PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

Ministry of Higher Education and Scientific Research

MOHAMED KHIDER UNIVERSITY - BISKRA

Faculty of Exact Sciences, Natural Sciences and Life

Department of Computer Science

Order N°: .....RTIC/M_2/2021

Memory Thesis

Presented to obtain the academic master's degree in Computer Science

Option: Networks and Technologies of Information and Telecommunications

IoT based solution for the prediction of Civil Engineering building

Presented by: MERIZIG Imane.

Defended on July 2021, in front of the jury composed of:

		President/ chairwoman
Pr. Sadek Labib terrissa	**Professor**	**Advisor**
		Examiner

2020/2021

Acknowledgements

Firstly, thank you, God, for guiding me through this path, giving me such wonderful family, friends, and mentors to complete this dissertation.

This dissertation could not have been possible without the blessings and support of many people in my life. I sincerely appreciate and gratefully acknowledge their help and backing. I would like to express an appreciation particularity to the following.

I would like to express my sincere gratitude to my supervisor Pr.Sadek Labib TERRISSA for the continuous support of my dissertation, for his patience, motivation, and his continuous positive energy from the very first day. His guidance and observations always helped me throughout the research. My cordial thanks extend to my dissertation committee members who accepted to evaluate this modest work.

I sincerely thank Ikram REMADNA for her advises and help during the development of this project.

My deepest gratitude goes to my family members who always encouraged me to do my best. Their assistance, along with their continuous motivation kept me going. I am forever thankful to my parents for being such understanding and supportive figures in my life. I can never thank them enough for their endless support and encouragement especially in difficult times.

Imane MERIZIG.

Abstract

Internet of things (IoT), Cloud computing (CC), Edge computing (EC) and machine learning (ML) plays an important role in Smart Civil Engineering (SCE).

Internet of things (IoT) is the branch of Information technology which generates a connectivity link between "Internet" and actual physical "Things", cloud and edge computing are a new computing paradigms, their purpose is to offering access to services, save and monitor transmitted data from IoT in any place and at any time, and machine learning is a branch in artificial intelligent (AI) which aim to predict and diagnosis the results.

The main objective of our system is IoT based for the prediction of Civil Engineering buildings, using Raspberry Pi and sensors to monitor buildings of civil engineering, then, we send data to Edge computing utilize the protocol IoT mosquitto to pre-processing and filtering data. After these, we save the send data from edge computing in cloud computing to do the prediction and diagnosis results, using the best models regression of machine learning Multi-Layer Perceptron (MLP) Regression and Linear regression (LR), which have the lowest value of root mean squared error (RMSE).

Keywords: IoT, Cloud computing, Edge computing, machine learning, smart civil engineering, mosquitto, MLP, Linear Regression, RMSE.

Résumé

Internet des objets (IDO), informatique de nuage, informatique périphérique et l'apprentissage automatique jouent un rôle important dans le génie civil intelligent (GSI).

Internet des objets (IDO) est la branche des technologies de l'information qui génère un lien de connectivité entre " Internet" et les " objets" physiques réels, l'informatique de nuage et l'informatique en périphérie sont des nouveaux paradigmes informatiques, leurs objectifs est d'offrir un accès aux services, de sauvegarder et de surveiller les données transmises par l'IOT en tout lieu et à tout moment. L'apprentissage automatique est une branche de l'intelligence artificielle (IA) qui vise à prédire et à diagnostiquer les résultats.

L'objectif principal de notre système est basé sur l'internet des objets pour la prédiction des structures de Génie Civil, en utilisant Raspberry Pi et des capteurs pour surveiller les structures de génie civil, puis, nous envoyons des données à l'informatique en périphérie par le protocole IOT ‘‘mosquitto’’ pour le prétraitement et le filtrage des données. Après cela, nous sauvegardons les données envoyées par informatique en périphérie dans l'informatique de nuage pour faire la prédiction et les résultats de diagnostic, en utilisant les meilleurs modèles de régression de l'apprentissage automatique Perceptron Multicouche Régression (PMR) et régression linéaire (RL), qui ont la plus faible valeur de la racine de l'erreur quadratique moyenne (REQM).

Mots-clés: IDO, informatique de nuage, informatique périphérique, apprentissage automatique, génie civil intelligent, mosquitto, PMR, régression linéaire, REQM.

ملخص

تلعب إنترنت الأشياء، الحوسبة السحابية، الحوسبة الإلكترونية والتعلم الالي دوراً مهماً في الهندسة المدنية الذكية.

إنترنت الأشياء هي أحد فروع تكنولوجيا المعلومات الذي يولد صلة اتصال بين " الإنترنت" و " الأشياء" المادية الفعلية، الحوسبة السحابية وحوسبة الحافة هما نموذجان جديدان للحوسبة، الهدف منهما هو إتاحة إمكانية الحصول على الخدمات، توفير ورصد البيانات المنقولة من طبقة انترنت الأشياء في أي مكان وفي أي وقت، والتعلم الآلي فرع في الذكاء الاصطناعي يهدف إلى التنبؤ بالنتائج وتشخيصها.

