الجمهورية الجزائرية الديموقراطية الشعبية
People's Democratic Republic of Algeria
وزارة التعليم العالي والبحث العلمي
Ministry Of Higher Education and Scientific Research

جامعة آكلي محند اولحاج البويرة
Akli Mohand Oulhadj university of brouira

كلية العلوم والتكنولوجيا
Faculty of Science and Technology

Department: Civil Engineering

Order N°: ..... /2024

MASTER THESIS

Presented by

MEROUCHI Fahima

BOUGHERBI Rabah

With a view to obtained the master's degree of science diploma

Branch: Civil Engineering

Topic

The implementation of artificial intelligence (AI) in civil engineering: review and case study

Presented and defended on 02 /07/ 2024, before the jury composed of:

arbaoui ahcene	MCA	University Akli Mohand Oulhadj, Bouira	President
BAKHTI Rachid	MCA	University Akli Mohand Oulhadj, Bouira	Supervisor
SAOUDI BRAHIM	MCA	University Akli Mohand Oulhadj, Bouira	Examiner

Abstract

this work delves into the realm of artificial intelligence (AI) and its profound impact on civil engineering practices. Through a meticulous review of existing AI algorithms, such as artificial neural networks, genetic algorithms, random forests, and others, we explore their implementation in civil engineering to evaluate and compute various parameters such as bearing capacity, structural health monitoring, and more. Additionally multiple AI frameworks are presented in this work, encompassing both open-source and commercial frameworks. A case study is included, where the bearing capacity of a shallow foundation is evaluated using the neural network algorithm. The bearing capacity is computed based solely on the outcomes of the dynamic probing test and soil density.

Key words: Artificial Intelligence (AI), Civil Engineering, Bearing Capacity, Artificial Neural Networks (ANN), Frameworks of artificial intelligence.

ملخص

يتناول هذا العمل مجال الذكاء الاصطناعي (AI) وتأثيره العميق على ممارسات الهندسة المدنية. من خلال عرض دقيق للخوارزميات الحالية في الذكاء الاصطناعي، مثل الشبكات العصبية الاصطناعية والخوارزميات الجينية وغابات العشوائية وغيرها، نستكشف تطبيقها في الهندسة المدنية لتقييم وحساب مختلف المعايير مثل قدرة التحمل ومراقبة صحة الهيكل والمزيد. بالإضافة إلى ذلك، تُقدَّم في هذا العمل عدة مكتبات رقمية في للذكاء الاصطناعي، بنوعيها المفتوحة المصدر والتجارية. يتضمن العمل دراسة حالة يتم فيها تقييم قدرة التحمل لأساس البنايات باستخدام خوارزمية الشبكة العصبية. تُحسب قدرة التحمل بناءً فقط على ة نتائج اختبار المجس الديناميكي وكثافة التربة.

الكلمات المفتاحية: الذكاء الاصطناعي(AI) ، الهندسة المدنية، قدرة التحمل، الشبكات العصبية(ANN) ، مكتبات الرقمية للذكاء الاصطناعي.

Acknowledgments

Today, I stand here filled with gratitude. First and foremost, to my parents, whose unwavering love and support provided the foundation for everything I've achieved. Thanks to their belief in me, I had the resources I needed to reach this milestone. I also extend my heartfelt thanks to my siblings, their encouragement a constant source of strength throughout these years.

Dr Bakhti Rachid, my deepest gratitude goes to you. Your guidance, patience, and willingness to help were instrumental in completing this work.

Finally, to my amazing friends, both old and new, your presence has enriched this journey beyond measure. You've helped me grow as a person and been there through thick and thin. To each of you, a heartfelt thank you.

M. Fahima

Acknowledgments

This work is the result of several years of study for this we thank GOD, the Almighty for giving us the strength and courage to carry out our work and for us helped me get to the end.

We deeply thank our dear parents for their moral, material and physical support during our studies. We would also like to sincerely thank all our teachers who guided us throughout our training particularly: our supervisor Dr, BAKHTI.R for his follow-up and his comments.

The jury members for the interest who have shown our work and who will do the honour of examining and judging it.

Our sincere thanks also go to the administrative staff of the civil engineering department of the university AKLI MOHAND OULHADJ BOUIRA for supporting us during our university career.

Finally, our sincere gratitude goes to all those who have contributed directly or indirectly to the development of our work.

B. Rabah

Contents

Chapter III: AI frameworks and libraries

Chapter IV: Computing the bearing capacity based on Neural Network

List of figures

List of tables

General Introduction

The world is drowning in data. From the ceaseless hum of sensor networks monitoring our cities to the colossal datasets generated by construction projects, the need for intelligent tools to analyse and interpret this information has become paramount. This insatiable demand for data mastery has propelled Artificial Intelligence (AI) to the forefront of innovation across diverse disciplines.

AI's journey began as a glimmer in the minds of pioneering scientists like Alan Turing and John McCarthy in the mid-20th century. Their vision laid the groundwork for a field that strives to create intelligent machines capable of mimicking human cognitive abilities. Over the decades, AI has experienced cycles of fervent optimism and disillusionment, punctuated by breakthroughs in symbolic AI, expert systems, and the advent of machine learning.

Today, machine learning (ML) stands as the bedrock of contemporary AI. It empowers algorithms to learn and improve from data without explicit programming. Deep learning (DL), a subfield of ML, leverages artificial neural networks (ANNs) – complex structures inspired by the human brain – to learn intricate patterns from massive datasets. These algorithms, the workhorses of AI, are revolutionizing various industries

However, implementing AI in civil engineering presents unique challenges. Data availability, quality, and security are paramount. Training AI models requires vast amounts of clean data, and ensuring the security of infrastructure-related data is critical. Additionally, fostering interdisciplinary collaboration between civil engineers and AI specialists is crucial to bridge the gap between theoretical capabilities and real-world applications.

Despite these challenges, the potential benefits of AI in civil engineering are immense. Imagine AI-powered predictive maintenance systems that anticipate infrastructure needs, smart infrastructure design that optimizes efficiency and sustainability, and even autonomous construction equipment that streamlines construction processes.

This study has a comprehensive set of objectives aimed at exploring the application of artificial intelligence (AI) in civil engineering:

The primary objective is to educate and familiarize readers with various AI techniques and algorithms, particularly focusing on machine learning models like neural networks, genetic algorithms, and particle swarm optimization. By delving into these methodologies, the study aims to enhance understanding of how AI can be utilized for predictive modelling and optimization in civil engineering projects.

The study seeks to review and analyse current implementations of AI in civil engineering, with a specific emphasis on practical applications in areas such as health monitoring, concrete analysis, building information modelling (BIM), and geotechnical engineering. By examining real-world use cases of AI in these domains, the research aims to showcase the benefits and challenges associated with integrating AI into civil engineering practices.

In addition, a review of popular AI frameworks and libraries has been delivered to facilitate the development of AI applications. By exploring frameworks like TensorFlow, PyTorch, and scikit-learn, the study aims to demonstrate how these tools streamline the development process, making it easier for engineers to implement AI solutions in civil engineering projects.

The final objective involves building a neural network Windows form application to calculate the bearing capacity of soil. This application designed to compare its predictions with real results obtained from penetrometer tests. By integrating AI into this specific task, the study aims to showcase how neural networks can be utilized to predict soil bearing capacity accurately, thereby enhancing the efficiency and accuracy of geotechnical engineering assessments.

Chapter I

AI algorithms

I.1 Introduction

Artificial intelligence (AI) is a branch of computer science focused on creating intelligent machines that can mimic human cognitive abilities. These abilities include reasoning, learning problem-solving, and decision-making. Think of AI as a giant information processor.it gathers data from various sources in the real world, analyses it, and uses that knowledge to perform practical tasks.in essence, AI provides a toolbox of methods, techniques, and tools to build models and solution that simulate intelligent behaviour. This intelligent behaviour can be modelled by natural processes.AI is rapidly developing field with the potential to revolutionize many aspects of our lives. by automating repetitive takes and offering new perspectives on complex problems, AI can boost our efficiency and deepen our understanding of the world[1].

The genesis of AI can be traced back to the mid-20th century, with seminal contributions from luminaries such as Alan Turing and John McCarthy. Turing's groundbreaking work laid the conceptual groundwork for computational thinking and machine intelligence, while McCarthy coined the term "artificial intelligence" in 1956, sparking a revolution in computing. Over the ensuing decades, AI experienced cycles of fervent optimism and disillusionment, punctuated by breakthroughs in symbolic AI, expert systems, and the advent of machine learning[2].

The resurgence of AI in the 21st century has been propelled by exponential advances in computational power, algorithmic sophistication, and the proliferation of big data. From the ascent of deep learning and neural networks to the proliferation of intelligent agents and autonomous systems, AI has transcended the realm of science fiction to become an indispensable force driving innovation and progress in virtually every sector of the global economy[2].

Within the vast landscape of AI techniques lie a plethora of methodologies, each tailored to address specific challenges and domains. Machine learning, the bedrock of contemporary AI, encompasses a spectrum of approaches, including supervised learning, unsupervised learning, and reinforcement learning. Neural networks, inspired by the architecture of the human brain, have emerged as the linchpin of deep learning, enabling machines to autonomously learn hierarchical representations of data.

Natural language processing (NLP) empowers machines to comprehend, generate, and manipulate human language, fostering applications in virtual assistants, language translation, and sentiment analysis. Computer vision algorithms endow machines with the ability to interpret and analyze visual information, fueling advancements in autonomous navigation, facial recognition, and medical imaging.

As we stand at the precipice of the AI revolution, we are confronted with boundless opportunities for innovation, progress, and societal transformation. Yet, with great power comes great responsibility. It behoves us, as stewards of this nascent discipline, to navigate the ethical, societal, and existential implications of AI with prudence, foresight, and a steadfast commitment to harnessing its power for the betterment of humanity. Only then can we unlock the full potential of artificial intelligence as a force for good in the world.

I.2 Top Artificial Intelligence Techniques

The rapid evolution of AI has spurred the development of a multitude of techniques, fundamentally altering our interactions with technology. Below are some of the foremost AI methodologies that have emerged from this evolution.

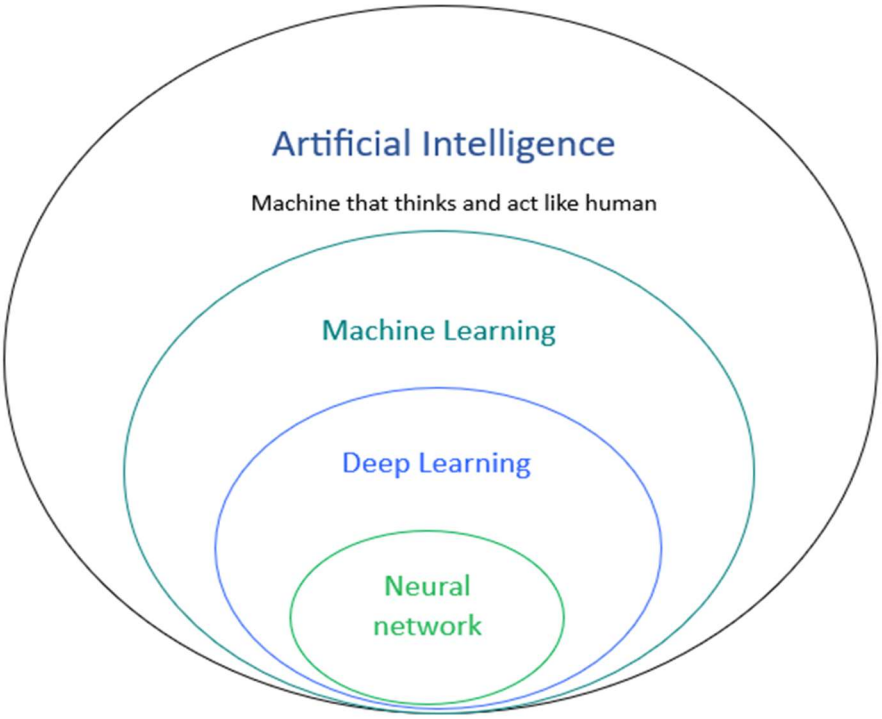

Figure I. 1: The Artificial Intelligence Realm.

I.2.1 Machine learning

machine learning is a subset of artificial intelligence and computer science, revolves around leveraging data and algorithms to empower AI systems to emulate human learning processes, thereby enhancing their precision over time[3].

Machine learning (ML) stands as a significant component within the field of artificial intelligence (AI), focusing on the exploration, formulation, and refinement of algorithms capable of self-learning from data and subsequently making predictions[4]. ML denotes the capacity of computers to acquire knowledge autonomously, without explicit programming instructions. These ML-based models exhibit both predictive and descriptive capabilities, enabling the extraction of insights from data. While ML operates within the broader domain of AI, its applicability spans diverse disciplines such as computer science, information theory, control computational complexity, probability and statistics, financial markets, and theory and philosophy[5].

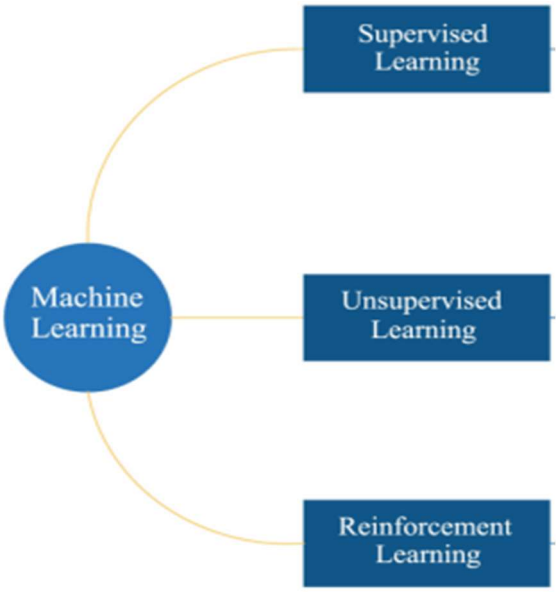

Figure I. 2: Machine learning categories.

I.2.2 Neural network

A neural network is a computational model inspired by the structure and functioning of the human brain, consisting of interconnected nodes called neurons. These neurons are organized into layers, including an input layer where data is introduced, one or more hidden layers where computation occurs, and an output layer that produces the results. Neural networks are

trained using a variety of algorithms, such as backpropagation, to adjust the weights of connections between neurons, enabling the network to learn and make predictions or decisions from input data. They are widely used in various fields, including machine learning, pattern recognition, image and speech recognition, and natural language processing[5].

- The neural network architecture consists of several layers, each with its specific role:

a. Input Layer

Neurons in this layer are responsible for representing the various features present in the input data[6].

b. Hidden Layers

Neurons within these layers carry out complex mathematical operations, often involving weighted sums and activation functions[5]. The hidden layers play a crucial role in enabling the network to comprehend intricate relationships within the data[6].

c. Activation Functions

Neurons utilize activation functions to introduce nonlinearity into the network. Among the commonly used functions are sigmoid, ReLU, and tanh[5].

d. Output Layer

Positioned at the end of the network, the final hidden layer connects to the output layer, where predictions are generated. The number of output neurons varies depending on the specific task at hand, such as regression or classification[6].

e. Training Process

Artificial neural networks (ANNs) learn by adjusting the connection. weights to minimize a designated loss function[4]. Through backpropagation, gradients of the loss with respect to the weights are computed, guiding the updates of the weights using optimization techniques like gradient descent. The primary aim is to identify weights that minimize the loss, which involves iteratively updating the weights to refine the predictions. Techniques such as stochastic gradient descent are commonly employed in this process[7].

Artificial neural networks demonstrate proficiency in modelling both linear and nonlinear relationships, offering flexibility for addressing complex tasks. Deep neural networks (DNNs), equipped with multiple hidden layers, excel particularly in capturing intricate patterns within the data. The design and training of ANNs require meticulous attention to architecture, hyperparameters, and data characteristics to mitigate challenges such as overfitting.

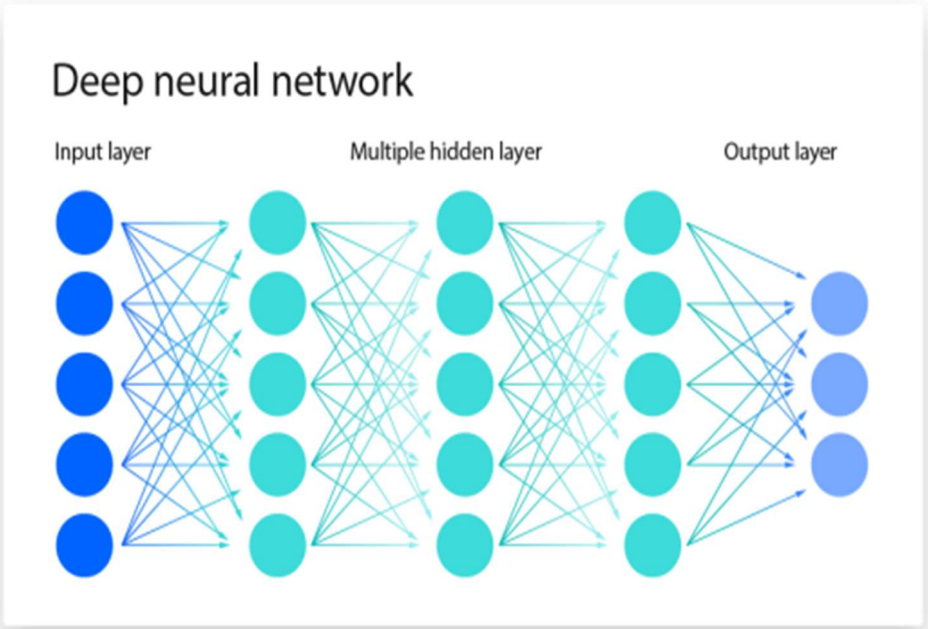

Figure I. 3: Basic structure of neural network.

I.2.3 Deep learning

Deep learning is a technique within the realm of artificial intelligence, which instructs computers to analyse data in a manner reminiscent of human cognitive processes[7]. Through deep learning, computers can discern intricate patterns within various types of data such as images, text, audio, and more, enabling them to generate precise analyses and predictions[4].

I.2.4 Big data

The term "big data" refers to datasets that surpass the capacities of conventional tools for management[8]. This concept is characterized by three primary factors often referred to as the three Vs: diversity, quantity, and speed. The diversity of data sources is expanding, arriving in larger quantities and at a faster pace, necessitating prompt processing and action[9].

I.2.5 Data mining /science

Data mining is a rapidly expanding field that constitutes a component of the knowledge discovery in databases (KDD) process. This process encompasses various preliminary steps such as data selection, data cleaning, preprocessing, and transformation before engaging in data mining activities[10]. Data mining entails the utilization of computer algorithms to unveil concealed patterns and unexpected associations within extensive datasets. Artificial intelligence (AI) encompasses a broader domain than machine learning. AI systems function as knowledge processing systems, incorporating fundamental techniques such as knowledge representation, knowledge acquisition, and inference, which includes search and control mechanisms.

