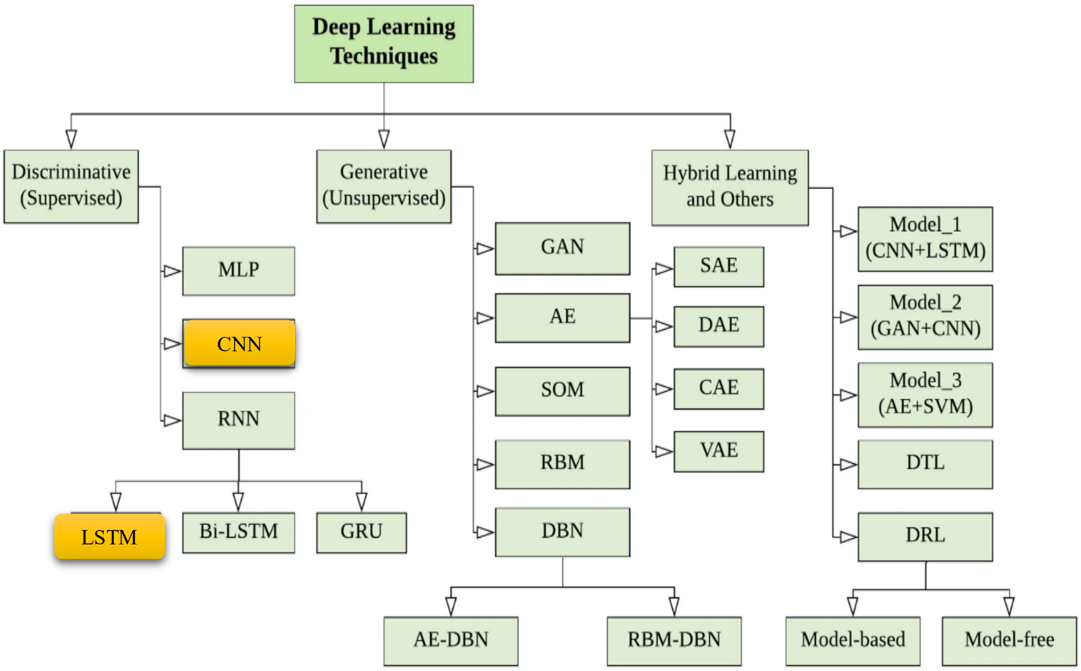


Figure 22*:* A taxonomy of DL techniques[24].

In the following, we briefly discuss one of these techniques shown in Figure 25 according to their learning capabilities.

Deep Networks for supervised or discriminative Learning

Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are deep learning algorithms that are particularly well-suited for processing structured data in the form of matrices or arrays. They are designed to automatically extract meaningful features from the data using convolutional operations.

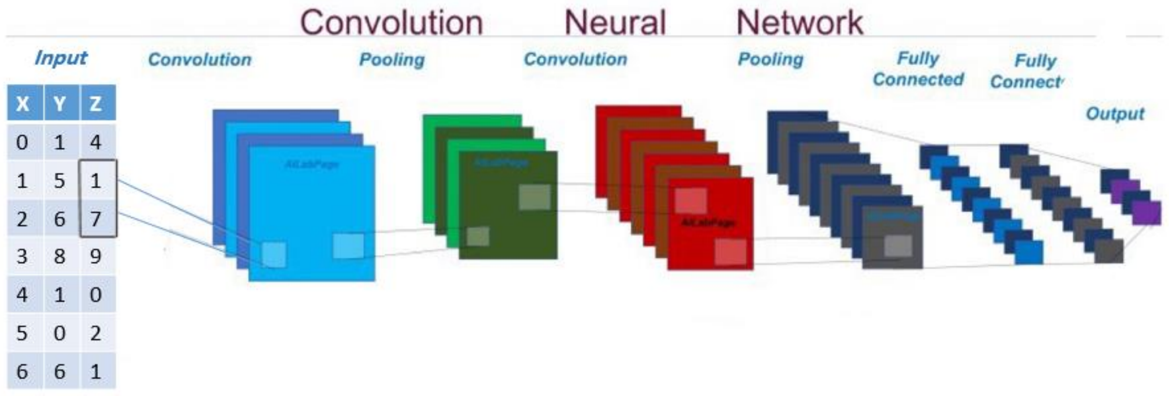
CNNs can handle different types of data based on their dimensionality. 3D CNNs are capable of processing three-dimensional data, such as videos or volumetric images. 2D CNNs are used for two-dimensional data, such as images or audio spectrograms. Lastly, 1D CNNs are suitable for one-dimensional data, such as signals or sequences.

The structure of a CNN typically consists of convolutional layers, pooling layers, and fully connected layers. The convolutional layer is responsible for feature extraction from the input data. It applies a set of filters to the input matrix, where each filter performs a convolution operation by calculating the dot product between the filter and a portion of the matrix. This convolution operation helps detect local patterns in the data.

After the convolutional layer, the results are usually passed through a pooling layer. This layer aims to reduce the dimensionality of the extracted features while preserving essential information. It performs an aggregation operation, such as maximum or average pooling, over specific regions of the matrix.

Next, fully connected layers are used, where each neuron is connected to all neurons in the previous layer. These layers are similar to those in traditional neural networks and allow for the combination of the extracted features to generate final outputs. Finally, an output layer is used to produce the desired results, such as classifications or predictions.

Training of CNNs is done through error backpropagation. The weights of the filters and layers in the network are iteratively adjusted to minimize the difference between the network's predictions and the ground truth associated with the training data. This process allows the network to adapt to the specific characteristics of the data and generalize to new data, improving its performance[37].

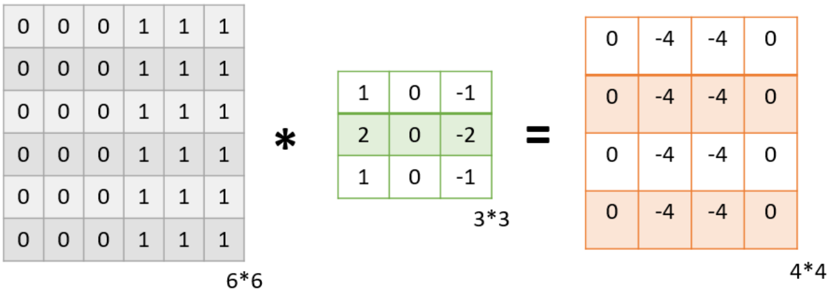


X	Y	Z
0	1	4
1	5	1
2	6	7
3	8	9
4	1	0
5	0	2
6	6	1

Figure 23: An example of a convolutional neural network.

Convolutional layer

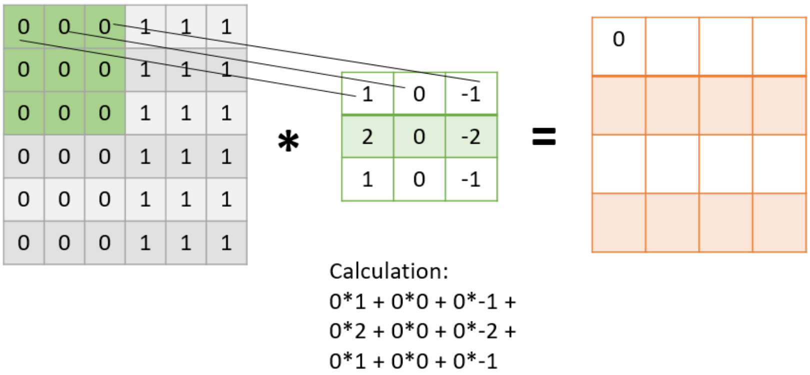
By noting (W) the input volume size, (F) the size of the convolutional layer neurons, (P) the amount of zero padding, then the number of neurons (N) that fit in a given volume is such that: $N = ((W - F + 2P)/\,(S))\ + 1$; [37].



$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 0 & -4 & -4 & 0 \\ 0 & -4 & -4 & 0 \\ 0 & -4 & -4 & 0 \\ 0 & -4 & -4 & 0 \end{bmatrix}$$

Figure 24: Convolutional layer.

Let’s get into some maths behind getting the feature map in the above image.

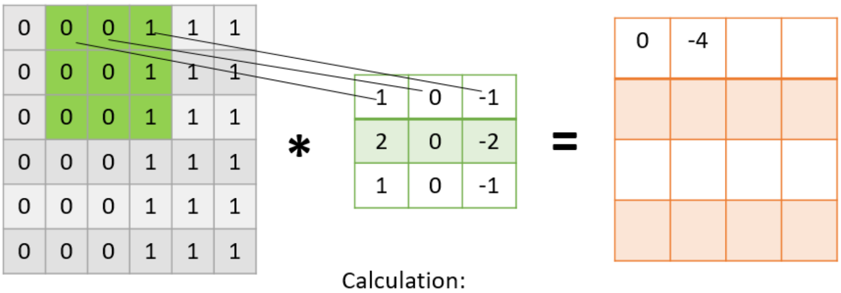


$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 0 & & & \\ & & & \\ & & & \\ & & & \end{bmatrix}$$

Calculation:
0*1 + 0*0 + 0*-1 +
0*2 + 0*0 + 0*-2 +
0*1 + 0*0 + 0*-1

As presented in the above figure, in the first step the filter is applied to the green highlighted part of the image, and the pixel values of the image are multiplied with the values of the filter (as shown in the figure using lines) and then summed up to get the final value.

In the next step, the filter is shifted by one column as shown in the below figure. This jump to the next column or row is known as stride and in this example, we are taking a stride of 1 which means we are shifting by one column[37].



0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1

*

1	0	-1
2	0	-2
1	0	-1

=

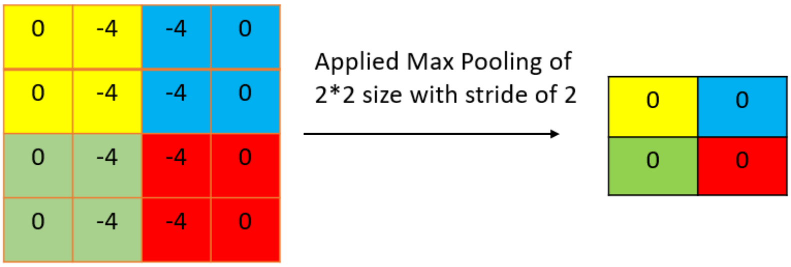
0	-4		

Calculation:
0*1 + 0*0 + 1*-1 +
0*2 + 0*0 + 1*-2 +
0*1 + 0*0 + 1*-1

Pooling Layer

The pooling layer is applied after the Convolutional layer and is used to reduce the dimensions of the feature map which helps in preserving the important information or features of the input image and reduces the computation time.

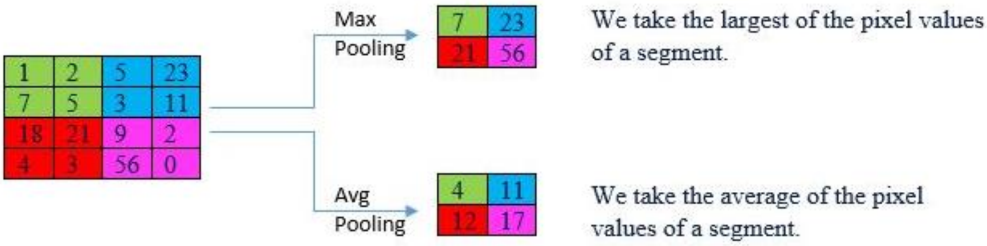
The most common types of Pooling are Max Pooling and Average Pooling. The below figure shows how Max Pooling works[37].



0	-4	-4	0
0	-4	-4	0
0	-4	-4	0
0	-4	-4	0

Applied Max Pooling of 2*2 size with stride of 2

0	0
0	0



1	2	5	23
7	5	3	11
18	21	9	2
4	3	56	0

7	23
21	56

We take the largest of the pixel values of a segment.

4	11
12	17

We take the average of the pixel values of a segment.

Fully Connected Layer

This layer connects the information extracted from the previous steps (Convolution layer and Pooling layers) to the output layer and eventually classifies the input into the desired label as shown in Figure 25.

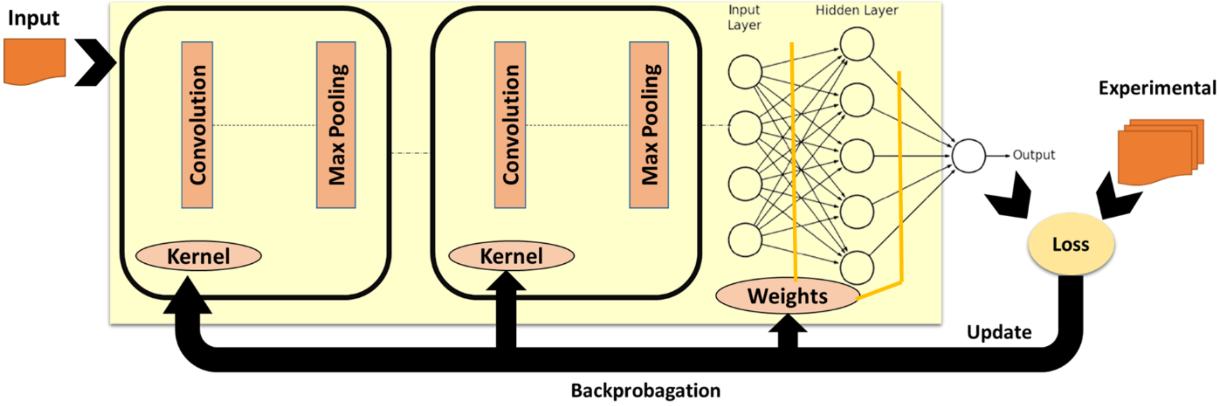


Figure 25 : The fully connected layer

Recurrent Neural Networks

A Recurrent Neural Network (RNN) is another popular neural network, which employs sequential or time-series data and feeds the output from the previous step as input to the current stage[45]. Like feedforward and CNN, recurrent networks learn from training input, however, distinguish by their "memory", which allows them to impact current input and output through using information from previous inputs. Unlike typical DNN, which assumes that inputs and outputs are independent of one another, the output of RNN is reliant on prior elements within the sequence. However, standard recurrent networks have the issue of vanishing gradients, which makes learning long data sequences challenging [24].RNN works on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer.