الهدف الرئيسي من نظامنا هو حل قائم على أساس التنبؤات بمباني الهندسة المدنية، باستخدام راسبيري باي وأجهزة الاستشعار لرصد مباني الهندسة المدنية، ثم نرسل البيانات إلى الحوسبة الالكترونية لتجهيزها وتصفيتها مسبقا باستخدام بروتكول انترنت الأشياء موسكيتو. بعد هذا، نحفظ البيانات المرسلة من حوسبة الحافة في الحوسبة السحابية للقيام بالتنبؤ وتشخيص النتائج، وذلك باستخدام أفضل نماذج الانحدار للتعلم الآلي الانحدار المتعدد الطبقات والانحدار الخطي، اللذين لديهما أدنى قيمة لخطأ المتوسط الجذري.

الكلمات المفتاحية: انترنت الأشياء، الحوسبة السحابية، حوسبة الحافة، التعلم الآلي، الهندسة المدنية الذكية، موسكيتو، الانحدار الخطي، الجذر.

Contents

List of figures

List of tables

Acronyms

IoT	Internet of Things.
SCE	Smart civil engineering.
SHM	structural health monitoring
MQTT	Message Queuing Telemetry Transport.
EC	Edge Computing.
ML	Machine Learning.
GBR	Gradient Boosting Regressor.
KNN	K-nearest neighbor.
MLPRegressor	Multi-layer perceptron regressor.
LR	Linear regression.
Relu	Rectified Linear Unit.
DB4S	DB Browser for SQLite.
AI	Artificial Intelligence.
RMSE	Root Mean Square Error.
MAE	Mean Absolute Error.
R^2	R-Squared.

Chapter 01: Basic concepts

Introduction

With the development of technologies, many new materials find their application in civil engineering to deal with the deteriorating infrastructure. Smart civil engineering is a promising example that deserves a wide focus, from research to application. To apply smart civil engineering we need to achieve maximum building efficiency, sustainability, architectural quality. Using computing paradigms.

In the last decade, we have witnessed a significant evolution of computing paradigms.

The most known, and consolidated one, is Cloud Computing, a paradigm born from the necessity of using computing as a utility, thus allowing easy development of new Internet services.

On the other hand, Internet of Things (IoT) is becoming widespread, due to drawbacks of Cloud Computing the emerging IoT brings in many new challenges that Cloud Computing has hard time to meet.

In this light, rather Edge computing interacting with Cloud and IoT, to tackling most of new challenges that Cloud Computing alone cannot address, such as bandwidth, latency, and connectivity.

In this chapter, we aim at explain the concept of smart civil engineering and the major modern computing paradigms used in this work, namely Cloud Computing, IoT and Edge computing.

1. Smart civil engineering (SCE)

Smart civil engineering (or SCE) the most challenging essential to the performance enhancement, operating efficiency, safety, and integrity of structural systems, innovation that pushes modern structural engineers to face problems and techniques that will most likely produce a deep change, especially for those located in regions prone natural hazardous events such as strong earthquakes, hurricanes, and tsunamis.

In this section we illustrate the definition of smart civil engineering, the reason for emergence, definition of structural health monitoring (SHM), Features and applications.

1.1. Definition of smart civil engineering

Smart civil engineering is a thinking buildings, perfectly integrated into its environment, connected to the global network, and able to receive, process, and take advantage of data and information to communicate with its users and share its infrastructure with that of the city and the transport systems, acting as founding element in the achievement of the future smart civil engineering. To apply this concept, should be connect buildings with internet using technology internet of things (IoT) to collect, analyze data and structural health monitoring (SHM) should be long-term and cost-effective as well. [1]

1.2.The reason for the emergence of smart civil engineering

Civil engineering is the professional practice of designing and developing infrastructure projects. The great thing about civil engineering in the real world is that it's an industry that perpetually adapts to the demands and desires of society. In recent years civil engineering has adapted to consider many contemporary environmental concerns, helping to address issues of pollution and scarce water supplies, while considering the potential of sustainable energy within the industry. But the ever-increasing world population and growth of urban environments has resulted in an unprecedented number of structures and infrastructure systems being built in regions that are exposed to natural hazards such as earthquakes, hurricanes, wind storms, flooding and fire. This led to a large increase in the number of material and human losses. Due of lack the sensing system for structural health monitoring of civil engineering. So smart civil engineering appeared to improve the future performance of civil engineering via: [2]

- System systematically reports on the condition of the structure by automatically;
- Making engineering-based judgments, recording history of past patterns and intensities, and providing early warning for excessive conditions or for impending failure without requiring human intervention;
- make the system capable of providing and facilitating self-diagnostic, real-time continuous sensing, advanced remote sensing, self-organizing, self-identification, or self-adaptation (decision making and alarm triggering) function";
- Enable smart (structural health monitoring) SHM structures to be capable of real-time continuous sensing of both external and internal condition changes and responding to these changes to improve performance without human intervention;