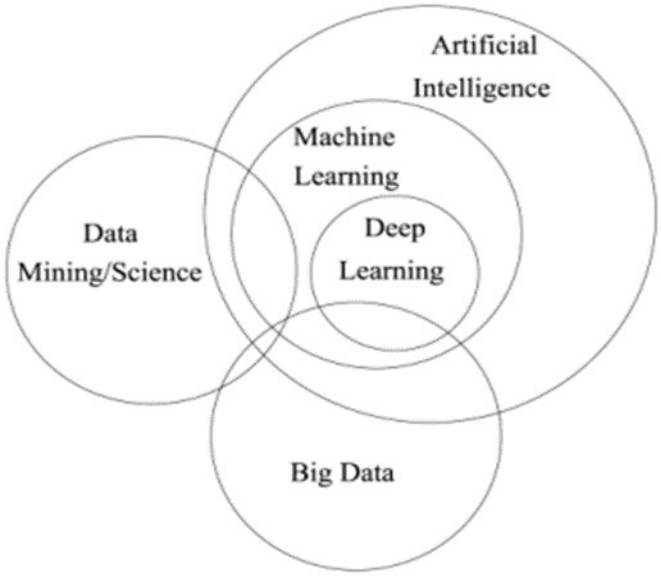

Figure I. 4: Illustration of the interrelation of different intelligent computational technique.

I.3 AI algorithms

An algorithm is a set of defined steps designed to perform a specific objective. This can be a simple process, such as a recipe to bake a cake, or a complex series of operations used in machine learning to analyze large datasets and make predictions. In the context of machine learning, algorithms are vital as they facilitate the learning process for machines, helping them to identify patterns and make decisions based on data.

And here's a multiple algorithm:

I.3.1 Genetic algorithm

The genetic algorithm (GA) is a metaheuristic technique, drawing inspiration from natural selection, and is categorized within the broader class of evolutionary algorithms (EA)[11]. GAs is frequently utilized to derive optimal solutions for optimization and search tasks, employing biologically inspired mechanisms like mutation, crossover, and selection. Various applications of GAs span domains such as enhancing decision tree efficacy, solving sudoku puzzles, optimizing hyperparameters, and conducting causal inference[12].

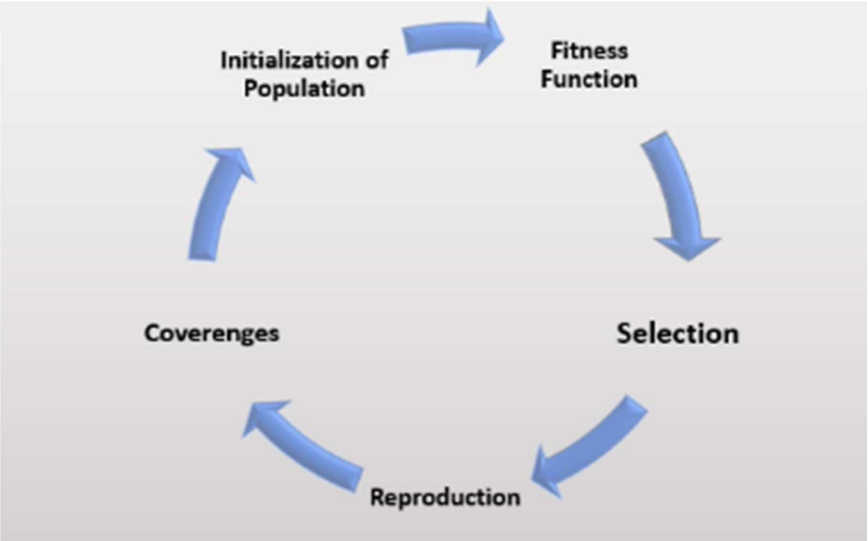

Figure I. 5: Genetic algorithm procedure.

Methodology of genetic algorithms

The methodology of a genetic algorithm (GA) typically involves several steps:

- Initialization:

Begin by creating a population of potential solutions (often referred to as individuals or chromosomes) randomly or using some heuristic method[12].

- Evaluation:

Everyone in the population is evaluated based on a fitness function, which quantifies how well it performs in solving the given problem[13]. The fitness function guides the selection process by providing a measure of the individual's quality.

- Selection:

Individuals from the current population are selected for reproduction based on their fitness scores. This selection process is typically biased towards individuals with higher fitness values, aiming to preserve and promote desirable traits in subsequent generations[13].

- Reproduction:

Selected individuals undergo reproduction processes, such as crossover and mutation, to create offspring. Crossover involves exchanging genetic material between parent individuals to generate new solutions, while mutation introduces random changes to maintain diversity in the population.

- Replacement:

The offspring replace some individuals in the current population, typically through strategies like elitism (where the best individuals are preserved) or generational replacement (where the entire population is replaced).

- Termination:

The algorithm terminates when a predefined stopping criterion is met, such as reaching a maximum number of generations, achieving a satisfactory solution, or exhausting computational resources[13].

- Iteration:

Steps 2 to 6 are repeated iteratively until the termination criterion is satisfied, with each iteration (generation) potentially leading to the improvement of solutions within the population.

By iteratively applying these steps, genetic algorithms explore the solution space, gradually refining candidate solutions to approach or reach optimal or near-optimal solutions for the given optimization problem.

I.3.2 Swarm intelligence

The inception of Swarm Intelligence (SI) traces back to 1992 when American scholars, Hack wood et al. introduced the concept within the framework of molecular automata systems, showcasing self-organization through interactions among neighbouring individuals in grid spaces. Initially, SI primarily drew inspiration from the collective behaviours observed in social insects and other animal groups. Presently, SI has evolved to encompass broader research into the collective behaviours of multi-component systems regulated by decentralized controls and self-organization. Meta-heuristics grounded in swarm intelligence emulate a collective of uncomplicated individuals, refining their solutions through interactions with both each other and the environment. This approach has demonstrated remarkable efficacy in addressing numerous challenging problems, rendering swarm intelligence a highly dynamic and vibrant research domain in recent years[14].

Figure I. 6: Swarm intelligence.

And here we have one of the swarm intelligence algorithms:

I.3.2.1 Particle swarm optimization

The concept of Particle Swarm Optimization (PSO) emerged through the collaborative efforts of Kennedy et al. drawing inspiration from the social behaviours observed in animals and insects, such as birds and fish[15]. PSO stands as a population-based global optimization method, where numerous independent solutions, referred to as particles, traverse through a multidimensional search space to identify the optimal solution. Each particle is characterized by a position vector and a velocity vector, which are iteratively adjusted based on the particle's optimal local vector and the current optimal global vector of the entire population[16].

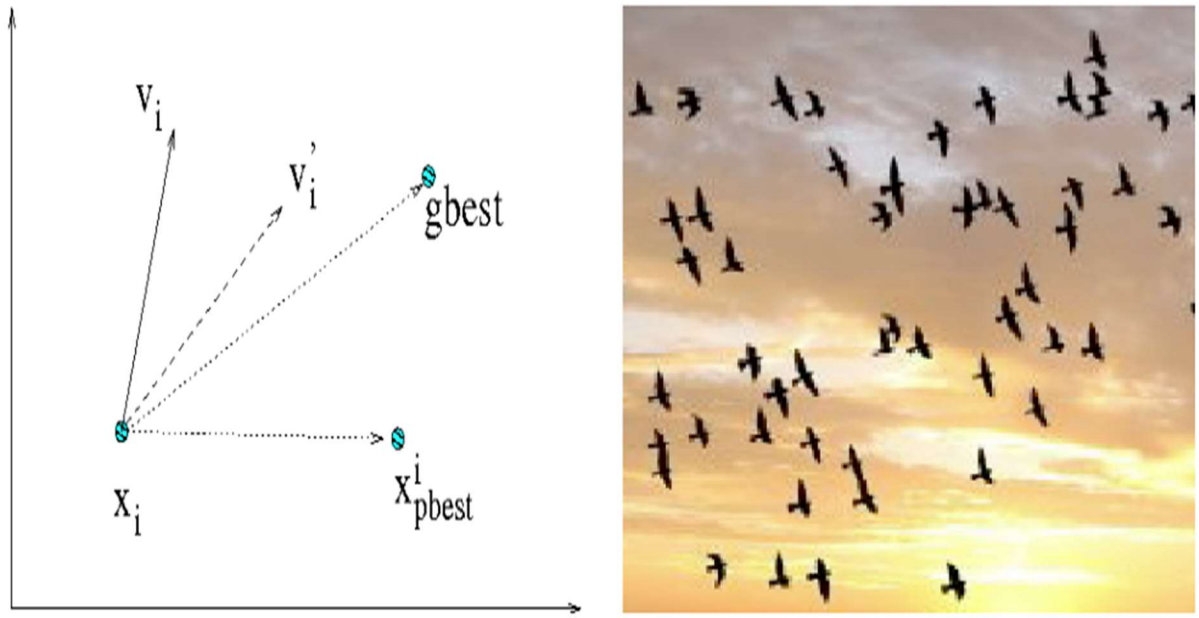

Figure I. 7: position in particle swarm optimization.

Methodology of particle swarm optimization

- Initialization:
  - The optimization process begins by initializing a population of potential solutions, often referred to as individuals, particles, or agents, within the search space of the problem.
  - Initial positions and velocities (if applicable) are assigned to these individuals randomly or using specific strategies to explore the search space effectively.

- Evaluation:
  - Everyone in the population evaluates its fitness by computing the objective function associated with its current position in the search space.
  - The fitness value represents how well the individual performs with respect to the optimization problem, serving as a measure of its quality.
- Interaction and Communication:
  - Individuals interact with each other through a process of sharing information or through environmental cues.
- Update:
  - Based on the evaluation results and information exchanged during interaction, individuals update their positions and/or velocities to search for better solutions.
  - The update rules are typically guided by mathematical equations or heuristics specific to the chosen swarm optimization algorithm.

And there are rules for both velocity and position of each particle are updated:

Velocity $\longrightarrow v_i(t) = \theta v_i(t-1) + c_1 r_1 \left(p_{best,i} - x_i(t-1)\right) + c_2 r_2 \left(g_{best,i} - x_i(t-1)\right)$

$\theta v_i(t-1) \longrightarrow$ Prevent the particle from drastically changing the direction.

$c_1 r_1 \left(p_{best,i} - x_i(t-1)\right) \rightarrow$ Make the particle tracks its best position.

$c_2 r_2 \left(g_{best,i} - x_i(t-1)\right) \rightarrow$ Make the particle tracks the best position found by the group.

Inertia weight θ :is a proportional agent that is related with the speed of last improvement; the value of θ is assumed to vary linearly from 0.9 to 0.4.

c_1 and c_2 : are the cognitive (individual) and social (group) learning rates.

r_1 and r_2 : are uniformly distributed random numbers in the range 0 and 1.

Position $\rightarrow x_i(t) = x_i(t-1) + v_i(t) \vdash_{if\ x_i(t) < x_{lb} \rightarrow x_i(t) = x_{lb}}^{if\ x_i(t) > x_{ub} \rightarrow x_i(t) = x_{ub}}$

- Iteration:
  - The process of evaluation, interaction, and update is repeated iteratively for a certain number of generations or until termination criteria are met.
  - During each iteration, individuals explore the search space further, potentially converging toward optimal or near-optimal solutions.
- Termination:
  - The optimization process terminates when predefined stopping criteria are satisfied, such as reaching a maximum number of iterations, achieving a satisfactory solution, or when improvements become negligible.
- Solution Extraction:
  - After termination, the best solution found by the swarm, or a set of best solutions, is extracted from the final population.
  - These solutions represent candidate solutions that approximate the optimal solution(s) of the optimization problem.
- Post-processing (Optional):
  - Additional post-processing steps may be applied to the extracted solutions, such as refining them further or ensuring that they satisfy any constraints imposed by the problem.
- Analysis and Validation:
  - The obtained solutions are analysed and validated to assess their quality and suitability for the optimization problem.
  - Results may be compared with known solutions or validated through testing in real-world scenarios.
- Parameter Tuning and Optimization:

Parameters of the swarm optimization algorithm, such as population size, communication range, or update coefficients, may be fine-tuned to improve performance or convergence speed.

I.3.3 Ant colony algorithm

Ant colony optimization (ACO) represents a metaheuristic approach mimicking the foraging actions of ant colonies to uncover the most efficient path towards food sources. Through their quest for sustenance, ants deposit pheromones along their routes, thereby enticing fellow ants to trail along the same path[16].

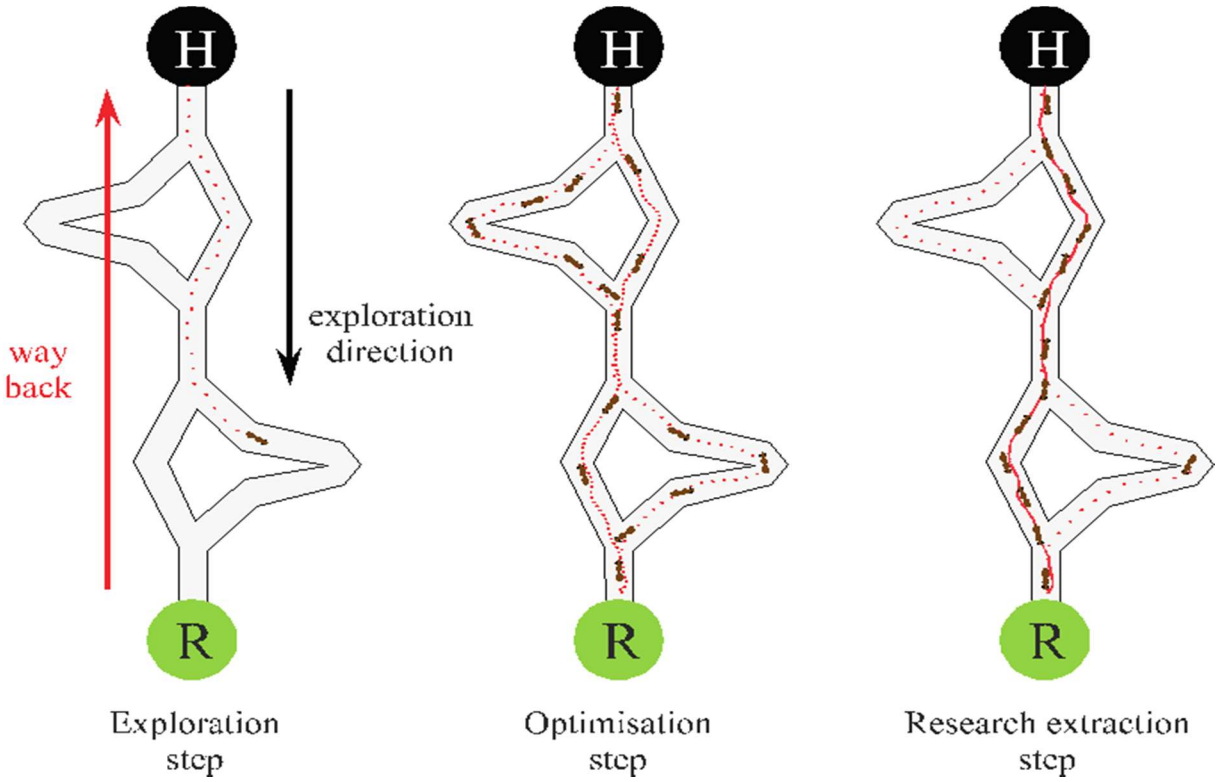

Figure I. 8: position in ant colony algorithm.

Methodology of ant colony algorithm

The ant colony optimization (ACO) algorithm works by simulating the foraging behaviour of ants to find the optimal solution to optimization problems[16], particularly those related to finding the shortest path. Here's how the algorithm typically operates:

- Initialization: Initially, a set of artificial ants is randomly placed in the problem space. These

ants represent potential solutions to the optimization problem.

- Solution Construction: Each ant begins to construct a solution by iteratively selecting paths.

based on the probability to select discrete values of design variables $p_j^k = \frac{\tau_j}{\sum_{j=1}^{m} \tau_j}$.

τ: pheromone trail.

k: the number of any possible solution.

This rule is guided by a combination of heuristic information (distance between nodes) and pheromone trails.

- Pheromone Update: As ants move along their paths, they deposit pheromone trails.

on the edges they traverse. The amount of pheromone deposited is typically proportional to the quality of the solution found. Pheromone evaporation may also occur to prevent stagnation and encourage exploration.

- Solution Evaluation: Once each ant has completed its solution construction, the

quality of each solution is evaluated based on a predefined objective function this function can be best f_{best} or worst f_{worst} (total distance travelled). Ants that find better solutions contribute more pheromone to the edges they traversed by $\tau_j^{new} \leftarrow \tau_j^{old} + \sum_k \Delta\tau_j^{(k)}$.

$$\Delta\tau j(k) = \frac{\zeta f_{best}}{f_{worst}}$$

The other ants, evaporate the pheromone of the other paths by $\tau_j^{new} \leftarrow (1-\rho)\tau_j^{old}$.

f_{best} →best objective function.

f_{worst} → worst objective function.

ζ→ scaling parameter.

ρ →Eevaporate rate.

- Global Update: After all ants have completed their tours, a global pheromone update.

is performed to adjust the pheromone levels on all edges. This update typically involves evaporating existing pheromone trails and depositing new pheromone based on the quality of the solutions found.

- Termination Criterion: The process of constructing solutions, updating pheromone

trails, and evaluating solutions iterates for a predefined number of iterations or until a termination criterion is met (a satisfactory solution is found).

By iteratively repeating these steps, the ACO algorithm gradually converges towards the optimal solution to the optimization problem. The pheromone trails effectively represent a form of indirect communication among the artificial ants, guiding them towards promising regions of the search space and facilitating the discovery of high-quality solutions.

I.3.4 Bee colony algorithm

The artificial bee colony algorithm (ABC) is a heuristic approach influenced by the honey gathering behaviour of bees. It assigns varying weights to neurons depending on their respective contributions to outcomes, with greater weights allocated to neurons with more significant impacts and lower weights to those with lesser impacts[17].

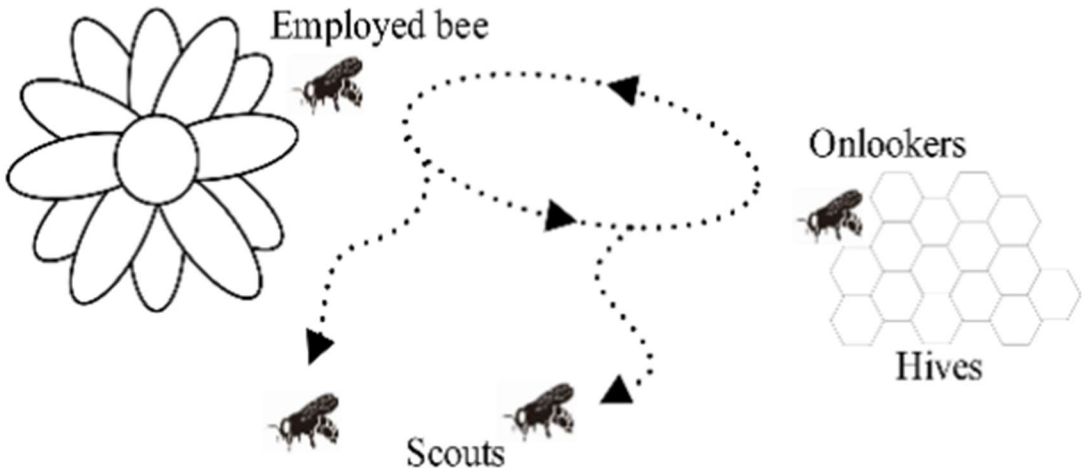

Figure I. 9: Schematic diagram of collecting nectar of bee colony.