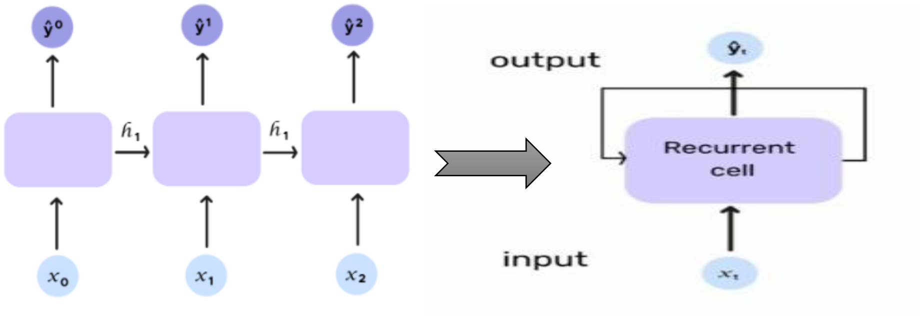


Figure 26: Recurrent Neural Network[46].

Working of Recurrent Neural Network

In Recurrent Neural networks, the information cycles through a loop to the middle-hidden layer.

The input layer 'x' takes in the input to the neural network and processes it and passes it onto the middle layer.

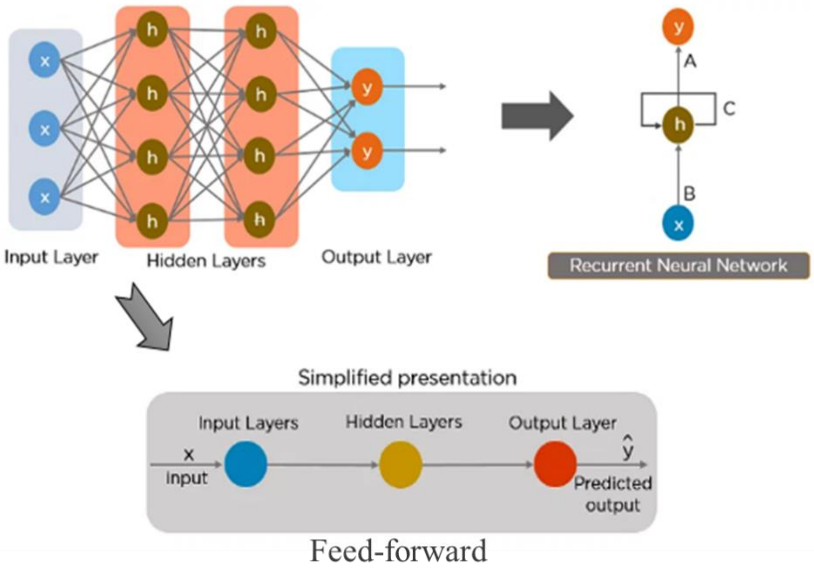


Figure 27: The main difference between a RNN and a feedforward network.

A feed-forward neural network allows information to flow only in the forward direction, from the input nodes, through the hidden layers, and to the output nodes. There are no cycles or loops in the network.

In a feed-forward neural network, the decisions are based on the current input. It doesn't memorize the past data, and there's no future scope. Feed-forward neural networks are used in general regression and classification problems[46].

The different applications of RNNs

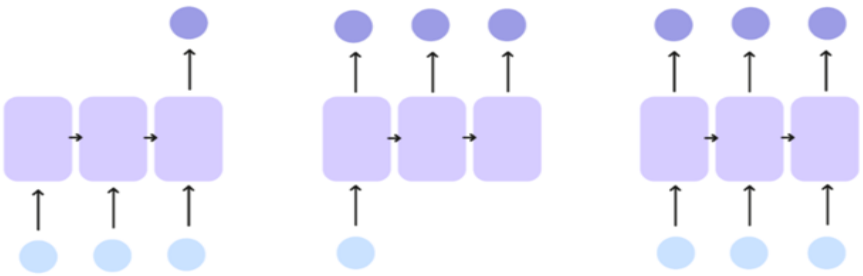


Figure 28: The different applications of RNNs[46].

1. **One to many**, The RNN receives a single input and returns multiple outputs, the classic example of this process is the image legend

2. **Many to one** There are several inputs and there is a single output. An illustration of this mode is the sentiment analysis of texts. This makes it possible to identify a feeling from a group of words, determine the word that is missing to finish the sentence received as input.

3. **Many to many,** Finally, you can take several inputs and get several outputs. We don't necessarily have the same number of input and output neurons. We can cite, here, the translation of text, but we can be ambitious and plan to finish a musical work with its beginning[46].

Two Issues of Standard RNNs

1- Exploding Gradient Problem

While training a neural network, if the slope tends to grow exponentially instead of decaying, this is called an Exploding Gradient. This problem arises when large error gradients accumulate, resulting in very large updates to the neural network model weights during the training process. Long training time, poor performance, and bad accuracy are the major issues in gradient problems[46].

2-Vanishing Gradient Problem

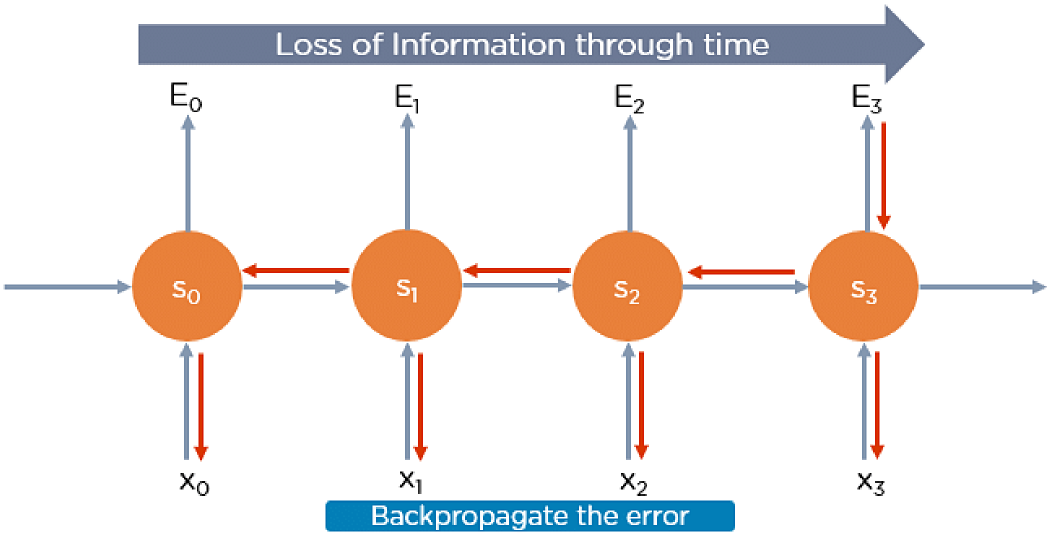
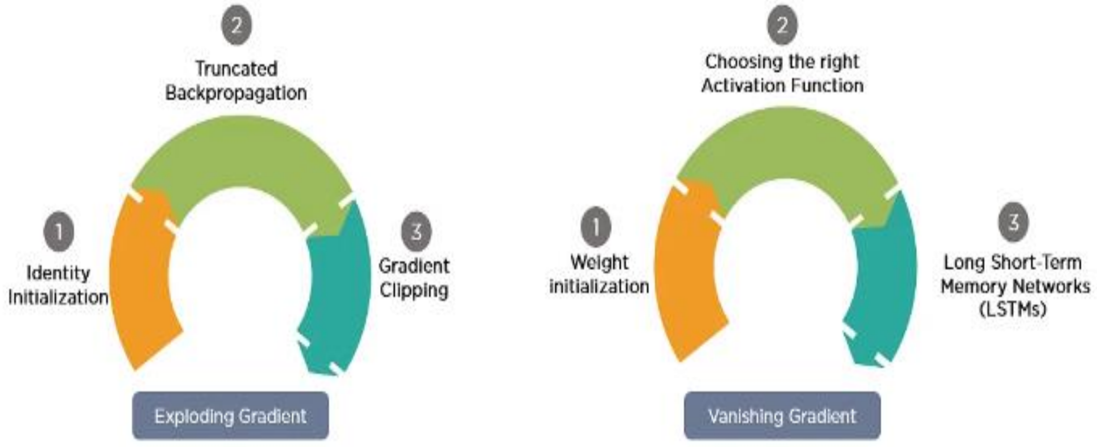


Figure 28: Vanishing Gradient Problem[46].

Recurrent Neural Networks enable you to model time-dependent and sequential data problems, such as stock market prediction, machine translation, and text generation. You will find, however, RNN is hard to train because of the gradient problem.

RNNs suffer from the problem of vanishing gradients. The gradients carry information used in the RNN, and when the gradient becomes too small, the parameter updates become insignificant. This makes the learning of long data sequences difficult[46].

Solution



Variant RNN Architectures

There are several variant RNN architectures that have been developed over the years to address the limitations of the standard RNN architecture. Here are a few examples :

1. Long Short-Term Memory (LSTM) Networks

LSTM is a type of RNN that is designed to handle the vanishing gradient problem that can occur in standard RNNs. It does this by introducing three gating mechanisms that control the flow of information through the network: the input gate, the forget gate, and the output gate. These gates allow the LSTM network to selectively remember or forget information from the input sequence, which makes it more effective for long-term dependencies[46].

2. Gated Recurrent Unit (GRU) Networks

GRU is another type of RNN that is designed to address the vanishing gradient problem. It has two gates: the reset gate and the update gate. The reset gate determines how much of the previous state should be forgotten, while the update gate determines how much of the new state should be remembered. This allows the GRU network to selectively update its internal state based on the input sequence[46].

3. Bidirectional RNNs:

Bidirectional RNNs are designed to process input sequences in both forward and backward directions. This allows the network to capture both past and future context, which can be useful for speech recognition and natural language processing tasks[46].

4. Encoder-Decoder RNNs:

Encoder-decoder RNNs consist of two RNNs: an encoder network that processes the input sequence and produces a fixed-length vector representation of the input and a decoder network that generates the output sequence based on the encoder's representation. This architecture is commonly used for sequence-to-sequence tasks such as machine translation[46].

5. Attention Mechanisms

Attention mechanisms are a technique that can be used to improve the performance of RNNs on tasks that involve long input sequences. They work by allowing the network to attend to different parts of the input sequence selectively rather than treating all parts of the input sequence equally. This can help the network focus on the input sequence's most relevant parts and ignore irrelevant information[46].

Common Activation Functions

Recurrent Neural Networks (RNNs) use activation functions just like other neural networks to introduce non-linearity to their models. Here are some common activation functions used in RNNs:

Sigmoid Function:

The sigmoid function is commonly used in RNNs. It has a range between 0 and 1, which makes it useful for binary classification tasks. The formula for the sigmoid function is[46]:

$$\sigma(x) = 1 / (1 + e^{(-x)}).$$

Hyperbolic Tangent (Tanh) Function:

The tanh function is also commonly used in RNNs. It has a range between -1 and 1, which makes it useful for non-linear classification tasks. The formula for the tanh function is[46]:

$$\text{Tanh}\ (x) = (e^{x} – e^{(-x)})\ /\ (e^{x} + e^{(-x)})).$$

Rectified Linear Unit (Relu) Function:

The ReLU function is a non-linear activation function that is widely used in deep neural networks. It has a range between 0 and infinity, which makes it useful for models that require positive outputs. The formula for the ReLU function is[46]:

$$\mathrm{ReLU}(x) = \max(0, x)$$

Leaky Relu Function:

The Leaky ReLU function is similar to the ReLU function, but it introduces a small slope to negative values, which helps to prevent "dead neurons" in the model. The formula for the Leaky ReLU function is[46]:

$$\text{Leaky ReLU}(x) = \max(0.01x, x)$$

Softmax Function:

The softmax function is often used in the output layer of RNNs for multi-class classification tasks. It converts the network output into a probability distribution over the possible classes. The formula for the softmax function is:

$$\mathrm{softmax}(x) = e^{x} / \sum(e^{x})$$

These are just a few examples of the activation functions used in RNNs. The choice of activation function depends on the specific task and the model's architecture[46].

Backpropagation Through Time

Backpropagation through time is when we apply a Backpropagation algorithm to a Recurrent Neural network that has time series data as its input.

In a typical RNN, one input is fed into the network at a time, and a single output is obtained. But in backpropagation, you use the current as well as the previous inputs as input. This is called a timestep and one timestep will consist of many time series data points entering the RNN simultaneously.

Once the neural network has trained on a timeset and given you an output, that output is used to calculate and accumulate the errors. After this, the network is rolled back up and weights are recalculated and updated keeping the errors in mind[46].

Long short-term memory (LSTM)

A long short-term memory (LSTM) network is a type of RNN model that avoids the vanishing gradient problem by adding 'forget' gates. LSTM is an implementation of the Recurrent Neural Network and was first proposed by Hochreiter et al. in 1997 [41]. Unlike the earlier described feed forward network architectures, LSTM can retain knowledge of earlier states and can be trained for work that requires memory or state awareness. LSTM partly addresses a major limitation of RNN, i.e., the problem of vanishing gradients by letting gradients to pass unaltered. As shown in the illustration in the following **Figure 29**.