- Evaluation and improvement of new and existing highway bridges for structural safety and remaining life;
- Design and development of ultrahigh performance concrete and high performance fiber reinforced cementations composites for sustainable infrastructure;
- Design and validation of smart civil engineering technologies for the structural health monitoring of civil structure and infrastructure systems; [3]

1.3. Definition of structural health monitoring (SHM)

Structural health monitoring is the process of implementing a damage identification strategy for civil and mechanical engineering infrastructure is referred to as structural health monitoring (SHM). This process involves the observation of a structure or mechanical system over time using periodically spaced measurements, the extraction of damage-sensitive features from these measurements and the statistical analysis of these features to determine the current state of system health. For long-term SHM, the output of this process is periodically updated information regarding the ability of the structure to continue to perform its intended function in light of the inevitable aging and damage accumulation resulting from the operational environments. Under an extreme event, such as an earthquake or unanticipated blast loading, SHM is used for rapid condition screening. This screening is intended to provide, in near real-time, reliable information about system performance during such extreme events and the subsequent integrity of the system. [4]

1.4. Features of smart civil engineering

The main features of smart civil engineering as below:

1.4.1. Systems are connected

The most fundamental feature of a smart civil engineering is the systems within it are linked. So, buildings, bridges, tunnel etc. are all connected. This is what makes a civil engineering "smart" – the ability of the systems within it to talk to one another.

1.4.2. The use of sensors

Sensors are an integral part of smart civil engineering and play an important role in collecting data to inform decisions about where to allocate resources.

1.4.3. Automation

Information is gathered and analyzed by the systems that have been put in place in a smart civil engineering – importantly, this is done constantly and in real time. This ongoing monitoring allows for automated adjustments that can control conditions across an entire civil engineering.

1.4.4. Data

Smart civil engineering generate a large volume of valuable data about their own use, which is something that regular civil engineering simply don't do. [5]

1.5. Applications of Smart Civil Engineering

Smart civil engineering play a vital role as far as the safety requirements are concerned in the design of various civil engineering infrastructures.

Smart civil engineering provide numerous applications, some of them are illustrated in TABLE 1.1: [6]

Applications	**Description**
Smart Concrete	Smart concrete has high potential and enhanced strength. Smart concrete can be prepared by adding carbon fibers for use in electro-magnetic shielding and for enhanced electrical conductivity of concrete. Smart concrete under loading and unloading process will loose and regain its conductivity, thus serving as a structural material as well as a sensor. Smart concrete plays a vital role in the construction of road pavements as a traffic-sensing recorder, and also melts ice on highways and air fields during snowfall in winter season by passing low voltage current through it.
Smart Buildings	A smart building is an intelligent space that will transform efficiency, comfort, and safety for people and assets. Smart buildings are those that incorporate sensors and intelligent systems to control building operations and facilities.

	Smart Buildings with intelligent solutions enable the high-performance workplace e-business strategies. Therefore, a smart building requires sensors to detect and monitor the number, presence and flow of people for a number of different requirements. Smart buildings contain a high level of electronic microprocessor based control systems that operate a wide range of services such as lighting, heat, ventilating and air conditioning, power, vertical transportation, fire and life safety, and security.
Smart Bridges	The use of smart materials permit the construction of smart bridges with a wider span to avoid the increased susceptibility to vibrations caused by ambient factors such as wind, rain or traffic. The critical deterioration of transportation infrastructure has put pressures on designers to think for new methods of rehabilitation and repair of bridges across the World. Thus, there has been a new trend in civil engineering called "smart structures", incorporating sensors in some of the most advanced building materials. The application of smart materials for bridges provides the following benefits: Less maintenance, real time monitoring of the response of the structure, monitoring the performance of the new advanced composite materials.

TABLE 1.1. Applications of Smart Civil Engineering.

2. Technologies used

In this work we use many technologies; we provide fundamentals about these technologies: Cloud Computing, internet of things and Edge computing.

2.1.Cloud Computing

Nowadays, Cloud Computing is being used by a large number of organizations. Many consider it a major development of the decade in computing. In this section, we submitted the fundamentals about Cloud Computing such as definition, characteristics and types.

2.1.1. Definitions of Cloud Computing

There are many definitions of Cloud Computing; we defined the most definitions;

Cloud Computing is the practice of leveraging a network of remote servers through the Internet to store, manage, and process data, instead of managing the data on a local server or computer. Since you are able to access the cloud on-demand, Cloud Computing allows for flexible availability of resources, including data storage and computing power in data centers more efficiently than solely relying on your own data center. [7]

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Cloud Computing deals with computation, software, data access and storage services that may not require end-user knowledge of the physical location and the configuration of the system that is delivering the services. Cloud Computing is a recent trend in IT that moves computing and data away from desktop and portable PCs into large data centers. [8]

2.1.2. Characteristics of Cloud Computing

There are five major characteristics of a Cloud Computing. Are below:

- **On-demand self-service**

Computing capabilities can be provided automatically when needed, without requiring any human interaction between consumer and service provider;

- **Broad network access**

Computing capabilities are available over the network and accessible through several mechanisms disposable for a wide range of client platforms (e.g., workstations, laptops, and mobile devices);

- **Resource pooling**

Computing resources are pooled to accommodate multiple consumers, dynamically allocating and deal locating them according to consumer demand. In addition, the provider resources are location independent, i.e. the consumer does not have any knowledge or control of their exact location;

- **Rapid elasticity**

Computing capabilities can flexibly be provided and released to scale in and out according to demand. Thus, the consumer has the perception of unlimited, and always adequate, computing capabilities;

- **Measured service**

Resource usage can be monitored and reported according to the type of service offered. This is particularly relevant in charge-per-use, or pay-per-user, services because it grants great transparency between the provider and the consumer of the service. [9]

2.1.3. Types of Cloud Computing

There are also several types of clouds. They are differentiated based on the way they are deployed and access provided to them. We list below four types of clouds.

2.1.3.1.Public Cloud

The computing infrastructure maintained by a provider is available to anyone. It is located usually in the provider's premises and controlled by the provider. As a consequence the infrastructure is shared simultaneously by many customers. There are both free and paid public clouds. For example, Google provides free storage space, office software, and email on its public cloud. Amazon's EC2 is a paid public cloud.

2.1.3.2.Private Cloud

The computing infrastructure is available for the exclusive use of a single organization. It may be a physically distributed set of interconnected computing systems belonging to the organization accessible to its members from anywhere, for example, from the far-flung branches of the organization. The infrastructure itself may be owned and maintained by the organization, outsourced to a third party, or it may be a combination of both. For example, a large bank with several branches may inter-connect their computing facilities to form a private cloud or request an infrastructure provider to design and maintain a cloud service for their exclusive use.

2.1.3.3.Community Cloud

The computing infrastructure is available for the exclusive use of a specific community of users with shared interests. For example, a group of universities may decide to cooperate and interconnect their computing infrastructure and create a community cloud which may be accessed by any of its members. The infrastructure of a community cloud may be owned and operated by each participating institution in-house or may be outsourced. The forerunner of the community cloud was called grid computing. A comparison of grid computing and Cloud Computing may be found in.

2.1.3.4.Hybrid Cloud

The computing infrastructure is a combination of two or more distinct entities, namely, private cloud, public cloud or community cloud. Each entity remains distinct but they are bound together by standardized protocols that permit data and application portability. For example, an organization may decide to keep part of its applications which it considers sensitive in its private cloud and execute other less sensitive applications on a public cloud. [10]

2.2.Internet of things (IoT)

Over the past decade, Cloud computing has been the predominant paradigm. According to this trend, computing, control, and data storage have been centralized and moved into the Cloud. On the other hand, Internet of Things (IoT) is now becoming widespread. The emerging IoT brings in many new challenges that Cloud Computing has a hard time to meet.

In this section, we provide fundamentals about Internet of Things such as definition, characteristics, architecture, application, components, protocol and applications. [9]

2.2.1. Definition of Internet of things (IoT)

IoT is a communication between devices. It enables data collection, storage and exchange, the devices can make decisions based on the data and perform tasks with very minimal human intervention. Unified infrastructure and AI-powered software used in conjunction with solutions from key technology enable facility owners to successfully deploy and exploit IoT solutions. The richer the set of available data and context, the greater the opportunities to boost efficiency, reliability, safety, and security. IoT relies on IP network to communicate the device data to the cloud or middleware platform.

2.2.2. Characteristics of IoT

The main features of Internet of Things are below:

- **Interconnectivity**

Everything in IoT can be interconnected with the global communication and information infrastructure;

- **Things-related services**

IoT is able to provide thing-related services within the constraints of things, such as privacy protection and semantic consistency between physical and virtual things;

- **Heterogeneity**

The devices in IoT can be based on different networks and/or hardware platforms. Moreover, they can interact with different service platforms and/or devices through different networks;

- **Constrained resources**

IoT usually involves devices characterized by energetic and computational constraints;

- **Dynamic changes and uncontrolled environment**

In IoT, the devices state (e.g., sleeping/awake, connected/disconnected) and context (e.g., location, speed) change dynamically. Therefore, IoT devices are part of an uncontrolled environment which is characterized by unstable surroundings and in which interactions among devices are unreliable due to both unstable network connectivity and device state dynamic changes. In addition, the number of devices can dynamically change;

- **Huge scale**

The number of devices that have to be managed and that have to communicate with each other is huge and it will be even more in the future. Moreover, the ratio of communications triggered by devices will steadily grow to the detriment of human-triggered communications. Even more critical will be the management and interpretation of data generated by such devices with the aim of sharing information with each other. [9]

2.2.3. Architecture of IoT

There's no single standard reference architecture for IoT as it encompasses a variety of technologies. This means that there's not one easy blueprint that can be followed for all possible implementations.