Methodology of bee colony algorithm:

The artificial bee colony (ABC) algorithm operates by simulating the foraging behaviour of honeybees to find optimal solutions to optimization problems.

- Initialization: The algorithm starts by randomly generating a population of artificial

bees, which represent potential solutions to the optimization problem.

- Employed Bees Phase: In this phase, each employed bee explores a solution by

adjusting one component (or parameter) of its current solution. This adjustment is typically made using a neighbourhood search strategy, such as randomly selecting a neighbouring solution or perturbing the current solution within a certain range.

- Onlooker Bees Phase: During this phase, onlooker bees select solutions to explore.

based on the quality of solutions discovered by employed bees. The probability of selecting a solution is proportional to its quality, with better solutions having a higher probability of being chosen.

- Scout Bees Phase: If an employed bee exhausts all possible solutions within its

neighbourhood without finding an improved solution, it becomes a scout bee. Scout bees explore new solutions randomly to introduce diversity into the population.

- Solution Evaluation: After employed and onlooker bees have explored solutions, each.

solution is evaluated based on a predefined objective function. This function quantifies the quality of the solution in terms of how well it satisfies the optimization criteria (minimizing a cost function or maximizing a performance metric).

- Update Solutions: Based on the evaluations, solutions are updated iteratively. Better

solutions replace inferior ones in the population, leading to an improvement in the overall quality of solutions over successive iterations.

- Termination Criterion: The algorithm continues to iterate through the phases until a

termination criterion is met. This criterion could be a maximum number of iterations, convergence to a satisfactory solution, or a predefined computational budget.

By iteratively repeating these steps, the ABC algorithm gradually converges towards an optimal or near-optimal solution to the optimization problem. The exploration of solutions by employed and onlooker bees, along with the introduction of diversity by scout bees, allows the algorithm to effectively search the solution space and find high-quality solutions.

I.3.5 Grey wolf algorithm

The Gray Wolf Optimization (GWO) algorithm is a novel meta-heuristic approach designed to address optimization challenges. It draws inspiration from the social structure and hunting behaviour of grey wolves in the wild. The GWO algorithm employs a hierarchical leadership model, featuring four distinct types of grey wolves: alpha, beta and delta[18].

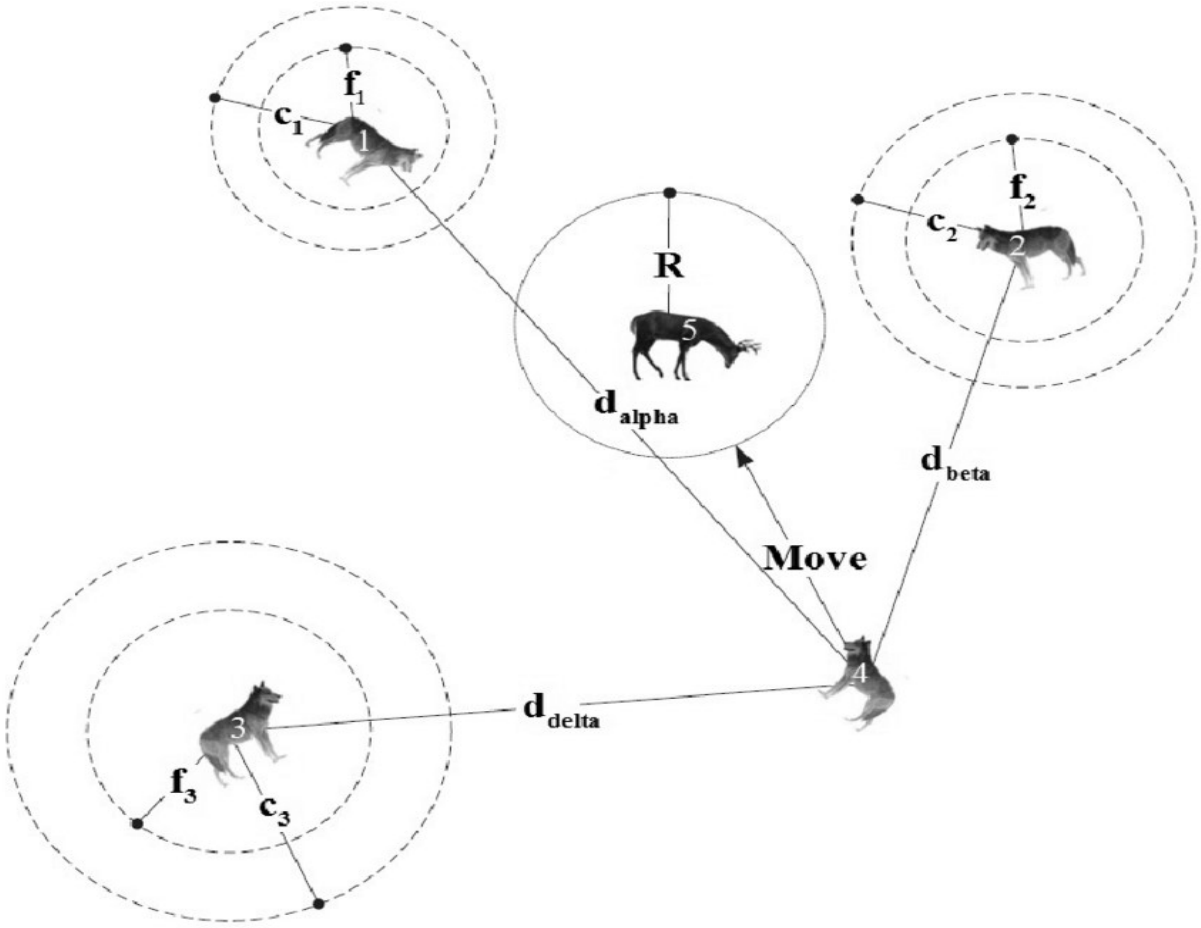

Figure I. 10: the position in GWO algorithm.

Methodology of Gray wolf algorithm:

The Grey Wolf Optimization (GWO) algorithm is a nature-inspired metaheuristic optimization algorithm based on the social hierarchy and hunting behaviour of grey wolves. Here's a general methodology of how the grey Wolf Algorithm typically operates:

- Initialization: Start by initializing a population of grey wolves, usually randomly, within the search space of the optimization problem. Each grey wolf represents a potential solution.

- Objective Function Evaluation: Evaluate the fitness or objective function value of each grey wolf in the population. The objective function represents the problem to be optimized.
- Pack Leadership: Determine the leadership hierarchy within the grey wolf pack. This is based on the fitness values of individual wolves. The alpha, beta, delta, and omega wolves represent the best, second-best, third-best, and worst solutions, respectively.
- Exploration and Exploitation: The alpha, beta, and delta wolves lead the pack in exploring the search space for better solutions. They guide the rest of the pack towards promising regions, balancing between exploration (searching for new solutions) and exploitation (refining existing solutions).
- Update Positions: Each wolf adjusts its position based on the positions of the alpha, beta, and delta wolves. This update is influenced by mathematical equations that simulate the social interactions and hunting behaviours of grey wolves.
- Boundary Handling: Ensure that the updated positions of the wolves remain within the boundaries of the search space, applying any necessary boundary handling techniques. Fitness Evaluation and Selection: After updating positions, evaluate the fitness of the new solutions. Select the alpha, beta, delta, and omega wolves based on the updated fitness values.
- Termination Criterion: Repeat steps 4-7 for a predefined number of iterations or until a termination criterion is met (reaching a satisfactory solution, reaching a maximum number of iterations).
- Solution Extraction: Extract the best solution found during the optimization process, typically, the position of the alpha wolf, as the optimal solution to the optimization problem.
- Result Analysis: Analyse the obtained solution and assess its quality in terms of the objective function value and its feasibility with respect to any constraints present in the problem.

By following these steps, the Grey Wolf Algorithm iteratively searches the solution space to find an optimal or near-optimal solution to the given optimization problem.

I.3.6 Random forest

Random forest is a popular algorithm in machine learning that falls under the umbrella of ensemble learning techniques. It functions by creating multiple decision trees during the training phase and then aggregates their predictions to make final decisions. In classification tasks, it selects the most common class among the trees' predictions, while in regression tasks, it calculates the average prediction from the individual trees. Each decision tree is constructed using a subset of the training data and a random subset of features, giving rise to the term "random forest."[19] This approach introduces randomness, which helps prevent overfitting and enhances the model's ability to generalize to unseen data. Another advantage of random forest is its capability to handle both numerical and categorical data, making it adaptable to diverse datasets. Overall, random forest stands out for its simplicity, efficiency, and robust performance in addressing complex classification and regression challenges.

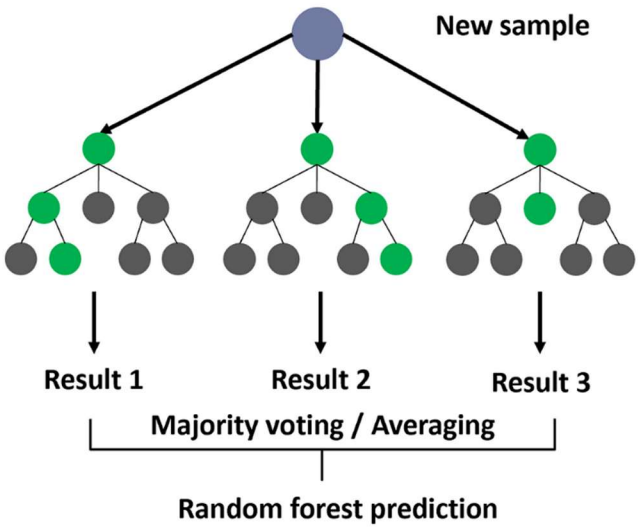

Figure I. 11: random forest prediction.

I.3.7 Extreme Gradient Boosting Trees (XGBoost)

Extreme Gradient Boosting Trees (XGBoost) stands out as a potent machine learning algorithm renowned for its prowess in predictive modeling endeavors. Falling under the ensemble learning paradigm, XGBoost employs a method known as gradient boosting. Through this technique, XGBoost constructs a sequence of decision trees, where each subsequent tree aims to rectify the errors of its predecessors. It iteratively enhances a predefined objective function, like minimizing loss or maximizing accuracy, by judiciously adding trees that most effectively

bolster the model's performance[20]. Additionally, XGBoost incorporates regularization strategies to forestall overfitting and amplifies its efficiency via parallel processing and hardware optimization. In sum, XGBoost garners acclaim for its remarkable predictive precision, rapid execution, and adaptability across a myriad of datasets and tasks.

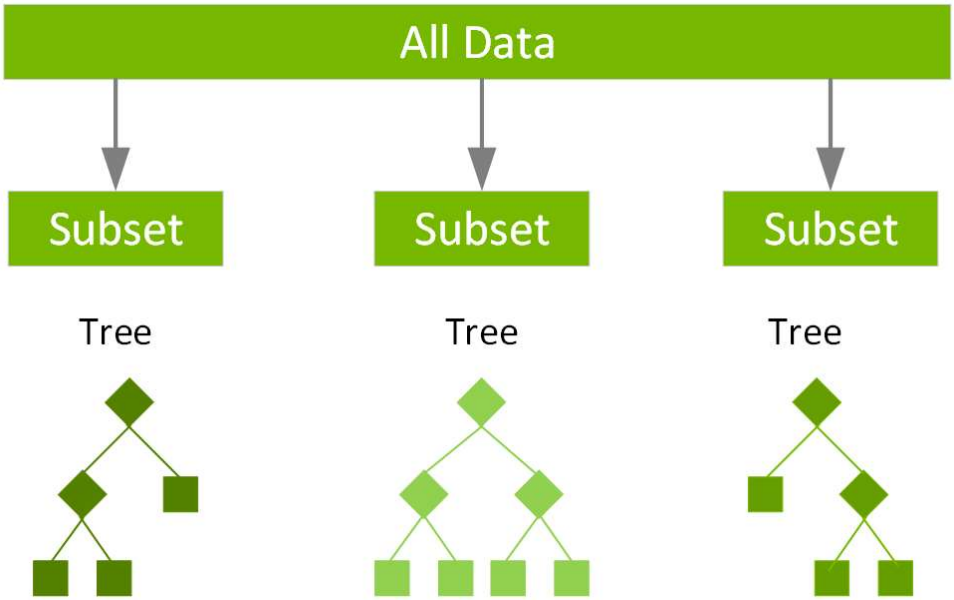

Figure I. 12: Gradient Boosting Trees.

I.4 Conclusion:

In conclusion, the field of artificial intelligence (AI) has experienced notable advancements driven by a wide array of techniques and algorithms, each contributing uniquely to addressing intricate problems across diverse domains. Principal among these techniques are machine learning and deep learning:

- Machine Learning: Renowned for its capacity to glean insights from data and facilitate predictive analysis, machine learning finds widespread utility in applications spanning recommendation systems to fraud detection.
- Deep Learning: Fueled by neural networks, deep learning excels in processing and comprehending intricate data such as images, audio, and text, catalyzing breakthroughs in fields like computer vision, natural language processing, and autonomous driving.

Complementing these techniques are several AI algorithms that play pivotal roles in tackling specific challenges:

- Genetic Algorithm: Inspired by natural selection, genetic algorithms iteratively refine solutions to optimization and search problems, with applications spanning scheduling, optimization, and design tasks.
- Ant Colony Algorithm: Modeled after ant behavior, these algorithms simulate cooperative foraging to solve optimization problems, particularly adept at addressing routing and scheduling challenges.
- Particle Swarm Optimization: Drawing inspiration from collective behavior in nature, particle swarm optimization algorithms iteratively refine solutions by simulating particle movement within a search space, commonly applied in optimization and function optimization tasks.
- Gray Wolf Algorithm: Inspired by the hunting tactics of gray wolves, this algorithm specializes in optimization problems, particularly in continuous optimization and engineering design endeavors.
- Random Forest: A widely embraced ensemble learning technique, random forests amalgamate predictions from multiple decision trees to bolster accuracy and robustness, extensively employed in classification, regression, and anomaly detection.

Importantly, these represent just a subset of the manifold AI algorithms available, each boasting distinct strengths and applications. As AI research advances, the continual development and enhancement of new algorithms promise to broaden the capabilities of AI systems, empowering them to address an even wider spectrum of challenges across diverse fields.

Chapter II

Overview of AI implementations in civil engineering

II.1 Introduction

Artificial intelligence (AI) within civil engineering can be delineated as a branch of computer science dedicated to exploring, advancing, and applying intelligent computing systems[1]. This facet holds pivotal importance in the digital transformation and enhancement of civil engineering, facilitating substantial advancements in automation, efficiency, and dependability, while establishing a direct correlation between physical and digital construction realms[2]. The primary aim of this domain lies in exploring diverse methodologies to emulate and execute cognitive functions akin to those of the human brain, fostering technological innovations and formulating relevant hypotheses. Essential theories and methodologies encompass symbolism, behaviourism, and connectionism. Since its inception in the 1950s, artificial intelligence has sparked numerous expectations and aspirations. Widely acknowledged as a transformative technology, AI offers an alternative avenue for tackling various challenges and ambiguities. Moreover, it finds application in intricate system modelling, identification, optimization, forecasting, and control. The evolution of AI can be segmented into five distinct phases: pre-1956, incubation, formation, the dark phase, knowledge application, and integrated development (1986-present)[21]. Recent strides in data collection and processing hardware have ushered in a revolutionary machine learning technique known as Deep Learning (DL) within the realm of AI. Given the significant economic footprint of the construction sector and its potential to drive national development, governments worldwide are intensifying efforts to integrate AI, leveraging big data to instruct complex algorithms, thereby bolstering the construction industry. Once trained, AI technologies exhibit agility in making swift predictions and extrapolations in the domain of digital construction, adeptly addressing multifaceted and nonlinear functional challenges.

II.2 Civil engineering domain

II.2.1 Structural Engineering

Focuses on the design and analysis of structures to ensure they can withstand loads and environmental conditions. This includes buildings, bridges, dams, and towers[22].

II.2.2 Geotechnical Engineering

Deals with the behaviour of earth materials and assessing the stability of soil and rock slopes for construction projects. Geotechnical engineers also work on foundation design for structures[23].

II.2.3 Transportation Engineering

Involves planning, design, and operation of transportation systems, including highways, railways, airports, and urban transit systems[24].

II.2.4 Environmental Engineering

Addresses environmental issues related to water and air quality, waste management, pollution control, and sustainable development[25].

II.2.5 Water Resources Engineering

Focuses on the management of water resources, including water supply systems, flood control, irrigation systems, and hydroelectric power generation[26].

II.2.6 Construction Engineering

Involves project management, scheduling, cost estimation, and quality control during the construction phase of infrastructure projects[22].

II.3 The implementation of AI in civil engineering

II.3.1 Structural Health Monitoring

AI techniques such as machine learning (ML) and computer vision are employed for monitoring the structural health of buildings, bridges, and other infrastructure. This includes detecting defects, assessing structural integrity, and predicting potential failures.

II.3.1.1 Variations of Ant Colony Optimization for the solution of the structural damage identification problem

This study addresses the problem of identifying structural stiffness coefficients in a damped spring-mass system using Ant Colony Optimization (ACO) algorithms. Structural damage identification is a complex inverse problem characterized by significant sensitivity to small perturbations in input data, leading to substantial variations in the final solution. Traditional methods for solving this ill-posed problem often fall short due to the high dimensionality and presence of numerous local optima.

The researches (Carlos E. Braun et al) presented various adaptations of the ACO metaheuristic, both as standalone methods and in combination with the Hooke-Jeeves (HJ) local search algorithm. These adaptations include different strategies for pheromone evaporation and deposit, as well as the frequency of invoking the local search. A discretization approach is employed to manage continuous domain design variables within ACO, maintaining the native algorithm's characteristics while integrating new heuristic information based on search history.

The experimental setup involves simulating damage across structural elements and evaluating performance using both noiseless and noisy synthetic data. The results demonstrate that the hybrid ACO-HJ method, which incorporates a rank-based pheromone deposit strategy and the newly introduced heuristic information, yields the most accurate and reliable solutions for damage identification. This method effectively balances exploration and exploitation, outperforming other ACO variations in both accuracy and convergence speed [27].

II.3.1.2 Wavelet-based multiresolution analysis coupled with deep learning to efficiently monitor cracks in concrete

This study focuses on the critical issue of identifying the initial formation of cracks within concrete structures. These cracks, hidden from external view, have the potential to propagate unnoticed until they culminate in structural failure. Such failures can have catastrophic consequences, particularly in sensitive infrastructures such as nuclear power plants, dams, and bridges. Detecting and monitoring cracks during their early development stages is imperative to avert such disasters and uphold the safety and stability of essential structures.