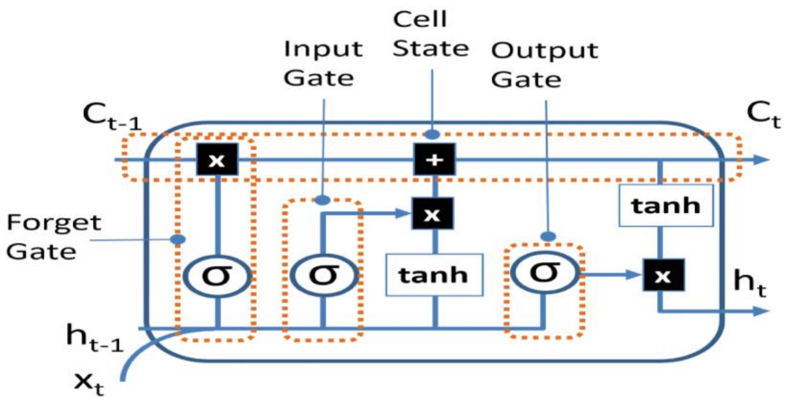
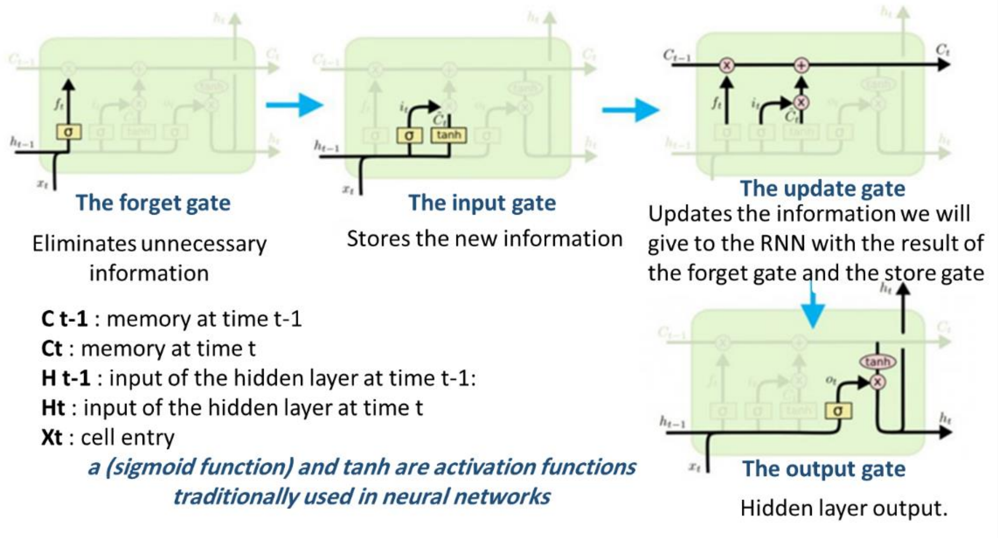


Figure 29: LSTM block with memory cell and gates[43].

Workings of LSTMs in RNN



The forget gate

Eliminates unnecessary information

The input gate

Stores the new information

The update gate

Updates the information we will give to the RNN with the result of the forget gate and the store gate

The output gate

Hidden layer output.

C t-1 : memory at time t-1
Ct : memory at time t
H t-1 : input of the hidden layer at time t-1:
Ht : input of the hidden layer at time t
Xt : cell entry

a (sigmoid function) and tanh are activation functions traditionally used in neural networks

Figure 30: How LSTMs work.

LSTMs work in a 3-step process.

Step 1: Decide How Much Past Data It Should Remember

The first step in the LSTM is to decide which information should be omitted from the cell in that particular time step. The sigmoid function determines this. It looks at the previous state (ht-1) along with the current input xt and computes the function.

$f_t = \sigma\ (\ W_f .\ (h_{t-1}\ , x_t\ ) + b_f\ )$

f_t = Forget gate, decide which information to delete that is not important from previous time step.

Step 2: Decide How Much This Unit Adds to the Current State

In the second layer, there are two parts. One is the sigmoid function, and the other is the tanh function. In the sigmoid function, it decides which values to let through (0 or 1). tanh function gives weightage to the values which are passed, deciding their level of importance (-1 to 1).

$i_t = \sigma\ (\ W_i .\ (h_{t-1}\ , x_t\ ) + b_i\ )$

$C_t = tanh\ (\ W_C .\ (h_{t-1}\ , x_t\ ) + b_C\ )$

i_t = input gate, determines which information to let through based on its significance in the current time step.

Step 3: Decide What Part of the Current Cell State Makes It to the Output

The third step is to decide what the output will be. First, we run a sigmoid layer, which decides what parts of the cell state make it to the output. Then, we put the cell state through tanh to push the values to be between -1 and 1 and multiply it by the output of the sigmoid gate.

$o_t = \sigma\ (\ W_o .\ (h_{t-1}\ , x_t\ ) + b_o\ )$

h_t= o_t * tanh (C_t)

o_t = output gate, allows the passed in information to impact the output in the current time step.

Types of Gates

Here are the different types of gates that we encounter in a typical recurrent neural network:

Input gate	Forget gate	Gate	Output gate
Write to cell or not?	Erase a cell or not?	How much to write to cell?	How much to reveal cell?

Tableau 1: Types of gates [37].

In standard LSTM, a typical LSTM cell is made of input, forget, and output gates and a cell activation component. These units receive the activation signals from different sources and control the activation of the cell by the designed multipliers, as described below [26].

Advantages of Recurrent Neural Network

Recurrent Neural Networks (RNNs) have several advantages over other types of neural networks, including:

- **Ability To Handle Variable-Length Sequences:**

  RNNs are designed to handle input sequences of variable length, which makes them well-suited for tasks such as speech recognition, natural language processing, and time series analysis.
- **Memory Of Past Inputs:** RNNs have a memory of past inputs, which allows them to capture information about the context of the input sequence. This makes them useful for tasks such as language modeling, where the meaning of a word depends on the context in which it appears.
- **Parameter Sharing:** RNNs share the same set of parameters across all time steps, which reduces the number of parameters that need to be learned and can lead to better generalization.
- **Non-Linear Mapping:** RNNs use non-linear activation functions, which allows them to learn complex, non-linear mappings between inputs and outputs.
- **Sequential Processing:** RNNs process input sequences sequentially, which makes them computationally efficient and easy to parallelize.
- **Flexibility:** RNNs can be adapted to a wide range of tasks and input types, including text, speech, and image sequences.
- **Improved Accuracy:** RNNs have been shown to achieve state-of-the-art performance on a variety of sequence modeling tasks, including language modeling, speech recognition, and machine translation.

These advantages make RNNs a powerful tool for sequence modeling and analysis, and have led to their widespread use in a variety of applications, including natural language processing, speech recognition, and time series analysis[46].

Disadvantages of Recurrent Neural Network

Although Recurrent Neural Networks (RNNs) have several advantages, they also have some disadvantages. Here are some of the main disadvantages of RNNs

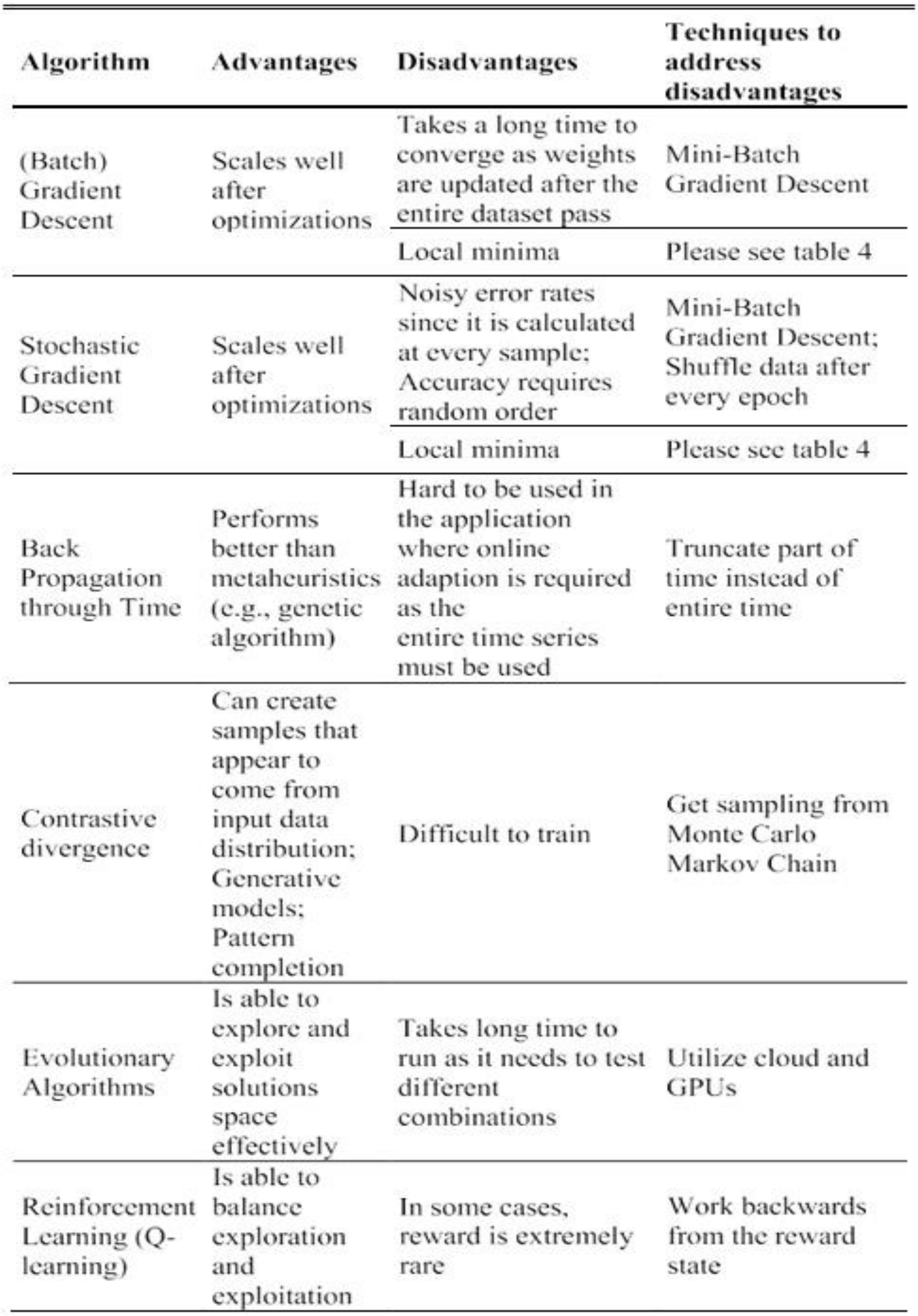
- Vanishing And Exploding Gradients: RNNs can suffer from the problem of vanishing or exploding gradients, which can make it difficult to train the network effectively. This occurs when the gradients of the loss function with respect to the parameters become very small or very large as they propagate through time.
- Computational Complexity: RNNs can be computationally expensive to train, especially when dealing with long sequences. This is because the network has to process each input in sequence, which can be slow.
- Difficulty In Capturing Long-Term Dependencies: Although RNNs are designed to capture information about past inputs, they can struggle to capture long-term dependencies in the input sequence. This is because the gradients can become very small as they propagate through time, which can cause the network to forget important information.
- Lack Of Parallelism: RNNs are inherently sequential, which makes it difficult to parallelize the computation. This can limit the speed and scalability of the network.
- Difficulty In Choosing the Right Architecture: There are many different variants of RNNs, each with its own advantages and disadvantages. Choosing the right architecture for a given task can be challenging, and may require extensive experimentation and tuning.
- Difficulty In Interpreting the Output: The output of an RNN can be difficult to interpret, especially when dealing with complex inputs such as natural language or audio. This can make it difficult to understand how the network is making its predictions.
- These disadvantages are important when deciding whether to use an RNN for a given task. However, many of these issues can be addressed through careful design and training of the network and through techniques such as regularization and attention mechanisms[46].

Comparison of DNN networks

Tableau 3: DNN network comparison table [43].

Netwok type	Architecture	Network model	Training Type	Training Algorithm	Common Application	Popular Dataset Sample
Feed Forward Neural Network	CNN	Discrim-inaive	Supervisde	Gradient descent based backprop-agation	Image recognition/Classification	MNIST
	Residual network	Discrim-inaive	Supervised	Gd based backprop-agation	Image recognition	ImageNet
	Autoencoder	Generative	Unsupervised	backprop-agation	Dimensionality Reduction; Encoding	MNIST
	Adversarial Networks	Discrim-inaive/Generative	Supervised	backprop-agation	Generate realistic fake data	CIFAR10
	RBM	Generative with Discrim-inaive finetuning	Unsupervised	Gd based Contrastive divergence	Dimentionallity Reduction ; Feature learning	MNIST
Recurrent nural network	LSTM	Discrim-inaive	Supervised	Gd and based backprop-agation through Time	NLP; Language Translation	MNIST Stoke Sequence
Radial Basis Function NN	RBT Network	Discrim-inaive	Super-vised /Unsupervised	k-means Clustering ; Least -square Function	Function approximation	Fisher's Iris data set
Kohonen selfOrganizing	Nodes arranged in hexagonal	Generative	Unsupe-rvised	Competitive Learning	Dimentionallity Reduction ;	SPAMbase

Comparison of deep learning algorithms



Algorithm	Advantages	Disadvantages	Techniques to address disadvantages
(Batch) Gradient Descent	Scales well after optimizations	Takes a long time to converge as weights are updated after the entire dataset pass	Mini-Batch Gradient Descent
		Local minima	Please see table 4
Stochastic Gradient Descent	Scales well after optimizations	Noisy error rates since it is calculated at every sample; Accuracy requires random order	Mini-Batch Gradient Descent; Shuffle data after every epoch
		Local minima	Please see table 4
Back Propagation through Time	Performs better than metaheuristics (e.g., genetic algorithm)	Hard to be used in the application where online adaption is required as the entire time series must be used	Truncate part of time instead of entire time
Contrastive divergence	Can create samples that appear to come from input data distribution; Generative models; Pattern completion	Difficult to train	Get sampling from Monte Carlo Markov Chain
Evolutionary Algorithms	Is able to explore and exploit solutions space effectively	Takes long time to run as it needs to test different combinations	Utilize cloud and GPUs
Reinforcement Learning (Q-learning)	Is able to balance exploration and exploitation	In some cases, reward is extremely rare	Work backwards from the reward state

Tableau 4: Comparaison of deep learning algorithms[43].