IoT architecture can actually vary significantly depending on the implementation; it needs to be open enough with open protocols so that it can support multiple network applications.

Even though there's no single IoT architecture that's universally agreed upon, the most basic and widely accepted format is a three-layer architecture. It was first introduced when the earliest research into the Internet of Things was being carried out. It proposes three layers: Perception, Network, and Application.

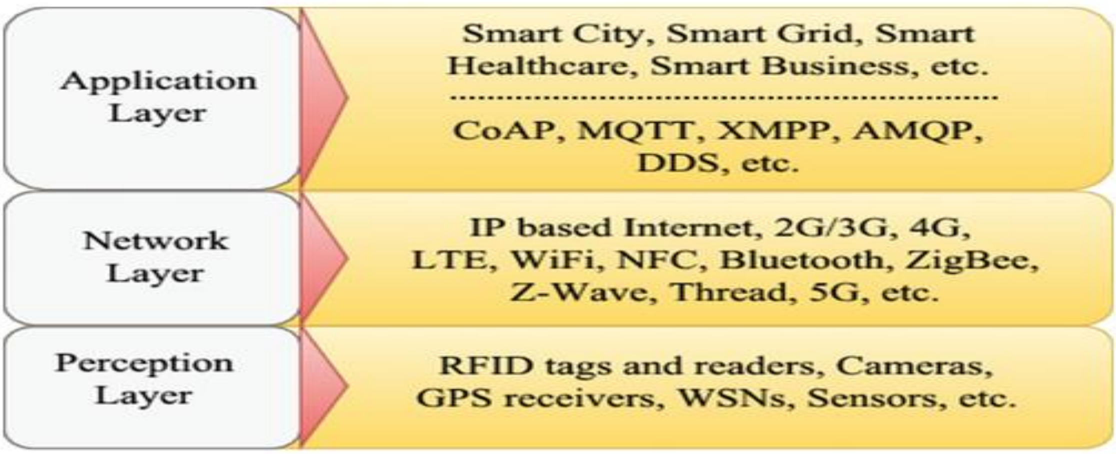

FIGURE 1.1. The layers of IoT architecture. [12]

2.2.3.1.Perception layer

This is the physical layer of the architecture. This is where the sensors and connected devices come into play as they gather various amounts of data as per the need of the project. These can be the edge devices, sensors, micro-controllers, and actuators that interact with their environment.

2.2.3.2.Network layer

The data that's collected by all of these devices needs to be transmitted and processed. That's the network layer's job. It connects these devices to other smart objects, servers, and network devices. It also handles the transmission of all of the data, the transmission can be done using RFID, ZigBee, 6LowPAN, Wi-Fi, 3G/4G, BLE etc. This layer can be associated to the physical and network layer of the TCP/IP model.

2.2.3.3.Application layer

The application layer is what the user interacts with. It's what is responsible for delivering application specific services to the user. This can be a smart home implementation, for example, where users tap a button in the app to turn on a coffee maker, it incorporates protocols like MQTT, CoAP, AMQP, DDS and HTTP. These protocols are very used in IoT systems. [13]

2.2.4. Components of IoT

Internet of Things (IoT) is the network of physical objects that contain embedded technology to communicate and sense or interact with their internal states or the external environment, contain 2 components:

2.2.4.1.Sensors

Sensors is an electronic device that produces electrical, optical, or digital data derived from a physical condition or event. Data produced from sensors is then electronically transformed, by another device, into information (output) that is useful in decision making done by intelligent devices or individuals (people).

2.2.4.2.Networks

Networks is to transmit the signals collected by sensors with all the different components of a typical network including routers, bridges in different topologies, including LAN, MAN and WAN. Connecting the different parts of networks to the sensors can be done by different technologies including Wi-Fi, Bluetooth, Low Power Wi-Fi , Wi-Max, regular Ethernet , Long Term Evolution (LTE) and the recent promising technology of Li-Fi (using light as a medium of communication between the different parts of a typical network including sensors). [14]

2.2.5. Protocol of IoT

In this section is explained the IoT protocol used in this work (MQTT). IoT protocols are application protocols that are commonly used for data exchange in IoT systems.

2.2.5.1.Message Queuing Telemetry Transport (MQTT)

The MQTT has a API that enables data exchanging between IoT-Ticket and the devices. It is possible to control a device, such as give actions, update de firmware, and change configurations using just the platform.

2.2.5.1.1. Definition of MQTT

MQTT is an open OASIS and ISO standard for client-server, publish/subscribe type messaging transport protocol. It was invented by Dr. Andy Stanford-Clark of IBM, and Arlen Nipper of Arcom, in 1999. The design principles of this protocol focus on minimizing network bandwidth and device resource requirements ensuring reliable delivery. It is capable of transmitting data over low-bandwidth or unreliable networks with very low consumption of power. Characteristics like lightweight, open, simple, and easy deployment make MQTT an ideal communication protocol for constraint environments. The protocol runs over TCP/IP, or over other network protocols that provide ordered, lossless, bi-directional connections.