Ahcene Arbaoui, Abdeldjalil Ouahabi, Sébastien Jacques,and Madina Hamiane employed a multi-step approach to crack detection in aging concrete samples.

Ultrasonic Inspection: First, they employed non-destructive ultrasonic testing to gather a specific ultrasonic signal that pinpoints the defect within the concrete sample.
Multiresolution Analysis with Wavelets: In a crucial step, they performed multiresolution analysis using wavelets on the captured ultrasonic signal. This analysis helped isolate and highlight the presence of cracks within the material. The analysis also generated a B-scan map, which essentially visualized the defect's location in space across different resolutions.
Deep Learning Classification: Finally, the multi-resolution image obtained from the wavelet analysis was fed into a deep learning algorithm based on Convolutional Neural Networks (CNNs). Architectures like AlexNet and VGG16 were specifically used within this deep learning model to automatically distinguish between cracked and non-cracked regions in the concrete sample.
Early Detection Capability: The approach employed facilitated the identification of crack initiation in its early stages, even before visible signs of concrete fracture appeared on the surface. This capability allowed for prompt maintenance interventions to prevent structural failure.

The methodology shows an exceptional accuracy, with crack detection accuracy surpassing 98% and loss function values below 0.1, proving its efficacy in identifying cracks in concrete structures. Economically, it required only an on-site portable ultrasonic device and a standard processor, making it practical for field deployment. This wavelet-based multiresolution analysis

combined with deep learning offers a cost-effective and efficient solution for monitoring cracks in civil engineering structures[28].

II.3.2 Building Information Modelling (BIM)

AI algorithms are used in BIM software for tasks such as clash detection, energy simulation, and construction scheduling. BIM also integrates with AI-driven analytics for better decision-making throughout the construction lifecycle.

II.3.2.1 Integrating Building Information modelling (BIM) and Artificial Intelligence (AI) for smart construction schedule, cost, quality, and safety management

The study aimed to confront the complexities surrounding the integration of Building Information Modelling (BIM) and Artificial Intelligence (AI) within smart construction management. Their objective was to overcome obstacles impeding the full exploitation of this integration's potential, including technical intricacies, interoperability hurdles, and organizational dynamics. Through a comprehensive analysis of existing literature, empirical data, and insights, the study sought to offer a nuanced comprehension of the current scenario and potential pathways for seamless integration. Ultimately, the goal was to devise strategic recommendations and best practices to guide stakeholders in harnessing this integration's transformative power for sustainable, efficient, and safe construction practices.

Specific challenges addressed encompassed augmenting data-driven decision-making, automating processes, optimizing resource allocation, and forecasting potential risks and delays via AI algorithms analyzing historical project data and real-time information. Additionally, proactive measures based on real-time data aimed to prevent schedule overruns.

Nitin Liladhar Rane underscored the significance of establishing robust data management protocols, nurturing a culture of innovation and collaboration, and tailoring strategies to align with the distinct requirements and objectives of construction stakeholders. By tackling these challenges and seizing opportunities, the study aimed to contribute to the ongoing dialogue on technology's transformative role in the construction industry, fostering sustainable growth and progress within the sector.

Literature Review: The researchers meticulously reviewed existing literature to grasp the current landscape of BIM and AI integration in construction. Synthesizing key findings and expert insights, they pinpointed challenges and opportunities.

Data Analysis: Empirical data analysis provided insights into technical complexities, interoperability issues, and organizational dynamics affecting BIM and AI integration. This informed their understanding of specific hurdles requiring attention.

Recommendation Development: Drawing from research findings, the team formulated actionable recommendations and guidelines tailored to industry stakeholders. These addressed issues like data standardization, interoperability, skill development, and organizational change management.

Strategic Planning: The study aimed to offer strategic recommendations and best practices to empower stakeholders in leveraging BIM and AI integration for sustainable, efficient, and safe construction practices. They crafted a roadmap to navigate challenges and capitalize on opportunities presented by this synergy.

Collaborative Ecosystem: Emphasizing collaboration, innovation, and adaptability, the researchers advocated for a culture of cooperation within the construction sector. Through knowledge sharing and technology adoption, they aimed to foster continuous improvement and competitiveness.

Continuous Improvement: Recognizing the importance of ongoing research and development, the study underscored the need to continually address challenges and unlock the full potential of BIM and AI integration in construction management. This focus on continuous enhancement aimed to enable stakeholders to fully capitalize on the benefits of integration[29].

II.3.3 Natural Disaster Management

AI technologies aid in disaster preparedness, response, and recovery. This includes using AI for early warning systems, damage assessment, evacuation planning, and infrastructure resilience analysis.

II.3.3.1 fire detection on a reinforcement concrete (RC)

Exposure to high temperatures, such as those experienced during a fire, has detrimental effects on the characteristics of building materials. Concrete, for instance, undergoes changes due to chemical reactions during fire exposure, resulting in a decline in its physical and microstructural properties. This often leads to spalling, characterized by the explosive detachment of concrete chunks from the structure during a fire. Spalling not only diminishes the overall cross-sectional area of a reinforced concrete (RC) member but also exposes the internal layers of concrete and steel reinforcement to the fire. Consequently, this accelerates the deterioration of strength and elastic modulus throughout the structural member.

To mitigate potential challenges arising from fire disasters, Naser and his team opted to develop an innovative solution aimed at predicting fire responses in reinforced concrete (RC) columns. Their approach involved the creation of a user-friendly tool, termed "Nomograms via Machine Learning," which provides simple visual aids for forecasting fire responses. This initiative began with the following steps:

Data Acquisition: To build their fire resistance prediction tool, the researchers meticulously compiled a dataset of 248 real-world fire tests involving reinforced concrete (RC) columns. This data, gleaned from various scientific sources, encompassed ten key factors that influence RC column behavior in a fire. These factors included the column's geometry, loading conditions, and material properties. Additionally, the dataset included a crucial output variable - the fire resistance of each tested column.

Machine Learning Techniques: The researchers leveraged the power of machine learning to create the core of their prediction tool. Specifically, they employed linear regression for nomograms based on regression analysis, and logistic regression for those based on classification. By training these models on the comprehensive dataset, they empowered them to predict the fire response and fire rating of RC columns based on the input features.

Visualizing Predictions: Building upon the machine learning models, the researchers developed innovative nomograms. These nomograms act as user-friendly visual aids that simplify the process of predicting fire resistance for RC columns. Notably, these nomograms incorporate various factors that are often overlooked in traditional engineering knowledge and

design codes. This inclusion significantly enhances the accuracy and practicality of these fire resistance prediction tools.

Performance Evaluation: To ensure the effectiveness of their creation, the researchers rigorously validated and analyzed the developed nomograms. This process assessed their ability to accurately predict the fire response and fire rating of RC columns. The study also compared the accuracy of these nomograms to existing methods, thereby demonstrating the clear advantage of the machine learning-based approach in boosting predictive capabilities.

Real-World Impact: The study culminates by exploring the broader implications of its findings for the civil engineering field. The potential to integrate these nomograms into building codes and standards is highlighted, paving the way for more robust fire safety practices. Additionally, the authors provide valuable recommendations for further research. These include exploring different types of nomograms and refining the accuracy of the predictive models through advanced regression techniques.

The study introduced a Simplified Visual Tool for Predicting Fire Response of RC Columns with significant outcomes. The regression-based nomogram, using a linear regression model, had an R-squared value of 0.64 and outperformed existing codes in predicting fire resistance. The classification-based nomogram, crafted with a logistic regression model, matched advanced machine learning models in accuracy, sensitivity, and specificity. Both nomograms surpassed traditional methods in comprehensive fire resistance evaluation. These tools offer accessible methods for forecasting and classifying fire response, providing practical value for engineers. Their integration into building regulations could enhance fire safety and structural design practices.

II.3.4 Geotechnical Engineering

AI techniques are used in geotechnical engineering for tasks such as slope stability analysis, soil classification, and predicting ground settlement. This assists in mitigating risks associated with foundation failures and landslides.

Artificial intelligence techniques have been applied to tackle various geotechnical challenges. Two studies are presented in this work as follow:

II.3.4.1 Predicting the ultimate bearing capacity of shallow footings resting on two distinct soil layers

A critical challenge in geotechnical engineering is predicting the ultimate bearing capacity of shallow footings resting on two distinct soil layers. This capacity determines the maximum load a foundation can withstand before failure. To address this challenge, researchers investigated the use of various machine learning and evolutionary techniques. These techniques included artificial neural networks (ANNs), genetic algorithms (GAs), and particle swarm optimization (PSO). The goal was to develop accurate predictive models for estimating the ultimate bearing capacity. The study then focused on optimizing these models and evaluating their performance to generate valuable insights for practical engineering applications
Hossein Moayedi, Arash Moatamediyan, Hoang Nguyen, Xuan Nam Bui, Dieu Tien Bui, and Ahmad Safuan A. Rashid, implemented a structured approach in their investigation aimed at predicting the ultimate bearing capacity of shallow footings on two-layered soil conditions. The essential steps involved in their methodology are outlined below:

Data Collection and Pre-processing: A dataset comprising 3515 full-scale numerical simulations of single shallow footings on two-layered soil conditions was acquired. Effective parameters influencing the ultimate bearing capacity, such as friction angle, dilation angle, unit weight, and elastic modulus, were identified. The dataset underwent pre-processing to identify dependable model inputs and outputs for training artificial intelligence models.

Model Development: Various non-linear intelligent models, including GA-ANN, ANFIS, DEA, FFNN, GRNN, and PSO-ANN, were employed to estimate the ultimate bearing capacity. A database incorporating fourteen inputs and one output (Fy) was compiled to facilitate model development. The models were optimized based on the influential parameters identified and the dataset obtained from Finite Element Method (FEM) simulations.

Model Evaluation: The performance of the developed models was evaluated by comparing their predictions with actual ultimate bearing capacity values. Evaluation metrics such as Root Mean Square Error (RMSE), R-squared (R^2), and Variance Accounted For (VAF) were utilized to gauge the accuracy and efficacy of the models in predicting bearing capacity.

Result Analysis: Results obtained from each technique were compared to discern the influence of significant parameters on the prediction of ultimate bearing capacity.

The study presented findings indicating the efficacy of various artificial intelligence and machine learning models in forecasting the ultimate bearing capacity of shallow footings on two-layered soil conditions. Here are key observations from their investigation:

Model Performance: Multiple models, encompassing GA-ANN, ANFIS, DEA, FFNN, GRNN, and PSO-ANN, demonstrated satisfactory predictive capabilities in estimating the ultimate bearing capacity. The PSO-ANN model emerged as notably superior and more dependable compared to alternative techniques, exhibiting high accuracy in predicting bearing capacity.

Evaluation Metrics: Assessment metrics such as Root Mean Square Error (RMSE), R-squared (R^2), and Variance Accounted For (VAF) were employed to appraise model performance. The PSO-ANN model showcased remarkable values of 0.01, 0.99, and 99.90 for RMSE, R^2, and VAF, respectively, on the training dataset, indicating its exceptional accuracy and reliability in predicting bearing capacity.

Comparison of Models: Models were compared based on their predictive prowess and accuracy in estimating ultimate bearing capacity. The PSO-ANN model consistently outperformed other models, underscoring its efficacy in resolving the complex engineering problem.

Ranking System: A novel ranking system termed CER (color intensity rating) was devised to evaluate the efficacy of proposed methods based on their outcomes. The PSO-ANN model attained high rankings in accuracy and performance, further affirming its superiority in forecasting ultimate bearing capacity.

Overall, the study concluded that the PSO-ANN model stood out as the most effective in predicting the ultimate bearing capacity of shallow footings on two-layered soil conditions. These insights offer valuable implications for future engineering applications and geotechnical research endeavors[30].

II.3.4.2 Predicting the ultimate axial bearing capacity of driven piles

In this study, the researchers aimed to address the challenge of accurately predicting the ultimate axial bearing capacity of driven piles. They noted that traditional methods and empirical equations often fall short in accurately estimating pile strength, particularly as the input parameters related to pile geometry and soil properties become more complex. To overcome these limitations, the researchers explored the potential of machine learning techniques, specifically Artificial Neural Network (ANN) and Random Forest (RF) algorithms, to develop a more precise and dependable predictive model for pile axial bearing capacity. By harnessing the power of these advanced computational methods, the study aimed to enhance prediction accuracy and efficiency in the realm of geotechnical engineering.
Tuan Anh Pham, Hai-Bang Ly, Van Quan Tran, Loi Van Giap, Huong-Lan Thi Vu, and Hong-Anh Thi Duong meticulously followed a structured approach in their research to forecast the ultimate axial bearing capacity of driven piles utilizing Artificial Neural Network (ANN) and Random Forest (RF) algorithms. Here's an outline of their methodology:

Data Collection: They compiled an extensive database comprising 2314 reports from static load tests conducted on driven piles. This dataset encompassed various parameters like pile diameter, length of pile segments, natural ground elevation, pile top elevation, guide pile segment stops driving elevation, pile tip elevation, average standard penetration test (SPT) values, and more.

Data Preparation: The dataset was meticulously divided into training (70%) and testing (30%) subsets to facilitate the development and validation phases of the machine learning models.

Model Development:

ANN: Employing Artificial Neural Network (ANN), the researchers constructed a predictive model for estimating the axial bearing capacity of piles.

RF: Additionally, they utilized the Random Forest (RF) algorithm to create a distinct predictive model for comparative analysis.

Model Evaluation: The performance of the ANN and RF models was evaluated using various error criteria such as mean absolute error (MAE), root mean squared error (RMSE), and coefficient of determination (R^2). The predicted outcomes from these machine learning models were juxtaposed with results obtained from five empirical equations sourced from literature and classical multi-variable regression.

Sensitivity Analysis: A sensitivity analysis was conducted to discern the most influential factors affecting the prediction of axial bearing capacity of piles. Key factors such as average SPT value and pile tip elevation emerged as significant contributors.

This research compared machine learning methods for predicting pile capacity. The Random Forest (RF) model significantly outperformed the Artificial Neural Network (ANN) model in accuracy. Even compared to traditional methods, RF showed better results. The study identified key factors affecting pile capacity and highlighted the potential of machine learning for pile design. It also suggests exploring even more advanced techniques for even better predictions[23].

II.3.5 Project Management

AI-powered project management tools assist in scheduling, resource allocation, risk analysis, and cost estimation, improving overall project efficiency and delivery.

II.3.5.1 Optimization of the energy consumption in activated sludge process using deep learning selective modelling.

The study tackled the challenge of optimizing energy consumption in wastewater treatment plants (WWTPs), with a specific focus on the activated sludge process. (Rafik Oulebsir, Abdelouahab Lefkir, Abdelhamid Safri, Abdelmalek Bermad) aimed to develop a method using artificial neural networks to create an optimal energy consumption model for WWTPs. This optimization is vital because WWTP operations incur significant energy costs, with aeration being identified as the most energy-intensive part of the wastewater treatment process.

In wastewater treatment plants, (Rafik Oulebsir, Abdelouahab Lefkir, Abdelhamid Safri, Abdelmalek Bermad) investigated a deep learning approach for optimizing energy consumption during the activated sludge process. This method involved a two-step data

selection process followed by deep neural network training. First, data points meeting environmental regulations were chosen. Then, from this pool, additional data points demonstrating optimal energy consumption based on pollution indicators were selected. Finally, the deep neural network model was trained using this curated data to estimate energy savings for unseen data.

The researchers' deep learning model achieved impressive results. During training, it demonstrated a strong correlation between predicted and actual values, with a coefficient of determination between 90% and 92%. This correlation remained high during testing, ranging from 74% to 82%. Furthermore, applying the optimized model to new data revealed significant energy savings, validating its effectiveness in optimizing energy consumption within wastewater treatment processes. This study underscores the potential of deep learning selective modelling for accurate energy consumption prediction in these facilities, paving the way for improved energy efficiency and reduced overall consumption in WWTPs[26].

II.4 The advantage of AI in civil engineering

AI offer numerous and diverse solutions to challenges that face engineers in various aspects of their work.

a. Efficiency

AI can automate repetitive tasks, such as drafting designs, analysing data, and generating reports, saving engineers time and enabling them to focus on more complex aspects of their projects.

b. Accuracy

AI algorithms can process large amounts of data and perform complex calculations with a high degree of accuracy, reducing errors in design, analysis, and decision-making. And this is just in the case of non-availability of data.

c. Optimization

AI can optimize designs, processes, and resource allocation to improve efficiency and reduce costs in construction projects. This includes optimizing structural designs, scheduling construction activities, and managing resources like materials and equipment.

d. Predictive Analytics

AI-powered predictive analytics can forecast project outcomes, identify potential risks, and recommend mitigation strategies based on historical data and real-time inputs, helping engineers make informed decisions and improve project outcomes.

e. Sustainability

AI can help engineers design more sustainable infrastructure by optimizing energy usage, reducing waste, and minimizing environmental impact. AI algorithms can analyse data to identify opportunities for improving energy efficiency, using renewable materials, and implementing eco-friendly construction practices.

f. Remote Monitoring and Management

AI-powered sensors and remote monitoring technologies can provide real-time data on the performance and condition of infrastructure, enabling engineers to remotely monitor and manage assets from anywhere, reducing the need for manual inspections and onsite visits.

g. Innovation

AI fosters innovation in civil engineering by enabling engineers to explore new design concepts, materials, and construction techniques. AI-powered generative design tools can generate novel design solutions that may not have been considered through traditional methods, leading to innovative and efficient designs.

h. Cost-Effectiveness

AI can help reduce costs in civil engineering projects by optimizing resource utilization, minimizing waste, and improving productivity. By streamlining processes and automating tasks, AI can help projects stay within budget and improve the overall return on investment.

i. Decision Support

AI can provide valuable insights and decision support to engineers throughout the project lifecycle, helping them evaluate alternative solutions, assess risks, and make data-driven decisions to achieve project objectives more effectively.