In **Tableau4**, we have compareded between different Deep learning algorithms.

In **Tableau 3**, we have compareded between different Deep Neural Network (DNN).

2.6 Applications of Deep Learning

During the past few years, deep learning has been successfully applied to numerous problems in many application areas. These include natural language processing, sentiment analysis, cybersecurity, business, virtual assistants, visual recognition, healthcare, robotics, and many more. In Figure 11, we have summarized several potential real-world application areas of deep learning.

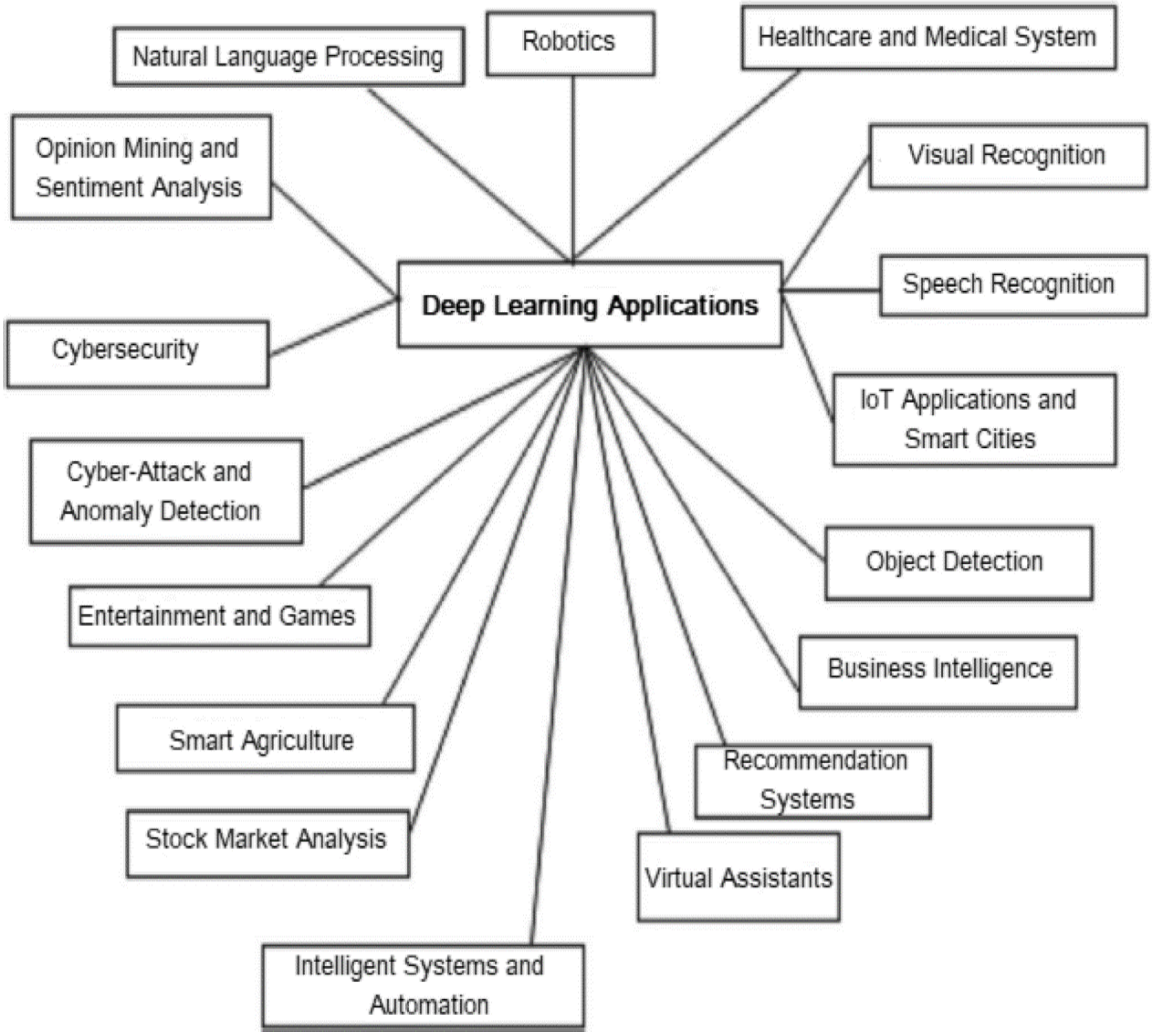

Natural Language Processing
Robotics
Healthcare and Medical System
Opinion Mining and Sentiment Analysis
Visual Recognition
Deep Learning Applications
Speech Recognition
Cybersecurity
IoT Applications and Smart Cities
Cyber-Attack and Anomaly Detection
Object Detection
Entertainment and Games
Business Intelligence
Smart Agriculture
Recommendation Systems
Stock Market Analysis
Virtual Assistants
Intelligent Systems and Automation

Figure 30: Several potential real-world application areas of deep learning[24].

2.7 Challenges and Opportunities

Challenges

As shown in the recent literature, data processing and analysis using deep learning delivers satisfactory performance. Yet a lot of issues have yet to be addressed when leveraging deep learning.

A. **Data Collection**

The performance of deep learning methods relies on data sources. Without sufficient clean data, the deep model cannot play a role, even if the architecture of the model is well designed. Therefore, how to implement the data collection equipment is a critical research issue. A cost effective, reliable, and trustworthy data collection paradigm plays an important role in developing practical deep learning based IoT applications [10].

B. **Model Training**

Training a deep network demands cumbersome tasks. As we know, the depths determine the capacity of a deep learning network to extract key features. However, the gradient vanishment problem appears when models grow deeper, which deteriorates the performance. To this end, Hinton et al. [23] propose an approach to pre-train models by stacking RBMs. In addition, the ReLU function applied as a substitute for the sigmoid function also contributes to the mitigation of the gradient vanishment problem. Overfitting is another serious problem that we face in training deep models. The key solution is to enter more data or reduce parameters of the model.

Opportunities

Despite the challenges, there are still opportunities in applying deep learning to solve problems:

Deep learning liberates our thought

In the past, we may hesitate to step into some unknown areas and encounter difficulties when we carry out some research due to the limitation of related professional knowledge. Now we can achieve some guess without the worry of data analysis tools. Deep learning gives us the ability to obtain and process data information. It means that we can boldly start more research and may promote the process of science and technology.

Deep architectures

Deep architectures have a strong ability in the representation of learning Features that describe characteristics of input data used to be designed manually in traditional methods. Deep learning allows machines to design features by themselves. improve the final system performance.

2.8 Conclusion

This chapter has provided us with an overview of AI-based modeling, highlighting the principles and capabilities of potential techniques such as machine learning and deep learning, which play a crucial role in the development of intelligent and high-performing systems. These techniques pave the way for significant advancements in solving complex problems and contribute to the emergence of smarter systems tailored to the needs of our society. Our specific objective is to apply these advanced AI techniques to our photocatalytic reactor simulation in order to enhance performance and optimize results.

CHAPTER III

Modeling of photocatalytic reactor

3.1 Introduction

By harnessing the power of deep learning, neural networks have the ability to learn from data by adjusting their weights and parameters. This enables them to capture intricate and abstract relationships among variables, which would be challenging using traditional methods. Many real-world problems involve non-linear interactions between variables, making exact analytical solutions unfeasible. Neural networks offer a promising solution by approximating and effectively modeling these non-linear relationships. As a result, neural networks have become a valuable tool for addressing regression problems, where the objective is to find a function that connects input variables to desired outputs.

In this chapter, we will utilize deep learning techniques to analyze data from photocatalytic reactors. We will employ neural networks, specifically CNN and LSTM models, to capture complex relationships and optimize the reactor's performance. Validating and comparing the performance of each model is crucial in order to make an informed decision. Through thorough experimentation and evaluation, we will assess their accuracy, efficiency, and generalization capabilities. By selecting the most promising and suitable model, we can minimize the time and costs associated with experimental trials while effectively achieving our objectives.

We will start by providing an overview of the development environment, programming language And libraries utilized in the implementation of our system.

3.2 Development environment

Google Colab: also known as Colaboratory or 'Colab', is a platform provided by Google that allows users to write and execute Python code directly in their web browser. It is based on Jupyter Notebook and is designed for training and research in machine learning. This platform enables users to train machine learning models directly in the cloud. Colab offers the following capabilities:

- Improve coding skills in the Python programming language.
- Develop deep learning applications using popular Python libraries such as Keras, TensorFlow, PyTorch, and OpenCV.
- Use a development environment (Jupyter Notebook) that requires no configuration.
- However, the standout feature of Colab compared to other services is its access to a free GPU (Graphics Processing Unit), which significantly accelerate the training of machine learning models. As a result, the execution is faster compared to local development environments.

Visual Studio Code (VS Code): is a versatile code editor that can be used for developing machine learning models. It provides a range of features and extensions that make it suitable for working with machine learning frameworks and libraries.

3.3 Programming Language and Libraries

Python: In recent years, Python has become the most widely used programming language among computer scientists. It has gained popularity in infrastructure management, data analysis, and software development. Python allows developers to focus on what they do rather than how they do it. It has freed developers from the constraints of older languages, making coding with Python faster compared to other languages [49].

Used Libraries:

TensorFlow: We utilized TensorFlow to define the basic components of the CNN-LSTM architecture. This library is specifically designed for implementing machine learning and deep learning algorithms. It offers great flexibility for developing neural networks [46].

Keras: Keras is a library that works in conjunction with TensorFlow. We employed Keras to implement different layers, activation functions, and prepare the training dataset [46].

NumPy: We used NumPy to handle input types based on the model's configuration. NumPy is designed for manipulating multidimensional arrays and provides mathematical functions to operate on these arrays. We used it specifically for image scanning and window extraction [47].

Matplotlib: Matplotlib is a Python library used for visualizing and plotting data. We utilized Matplotlib to visualize our images in the form of graphs [58].

Sklearn: Sklearn is one of the most useful libraries for machine learning in Python. It provides various efficient tools for machine learning and statistical modeling, including classification, regression, clustering, and dimensionality reduction [49].

Pandas: Pandas is an open-source data analysis and manipulation tool. It is fast, powerful, flexible, and easy to use. Pandas is built on top of the Python programming language [48].

Tkinter: Tkinter is an open-source portable Graphical User Interface (GUI) library designed to be used in Python scripts. Tkinter is based on the Tk library, which is also used by Tcl/Tk and Perl and is implemented in C. Therefore, we can say that Tkinter is implemented using multiple layers [50].

3.4 The Modeling Approach Followed

To achieve the modeling of the photocatalytic reactor, we followed the following steps:

a) **Data set:** Responsible for acquiring and receiving data from various sources, we receive the dataset from the research laboratory of the Department of Physics at Ibn Khaldoun University in Tiaret. The dataset consists of three sets of conditions: *pollutant concentration, light intensity, and material mass.* Each dataset contains 900 values.

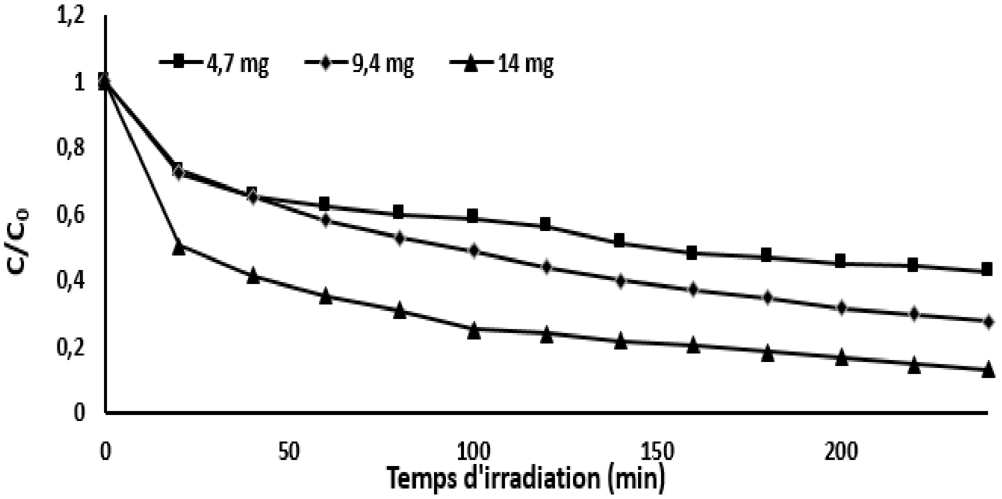


Figure 6: Effect of the deposited material mass on the substrate[16].