2.2.5.1.2. MQTT control packet

MQTT uses 14 different control packets (see TABLE 1.2 for their description) and they are numbered from 1 to 14 with maximal size of 256 MB. Every control packet has a fixed header (with the type of the control packet, flags specific to it, and remaining length), while some of them have a variable header and/or payload.

Control packet	Description
CONNECT	Client requests a connection to a server
CONNACK	Server acknowledges the connection request to the client
PUBLISH	Message publishing
PUBACK	Publish acknowledgement
PUBREC	Publish received (QoS 2 publish received, part 1)
PUBREL	Publish release (QoS 2 publish received, part 2)
PUBCOMP	Publish complete (QoS 2 publish received, part 3)
SUBSCRIBE	Client subscribes to topics
SUBACK	Server acknowledges the subscription request to the client
UNSUBSCRIBE	Client unsubscribes from topics
UNSUBACK	Server acknowledges the unsubscription request to the client
PINSREQ	Client sends a PING request to the server
PINGRESP	Server sends a PING response to the client
DISCONNECT	Client sends a disconnect notification to the server for clear disconnection

TABLE 1.2. Control Packets in MQTT. [15]

2.2.5.1.3. MQTT Fundamentals

In this protocol two types of clients have been defined. The clients that send messages are called publishers. Other clients that receive the messages are called subscribers. These two types of clients never communicate directly with each other. In order to exchange messages, they utilize a central point that plays the role of a server and is called a broker. The role of the broker is to receive the messages sent by the publishers and to forward them to the subscribers. Publishers send messages on certain topics identified by their topic names. Subscribers use topic filters to subscribe/unsubscribe to/from specific topics by sending SUBSCRIBE/UNSUBSCRIBE control packets. When a broker receives the message, it determines the topic to which it needs to be sent and then transmits the message to those clients (subscribers) who have subscribed to that particular topic (FIGURE 1.2). Any message from the publisher to the broker, and from the broker to the subscribers is carried by the PUBLISH control packet. Note, that the publisher does not receive any information on how many subscribers received the published message.

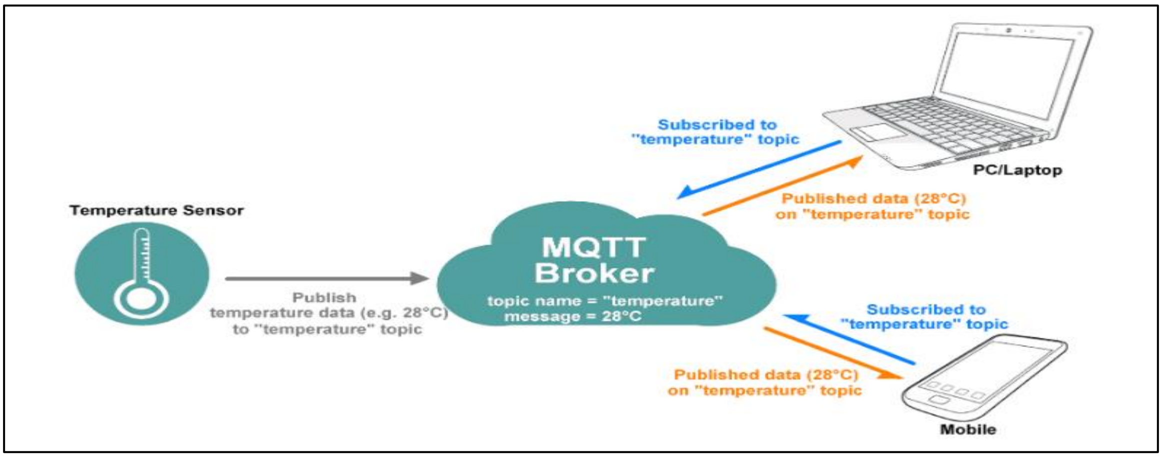

FIGURE 1.2. MQTT publish/subscribe model. [16]

2.2.6. Applications of IoT

In this part we considers a few among many of the IoT application domains that have emerged in recent years. Focuses mainly on some of the domains that have great potential for exponential growth in the fourth industrial revolution: [12]

Applications	Description
Smart home	Is a house or living environment where household appliances like washing machines and other everyday devices can be remotely monitored and controlled using smartphones, tablets, or laptop computers from anywhere in the World via the Internet or private network. This might enhance home care and monitoring, access control, energy efficiency, convenience and quality of everyday life.
Smart grid	This is an energy delivery concept that promises to optimally and efficiently deliver the highest quality of energy at the lowest cost possible. It will provide more accurate monitoring and control adaptation, where consumers can analyze their consumption pattern via the 2-way communication between their smart meters and the operators. Typical smart grid cyber assets are: smart meters, transformers, and data

	centers.
Smart city	Is the city that employs digital technologies to improve services in key sectors of the economy like healthcare, water, energy, transport, and waste-water treatment for the well-being of its citizens. A smart city is also expected to be proactive to global challenges. Examples of smart city: power-generation/distribution systems, Intelligent Transportation System (ITS), street lights, water/waste-water treatment facilities, water distribution systems, WSNs and communication systems.