II.5 challenges that civil engineers face when implementing AI

While the integration of Artificial Intelligence (AI) in civil engineering holds immense potential, it also presents several challenges that civil engineers may encounter during implementation[31]:

a. Data Availability and Quality

AI models rely heavily on data for training and decision-making. Obtaining large and relevant datasets specific to civil engineering tasks can be challenging. Additionally, ensuring the quality, accuracy, and reliability of the data is crucial for the effectiveness of AI algorithms[32].

b. Interdisciplinary Knowledge

Implementing AI in civil engineering often requires interdisciplinary knowledge spanning engineering, computer science, and data science. Civil engineers may need to collaborate with experts from other fields to develop and deploy AI solutions effectively.

c. Complexity of Infrastructure Projects

Civil engineering projects involve complex and multifaceted challenges, including structural design, environmental considerations, and regulatory requirements. Developing AI algorithms that can address these complexities and provide holistic solutions is a significant challenge.

d. Regulatory and Safety Standards

Civil engineering projects must comply with strict regulatory and safety standards to ensure public safety and environmental protection. Integrating AI into the design, construction, and maintenance processes while adhering to these standards requires careful consideration and validation[32].

e. Ethical and Social Implications

AI applications in civil engineering raise ethical concerns related to privacy, fairness, and accountability. Engineers must consider the social implications of AI technologies, such as job displacement, equity in access to infrastructure, and unintended consequences of AI-driven decisions[33].

f. Adoption and Cultural Resistance

Introducing AI into traditional engineering practices may face resistance from stakeholders who are unfamiliar or sceptical about the technology. Civil engineers may encounter challenges in gaining acceptance, trust, and adoption of AI solutions within their organizations and the broader industry[33].

g. Cost and Resource Constraints

Implementing AI technologies often requires significant investments in infrastructure, software, and training. Small firms or organizations with limited resources may face challenges in acquiring and deploying AI solutions effectively[32].

h. Interpretability and Transparency

AI models can sometimes be opaque and difficult to interpret, especially for complex tasks like structural design or risk assessment. Ensuring transparency and explainability in AI-driven decisions is essential for gaining trust and acceptance among stakeholders.

i. Security and Privacy Risks

AI systems in civil engineering may be vulnerable to cybersecurity threats, such as data breaches, malicious attacks, or algorithm manipulation. Protecting sensitive project data and ensuring the security of AI systems is critical for maintaining the integrity and reliability of infrastructure projects[34].

j. Long-term Maintenance and Support

Civil engineering projects have long lifecycles, and AI systems must be maintained, updated, and supported over time. Ensuring the scalability, reliability, and sustainability of AI solutions throughout the project lifecycle is a significant challenge for civil engineers.

II.6 Future implication of AI in civil engineering

The future implications of Artificial Intelligence (AI) in civil engineering are vast and transformative, shaping the way infrastructure is designed, constructed, and maintained. Some key future implications include[35]:

a. Smart Infrastructure Design

AI will enable the creation of smarter, more efficient infrastructure designs. Generative design algorithms can explore a wide range of design options, optimizing for factors such as cost, sustainability, and resilience. AI will also facilitate the integration of sensors and IoT devices into infrastructure, enabling real-time monitoring and adaptive functionality.

b. Predictive Maintenance

AI-powered predictive maintenance systems will become standard in civil engineering. By analysing sensor data and historical maintenance records, AI algorithms will predict when infrastructure components are likely to fail and recommend proactive maintenance actions, reducing downtime and extending asset lifespans.

c. Autonomous Construction

Construction sites will increasingly utilize autonomous equipment and robotics, guided by AI algorithms. AI-driven construction machinery will perform tasks such as excavation, material handling, and assembly with greater precision and efficiency, while AI-powered drones will assist in site surveying, inspection, and monitoring.

d. Virtual Design and Construction (VDC)

AI will enhance Virtual Design and Construction processes, allowing engineers to create digital twins of infrastructure projects and simulate construction processes virtually. AI algorithms will optimize construction sequencing, resource allocation, and logistics planning, minimizing delays and cost overruns.

e. Enhanced Risk Management

AI will improve risk management in civil engineering by analysing vast amounts of data to identify potential risks and vulnerabilities in infrastructure projects. AI-driven risk assessment

tools will help engineers make informed decisions to mitigate risks and enhance project resilience against natural disasters, climate change, and other threats.

f. Adaptive Infrastructure

AI will enable infrastructure to adapt dynamically to changing environmental conditions and user needs. Smart transportation systems will optimize traffic flow in real-time, while AI-controlled energy grids will balance supply and demand efficiently. Adaptive infrastructure will enhance safety, efficiency, and sustainability across various domains.

g. Sustainable Development

AI will play a crucial role in advancing sustainable development goals in civil engineering. AI algorithms will optimize energy usage, reduce waste, and minimize environmental impact in infrastructure projects. AI-driven simulations and modelling tools will inform decision-making to design more resilient and eco-friendly infrastructure solutions.

h. Augmented Engineering Workforce

AI will augment the capabilities of civil engineering professionals, enabling them to focus on high-level tasks that require creativity and expertise. AI-powered tools will automate routine tasks, facilitate design exploration, and provide decision support, empowering engineers to innovate and solve complex challenges more effectively.

i. Global Connectivity and Collaboration

AI will facilitate global connectivity and collaboration in civil engineering. Cloud-based AI platforms will enable engineers to share data, collaborate on projects remotely, and access advanced simulation and analysis tools from anywhere in the world, fostering innovation and knowledge exchange across borders.

j. Ethical and Societal Implications

As AI becomes increasingly integrated into civil engineering practices, addressing ethical and societal implications will be crucial. Engineers must consider issues such as equity, privacy, and transparency to ensure that AI technologies benefit society as a whole and contribute to sustainable and inclusive development[33].

II.7 Conclusion

The rise of Artificial Intelligence (AI) is significantly reshaping civil engineering across health monitoring, concrete analysis with ultrasonics, Building Information Modeling (BIM), and geotechnical applications.

In health monitoring, AI algorithms become powerful allies, analyzing sensor data from structures to identify anomalies and predict potential failures. This proactive approach allows for optimized maintenance schedules, ultimately enhancing safety and extending the lifespan of infrastructure.

For concrete evaluation using ultrasonics, AI steps in to interpret ultrasonic signals, pinpointing defects, cracks, and voids within concrete structures with greater accuracy. This automation improves efficiency and cost-effectiveness compared to traditional manual inspections.

BIM, when combined with AI, creates a revolutionary approach to infrastructure projects. AI analyzes vast amounts of BIM data, optimizing designs, streamlining workflows, and mitigating risks throughout the project lifecycle. These fosters enhanced collaboration among stakeholders and leads to better-informed decision-making.

In geotechnical engineering, AI models become valuable tools for predicting soil behavior, analyzing geological data, and optimizing foundation designs. By leveraging AI's capabilities, engineers can mitigate risks associated with ground instability, optimize construction techniques, and ensure the long-term stability of infrastructure projects.

In conclusion, integrating AI into civil engineering unlocks transformative benefits across various domains. From health monitoring and concrete analysis to BIM and geotechnical applications, AI empowers civil engineers to improve safety, efficiency, and sustainability. This ultimately leads to the delivery of infrastructure projects that meet the demands of the modern world.

Chapter III

AI frameworks and libraries

III.1 Introduction

AI frameworks and libraries are foundational components of the artificial intelligence ecosystem, empowering developers and researchers to create and deploy machine learning models effectively. While both frameworks and libraries facilitate AI development, they serve distinct purposes and offer different levels of abstraction.

Frameworks provide comprehensive infrastructures for building and training machine learning models. They typically offer a wide range of functionalities, including high-level APIs for constructing neural networks, optimization algorithms for training models, and tools for deploying them in production environments. Frameworks abstract away low-level implementation details, enabling developers to focus on model architecture and experimentation. They provide a structured framework for organizing code and managing dependencies, making it easier to scale projects and collaborate with others.

On the other hand, libraries are collections of pre-written code modules that address specific tasks or functionalities within AI. Unlike frameworks, libraries do not impose a rigid structure or workflow on developers. Instead, they offer a set of tools and utilities for performing common tasks such as data preprocessing, feature extraction, or model evaluation. Libraries can be used independently or in conjunction with frameworks to augment their capabilities or address specialized requirements. They provide a more flexible and modular approach to AI development, allowing developers to mix and match components based on their specific needs.

In summary, while both frameworks and libraries are essential components of AI development, they serve different roles and cater to different aspects of the workflow. Frameworks provide comprehensive infrastructures for building and training models, while libraries offer specialized tools and utilities for performing specific tasks within AI. By leveraging frameworks and libraries in combination, developers can harness the full power of artificial intelligence to solve complex problems and drive innovation in various domains.

III.2 Type of frameworks and libraries

AI frameworks and libraries are diverse tools that cater to various aspects of artificial intelligence development. Here's some different types:

a. Deep Learning Frameworks

These are specialized tools designed for building, training, and deploying deep neural networks. They offer APIs for defining network architectures, optimizing models, and running computations efficiently, often leveraging GPUs for acceleration. Examples include TensorFlow, PyTorch, and Keras[36].

b. Machine Learning Frameworks

Unlike deep learning frameworks, these cover a broader spectrum of machine learning techniques beyond deep neural networks. They provide support for classical algorithms such as linear regression, decision trees, and clustering, along with utilities for data preprocessing, feature engineering, and model evaluation. Examples include scikit-learn, Apache Spark MLlib, and XGBoost[36].

c. Natural Language Processing (NLP) Libraries

NLP libraries focus on processing and understanding human language. They offer tools and models for tasks like text classification, sentiment analysis, named entity recognition, and machine translation. Examples include NLTK, SpaCy, and Gensim[37].

d. Computer Vision Libraries

These libraries specialize in tasks related to image and video processing, such as object detection, image segmentation, and facial recognition. They provide pre-trained models, algorithms, and utilities for analysing visual data. Examples include OpenCV, Dlib, and SimpleCV.

e. Reinforcement Learning Libraries

These are tools for developing and training agents that learn to make decisions through interaction with an environment. They offer algorithms, environments, and utilities for reinforcement learning tasks. Examples include OpenAI Gym, Stable Baselines, and RLLib.

f. AutoML Libraries

AutoML libraries automate various stages of the machine learning pipeline, including data preprocessing, feature engineering, model selection, and hyperparameter tuning. They enable developers to build and deploy models with minimal manual intervention. Examples include Google AutoML, H2O.ai, and TPOT.

g. Model Interpretability Libraries

These libraries help interpret and explain the predictions of machine learning models, enhancing their transparency and trustworthiness. They offer techniques for visualizing model behavior, identifying influential features, and understanding model decisions. Examples include Lime, SHAP, and eli5.

h. Probabilistic Programming Frameworks

These frameworks enable the specification and inference of probabilistic models. They are useful for tasks such as Bayesian inference, probabilistic graphical models, and uncertainty estimation. Examples include Pyro, Edward, and Stan.

i. General AI Development Platforms

These platforms offer end-to-end solutions for AI development, encompassing data preparation, model building, deployment, and monitoring. They often integrate multiple frameworks and libraries into a unified environment. Examples include Google AI Platform, Microsoft Azure ML, and AWS SageMaker.

III.3 Open-Source software vs Commercial software

AI frameworks and libraries come in both open-source and commercial variants. Open-source tools are often freely available and offer transparency, flexibility, and community-driven development. Commercial solutions may offer additional features, support, and services, but at a cost.

III.3.1 top open-source AI frameworks and libraries

III.3.1.1 TensorFlow

TensorFlow is an open-source deep learning framework developed by researchers and engineers at Google Brain, Google's AI research division. It was initially released in 2015 and has since become one of the most widely used frameworks for building and training machine learning models.

TensorFlow provides a flexible and scalable infrastructure for defining, training, and deploying machine learning models, particularly deep neural networks. Its key feature is its computational graph abstraction, which allows users to define complex mathematical computations as a directed graph of nodes, where each node represents a mathematical operation and each edge represents the flow of data (tensors) between nodes[38].

TensorFlow is primarily written in C++ for performance and efficiency, with Python serving as the main programming language for its high-level API and interface. Python is widely used for tasks such as defining model architectures, specifying training procedures, and interacting with the TensorFlow library. Additionally, TensorFlow supports other programming languages such as C++, Java, and JavaScript through its TensorFlow Serving and TensorFlow Lite components, enabling model deployment and inference in various environments.

Python's popularity and versatility have contributed to TensorFlow's widespread adoption among researchers, developers, and practitioners in both academia and industry. Its Python API provides a user-friendly interface for building and experimenting with complex machine learning models while leveraging the computational power and efficiency of the underlying C++ implementation.

TensorFlow supports a wide range of tasks in artificial intelligence, including:

a. Deep Learning: TensorFlow provides a comprehensive set of tools and APIs for building and training deep neural networks, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformer models.
b. Machine Learning: Beyond deep learning, TensorFlow supports traditional machine learning algorithms and techniques, such as linear regression, logistic regression, decision trees, and support vector machines.
c. Natural Language Processing (NLP): TensorFlow offers specialized modules and models for NLP tasks, such as text classification, sentiment analysis, named entity recognition, machine translation, and language generation.
d. Computer Vision: TensorFlow includes pre-trained models and tools for computer vision tasks, including image classification, object detection, image segmentation, and facial recognition.

e. Reinforcement Learning: TensorFlow provides support for reinforcement learning algorithms, enabling the development and training of agents that learn to make decisions through interaction with an environment.

f. Distributed Computing: TensorFlow includes features for distributed computing, allowing users to scale their machine learning workloads across multiple CPUs or GPUs, as well as distributed clusters of machines.

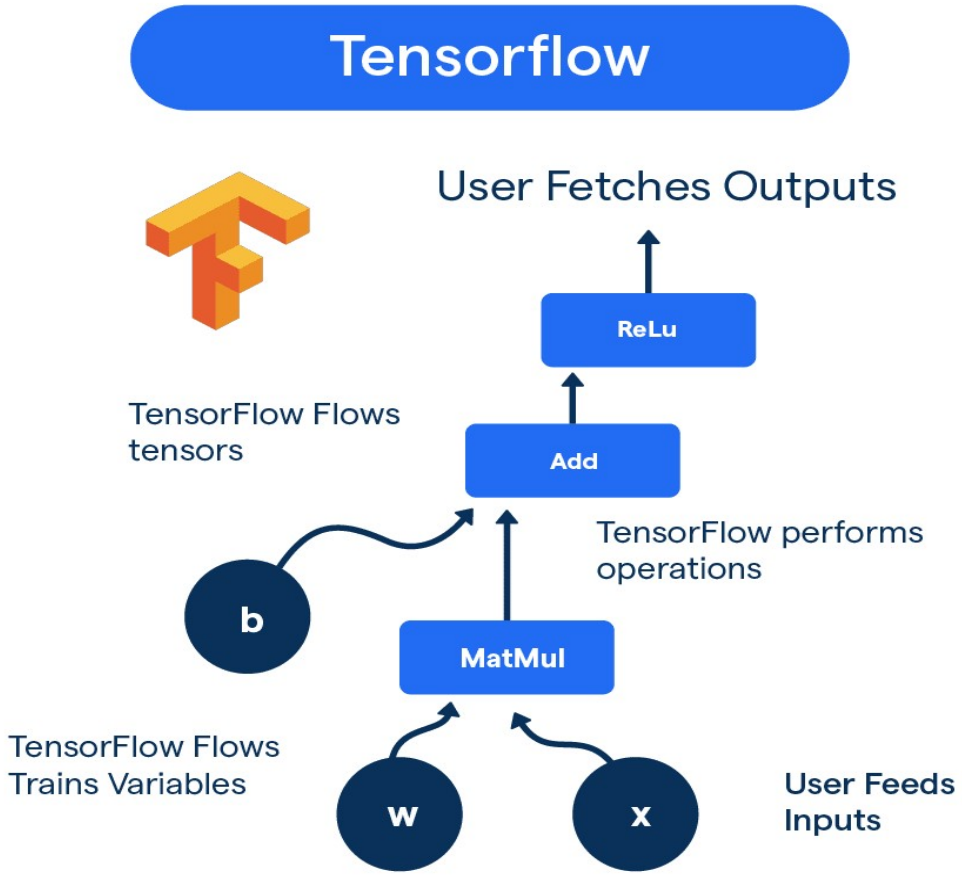

Figure III. 1: TensorFlow: User Fetches Outputs.

III.3.1.2 PyTorch

PyTorch, an open-source deep learning framework, originates primarily from the Facebook AI Research (FAIR) lab and was introduced to the public in 2016. Its rapid adoption by researchers and practitioners is owed to its dynamic computation graph, adaptable structure, and intuitive interface[39].

PyTorch serves as a Python-centric platform for constructing and training machine learning models, especially those involving deep neural networks. In contrast to frameworks utilizing static computation graphs, PyTorch embraces a dynamic approach, affording greater flexibility and simplified debugging. This dynamic nature means that computational graphs are generated on-the-fly during runtime, offering advantages for tasks featuring intricate or evolving architectures.

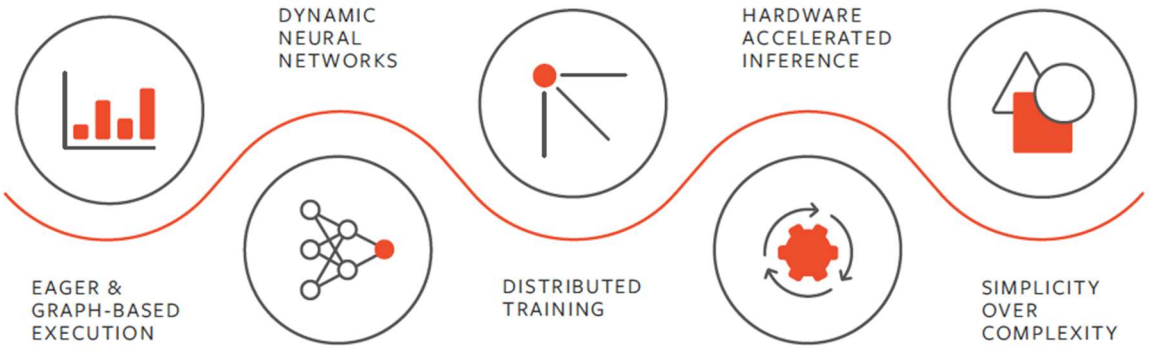

Figure III. 2: Concepts of PyTorch.

III.3.1.3 Scikit-learn

Developed by David Cournapeau, Scikit-learn is a Python library tailored for machine learning tasks, particularly data mining and analysis. Its intuitive interface, leveraging well-known scientific computing packages like NumPy, SciPy, and matplotlib, caters to both novices and experts in the field. Offering a plethora of machine learning algorithms, alongside tools for model evaluation, integration with other Python libraries, and robust preprocessing functionalities, Scikit-learn finds utility across various domains:

a. Data Science: Renowned for its versatility, Scikit-learn is a go-to tool in data science, facilitating tasks such as classification, regression, clustering, and dimensionality reduction. Its extensive toolkit proves invaluable for deciphering complex datasets.
b. Research: Researchers rely on Scikit-learn to prototype novel machine learning algorithms, benchmark their performance against existing methods, and conduct experiments. Its user-friendly interface expedites the experimentation process.

c. Education: Frequently integrated into academic curricula, Scikit-learn aids in teaching machine learning concepts comprehensively. Its simplicity allows students to implement algorithms covered in their courses with ease.

d. Industry Applications: From predictive modeling to fraud detection, Scikit-learn finds widespread usage in industries spanning finance, healthcare, e-commerce, and beyond. Its efficiency and adaptability make it adept at addressing diverse business challenges, including natural language processing, image recognition, and anomaly detection.