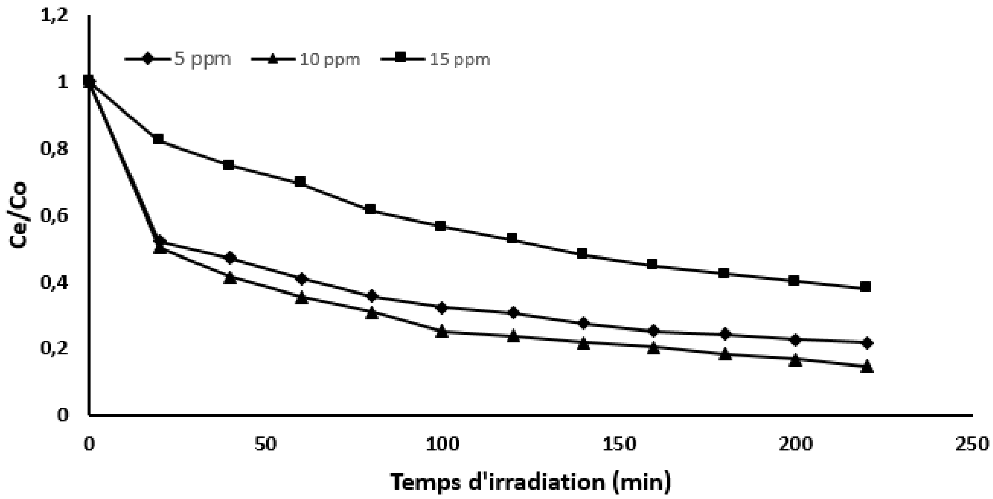


Figure 7: Effect of pollutant concentration (BM)[16].

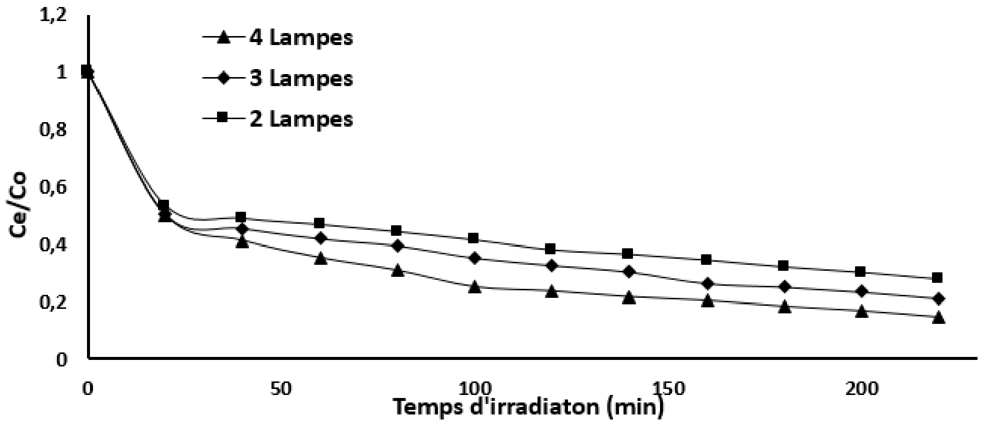


Figure 8: Effect of light intensity[16].

b) Data Processing: Involves processing the inputted data to extract pertinent information and perform necessary transformations. This stage encompasses data cleaning, filtering, and feature extraction to prepare the data for analysis. we normalize the data using the scaler library from Keras machine learning models. This step ensures that the data is on a consistent scale and ready for further analysis and modeling.

c) Deep Learning Models and evaluation:

First, we have developed three CNN sub-models, than the models are evaluated to assess their performance and accuracy. This evaluation involves using separate test datasets that were not part of the training phase, helping measure the models' ability to generalize to unseen data and identify areas for improvement:

c.a) Convolution neural network models:

In the following section, we will introduce the four most successful models that have outperformed the others.

1) CNN material mass sub-model

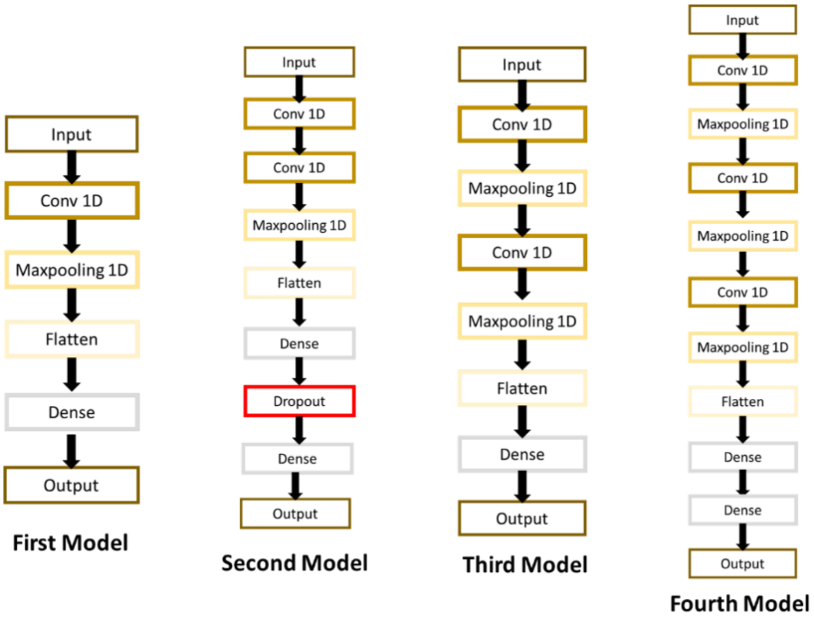


Figure 32: The four most successful models of material mass.

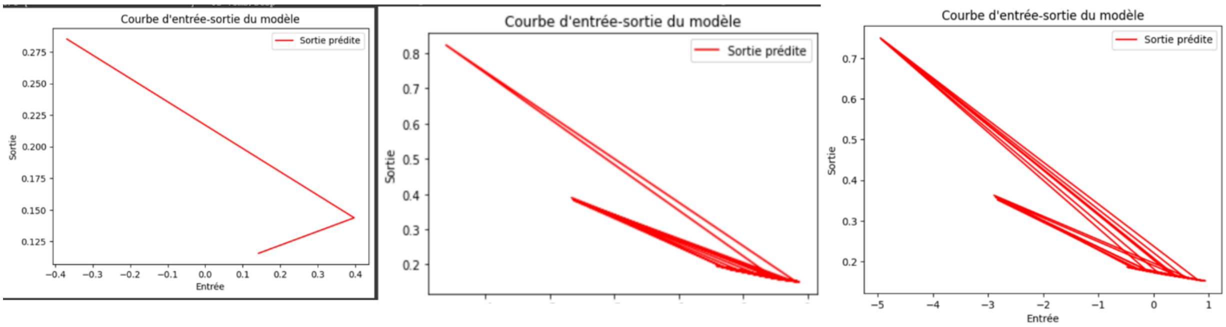


Figure 33: the prediction of most successful models of material mass.

In these graphs, the x-axis represents time and the y-axis represents the predicted pollutant concentration. We can observe that there are certain x-values that have two different predicted y-values. This means that our model has taken values from the same interval of time but produced different predictions, confirming that our model is flawed.

Table 5: A list of hyperparameters in the 1D CNN mass material submodel.

Model		Hyperparameters
1	Convolution layer	Kernel size = 1, filters = 64, strides = 1, padding = valid, activation = Relu
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	connected layer	Neurons = 50, activation function = relu
2	Convolution layer	Kernel size = 1, filters = 64, strides = 1, padding = valid, activation = relu
	Convolution layer	Kernel size = 1, filters = 32, strides = 1, padding = valid, activation = relu
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Connected layer	Neurons = 64, activation function = relu
	Dropout	rate=0.2
	Connected layer	Neurons = 50, activation function = relu
3	Convolution layer	Kernel size = 1, filters = 64, strides = 1, padding = valid, activation =sigmoid
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Convolution layer	Kernel size = 1, filters = 50, strides = 1, padding = valid, activation =relu
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Connected layer	Neurons = 128, activation function = relu
4	Convolution layer	Kernel size = 1, filters = 64, strides = 1, padding = valid, activation =sigmoid
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Convolution layer	Kernel size = 1, filters = 50, strides = 1, padding = valid, activation =relu
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Convolution layer	Kernel size = 1, filters = 32, strides = 1, padding = valid, activation =sigmoid
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Connected layer	Neurons = 50, activation function = relu
	Connected layer	Neurons = 50, activation function = relu

2) CNN pollutant concentration sub-model :

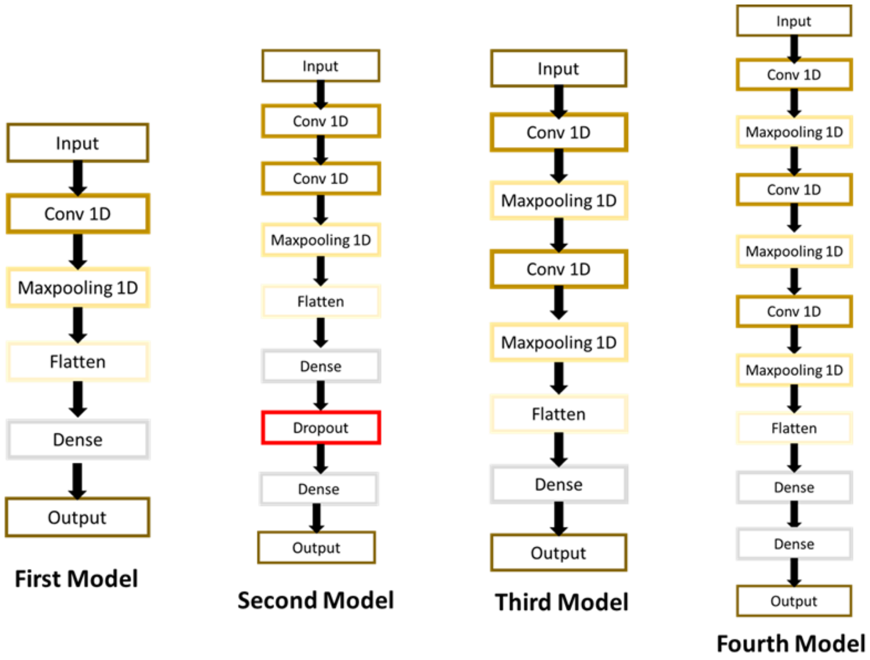


Figure 34: The four most successful models of pollutant concentration.

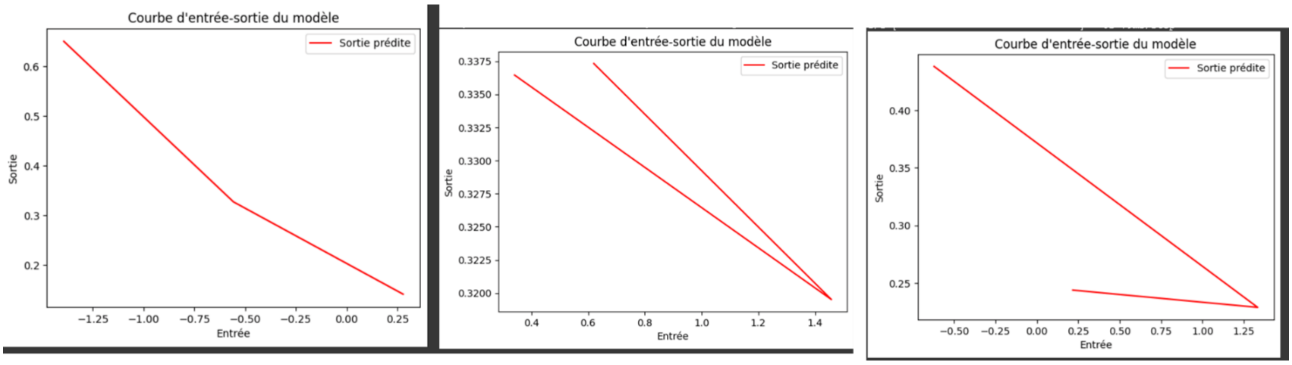


Figure 35: The prediction of most successful models of pollutant concentration.

In these graphs, the x-axis represents time and the y-axis represents the predicted pollutant concentration. We can observe that there are certain x-values that have two different predicted y-values. This means that our model has taken values from the same interval of time but produced different predictions, confirming that our model is flawed.

Table 6: A list of hyperparameters in the 1D CNN pollutant concentration submodel.

Model		Hyperparameters
1	Convolution layer	Kernel size = 1, filters = 64, strides = 1, padding = valid, activation = relu
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	connected layer	Neurons = 50, activation function = relu
2	Convolution layer	Kernel size = 1, filters = 64, strides = 1, padding = valid, activation = relu
	Convolution layer	Kernel size = 1, filters = 32, strides = 1, padding = valid, activation = relu
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Connected layer	Neurons = 64, activation function = relu
	Dropout	rate=0.2
	Connected layer	Neurons = 50, activation function = relu
3	Convolution layer	Kernel size = 1, filters = 64, strides = 1, padding = valid, activation =sigmoid
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Convolution layer	Kernel size = 1, filters = 50, strides = 1, padding = valid, activation =relu
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Connected layer	Neurons = 128, activation function = relu
4	Convolution layer	Kernel size = 1, filters = 64, strides = 1, padding = valid, activation =sigmoid
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Convolution layer	Kernel size = 1, filters = 50, strides = 1, padding = valid, activation =relu
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Convolution layer	Kernel size = 1, filters = 32, strides = 1, padding = valid, activation =sigmoid
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Connected layer	Neurons = 50, activation function = relu
	Connected layer	Neurons = 50, activation function = relu

3) CNN light intensity sub-model

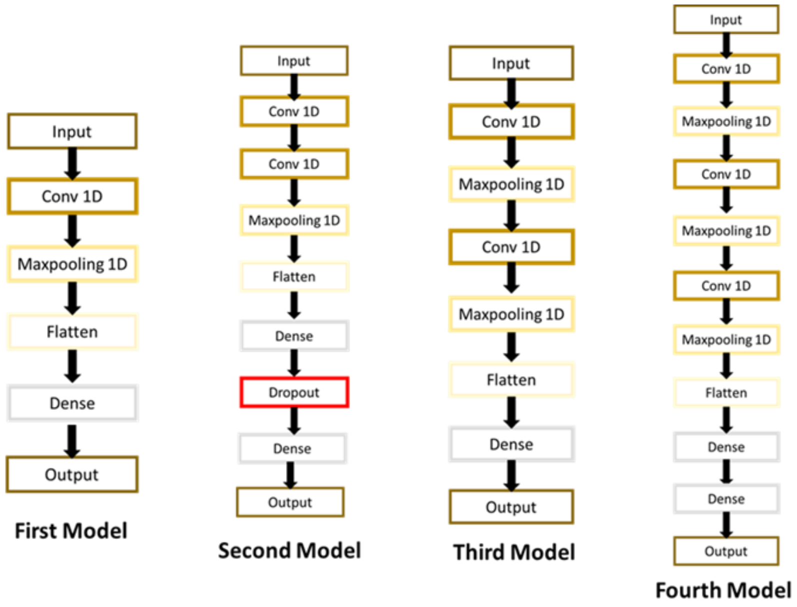


Figure 36: The four most successful models of light intensity.