TABLE 1.3. Applications of IoT.

2.3.Edge computing (EC)

Edge computing is an emerging paradigm born of necessity to move the computation at the edge of the network. Although the first appearance of Edge computing in the literature is previous to the Cloud, the increasing interest for Edge computing starts with the emerging of IoT and related new challenges.

In this section, we first submitted the main definition of Edge computing, then, we illustrate the architecture, characteristics, benefits, applications.

2.3.1. Definition of Edge computing

Edge computing is a technology developed in the context of high bandwidth and time sensitive IoT integration. As devices access the Internet of Things in large scale, the massive volumes of data generated on the terminals can provide commercial values but are very challenging in data processing.

Edge computing has been considered as one of the new technical trends for the development of IoT. The application of edge computing has been recognized as a significant means to achieve efficiency of network operations and latency reduction for better end user experience. It also satisfies the needs in terms of agile connectivity, real-time services, data optimization, application intelligence, and privacy protection. [17]

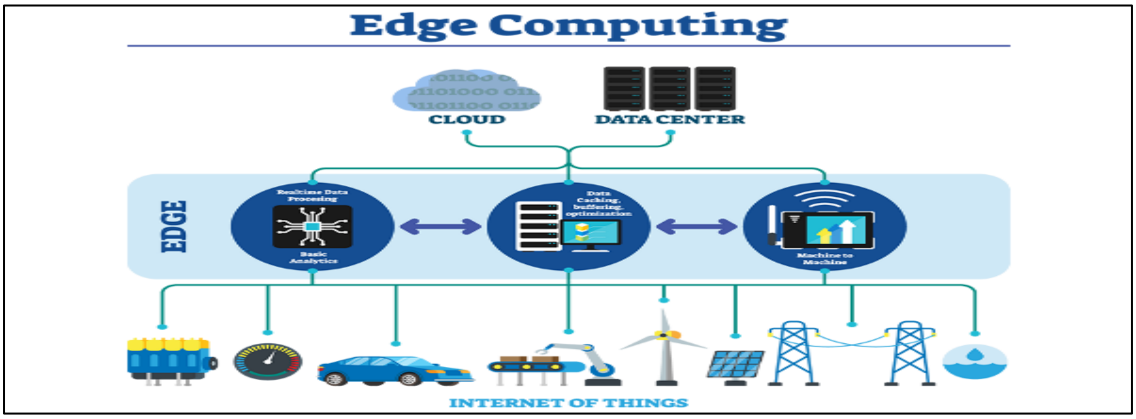

FIGURE 1.3. Edge computing based IoT. [17]

2.3.2. Architecture of edge computing

Architecture of edge computing, consists of three layers: IoT devices, Edge Computing, and Cloud Computing. All IoT devices are end users for edge computing. In this architecture, IoT can benefit from both edge computing and Cloud Computing, because of the characteristics of the two structures (i.e., high computational capacity and large storage).

Considering can deployed at different distances from the end users, the edge architecture is divided into a hierarchy, defining functions based on distance and resources. Thus, a hierarchical model is suitable for describing the network structure of edge computing.

The bottom layer consist smart devices (i.e., those with general purpose computing/storage capabilities) interacting directly with ECs or regular devices(e.g., a few ordinary cameras) interfaced locally with an ”edge computing device” that then interacts with the ECs.

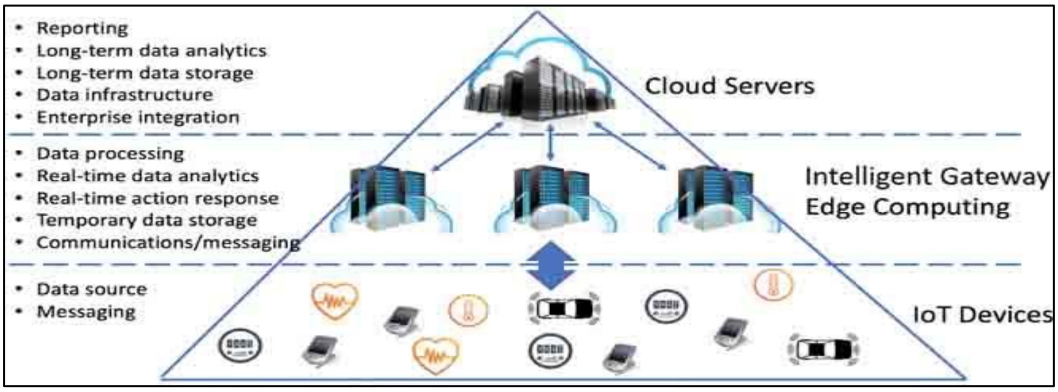

FIGURE 1.4. The basic edge computing architecture. [18]

2.3.3. Characteristics of Edge Computing

Edge computing possesses several characteristics similar to the Cloud Computing. However, the distinguishing characteristics of Edge computing that makes it unique are as follows:

2.3.3.1.Dense Geographical Distribution

Edge computing brings the Cloud services closer to the user by deploying numerous computing platforms in the edge networks. The dense geographical distribution of the infrastructure assists in the following ways:

a) The network administrators can facilitate location-based mobility services without traversing the entire WAN;

b) Big data analytics can be performed rapidly with better accuracy;

c) The Edge systems enable the real-time analytics on a large scale.