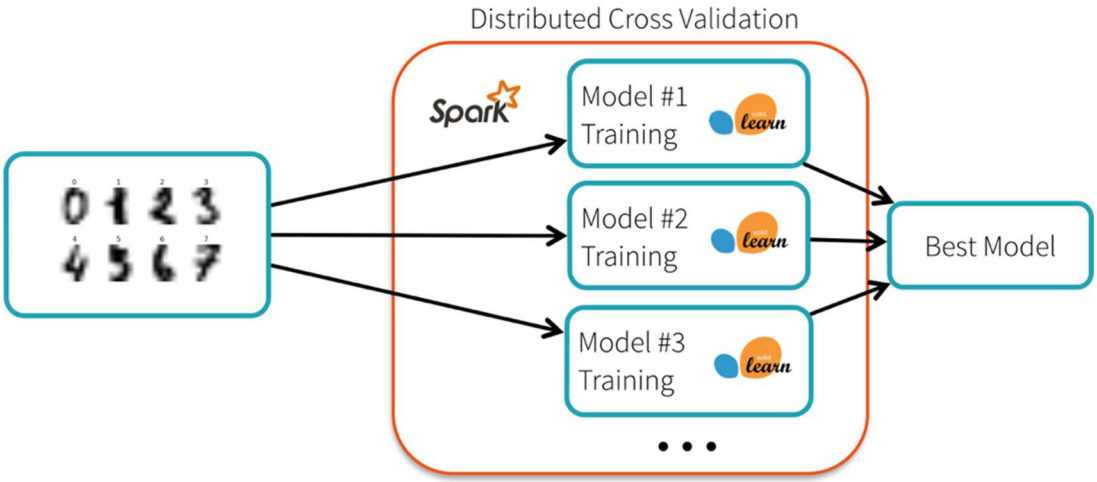

Figure III. 3: Distributed Cross Validation.

III.3.1.4 Microsoft Cognitive Toolkit (CNTK)

The Microsoft Cognitive Toolkit (CNTK) is a is a free and open-source sophisticated deep learning platform conceived and crafted by Microsoft, specifically tailored for constructing neural networks. Originating from the inventive minds at Microsoft Research, it made its debut in April 2016. CNTK distinguishes itself with its emphasis on efficiency, scalability, and versatility, rendering it ideal for honing deep learning models in diverse fields such as computer vision, speech recognition, and natural language processing. Notably, it accommodates both CPU and GPU computing, catering to a wide spectrum of users. Its comprehensive suite of features equips researchers and developers with powerful tools for exploring and implementing deep learning solutions[40].

However, it's important to note that getting accustomed to Microsoft CNTK might require more effort due to its potentially steeper learning curve in comparison to simpler frameworks designed for beginners.

Figure III. 4: Microsoft Cognitive Toolkit Logo.

III.3.1.5 PyBrain

PyBrain, an open-source machine learning library for Python, originated from the Swiss AI Lab IDSIA (Istituto Dalle Molle di Studi sull'Intelligenza Artificiale) starting around 2007. Engineered to be flexible and user-friendly, it caters to various machine learning tasks, including supervised learning, unsupervised learning, and reinforcement learning.

This framework is prized for its lightweight nature, making it ideal for experimentation and rapid prototyping. It boasts support for a diverse array of machine learning algorithms, further enhancing its versatility.

Furthermore, PyBrain's utility extends to educational settings, facilitating comprehension and hands-on exploration of machine learning concepts.

However, it's worth noting that PyBrain may have limitations such as sparse documentation and a smaller user community compared to mainstream libraries. Additionally, it might lack certain advanced features present in other frameworks.

Figure III. 5: PyBrain Logo.

III.3.1.6 Caffe

Caffe emerged as a deep learning platform stemming from the research efforts of Yangqing Jia during his doctoral program at the University of California, Berkeley, approximately in 2013. This framework is extensively employed for both the training and deployment phases of deep learning models. Its notable attributes include remarkable speed, modular design, and a flexible architecture, rendering it highly adaptable across diverse domains. Its widespread adoption spans academic research, industrial sectors, and extensive applications in fields like computer vision, natural language processing, and robotics[40].

Figure III. 6: caffe Logo.

III.3.2 top Commercial AI frameworks and libraries

III.3.2.1 OpenAI

OpenAI, founded in December 2015 by Elon Musk, Sam Altman, Greg Brockman, Ilya Sutskever, Wojciech Zaremba, and John Schulman, is committed to propelling artificial intelligence forward responsibly for the betterment of humanity[41].

The organization's portfolio includes diverse AI models like the GPT series, notably GPT-3, renowned for their proficiency in natural language processing, text generation, and comprehension. These models find broad utility across various sectors, including industries, research endeavors, and educational initiatives[42].

To incorporate OpenAI's models into your codebase, you'll engage their APIs, typically entailing the transmission of requests to their servers and the subsequent reception of responses. This interaction facilitates the integration of cutting-edge AI capabilities seamlessly into your applications[41].

Figure III. 7: OpenAI Logo.

III.3.2.2 OpenNN

OpenNN is a comprehensive framework dedicated to the creation, training, and deployment of neural network models. Crafted predominantly in C++, it shines for its remarkable efficiency and rapid processing capabilities. At its core, OpenNN offers an intuitive interface facilitating network configuration alongside a repertoire of functionalities including support for diverse network architectures, optimization algorithms, and data preprocessing utilities. Noteworthy is OpenNN's prowess in efficiently managing extensive datasets and intricate models. Its applications extend across both research pursuits and real-world scenarios where data-driven

decision-making powered by AI is imperative. These domains encompass finance, healthcare, manufacturing, and various others, reflecting its versatility and relevance across diverse industries.

Figure III. 8: OpenNN Logo.

III.3.2.3 IBM Watson

IBM Watson stands as an AI and machine learning platform brought to fruition by IBM. Its inception can be traced back to the early 2000s when a dedicated team of researchers and engineers within IBM embarked on its development journey. The platform's core architecture predominantly relies on Java, supplemented by modules written in languages like C++, Python, and JavaScript, ensuring support for diverse functionalities and seamless integrations[47].

IBM Watson stands out as a versatile suite of AI and machine learning solutions developed by IBM. Offering a diverse array of tools and functionalities, it empowers users to craft and implement AI-driven applications spanning various domains such as natural language processing, computer vision, and predictive analytics.

An advantage of IBM Watson lies in its seamless integration with IBM Cloud, streamlining the process of deploying and overseeing AI applications. Moreover, its strength is bolstered by IBM's wealth of experience in the field, ensuring that users can rely on its robust and dependable AI capabilities.

Nonetheless, it's worth noting that the pricing structure of IBM Watson services and consulting might present a hurdle for smaller enterprises or organizations constrained by budget considerations. While the suite offers extensive features, the affordability factor may influence the decision-making process for those in search of AI solutions and support services.

Figure III. 9: IBM Waston Logo.

III.3.2.4 Hugging Face

Hugging Face stands out as a significant player in the AI tools landscape, founded by Clément Delangue and Julien Chaumond in 2016. The foundation of the platform predominantly relies on Python, harnessing its adaptability and the rich ecosystem it offers for AI advancement.

Hugging Face is distinguished as a premier supplier of intuitive AI utilities, celebrated particularly for their "Transformers" library. This library plays a pivotal role in enabling sophisticated machine learning endeavors, particularly in tasks related to language processing and chatbot creation. Additionally, they provide utilities for image and sound generation, as well as for enhancing data management in AI models and simplifying the process of updating large-scale AI models.

Of significant note, Hugging Face enhances accessibility to their suite of tools by offering web-friendly versions. This accessibility empowers individuals of all skill levels, from beginners to seasoned professionals, to explore AI experimentation across a broad spectrum of domains, including natural language processing and computer vision.

Figure III. 10: Hugging Face Logo.

III.4 The difference between open source and commercial AI frameworks and libraries

III.4.1 Open-source AI frameworks

Open-source AI frameworks, governed by open-source licenses, offer users the freedom to utilize the software without restrictions [47].

Advantages of open-source frameworks include:

- Typically free, making them cost-effective for small projects and startups.
- Boasting vibrant communities, they serve as valuable hubs for learning and issue resolution.
- Users can delve into the source code, enabling greater control over AI implementations.

However, there are drawbacks to consider:

- Support may be limited. Although community assistance exists, it may not match the responsiveness or comprehensiveness of commercial support.
- Some open-source frameworks pose complexity, presenting challenges for beginners to fully comprehend[43].

III.4.2 Commercial AI frameworks

Commercial AI frameworks, crafted by companies under proprietary licenses, impose limitations on users' actions and may incur additional charges [53].

Advantages of commercial frameworks include:

- Dedicated support teams ensure swift resolution of issues.
- Emphasis on user-friendliness enhances accessibility for developers of varying expertise.
- Advanced features and optimizations tailored to specific needs are often found in commercial frameworks.

However, there are drawbacks to consider:

- High costs may deter smaller or bootstrapped projects.
- Adoption of a commercial framework may result in vendor dependency, constraining flexibility.

III.5 the integration of frame works and libraries into codebase

Integrating any framework into your code generally follows a similar process[44]:

a. Installation

Begin by installing the framework on your system, which typically entails downloading the framework package or utilizing package management tools such as pip (for Python) or npm (for Node.js). Ensure to adhere to the installation instructions outlined in the framework's documentation.

b. Initialization

Incorporate the framework into your codebase by importing it or including its components. This process varies depending on the programming language and framework utilized and may involve import statements, header file inclusions, or initialization of framework components.

c. Configuration

Tailor the framework to meet your specific requirements by configuring parameters, defining options, or specifying resource paths like models or datasets. Consult the framework's documentation for detailed guidance on configuration options.

d. Data Preparation

Prepare your data for utilization with the framework by loading it from files, databases, or APIs. Additionally, preprocess the data, such as normalization or scaling, and segment it into training, validation, and testing sets if applicable.

e. Model Building

Construct the model architecture using the framework's APIs or tools, specifying layers, neurons, connections, and other components. Depending on your task, you can opt for pre-trained models or design custom ones.

f. Training

Train the model using the training data, which involves feeding it into the model, adjusting parameters like weights and biases through optimization techniques like gradient descent, and evaluating the model's performance over multiple iterations or epochs.

g. Validation

Assess the trained model's performance and generalization ability using validation data, facilitating the detection of overfitting and enabling fine-tuning of model parameters if required.

h. Testing

Evaluate the model's performance on unseen data by testing it using separate testing data sets, providing insights into its real-world performance.

i. Deployment

Deploy the trained model for inference or production use, such as saving it to disk, bundling it with your application, or deploying it to cloud services for online inference.

j. Monitoring and Maintenance

Continuously monitor the deployed model's performance and undertake periodic maintenance tasks as necessary. This may involve retraining the model with new data, updating parameters, or scaling infrastructure to meet increasing demands.

III.6 Conclusion

Pre-written code libraries and frameworks act as accelerators in software development. Frameworks enforce a structured approach, defining the application's core architecture. In contrast, libraries function as modular building blocks for specific tasks.

The selection process hinges on your project's specific needs. Frameworks excel at building applications within a predefined structure, while libraries provide flexibility by allowing you to integrate functionalities into existing projects.

Open-source libraries boast free access and modifiability, fostering a collaborative environment and customization. However, dedicated support might be limited. On the other hand, commercial libraries frequently come with comprehensive support and documentation, but require licensing fees and may restrict modifications.

Ultimately, the optimal choice revolves around factors like project requirements, budget, and the desired level of control. When selecting frameworks and libraries for your software development projects, carefully consider the trade-offs between flexibility, support, and cost.

Chapter IV

Computing the bearing capacity based on Neural Network

IV.1 Introduction

The calculation of bearing capacity is a fundamental aspect of geotechnical engineering, which involves determining the maximum load that soil can support without failure. This process requires a deep understanding of soil properties and behavior under various loading conditions. Engineers use a variety of methods and empirical formulas to estimate bearing capacity, ensuring that foundations are safe and efficient. Accurate computation is crucial as it directly influences the design and safety of structures such as buildings, bridges, and other infrastructures.

Penetration tests, such as the Standard Penetration Test (SPT) and the Cone Penetration Test (CPT) and the Dynamic Probing Test (DPT) are widely used in geotechnical investigations to assess soil properties. These tests provide valuable data on soil density, strength, and stratification by measuring the resistance of soil to penetration by a standard probe. The results from these tests help engineers to understand the subsurface conditions and to make informed decisions about foundation design and construction practices. Penetration tests are essential for obtaining accurate field data, which in turn enhances the reliability of bearing capacity calculations.

Recent advancements in artificial intelligence (AI) have introduced innovative approaches to predicting bearing capacity, often matching the accuracy of traditional methods while requiring significantly less data and resources. AI models, such as neural networks, trained on historical data to identify patterns and make predictions. These technologies can provide rapid and cost-effective solutions, particularly in scenarios where extensive field testing is impractical or budget constraints are significant. By leveraging AI, engineers can achieve reliable results with minimal data inputs, revolutionizing the field of geotechnical engineering and optimizing resource allocation. In this chapter, an AI model will be suggested to compute the bearing capacity of shallow foundation using only the density of the soil and the equivalent dynamic point resistance value .

IV.2 Bearing capacity computing

IV.2.1 Difinition

Bearing capacity computing refers to the process of calculating the maximum load (weight) that the soil can support without experiencing failure. This is a critical aspect of geotechnical engineering, ensuring the stability of foundations for structures like buildings, bridges, and retaining walls[45].

IV.2.2 Computing methods

There are various methods for calculating bearing capacity, each with its own complexity and accuracy. The choice of method depends on several factors, including the type of foundation being designed (shallow or deep), the soil properties at the site (cohesive or non-cohesive), and the desired level of precision.

IV.2.2.1 Terzaghi Equation

widely used method for calculating the ultimate bearing capacity of shallow foundations in cohesive soils (clay). It provides a preliminary estimate of the maximum load the soil can support without failure[46].

$$Qu = c' Nc + q' Nq + 0.5\gamma' B N\gamma$$

Qu⟶ Ultimate bearing capacity of the soil (force per unit area).

c' ⟶ Effective cohesion of the soil (strength that resists shear deformation).

Nc ⟶ Shape factor depending on the foundation geometry (e.g., strip footing, square footing).

q' ⟶ Effective overburden pressure (weight of the soil above the foundation level).

Nq ⟶ Bearing capacity factor considering the soil's angle of internal friction (resistance to deformation).

B ⟶ Width of the foundation.

γ' ⟶ Effective unit weight of the soil (weight of soil solids minus the buoyant force of water).

Nγ ⟶ Shape factor for the depth term.

IV.2.2.2 Computing the bearing capacity from DPT

The general bearing capacity equation according to the standard DTR BC 2.331 and the standard DTU 13.23 is giving by:

$$q_u = \frac{R_d}{\beta} \; ; 5 \leq \beta \leq 7$$

With:

$R_d \rightarrow$ Dynamic point resistance (bar)

$\beta \rightarrow$Lift coefficient between 5 and 7 (in general β=5)

And by calculating the q_u we could calculate the q_{adm} which it is giving by

$$q_{adm} = \frac{q_u - \gamma_1 D}{F} + \gamma_1 D \qquad F \geq 3$$

$\gamma_1 \rightarrow$ The soil density

$D \rightarrow$ The embedding depth

$F \rightarrow$ safety factor ≥3

IV.2.2.3 Meyerhof Equation

Is an advancement over the Terzaghi Equation for calculating the ultimate bearing capacity of shallow foundations. It considers a wider range of factors and provides a more refined estimate compared to the simpler Terzaghi approach[47].

IV.2.2.4 Vesić Equation

Is another method for calculating the ultimate bearing capacity of shallow foundations. It builds upon the concepts of Terzaghi and Meyerhof but offers a more comprehensive and versatile approach, particularly for complex scenarios[48].

IV.2.2.5 Geotechnical software

Geotechnical software computing methods refer to the use of specialized software programs to analyze the bearing capacity of foundations and other geotechnical engineering problems. These software tools offer powerful capabilities that go beyond the limitations of simpler hand calculations like the Terzaghi, Meyerhof, and Vesić equations[49].

IV.2.3 The importance of bearing capacity computing

a. Safe foundation design

By knowing the bearing capacity of the soil, engineers can design foundations strong enough to support the weight of the structure without causing[45]:

- Excessive settlement: Sinking of the foundation due to soil compression.
- Shear failure: Collapse of the soil due to exceeding its strength.

b. Cost optimization

Accurate bearing capacity calculations can help optimize foundation design. By knowing the soil's capacity, engineers can choose a foundation type and size that matches the load requirements, potentially reducing construction costs.

IV.2.4 The factors that affect the bearing capacity computing

IV.2.4.1 Soil Properties

a. Type

- **Sand**

Generally good bearing capacity when dense and dry. Loose sand or saturated sand with water flow can have lower capacity.

- **Clay**

Can have high bearing capacity when dry and stiff, but strength reduces significantly with moisture content.

- **Rock**

Offers the highest bearing capacity but can be expensive to excavate for foundations.

- **Density**

Denser soil particles are packed closer together, leading to increased friction and resistance to deformation, resulting in higher bearing capacity. Loose soil has lower capacity.

b. Shear Strength

This is the soil's internal resistance to deformation or shearing. Higher shear strength indicates a stronger soil and a higher bearing capacity. Clays have higher shear strength than sands when dry, but this can change with moisture content[50].

c. Moisture Content

Saturated soil (filled with water) can have lower bearing capacity because water reduces the friction between soil particles. Clay soils are particularly affected by moisture as they become softer and lose strength.

IV.2.4.2 Depth of Foundation

Generally, the deeper a foundation is placed, the higher the bearing capacity. This is because deeper soil layers experience greater confining pressure from the soil above them, making them more resistant to deformation. However, there's a point of diminishing returns, and very deep foundations might not be practical or cost-effective.

IV.2.4.3 Groundwater Conditions

The presence of groundwater can significantly reduce bearing capacity. Water fills voids between soil particles, reducing friction and internal strength. Additionally, groundwater can create uplift pressure on the foundation, potentially causing it to rise.

IV.2.4.4 Structure Size and Weight

The total load exerted by the structure on the soil and its distribution (even or uneven) needs to be considered. A heavier structure will require a foundation with a higher bearing capacity to avoid excessive settlement or failure. The distribution of weight is also important. Uneven weight distribution can cause the foundation to tilt or settle unevenly.

IV.3 dynamic penetration test

IV.3.1 definition

DPT is an in-situ test, meaning it measures the properties of the soil directly at its location within the ground. It's a fast and economical method for geotechnical investigations, providing valuable information about the subsurface conditions[51].

Figure IV. 1: dynamic penetration test machine.

IV.3.2 the process of dynamic penetration test

IV.3.2.1 Equipment

- DPT uses a multiple penetrometer with a metal cone tip.
- The penetrometer is connected to a drive rod and a hammer mechanism.

IV.3.2.2 Test Procedure

- The penetrometer is driven into the ground by repeatedly lifting and dropping a weight (typically 50kg or 63.5kg) from a specific height (usually 500mm or 750mm).
- The number of blows required for the cone to penetrate a certain depth (every 10cm/or 20cm) is recorded.
- The test continues until a predetermined depth is reached or a refusal criteria is met (exceeding a maximum number of blows).

IV.3.2.3 Data Analysis

The number of blows per depth interval is plotted on a graph, creating a DPT profile. This profile reflects the relative ease or difficulty of penetrating the soil layers, indicating variations in their strength and density[52].