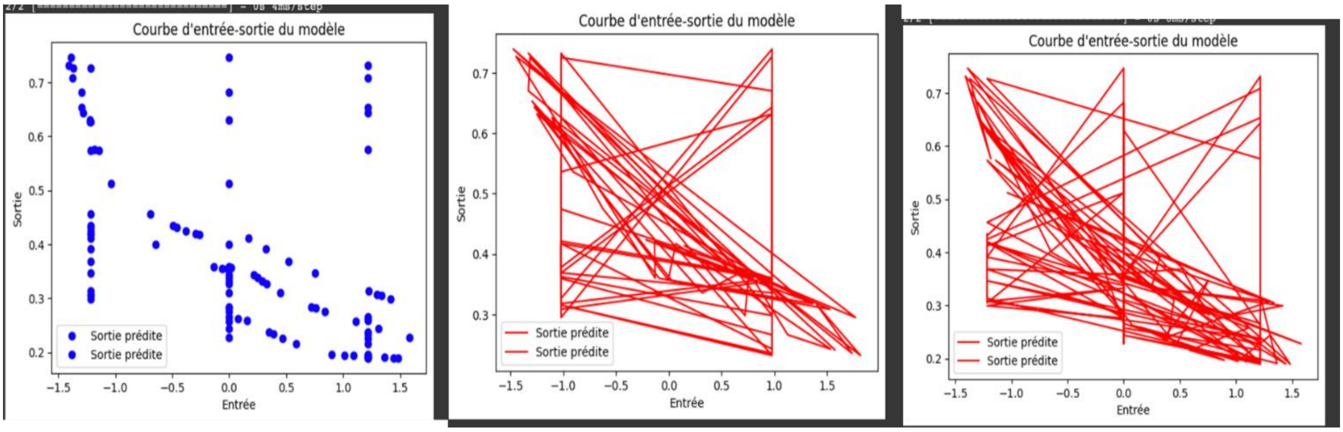


Figure 37: The most successful models of light intensity.

In these graphs, the x-axis represents time and the y-axis represents the predicted pollutant concentration. We can observe that there are certain x-values that have different predicted y-values. This means that our model has taken values from the same interval of time but produced different predictions, confirming that our model is flawed.

Table 7: A list of hyperparameters in the 1D CNN light intensity submodel.

Model		Hyperparameters
1	Convolution layer	Kernel size = 1, filters = 64, strides = 1, padding = valid, activation = relu
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	connected layer	Neurons = 50, activation function = relu
2	Convolution layer	Kernel size = 1, filters = 64, strides = 1, padding = valid, activation = relu
	Convolution layer	Kernel size = 1, filters = 32, strides = 1, padding = valid, activation = relu
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Connected layer	Neurons = 64, activation function = relu
	Dropout	rate=0.2
	Connected layer	Neurons = 50, activation function = relu
3	Convolution layer	Kernel size = 1, filters = 64, strides = 1, padding = valid, activation =sigmoid
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Convolution layer	Kernel size = 1, filters = 50, strides = 1, padding = valid, activation =relu
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Connected layer	Neurons = 128, activation function = relu
4	Convolution layer	Kernel size = 1, filters = 64, strides = 1, padding = valid, activation =sigmoid
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Convolution layer	Kernel size = 1, filters = 50, strides = 1, padding = valid, activation =relu
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Convolution layer	Kernel size = 1, filters = 32, strides = 1, padding = valid, activation =sigmoid
	Poolling layer	method: maxpooling, pool size: 2, stride = 2, padding = valid Fully
	Connected layer	Neurons = 50, activation function = relu
	Connected layer	Neurons = 50, activation function = relu

When dealing with non-linear regressions, convolutional neural networks (CNNs) have certain limitations that should be considered:

- **Modeling complexity:** CNNs are primarily designed for image classification tasks and may not be the most suitable choice for regression problems. The complex nature of CNN models can make it challenging to capture intricate non-linear relationships within regression data.
- **Data representation:** CNNs excel at extracting local spatial features in images, but this may not be optimal for representing regression data, which often involves more abstract and global interactions among variables.
- **Sample size:** CNNs typically require a large amount of data to achieve reliable results. If the regression dataset is relatively small, it can be difficult to train a CNN model that is robust and generalizable.
- **Architecture selection:** The performance of a CNN model heavily relies on the chosen architecture, including the number of layers, filter sizes, activation functions, etc. If the architectures you have experimented with did not yield satisfactory results, it may be necessary to reconsider and fine-tune these architectural choices.

It is important to note that these critiques do not imply that CNNs are ineffective for non-linear regressions. However, they highlight the need for careful adaptation and optimization to achieve reliable results in regression tasks. Exploring alternative approaches, such as LSTM, may be beneficial depending on the specific characteristics of your dataset.

c.b) Long Short Term memory models

1) LSTM material mass sub-model

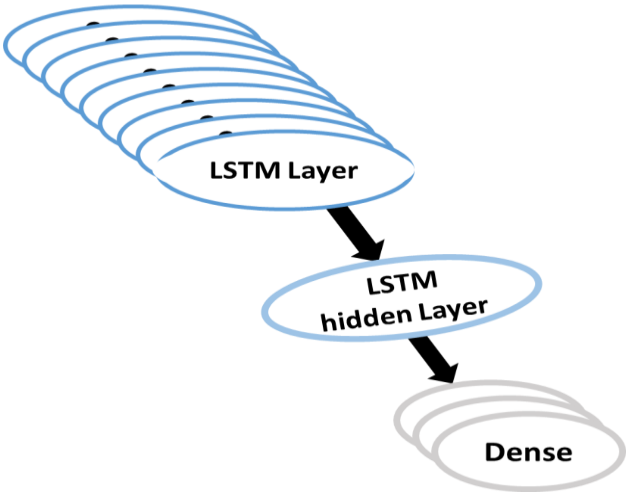


Figure 38: LSTM material mass sub-model.

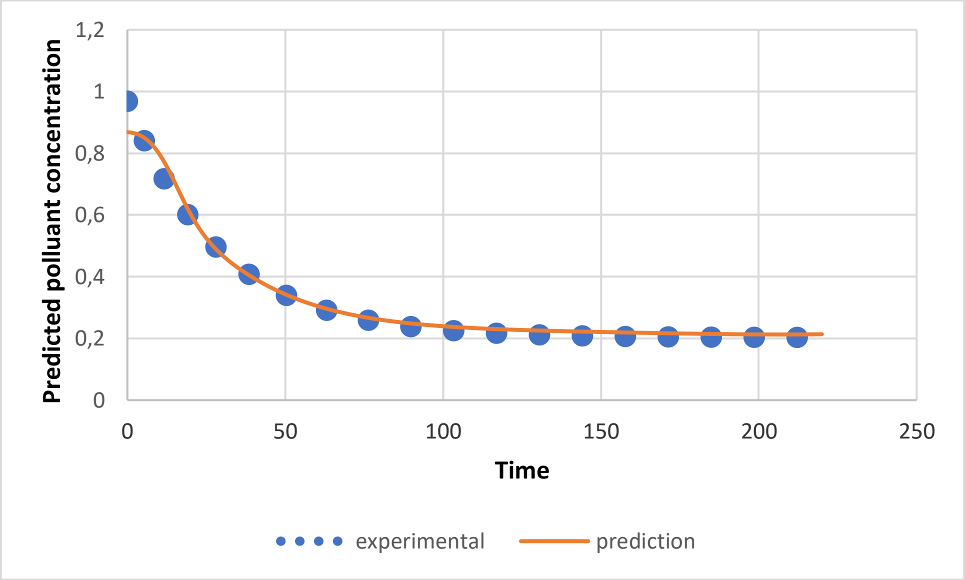


Figure 39: Experimental and prediction LSTM sub-model for material mass.

Table 8: A list of hyperparameters in LSTM material mass submodel.

Model		Hyperparameters
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	Connected layer	Neurons = 50, activation function = linear
	Connected layer	Neurons = 50, activation function = linear
	Connected layer	Neurons = 50, activation function = linear

2) LSTM pollutant concentration sub-model

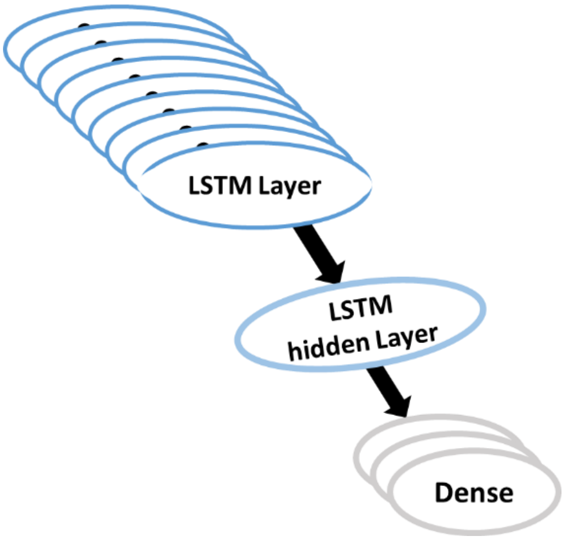


Figure 40 : LSTM pollutant concentration sub-model.

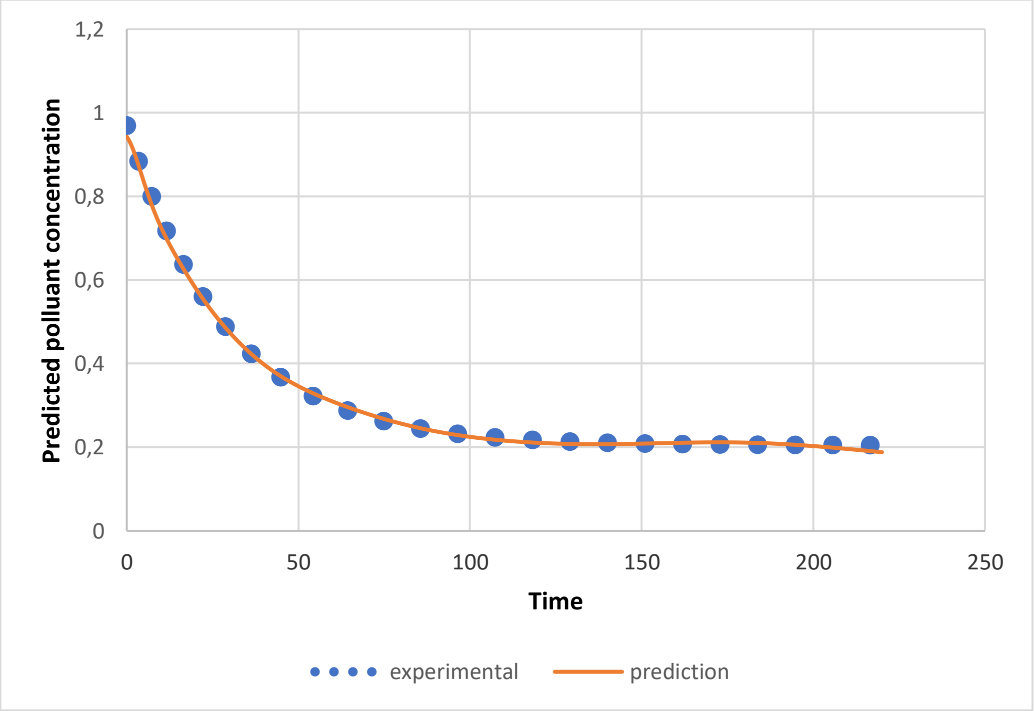


Figure 41: Experimental and prediction LSTM sub-model for pollutant concentration.

Table 9: A list of hyperparameters in LSTM pollutant concentration submodel.