IV.3.3 Advantages

- **Fast and cost-effective:** DPT can be performed quickly and at a lower cost compared to some other geotechnical tests.
- **Minimal equipment and manpower:** The test requires relatively simple equipment and minimal manpower, making it suitable for various field conditions.
- **Minimal ground disturbance:** DPT creates minimal disturbance to the ground compared to some drilling or excavation techniques.

IV.3.4 Limitations

- **Limited depth:** DPT is typically limited to investigating shallower depths (up to 15-20 meters) compared to some drilling methods.
- **Qualitative assessment:** DPT provides a qualitative assessment of soil strength based on penetration resistance. It may not provide direct correlations to absolute soil strength parameters without calibration with other tests.
- **Soil type limitations:** DPT might not be suitable for very soft or loose soils or encountering obstacles like cobbles or boulders.

IV.4 The suggested computer code

IV.4.1 Inputs layer

the algorithm has two different inputs, first we have the type of neurons

```
Network                    NeuronType                Hidden
1   <Serializable>
    16 références
2   Public Enum NeuronType
3       Hidden
4       Output
5   End Enum
```

and second, we have a set of data of soil properties Rd and the unit weight γ as inputs.

For the training process

```
Private Sub RandomizeSamples()
    If (Me.InvokeRequired) Then
        Me.Invoke(Sub() Me.RandomizeSamples())
    Else
        Static rnd As New Random(Me.GetHashCode)
        Dim a As New List(Of Double),
            b As New List(Of Double), r As New List(Of Double)
        Me.tbSampleSetA.Clear()
        Me.tbSampleSetB.Clear()
        Me.tbResult.Clear()

        a.Add(40)
        b.Add(1.5)
        r.Add(1.12)

        a.Add(41)
        b.Add(1.52)
        r.Add(1.15)

        a.Add(42)
        b.Add(1.54)
        r.Add(1.2)

        a.Add(45)
        b.Add(1.58)
        r.Add(1.3)

        a.Add(100)
        b.Add(1.7)
        r.Add(1.86)

        a.Add(125)
        b.Add(1.75)
        r.Add(1.99)

        a.Add(160)
        b.Add(1.77)
        r.Add(2.13)

        r.Add(1.2)

        a.Add(45)
        b.Add(1.58)
        r.Add(1.3)

        a.Add(100)
        b.Add(1.7)
        r.Add(1.86)

        a.Add(125)
        b.Add(1.75)
        r.Add(1.99)

        a.Add(160)
        b.Add(1.77)
        r.Add(2.13)

        a.Add(170)
        b.Add(1.79)
        r.Add(2.21)

        a.Add(184)
        b.Add(1.8)
        r.Add(2.45)

        a.Add(300)
        b.Add(2)
        r.Add(5.3)

        Me.tbSampleSetA.Text = String.Join(" ", a)
        Me.tbSampleSetB.Text = String.Join(" ", b)
        Me.tbResult.Text = String.Join(" ", r)
        Me.CreateCharts(r)
    End If
End Sub
```

For the query process

```
Private Sub RandomizeSamples2()
    If (Me.InvokeRequired) Then
        Me.Invoke(Sub() Me.RandomizeSamples2())
    Else

        Dim a As New List(Of Double),
            b As New List(Of Double), r As New List(Of Double)
        Me.tbSampleSetA.Clear()
        Me.tbSampleSetB.Clear()
        Me.tbResult.Clear()

        a.Add(50)
        b.Add(1.5)
        r.Add(1.12)

        a.Add(52)
        b.Add(1.52)
        r.Add(1.15)

        a.Add(60)
        b.Add(1.54)
        r.Add(1.2)

        a.Add(62)
        b.Add(1.58)
        r.Add(1.3)

        a.Add(80)
        b.Add(1.61)
        r.Add(1.86)

        a.Add(90)
        b.Add(1.65)
        r.Add(1.99)


        a.Add(95)
        b.Add(1.68)
        r.Add(2.13)

        a.Add(94)
        b.Add(1.79)
        r.Add(2.21)

        a.Add(115)
        b.Add(1.8)
        r.Add(2.45)

        a.Add(200)
        b.Add(2)
        r.Add(5.3)

        Me.tbSampleSetA.Text = String.Join(" ", a)
        Me.tbSampleSetB.Text = String.Join(" ", b)
        Me.tbResult.Text = ""
        Me.CreateCharts(r)
    End If
End Sub
```

IV.4.2 hidden layer (processing)

This is the core phase where the algorithm performs operations on the inputs using a series of computational steps. This phase is guided by logical and arithmetic calculations to process the data effectively.

- **Decision making:** the algorithm take the inputs (the values of Rd and) and start calculating the bearing capacity of soil.

```
Public Class Neuron
    2 références
    Public Property ID As String
    4 références
    Public Property Type As NeuronType
    14 références
    Public Property Inputs As Double()
    9 références
    Public Property Weights As Double()
    4 références
    Public Property BiasWeight As Double
    9 références
    Public Property [Error] As Double
    0 références
    Sub New()
        Me.ID = Guid.NewGuid().ToString
    End Sub
    3 références
    Sub New(Type As NeuronType)
        Me.Type = Type
        Me.Inputs = New Double(2) {}
        Me.Weights = New Double(2) {}
        Me.ID = Guid.NewGuid().ToString
    End Sub
    1 référence
    Public Sub Randomize()
        Me.Weights(0) = Me.Random.NextDouble()
        Me.Weights(1) = Me.Random.NextDouble()
        Me.BiasWeight = Me.Random.NextDouble()
    End Sub
    3 références
    Public Sub Cycle()
        Me.Weights(0) += Me.Error * Inputs(0)
        Me.Weights(1) += Me.Error * Inputs(1)
        Me.BiasWeight += Me.Error
    End Sub

    Public ReadOnly Property Output As Double
        Get
            Return Sigmoid.Output(Me.Weights(0) * Me.Inputs(0) + Me.Weights(1) * Me.Inputs(1) + Me.BiasWeight)
        End Get
    End Property
    3 références
    Public Function Random() As Random
        Static rnd As New Random(Me.GetHashCode)
        Return rnd
    End Function
    0 références
    Public Overrides Function ToString() As String
        Return String.Format("[{0}] Output: {1} Bias: {2} Error: {3}", Me.Type, Me.Output, Me.BiasWeight, Me.Error)
    End Function
End Class
```

- **Looping:** the algorithm displays every 100 operations the new output

```
Public Sub Train(x As Double(), y As Double(), Results As Double())
    Try
        Dim epoch As Integer = 0

        Dim NH1 As Neuron = Me.GetNeuron(NeuronType.Hidden, 0)
        Dim NH2 As Neuron = Me.GetNeuron(NeuronType.Hidden, 1)
        Dim NOUT As Neuron = Me.GetNeuron(NeuronType.Output)

        Me.Training = True
        RaiseEvent TrainingStarted()

        Do
            epoch += 1
            For i As Integer = 0 To Results.Length - 1
                NH1.Inputs = New Double() {x(i), y(i)}
                NH2.Inputs = New Double() {x(i), y(i)}
                NOUT.Inputs = New Double() {NH1.Output, NH2.Output}

                If (epoch Mod 100 = 0 And epoch <> 0) Then
                    RaiseEvent TrainingCycle(i, x(i), y(i), NOUT.Output)
                End If

                NOUT.Error = Sigmoid.F(NOUT.Output) * (Results(i) - NOUT.Output)
                NOUT.Cycle()

                NH1.Error = Sigmoid.F(NH1.Output) * NOUT.Error * NOUT.Weights(0)
                NH2.Error = Sigmoid.F(NH2.Output) * NOUT.Error * NOUT.Weights(1)

                NH1.Cycle()
                NH2.Cycle()
            Next
        Loop Until Not Me.Training
    Catch ex As Exception
        Debugger.Break()
    Finally
        RaiseEvent TrainingStopped()
    End Try
End Sub
```

IV.4.3 activation function

```
Public Class Sigmoid
    1 référence
    Public Shared Function Output(value As Double) As Double
        Return 1 / (1 + Math.Exp(-value))
    End Function
    3 références
    Public Shared Function F(value As Double) As Double
        Return value * (1 - value)
    End Function
End Class
```

IV.4.4 Output layer

After processing the inputs, the algorithm produces an output. This output is the result of the algorithm's operations.

```
Public Class Simple
    3 références
    Public Property Training As Boolean
    10 références
    Public Property Neurons As List(Of Neuron)
    <NonSerialized> Public Event TrainingStarted()
    <NonSerialized> Public Event TrainingStopped()
    <NonSerialized> Public Event TrainingCycle(index As Integer, x As Double, y As Double, output As Double)
    1 référence
    Sub New()
        Me.Neurons = New List(Of Neuron)
        Me.Reset()
    End Sub
    2 références
    Public Sub Reset()
        Me.Neurons.Clear()
        Me.Neurons.Add(New Neuron(NeuronType.Hidden))
        Me.Neurons.Add(New Neuron(NeuronType.Hidden))
        Me.Neurons.Add(New Neuron(NeuronType.Output))
        Me.Randomize()
    End Sub
    1 référence
    Public Sub Abort()
        Me.Training = False
    End Sub
```

```
    Public Sub Resolve(x As Double(), y As Double())
        If (x.Length = y.Length) Then
            Dim NH1 As Neuron = Me.GetNeuron(NeuronType.Hidden, 0)
            Dim NH2 As Neuron = Me.GetNeuron(NeuronType.Hidden, 1)
            Dim NOUT As Neuron = Me.GetNeuron(NeuronType.Output)
            For i As Integer = 0 To x.Length - 1
                NH1.Inputs = New Double() {x(i), y(i)}
                NH2.Inputs = New Double() {x(i), y(i)}
                NOUT.Inputs = New Double() {NH1.Output, NH2.Output}
                RaiseEvent TrainingCycle(i, x(i), y(i), NOUT.Output)
            Next
        End If
    End Sub
    0 références
    Public Function Resolve2(x As Double, y As Double) As Double

        Dim NH1 As Neuron = Me.GetNeuron(NeuronType.Hidden, 0)
        Dim NH2 As Neuron = Me.GetNeuron(NeuronType.Hidden, 1)
        Dim NOUT As Neuron = Me.GetNeuron(NeuronType.Output)

        NH1.Inputs = New Double() {x, y}
        MsgBox(NH1.Output)
        NH2.Inputs = New Double() {x, y}
        NOUT.Inputs = New Double() {NH1.Output, NH2.Output}
        Return NOUT.Output

    End Function
```

IV.4.4.1 End learning button

In this computer code, the user can end the training only by clicking the corresponding button when the outputs seems acceptable comparing to the reference data .

IV.5 The Graphical User Interface of the code

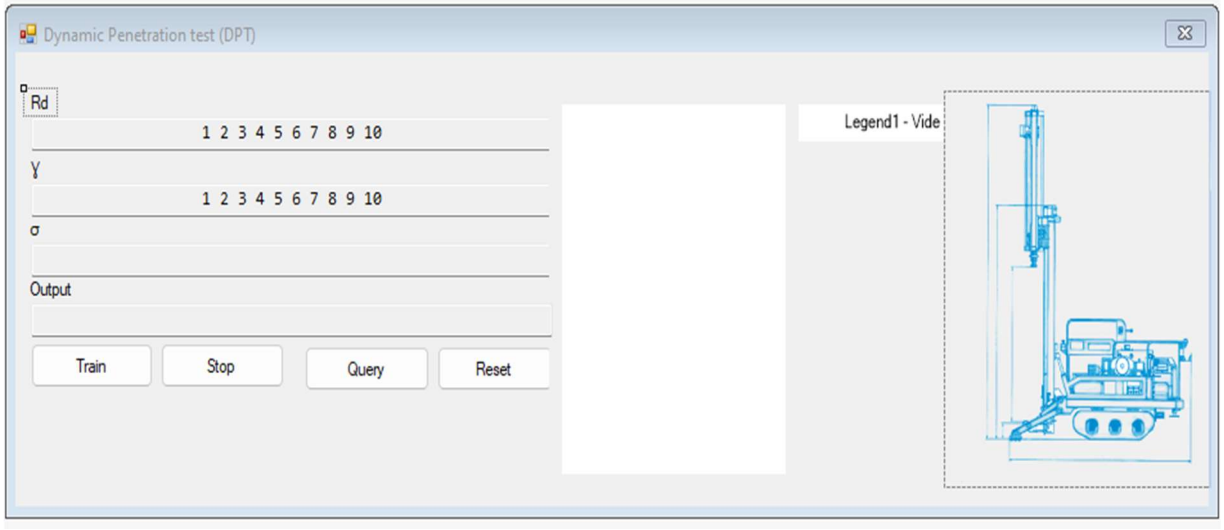

Figure IV. 2: GUI of the dynamic penetration test.

Graphical User Interfaces (GUIs) offer a user-friendly alternative. These interfaces rely on visual elements like icons, menus, and buttons, making it easy to interact with computer programs.

This windows form (WinForms) has been developed in order to represent the outputs and the inputs in a chart.

IV.5.1 Input Fields

- Rd represented by a label and a text box.
- γ the unit weight represented by a label and a text box.
- σ bearing capacity represented by a label and a text box.
- Penetrometer represented by a picture box.

IV.5.2 buttons

```
Public Class frmMain
    10 références
    Public Property Network As Simple
    6 références
    Public Property Buffer As List(Of Double)
    1 référence
    Public Property Multiplier As Integer = 100
    0 références
    Private Sub frmMain_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Me.Initialize()
        Me.btnQuery.PerformClick()
    End Sub
    0 références
    Private Sub btnTrain_Click(sender As Object, e As EventArgs) Handles btnTrain.Click
        Call New Threading.Thread(AddressOf Me.StartTraining) With {.IsBackground = True}.Start()
    End Sub
    0 références
    Private Sub btnAbort_Click(sender As Object, e As EventArgs) Handles btnAbort.Click
        Me.Network.Abort()
        Me.Network.SaveAs(".\Network.bin")
    End Sub
    0 références
    Private Sub btnReset_Click(sender As Object, e As EventArgs) Handles btnReset.Click
        Me.Network.Reset()
    End Sub
    0 références
    Private Sub btnQuery_Click(sender As Object, e As EventArgs) Handles btnQuery.Click
        Me.RandomizeSamples2()
        Me.Network.Resolve(Me.StringToDoubles(Me.tbSampleSetA.Text), Me.StringToDoubles(Me.tbSampleSetB.Text))
    End Sub
```

```
Private Sub Initialize()
    Me.Buffer = New List(Of Double)
    If (Not File.Exists(".\Network.bin")) Then
        Me.Network = New Simple
    Else
        Me.Network = File.ReadAllBytes(".\Network.bin").Deserialize(Of Simple)()
    End If
    AddHandler Network.TrainingStarted, AddressOf Me.TrainingStarted
    AddHandler Network.TrainingStopped, AddressOf Me.TrainingStopped
    AddHandler Network.TrainingCycle, AddressOf Me.TrainingCycle
End Sub
1 référence
Private Sub StartTraining()
    Me.RandomizeSamples()
    Me.Network.Train(Me.StringToDoubles(Me.tbSampleSetA.Text), Me.StringToDoubles(Me.tbSampleSetB.Text)
        , Me.StringToDoubles(Me.tbResult.Text))
End Sub
2 références
Private Sub TrainingStarted()
    If (Me.InvokeRequired) Then
        Me.Invoke(Sub() Me.TrainingStarted())
    Else
        Me.btnTrain.Enabled = False
        Me.btnQuery.Enabled = False
        Me.btnReset.Enabled = False
    End If
End Sub
```

```
Private Sub TrainingStopped()
    If (Me.InvokeRequired) Then
        Me.Invoke(Sub() Me.TrainingStopped())
    Else
        Me.btnTrain.Enabled = True
        Me.btnQuery.Enabled = True
        Me.btnReset.Enabled = True
    End If
End Sub
```

- "Train" button to initiate the training.
- "stop" button to stop the training.
- "Query" button to give a direct result.
- "Reset " button to reset the training.

IV.5.3. output:

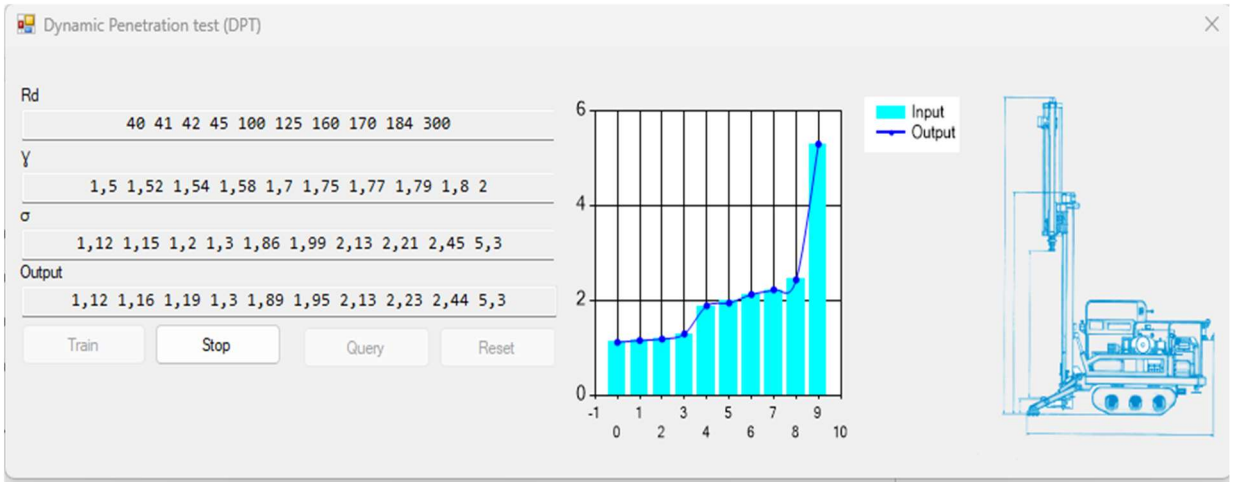

Figure IV. 3: GUI shows the outputs and the chart.

Calculated bearing capacity value displayed in the Output text box and draw a chart that shows the variation of the output and how it is getting closer to the real value.

Creating a chart

```
Private Sub CreateCharts(data As List(Of Double))
    Me.Buffer.Clear()
    Me.Chart.Series.Clear()
    Me.Chart.Series.Add("Input")
    Me.Chart.Series("Input").Color = Color.Cyan
    Me.Chart.Series("Input").XValueType = ChartValueType.Double
    Me.Chart.Series("Input").ChartType = SeriesChartType.Column
    Me.Chart.Series.Add("Output")
    Me.Chart.Series("Output").Color = Color.Blue
    Me.Chart.Series("Output").XValueType = ChartValueType.Double
    Me.Chart.Series("Output").ChartType = SeriesChartType.Spline
    Me.Chart.Series("Output").MarkerStyle = MarkerStyle.Circle
    Me.Chart.Series("Output").MarkerSize = 5
    Me.Chart.Series("Output").MarkerBorderWidth = 1
    For j As Integer = 0 To data.Count - 1
        Me.Chart.Series("Input").Points.AddXY(j, data(j))
        Me.Buffer.Add(0)
    Next
End Sub
```

Running the chart

```
Private Sub TrainingCycle(index As Integer, x As Double, y As Double, output As Double)
    If (Me.InvokeRequired) Then
        Me.Invoke(Sub() Me.TrainingCycle(index, x, y, output))
    Else
        Me.Buffer(index) = Math.Round(output * 100, 2)
        If (index = 9) Then
            Me.Chart.Series("Output").Points.Clear()
            For i As Integer = 0 To 9
                Me.Chart.Series("Output").Points.AddXY(i, Me.Buffer(i))
            Next
            Me.tbNetValues.Text = String.Join(" ", Me.Buffer.Select(Function(v) Math.Round(v, 2)))
            Me.Chart.Refresh()
        End If
    End If
End Sub
```

IV.6 Results

IV.6.1 the inputs

Here we are representing the inputs we use in in the code

Sigma $(\sigma)(bar)$	Rd(bar)	gamma(γ) (bar)
1,12	40	1,5
1,15	41	1,5
1,2	42	1,5
1,3	45	1,5
1,4	50	1,5
1,42	52	1,5
1,44	58	1,6
1,5	55	1,6
1,58	69	1,6
1,63	77	1,6
1,65	80	1,6
1,7	85	1,6
1,75	90	1,6
1,86	100	1,7
1,87	101	1,7
1,9	105	1,7
1,92	110	1,7
1,94	115	1,75
1,95	118	1,75
1,99	125	1,75
2	130	1,75
2,09	155	1,75
2,13	160	1,79
2,2	169	1,79
2,21	170	1,79

2,36	175	1,79
2,4	179	1,8
2,45	184	1,8
2,5	190	1,8
2,7	201	1,8
2,99	215	1,9
3	220	1,9
3,16	241	2
3,34	250	2
5,3	300	2
6	350	2

Table IV. 1: The inputs we use in the WinForms code.

IV.6.2 The Outputs variation

References (σ)(bar)	Outputs (1) (bar)	Outputs (2) (bar)	Outputs (3) (bar)	Outputs (4) (bar)
1.12	1.24	1.17	1.16	1.12
1.15	1.25	1.19	1.17	1.16
1.2	1.25	1.2	1.19	1.19
1.3	1.27	1.24	1.25	1.29
1.86	1.6	1.78	1.85	1.86
1.99	1.81	1.95	1.98	1.96
2.13	2.19	2.2	2.18	2.18
2.21	2.32	2.28	2.26	2.26
2.45	2.52	2.42	2.38	2.39
5.3	5.28	5.3	5.3	5.3

Table IV. 2: This table shows how the network's outputs change over time during training.

This table represent the variation of the outputs with the time

- Outputs (1) were taken 1 minute after the training start
- Output (2) was taken 2 minutes after the first output
- Output (3) was taken 3 minutes after the second output
- The last output was taken after 10 min

IV.7 Conclusion

AI is increasingly becoming an integral part of civil engineering, transforming traditional practices with its ability to provide accurate and reliable predictions. By utilizing machine learning algorithms and neural networks, AI can analyze minimal data to yield results that closely mirror those obtained from conventional methods. This technological advancement not only streamlines the engineering process but also reduces the need for extensive and costly field investigations. As AI continues to evolve, it promises to make the work of civil engineers more efficient and effective, enabling them to design safer and more reliable structures with fewer resources and at a lower cost. The integration of AI in civil engineering signifies a significant leap forward, paving the way for more innovative and resource-efficient engineering solutions.

General Conclusion

General Conclusion

In conclusion, this work highlights the pivotal role of artificial intelligence (AI) techniques and algorithms in revolutionizing civil engineering practices. By leveraging machine learning algorithms such as artificial neural networks, genetic algorithms, and particle swarm optimization, researchers have made significant strides in predicting critical parameters and optimizing construction processes. The implementation of AI in civil engineering, from health monitoring to concrete analysis and geotechnical applications, has enhanced safety, efficiency, and sustainability in infrastructure projects.

Looking towards the future, the continued evolution of AI promises to further streamline engineering processes, reduce costs, and improve the reliability of structures. However, challenges such as data availability and quality, interdisciplinary knowledge requirements, and ethical considerations must be addressed to fully realize the potential of AI in civil engineering.

Furthermore, the study sheds light on the top AI frameworks and libraries that play a crucial role in facilitating AI implementation in civil engineering. Frameworks like TensorFlow, PyTorch, and scikit-learn, along with libraries such as Keras and OpenCV, provide powerful tools for developing and deploying AI models in construction management and structural analysis. By harnessing these frameworks and libraries, civil engineers can unlock transformative benefits and drive innovation in the construction industry.

[1] S. J. Russell, P. Norvig, and E. Davis, *Artificial intelligence: a modern approach*, 3rd ed. in Prentice Hall series in artificial intelligence. Upper Saddle River: Prentice Hall, 2010.

[2] S. Bringsjord and N. S. Govindarajulu, "Artificial Intelligence," in *The Stanford Encyclopedia of Philosophy*, Summer 2024., E. N. Zalta and U. Nodelman, Eds., Metaphysics Research Lab, Stanford University, 2024. Accessed: Jun. 05, 2024. [Online]. Available: https://plato.stanford.edu/archives/sum2024/entries/artificial-intelligence/

[3] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: concepts, tools, and techniques to build intelligent systems*, Second edition. Beijing [China] ; Sebastopol, CA: O'Reilly Media, Inc, 2019.

[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[5] I. N. da Silva, D. H. Spatti, R. A. Flauzino, L. H. B. Liboni, and S. F. dos R. Alves, *Artificial Neural Networks: A Practical Course*. Springer, 2016.

[6] S. Agatonovic-Kustrin and R. Beresford, "Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research," *J. Pharm. Biomed. Anal.*, vol. 22, no. 5, pp. 717–727, Jun. 2000, doi: 10.1016/S0731-7085(99)00272-1.

[7] "What Is Deep Learning? Definition, Examples, and Careers," Coursera. Accessed: Jun. 05, 2024. [Online]. Available: https://www.coursera.org/articles/what-is-deep-learning

[8] A. De Mauro, M. Greco, and M. Grimaldi, "What is big data? A consensual definition and a review of key research topics," *AIP Conf. Proc.*, vol. 1644, no. 1, pp. 97–104, Feb. 2015, doi: 10.1063/1.4907823.

[9] M. Al-Mekhlal and A. Ali Khwaja, "A Synthesis of Big Data Definition and Characteristics," in *2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, Aug. 2019, pp. 314–322. doi: 10.1109/CSE/EUC.2019.00067.

[10] J. Han, M. Kamber, and J. Pei, *Data mining: concepts and techniques*, 3rd ed. in Morgan Kaufmann series in data management systems. Amsterdam Boston: Elsevier/Morgan Kaufmann, 2012.

[11] S. Forrest, "Genetic algorithms," *ACM Comput. Surv.*, vol. 28, no. 1, pp. 77–80, Mar. 1996, doi: 10.1145/234313.234350.

[12] S. N. Sivanandam and S. N. Deepa, "Genetic Algorithms," in *Introduction to Genetic Algorithms*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 15–37. doi: 10.1007/978-3-540-73190-0_2.

[13] O. Kramer, "Genetic Algorithms," in *Genetic Algorithm Essentials*, O. Kramer, Ed., Cham: Springer International Publishing, 2017, pp. 11–19. doi: 10.1007/978-3-319-52156-5_2.

[14] A. Chakraborty and A. K. Kar, "Swarm Intelligence: A Review of Algorithms," in *Nature-Inspired Computing and Optimization: Theory and Applications*, S. Patnaik, X.-S. Yang, and K. Nakamatsu, Eds., Cham: Springer International Publishing, 2017, pp. 475–494. doi: 10.1007/978-3-319-50920-4_19.

[15] D. Bratton and J. Kennedy, “Defining a Standard for Particle Swarm Optimization,” in *2007 IEEE Swarm Intelligence Symposium*, Apr. 2007, pp. 120–127. doi: 10.1109/SIS.2007.368035.

[16] H. Abdallah, H. M. Emara, H. T. Dorrah, and A. Bahgat, “Using Ant Colony Optimization algorithm for solving project management problems,” *Expert Syst. Appl.*, vol. 36, no. 6, pp. 10004–10015, Aug. 2009, doi: 10.1016/j.eswa.2008.12.064.

[17] D. Karaboga, “Artificial bee colony algorithm,” *Scholarpedia*, vol. 5, no. 3, p. 6915, Mar. 2010, doi: 10.4249/scholarpedia.6915.

[18] Y. Hou, H. Gao, Z. Wang, and C. Du, “Improved Grey Wolf Optimization Algorithm and Application,” *Sensors*, vol. 22, no. 10, Art. no. 10, Jan. 2022, doi: 10.3390/s22103810.

[19] J. S. Rhodes, A. Cutler, and K. R. Moon, “Geometry- and Accuracy-Preserving Random Forest Proximities,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 9, pp. 10947–10959, Sep. 2023, doi: 10.1109/TPAMI.2023.3263774.

[20] A. Ibrahem Ahmed Osman, A. Najah Ahmed, M. F. Chow, Y. Feng Huang, and A. El-Shafie, “Extreme gradient boosting (Xgboost) model to predict the groundwater levels in Selangor Malaysia,” *Ain Shams Eng. J.*, vol. 12, no. 2, pp. 1545–1556, Jun. 2021, doi: 10.1016/j.asej.2020.11.011.

[21] Y. Huang, J. Li, and J. Fu, “Review on Application of Artificial Intelligence in Civil Engineering,” *Comput. Model. Eng. Sci.*, vol. 121, no. 3, pp. 845–875, 2019, doi: 10.32604/cmes.2019.07653.

[22] I. H. El-adaway, G. Ali, R. Assaad, A. Elsayegh, and I. S. Abotaleb, “Analytic Overview of Citation Metrics in the Civil Engineering Domain with Focus on Construction Engineering and Management Specialty Area and Its Subdisciplines,” *J. Constr. Eng. Manag.*, vol. 145, no. 10, p. 04019060, Oct. 2019, doi: 10.1061/(ASCE)CO.1943-7862.0001705.

[23] T. A. Pham, H.-B. Ly, V. Q. Tran, L. V. Giap, H.-L. T. Vu, and H.-A. T. Duong, “Prediction of Pile Axial Bearing Capacity Using Artificial Neural Network and Random Forest,” *Appl. Sci.*, vol. 10, no. 5, p. 1871, Mar. 2020, doi: 10.3390/app10051871.

[24] L. S. Iyer, “AI enabled applications towards intelligent transportation,” *Transp. Eng.*, vol. 5, p. 100083, Sep. 2021, doi: 10.1016/j.treng.2021.100083.

[25] K. Yetilmezsoy, B. Ozkaya, and M. Çakmakci, “ARTIFICIAL INTELLIGENCE-BASED PREDICTION MODELS FOR ENVIRONMENTAL ENGINEERING,” *NEURAL Netw. WORLD*, vol. 21, no. 3, 2011, doi: 10.14311/nnw.2011.21.012.

[26] R. Oulebsir, A. Lefkir, A. Safri, and A. Bermad, “Optimization of the energy consumption in activated sludge process using deep learning selective modeling,” *Biomass Bioenergy*, vol. 132, p. 105420, Jan. 2020, doi: 10.1016/j.biombioe.2019.105420.

[27] C. E. Braun, L. D. Chiwiacowsky, and A. T. Gómez, “Variations of Ant Colony Optimization for the Solution of the Structural Damage Identification Problem,” *Procedia Comput. Sci.*, vol. 51, pp. 875–884, 2015, doi: 10.1016/j.procs.2015.05.218.

[28] A. Arbaoui, A. Ouahabi, S. Jacques, and M. Hamiane, “Wavelet-based multiresolution analysis coupled with deep learning to efficiently monitor cracks in concrete,” *Frat. Ed Integrità Strutt.*, vol. 15, no. 58, pp. 33–47, Sep. 2021, doi: 10.3221/IGF-ESIS.58.03.

[29] N. Rane, “Integrating Building Information Modelling (BIM) and Artificial Intelligence (AI) for Smart Construction Schedule, Cost, Quality, and Safety Management: Challenges and Opportunities,” *SSRN Electron. J.*, 2023, doi: 10.2139/ssrn.4616055.

[30] H. Moayedi, A. Moatamediyan, H. Nguyen, X.-N. Bui, D. T. Bui, and A. S. A. Rashid, “Prediction of ultimate bearing capacity through various novel evolutionary and neural network models,” *Eng. Comput.*, vol. 36, no. 2, pp. 671–687, Apr. 2020, doi: 10.1007/s00366-019-00723-2.

[31] R. Dzhusupova, J. Bosch, and H. H. Olsson, “Challenges in developing and deploying AI in the engineering, procurement and construction industry,” in *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*, Jun. 2022, pp. 1070–1075. doi: 10.1109/COMPSAC54236.2022.00167.

[32] H. Baker, M. R. Hallowell, and A. J.-P. Tixier, “AI-based prediction of independent construction safety outcomes from universal attributes,” *Autom. Constr.*, vol. 118, p. 103146, Oct. 2020, doi: 10.1016/j.autcon.2020.103146.

[33] “An Introduction to Ethics in Robotics and AI.” Accessed: Jun. 22, 2024. [Online]. Available: https://library.oapen.org/handle/20.500.12657/41303

[34] G. Canal *et al.*, “Building Trust in Human-Machine Partnerships,” *Comput. Law Secur. Rev.*, vol. 39, p. 105489, Nov. 2020, doi: 10.1016/j.clsr.2020.105489.

[35] A. Barredo Arrieta *et al.*, “Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI,” *Inf. Fusion*, vol. 58, pp. 82–115, Jun. 2020, doi: 10.1016/j.inffus.2019.12.012.

[36] V. Liermann, “Overview Machine Learning and Deep Learning Frameworks,” in *The Digital Journey of Banking and Insurance, Volume III: Data Storage, Data Processing and Data Analysis*, V. Liermann and C. Stegmann, Eds., Cham: Springer International Publishing, 2021, pp. 187–224. doi: 10.1007/978-3-030-78821-6_12.

[37] D. Rothman, *Transformers for natural language processing: build, train, and fine-tune deep neural network architectures for NLP with Python, Hugging Face, and OpenAI´s GPT3, ChatGPT, and GPT-4*, Second edition. in Expert Insight. Birmingham Mumbai: Packt, 2022.

[38] Z. Wang, K. Liu, J. Li, Y. Zhu, and Y. Zhang, “Various Frameworks and Libraries of Machine Learning and Deep Learning: A Survey,” *Arch. Comput. Methods Eng.*, vol. 31, no. 1, pp. 1–24, Jan. 2024, doi: 10.1007/s11831-018-09312-w.

[39] Z. Wang, K. Liu, J. Li, Y. Zhu, and Y. Zhang, “Various Frameworks and Libraries of Machine Learning and Deep Learning: A Survey,” *Arch. Comput. Methods Eng.*, vol. 31, no. 1, pp. 1–24, Jan. 2024, doi: 10.1007/s11831-018-09312-w.

[40] R. Elshawi, A. Wahab, A. Barnawi, and S. Sakr, “DLBench: a comprehensive experimental evaluation of deep learning frameworks,” *Clust. Comput.*, vol. 24, no. 3, pp. 2017–2038, Sep. 2021, doi: 10.1007/s10586-021-03240-4.

[41] J. Arroyo, C. Manna, F. Spiessens, and L. Helsen, “An Open-AI gym environment for the Building Optimization Testing (BOPTEST) framework,” presented at the Building Simulation 2021, in Building Simulation, vol. 17. IBPSA, 2021, pp. 175–182. doi: 10.26868/25222708.2021.30380.

[42] P. Gawłowicz and A. Zubow, “ns-3 meets OpenAI Gym: The Playground for Machine Learning in Networking Research,” in *Proceedings of the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, in MSWIM ’19. New York, NY, USA: Association for Computing Machinery, Nov. 2019, pp. 113–120. doi: 10.1145/3345768.3355908.

[43] A. T. Dang, R. Qaddoura, A. M. Al-Zoubi, H. Faris, and P. A. Castillo, “EvoCC: An Open-Source Classification-Based Nature-Inspired Optimization Clustering Framework in Python,” in *Applications of Evolutionary Computation*, J. L. Jiménez Laredo, J. I. Hidalgo, and K. O. Babaagba, Eds., Cham: Springer International Publishing, 2022, pp. 77–92. doi: 10.1007/978-3-031-02462-7_6.

[44] S. Yu, “The study of integrated frameworks for library and digital archives,” *Electron. Libr.*, vol. 24, no. 5, pp. 608–618, Jan. 2006, doi: 10.1108/02640470610707222.

[45] A. I. Lawal and S. Kwon, “Development of mathematically motivated hybrid soft computing models for improved predictions of ultimate bearing capacity of shallow foundations,” *J. Rock Mech. Geotech. Eng.*, vol. 15, no. 3, pp. 747–759, Mar. 2023, doi: 10.1016/j.jrmge.2022.04.005.

[46] A. S. Kumbhojkar, “Numerical Evaluation of Terzaghi’s Nγ,” *J. Geotech. Eng.*, vol. 119, no. 3, pp. 598–607, Mar. 1993, doi: 10.1061/(ASCE)0733-9410(1993)119:3(598).

[47] S. Van Baars, “Numerical check of the Meyerhof bearing capacity equation for shallow foundations,” *Innov. Infrastruct. Solut.*, vol. 3, no. 1, p. 9, Nov. 2017, doi: 10.1007/s41062-017-0116-1.

[48] M. S. Stanković, M. L. Zlatanović, and N. O. Vesić, “Basic equations of G-almost geodesic mappings of the second type, which have the property of reciprocity,” *Czechoslov. Math. J.*, vol. 65, no. 3, pp. 787–799, Sep. 2015, doi: 10.1007/s10587-015-0208-z.

[49] K.-K. Phoon *et al.*, “Geotechnical uncertainty, modeling, and decision making,” *Soils Found.*, vol. 62, no. 5, p. 101189, Oct. 2022, doi: 10.1016/j.sandf.2022.101189.

[50] S. Adarsh, R. Dhanya, G. Krishna, R. Merlin, and J. Tina, “Prediction of Ultimate Bearing Capacity of Cohesionless Soils Using Soft Computing Techniques,” *Int. Sch. Res. Not.*, vol. 2012, no. 1, p. 628496, 2012, doi: 10.5402/2012/628496.

[51] J. A. Howie, C. R. Daniel, R. S. Jackson, B. Walker, and A. Sy, “Comparison of energy measurement methods in the standard penetration test : final report, appendices I, II, III, and IV.” Accessed: Jun. 22, 2024. [Online]. Available: https://open.library.ubc.ca/soa/cIRcle/collections/facultyresearchandpublications/52383/items/1.0048557

[52] K.-J. Melzer and U. Smoltczyk, “Dynamic penetration testing State-of-the-art report,” in *Penetration Testing, volume 1*, Routledge, 1982.

[53] https://www.datacamp.com/blog/top-ai-frameworks-and-libraries