Model		Hyperparameters
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM LSTM	Neurons = 50, activation function = tanh Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	Connected layer	Neurons = 50, activation function = linear
	Connected layer	Neurons = 50, activation function = linear
	Connected layer	Neurons = 50, activation function = linear

3) LSTM light intensity sub-model

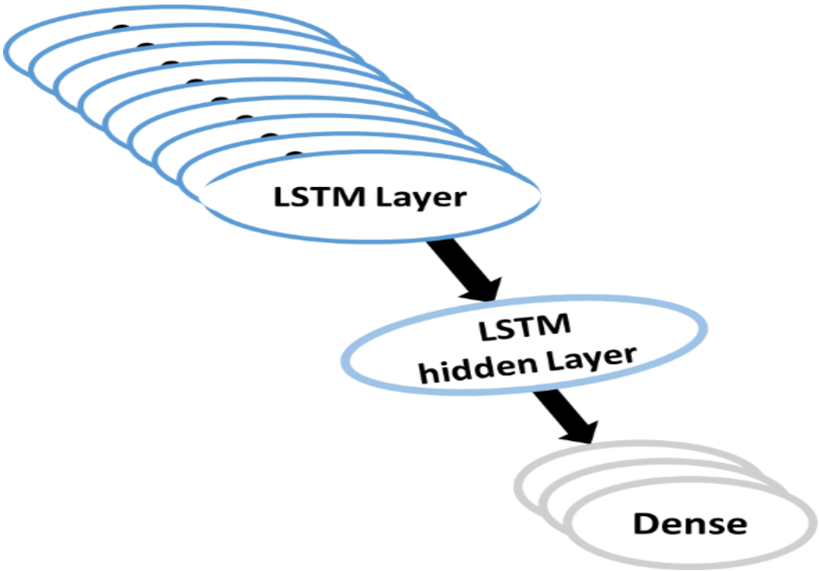


Figure 42: LSTM light intensity sub-model.

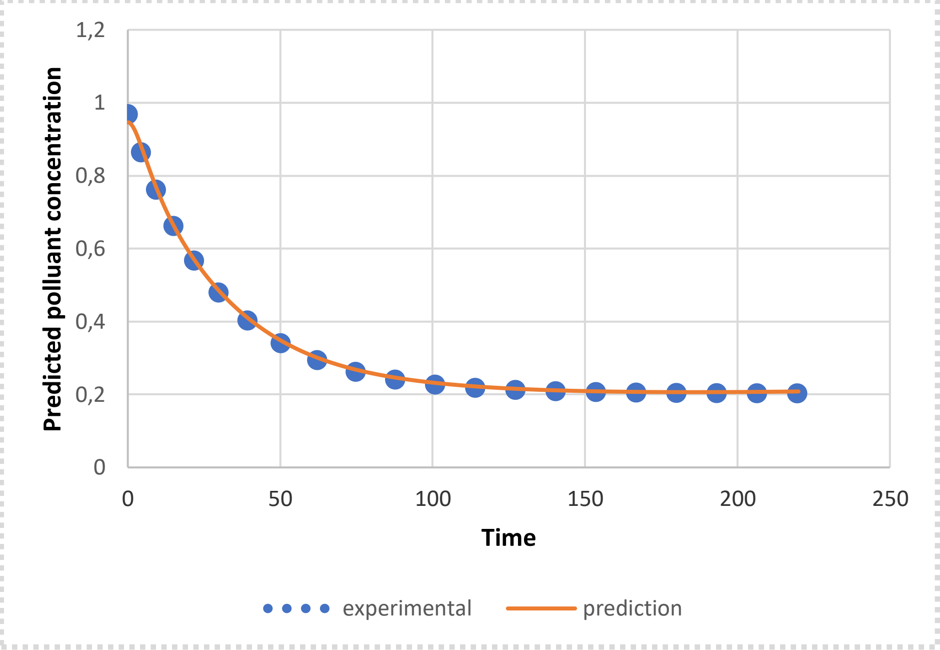


Figure 43: Experimental and prediction LSTM sub-model for light intensity.

Table 10: A list of hyperparameters in LSTM light intensity submodel.

Model		Hyperparameters
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	LSTM	Neurons = 50, activation function = tanh
	Connected layer	Neurons = 50, activation function = linear
	Connected layer	Neurons = 50, activation function = linear
	Connected layer	Neurons = 50, activation function = linear

d)Model Evaluation:

We assessed and validated the model's effectiveness using performance metrics.

i. **Mean Squared Error (MSE):** It represents the average of the squared difference between the actual values and the predicted values in the dataset. It measures the variance of the residual values. It is given by Equation 7.

$$MSE = \frac{1}{n}\sum_{1}^{n}(y-\hat{y})^2 \qquad (7)$$

ii. **Mean Absolute Error (MAE):** It represents the average of the absolute difference between the actual values and the predicted values in the dataset. It measures the average of the residual values in the dataset. It is given by Equation 8.

$$MAE = \frac{1}{n}\sum_{1}^{n}|y-\hat{y}| \qquad (8)$$

iii. Root Mean Square Error (RMSE): It measures the standard deviation of the residuals. It is given by Equation 9.

$$RMSE = \sqrt{\frac{1}{n}\sum_{1}^{n}(y-\hat{y})} \qquad (9)$$

Table 11: The detailed error rates for each LSTM are as follows:

Model err	MSE	MAE	RMSE
Material mass	0.005	0.002	0.0003
Pollutant concentration	0.005	0.003	0.0001
Light intensity	0.004	0.001	0.0002

e)Output Generation: Once the models have been trained and evaluated, they can be deployed to generate useful outputs or predictions based on new input data. we generate visual representations, such as graphs, to present and interpret the results obtained from the model.

Interpolation refers to the process of estimating values within a known range of data points. It involves using the existing data points to infer or predict values at intermediate positions. In other words, interpolation fills in the gaps between observed data points by assuming a smooth or continuous relationship between them. It is commonly used when there is missing or incomplete data within a given range.

Extrapolation, on the other hand, involves estimating values outside the known range of data points. It is the process of extending the trend or pattern observed in the existing data beyond the available range. Extrapolation assumes that the observed trend will continue in the same manner beyond the existing data points. However, it is important to note that extrapolation carries a higher level of uncertainty and can be less reliable compared to interpolation, especially when making predictions far outside the range of the observed data.

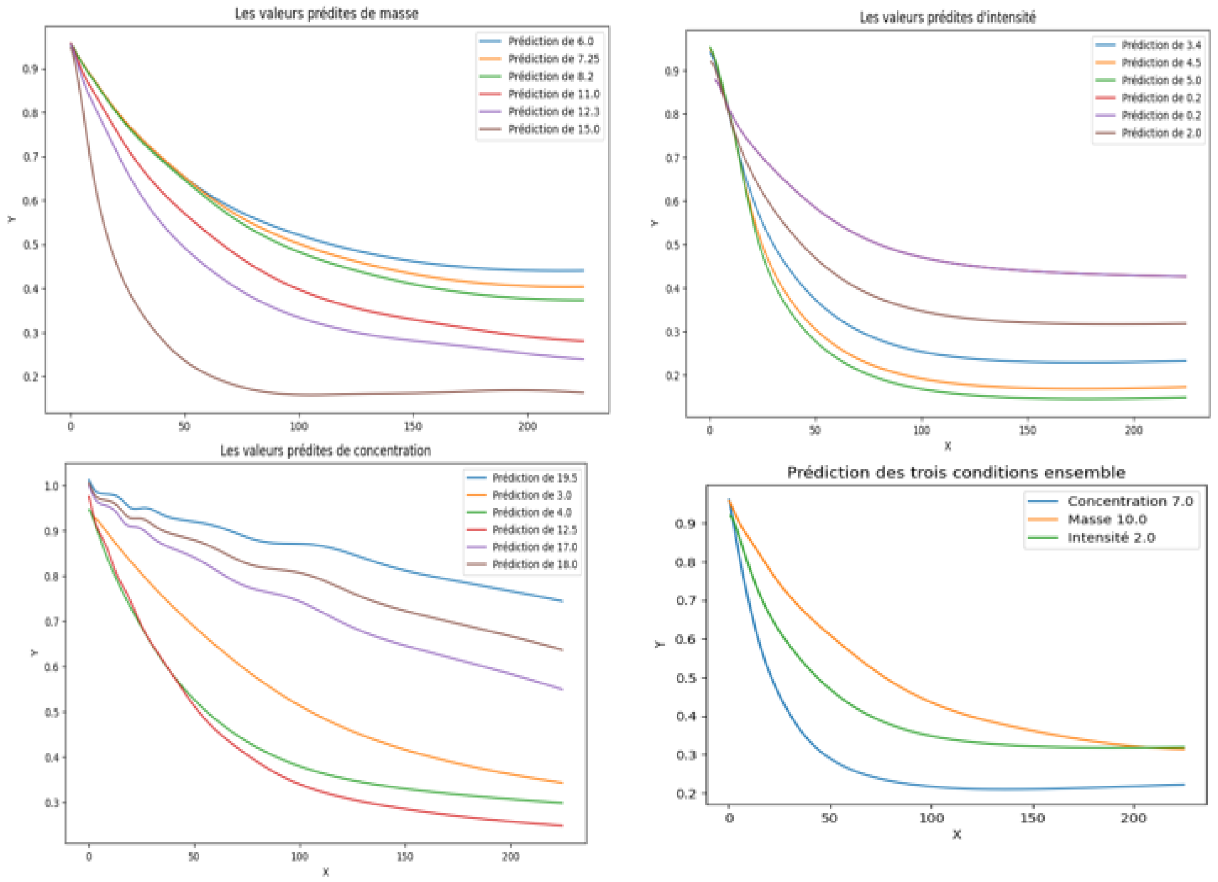


Figure 44: Predictions of LSTM model.

e) Feedback Loop: In certain cases, the system may incorporate a feedback loop where the generated outputs provide feedback to enhance the overall system performance. This may involve retraining the models with new data, adjusting parameters, or implementing system-level changes based on observed results.

Overall, the system architecture follows a cyclical process, commencing with data input, followed by processing, training, evaluation, and output generation. This iterative approach facilitates continuous learning and improvement of the system's performance over time.

User interface

Following the evaluation and confirmation of LSTM's suitability for this complex non-linear regression problem, we have implemented the model within a user-friendly interface to enhance its usability and accessibility.

This interface consists of four buttons, each leading to a separate interface based on the user's specific needs.

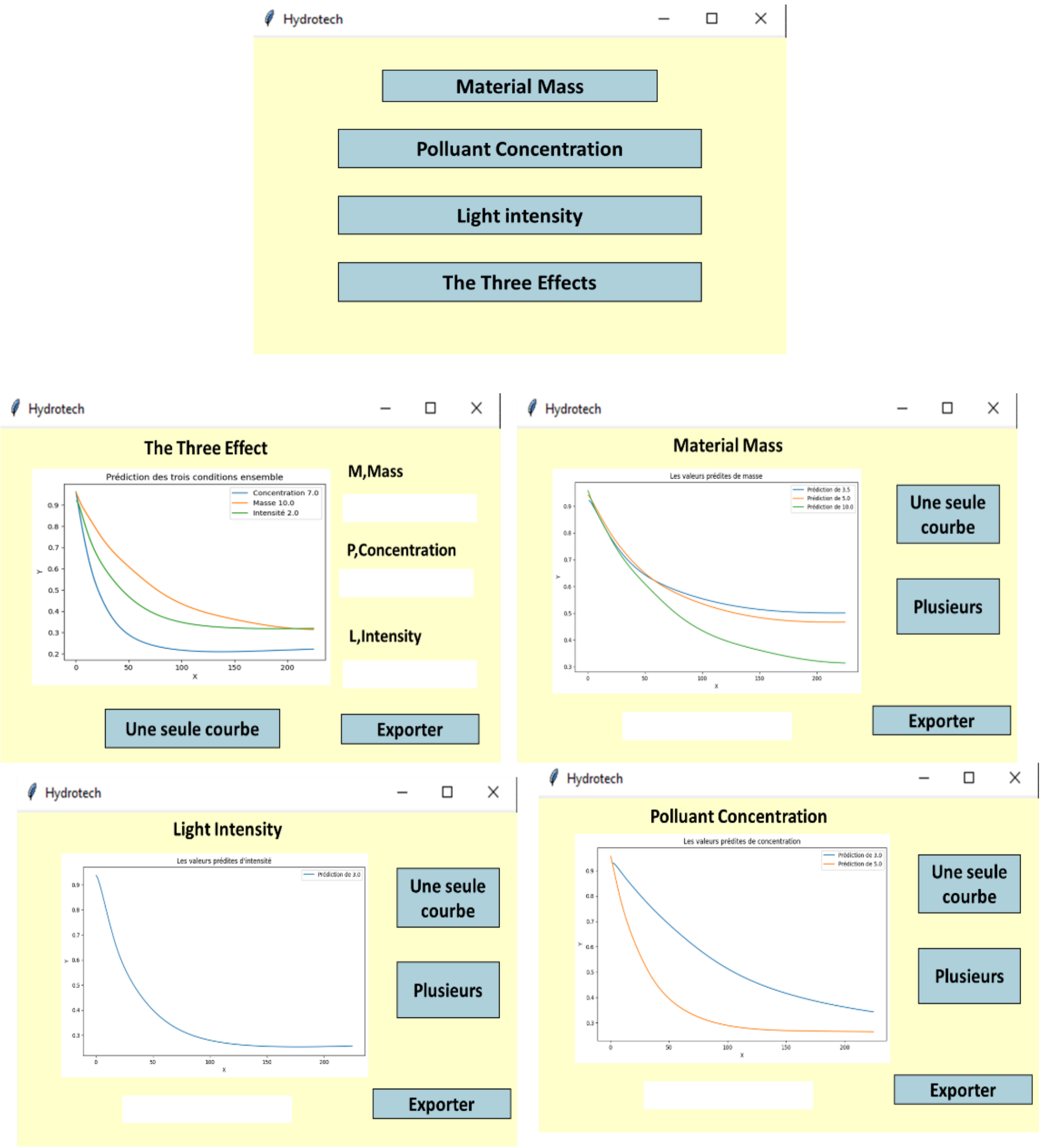


Figure 46: user interface.

3.5 Conclusion

We have developed and validated a deep learning model capable of solving a complex nonlinear problem. The model was effective in both the training and validation phases, demonstrating the ability to use LSTM for nonlinear regression.

Our modeling approach allows for time and cost savings.

This work leads to the conclusion that LSTM is a promising modeling method that can provide a significant solution to nonlinear challenges.

Conclusion

The field of Machine Learning and Artificial Intelligence witnessed a surge in interest with the emergence of neural networks. These networks, equipped with efficient learning methods and network structures, became widely utilized. However, over time, the excitement around neural networks diminished.

It was in 2006 that Hinton et al. introduced a groundbreaking concept called Deep Learning, which revolutionized the field. Deep Learning builds upon the foundations of artificial neural networks, enabling the exploration of complex non-linear problems.

This study delves into the realm of deep learning, focusing on its application in solving non-linear problems. By selecting the prediction of water pollutant degradation lifetime as a real-world problem, it demonstrates the efficacy of deep learning in tackling complex challenges. The findings of this research open up new and fascinating avenues, serving as a valuable reference for future research and implementation in relevant domains.

References

Bibliography

[1] M. Guo, M. Zhang, R. Liu, X. Zhang, et G. Li, « State-of-the-Art Advancements in Photocatalytic Hydrogenation: Reaction Mechanism and Recent Progress in Metal-Organic Framework (MOF)-Based Catalysts », *Adv. Sci.*, vol. 9, nº 1, p. 2103361, oct. 2021, doi: 10.1002/advs.202103361.

[3] S. Al Jitan, G. Palmisano, et C. Garlisi, « Synthesis and Surface Modification of TiO2-Based Photocatalysts for the Conversion of CO2 », *Catalysts*, vol. 10, nº 2, Art. nº 2, févr. 2020, doi: 10.3390/catal10020227.

[4] D. Chen *et al.*, « Photocatalytic degradation of organic pollutants using TiO2-based photocatalysts: A review », *J. Clean. Prod.*, vol. 268, p. 121725, sept. 2020, doi: 10.1016/j.jclepro.2020.121725.

[5] S. Cruz *et al.*, « Design and testing of a microchannel reactor for the PROX reaction », *Chem. Eng. J.*, vol. 167, nº 2-3, p. 634-642, mars 2011, doi: 10.1016/j.cej.2010.08.088.

[8] T. Fawzi, S. Rani, S. C. Roy, et H. Lee, « Photocatalytic Carbon Dioxide Conversion by Structurally and Materially Modified Titanium Dioxide Nanostructures », *Int. J. Mol. Sci.*, vol. 23, nº 15, p. 8143, juill. 2022, doi: 10.3390/ijms23158143.

[9] D. Chen *et al.*, « Photocatalytic degradation of organic pollutants using TiO2-based photocatalysts: A review », *J. Clean. Prod.*, vol. 268, p. 121725, sept. 2020, doi: 10.1016/j.jclepro.2020.121725.

[10] N. Thomas, D. D. Dionysiou, et S. C. Pillai, « Heterogeneous Fenton catalysts: A review of recent advances », *J. Hazard. Mater.*, vol. 404, p. 124082, févr. 2021, doi: 10.1016/j.jhazmat.2020.124082.

[11] B. M. Rajbongshi, « 7 - Photocatalyst: mechanism, challenges, and strategy for organic contaminant degradation », in *Handbook of Smart Photocatalytic Materials*, C. Mustansar Hussain et A. K. Mishra, Éd., Elsevier, 2020, p. 127-149. doi: 10.1016/B978-0-12-819049-4.00011-8.

[15] J. Szyman, « Multiple Steady States in the Photocatalytic Reactor for Colored Compounds Degradation », *Molecules*, vol. 26, nº 13, p. 3804, juin 2021, doi: 10.3390/molecules26133804.

[16] Z. Warren, T. Tasso Guaraldo, A. S. Martins, J. Wenk, et D. Mattia, « Photocatalytic foams for water treatment: A systematic review and meta-analysis », *J. Environ. Chem. Eng.*, vol. 11, nº 1, p. 109238, févr. 2023, doi: 10.1016/j.jece.2022.109238.

[18] X. Chen et S. S. Mao, « Titanium Dioxide Nanomaterials: Synthesis, Properties, Modifications, and Applications », *Chem. Rev.*, vol. 107, nº 7, p. 2891-2959, juill. 2007, doi: 10.1021/cr0500535.

[19] A. Bouazza, S. Bassaid, A. Dehbi, A. Guarnaccio, et M. D'Auria, « Synthesis and characterization of TiO2–x% curcumin nanocomposites (with x = 3, 5, and 10): application to the photocatalytic degradation of methylene blue », *React. Kinet. Mech. Catal.*, avr. 2023, doi: 10.1007/s11144-023-02400-4.

[20] M. L. Mocwana *et al.*, « Photocatalytic Degradation of Methylene Blue and Ortho-Toluidine Blue: Activity of Lanthanum Composites LaxMOy (M: Fe, Co, Ni) », *Catalysts*, vol. 12, nº 11, Art. nº 11, nov. 2022, doi: 10.3390/catal12111313.

[21] R. Li, T. Li, et Q. Zhou, « Impact of Titanium Dioxide (TiO2) Modification on Its Application to Pollution Treatment—A Review », *Catalysts*, vol. 10, nº 7, Art. nº 7, juill. 2020, doi: 10.3390/catal10070804.

[22] S. J. Armaković, M. M. Savanović, et S. Armaković, « Titanium Dioxide as the Most Used Photocatalyst for Water Purification: An Overview », *Catalysts*, vol. 13, nº 1, p. 26, déc. 2022, doi: 10.3390/catal13010026.

[23] A. Enesca et L. Isac, « The Influence of Light Irradiation on the Photocatalytic Degradation of Organic Pollutants », *Materials*, vol. 13, nº 11, p. 2494, mai 2020, doi: 10.3390/ma13112494.

[24] I. H. Sarker, « Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions », *SN Comput. Sci.*, vol. 2, nº 6, p. 420, août 2021, doi: 10.1007/s42979-021-00815-1.

[29] G. Jain, D. Mittal, D. Thakur, et M. K. Mittal, « A deep learning approach to detect Covid-19 coronavirus with X-Ray images », *Biocybern. Biomed. Eng.*, vol. 40, nº 4, p. 1391-1405, oct. 2020, doi: 10.1016/j.bbe.2020.08.008.

[30] M. Toğaçar, B. Ergen, et Z. Cömert, « COVID-19 detection using deep learning models to exploit Social Mimic Optimization and structured chest X-ray images using fuzzy color and

stacking approaches », *Comput. Biol. Med.*, vol. 121, p. 103805, juin 2020, doi: 10.1016/j.compbiomed.2020.103805.

[31] S. Suganyadevi, V. Seethalakshmi, et K. Balasamy, « A review on deep learning in medical image analysis », *Int. J. Multimed. Inf. Retr.*, vol. 11, n° 1, p. 19-38, mars 2022, doi: 10.1007/s13735-021-00218-1.

[32] N. S. Punn, S. K. Sonbhadra, et S. Agarwal, « COVID-19 Epidemic Analysis using Machine Learning and Deep Learning Algorithms ». medRxiv, p. 2020.04.08.20057679, 1 juin 2020. doi: 10.1101/2020.04.08.20057679.

[34] I. H. Sarker, « Machine Learning: Algorithms, Real-World Applications and Research Directions », *Sn Comput. Sci.*, vol. 2, n° 3, p. 160, 2021, doi: 10.1007/s42979-021-00592-x.

[40] N. Ketkar et J. Moolayil, *Deep Learning with Python: Learn Best Practices of Deep Learning Models with PyTorch*. Berkeley, CA: Apress, 2021. doi: 10.1007/978-1-4842-5364-9.

[43] A. Shrestha et A. Mahmood, « Review of Deep Learning Algorithms and Architectures », *IEEE Access*, vol. 7, p. 53040-53065, 2019, doi: 10.1109/ACCESS.2019.2912200.

[45] X. Ma *et al.*, « A Survey on Deep Learning Empowered IoT Applications », *IEEE Access*, vol. 7, p. 181721-181732, 2019, doi: 10.1109/ACCESS.2019.2958962.

Webography

[2] « LA-PHOTOCATALYSE-FR.pdf ». Consulté le: 13 mars 2023. [En ligne]. Disponible sur: https://sct-cleantech.com/wp-content/uploads/2022/04/LA-PHOTOCATALYSE-FR.pdf

[6] G. Grillo, P. Cintas, M. Colia, E. Calcio Gaudino, et G. Cravotto, « Process intensification in continuous flow organic synthesis with enabling and hybrid technologies », *Front. Chem. Eng.*, vol. 4, 2022, Consulté le: 20 mars 2023. [En ligne]. Disponible sur: https://www.frontiersin.org/articles/10.3389/fceng.2022.966451

[7] « Titanium Dioxide Nanomaterials: Synthesis, Properties, Modifications, and Applications | Chemical Reviews ». https://pubs.acs.org/doi/abs/10.1021/cr0500535 (consulté le 13 mars 2023).

[12] « les cinq étapes des réactions photocatalytiques – Recherche Google ». https://www.google.com/search?q=les+cinq+%C3%A9tapes+des++r%C3%A9actions+photocatalytiques&tbm=isch&ved=2ahUKEwiuh6-z5tn9AhW3JGIAHa2FAk8Q2-cCegQIABAA&oq=les+cinq+%C3%A9tapes+des++r%C3%A9actions+photocatalytiques&gs_lcp=CgNpbWcQA1AAWABgkAhoAHAAeACAAaAEiAGgBJIBAzUtMZgBAKABAaoBC2d3cy13aXotaW1nwAEB&sclient=img&ei=6Y4PZO7uI7fJiLMPrYuK-AQ&bih=658&biw=747&rlz=1C1YTUH_frDZ1041DZ1041#imgrc=ypO6SgLz8FyDdM (consulté le 13 mars 2023).

[13] « Nano Metal-Containing Photocatalysts for the Removal of Volatile Organic Compounds: Doping, Performance, and Mechanisms - PMC ». https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9027785/ (consulté le 20 mars 2023).

[14] « The main reactions that occur in photocatalytic reactor – Recherche Google ». https://www.google.com/search?q=The+main+reactions+that+occur+in+photocatalytic+reactor&rlz=1C1YTUH_frDZ1041DZ1041&sxsrf=AJOqlzU0PnHn89caIrwznvaVzFVl5HIk3g:1678743012632&source=lnms&tbm=isch&sa=X&ved=2ahUKEwiQoL-H7dn9AhU_q4QIHembDj0Q_AUoAXoECAEQAw&biw=747&bih=658&dpr=1.25 (consulté le 13 mars 2023).

[17] « Photocatalytic Degradation of Organic Pollutants in Water | IntechOpen ». https://www.intechopen.com/chapters/42060 (consulté le 13 mars 2023).

[25] « Neural Networks - History ». https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/History/history2.html (consulté le 2 mai 2023).

[26] « Machine learning, explained | MIT Sloan ». https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained (consulté le 6 juin 2023).

[27] G. Saint-Cirgue, « Comment fonctionne le Machine Learning ? », *Machine Learnia*, 11 mars 2019. https://machinelearnia.com/comment-fonctionne-machine-learning/ (consulté le 8 juin 2023).

[28] « CHEST X-RAY IMAGE CLASSIFICATION USING FASTER R-CNN - Malaysian Journal of Computing (MJoC) ». https://mjoc.uitm.edu.my/main/index.php/journal/19-volume-4-1-2019/51-6095 (consulté le 24 avril 2023).

[33] D. Johnson, « Reinforcement Learning: What is, Algorithms, Types & Examples », 27 mai 2023. https://www.guru99.com/reinforcement-learning-tutorial.html (consulté le 16 juin 2023).

[35] « What are Neural Networks? | IBM ». https://www.ibm.com/topics/neural-networks (consulté le 3 juin 2023).

[36] « Introduction to Neural Network: Build your own Network ». https://www.analyticsvidhya.com/blog/2023/02/introduction-to-neural-network-build-your-own-network/ (consulté le 8 mai 2023).

[37] « CS 230 - Deep Learning Tips and Tricks Cheatsheet ». https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-deep-learning-tips-and-tricks (consulté le 24 avril 2023).

[38] baeldung, « Multi-Layer Perceptron vs. Deep Neural Network | Baeldung on Computer Science », 29 juin 2022. https://www.baeldung.com/cs/mlp-vs-dnn (consulté le 8 juin 2023).

[39] J. Nabi, « Estimators, Loss Functions, Optimizers —Core of ML Algorithms », *Medium*, 8 juillet 2019. https://towardsdatascience.com/estimators-loss-functions-optimizers-core-of-ml-algorithms-d603f6b0161a (consulté le 16 juin 2023).

[41] « Papers with Code - SGD Explained ». https://paperswithcode.com/method/sgd (consulté le 6 juin 2023).

[42] « optimization - What’s the difference between momentum based gradient descent and Nesterov’s accelerated gradient descent? - Cross Validated ». https://stats.stackexchange.com/questions/179915/whats-the-difference-between-momentum-based-gradient-descent-and-nesterovs-acc (consulté le 16 juin 2023).

[44] « Deep Learning with TensorFlow [Book] ». https://www.oreilly.com/library/view/deep-learning-with/9781786469786/ (consulté le 27 mai 2023).

[46] alexandre, « Recurrent Neural Networks: The complete Guide », *Data Science Courses | DataScientest*, 19 janvier 2023. https://datascientest.com/en/recurrent-neural-network-what-is-it (consulté le 21 mai 2023).

[46] https://www.tensorflow.org/?hl=fr (consulté le 21 mai 2023).

[47] https://numpy.org/ (consulté le 21 mai 2023).

[48] https://pandas.pydata.org/ (consulté le 21 mai 2023).

[49] https://scikit-learn.org/stable/index.html (consulté le 21 mai 2023).

[50] https://www.cs.mcgill.ca/~hv/classes/MS/TkinterPres/ (consulté le 21 mai 2023).

[51] https://matplotlib.org/ (consulté le 21 mai 2023).