Examples include sensor networks to monitor the environment and pipeline monitoring.

2.3.3.2. Location Awareness

The location-awareness attribute of Edge computing allows the mobile users to access services from that Edge server closest to their physical location. Users can employ various technologies such as cell phone infrastructure, GPS, or wireless access points to find the location of electronic devices. This location awareness can be used by several Edge computing applications such as Fog-based vehicular safety applications and Edge-based disaster management.

2.3.3.3.Low Latency

Edge Computing paradigms bring the computation resources and services closer to the users, which reduces the latency in accessing the services. The low latency of Edge computing enables the users to execute their resource-intensive and delay-sensitive applications on the resource-rich Edge devices (e.g. router, access point, base station, or dedicated server). [19]

2.3.3.4.Open architectures

Edge computing resources deploy open architectures that leverage standardized protocols (e.g. OPC UA, MQTT) and semantic data structures (e.g. Sparkplug) that reduce integration costs and increase vendor interoperability.

2.3.3.5.Data pre-processing and filtering

Edge computing resources can pre-process data at the edge and only send relevant information to the cloud, which reduces data transmission and storage costs. An example of data pre-processing and filtering is an intelligent edge computing device running an edge agent that pre-processes data at the edge before sending it to the cloud, thus reducing bandwidth costs (e.g. AWS). [20]

2.3.4. Benefits of Edge Computing

Edge computing has emerged as one of the most effective solutions to network problems associated with moving huge volumes of data generated in today's world. Here are some of the most important benefits of edge computing:

2.3.4.1.Eliminates Latency

Latency refers to the time required to transfer data between two points on a network. Large physical distances between these two points coupled with network congestion can cause delays. As edge computing brings the points closer to each other, latency issues are virtually nonexistent.

2.3.4.2.Saves Bandwidth

Bandwidth refers to the rate at which data is transferred on a network. As all networks have a limited bandwidth, the volume of data that can be transferred and the number of devices that can process this is limited as well. By deploying the data servers at the points where data is generated, edge computing allows many devices to operate over a much smaller and more efficient bandwidth.

2.3.4.3.Reduces Congestion

Although the Internet has evolved over the years, the volume of data being produced everyday across billions of devices can cause high levels of congestion. In edge computing, there is a local storage and local servers can perform essential edge analytics in the event of a network outage. [21]

2.3.5. Applications of Edge computing

Prime use cases edge computing, which take full advantage of edge technology, include: [22]

Applications	Description
Autonomous Vehicles	The decision to stop for a pedestrian crossing in front of an autonomous vehicle (AV) must be made immediately. Relying on a remote server to handle this decision is not reasonable. Additionally, vehicles that utilize edge technology can interact more efficiently because they can communicate with each other first as opposed to sending data on accidents, weather conditions, traffic, or detours to a remote server first. Edge computing can help.
Healthcare Devices	Health monitors and other wearable healthcare devices can keep an eye on chronic conditions for patients. It can save lives by instantly alerting caregivers when help is required. Additionally, robots assisting in surgery must be able to quickly analyze data in order to assist safely, quickly, and

	accurately. If these devices rely on transmitting data to the cloud before making decisions, the results could be fatal.
Smart Speakers	Smart speakers can gain the ability to interpret voice instructions locally in order to run basic commands. Turning lights on or off, or adjusting thermostat settings, even if internet connectivity fails would be possible.

TABLE 1.4. Applications of edge computing.

Conclusion

In this chapter we provided an importance of realize smart civil engineering and to achieve maximum building efficiency, sustainability, using technology IoT, structural health monitoring and smart sensors. Then we gave a detailed explanation of technologies used: Cloud Computing, IoT and Edge computing.

In the next chapter we provide the main about machine learning and some previous studies.

Chapter 02: Machine learning and previous studies

Introduction

In recent decades, there has been rapid modernization, industrialization and infrastructural development across the world. The stability of soil slopes is a major source concern for civil engineers. Fearing loss of life and property due to the collapse of the slope and the infiltration of rainfall into the debris. Thus, there is a desire need of early-warning systems for large-scale soil slopes in crowded cities and the methods of possible to predict the value of the desired variables.

Machine learning algorithms have in the past years developed to become a working horse in brain imaging and the computational neurosciences, as they are instrumental for mining vast amounts of neural data of ever increasing measurement precision and detecting minuscule signals from an overwhelming noise floor.

In this chapter, we illustrate the definition of machine learning, architecture, types, algorithms and the applications. after these, we illustrate some relevant previous studies that are concerned with the field of smart civil engineering with early warning systems based Sensors, Structural health monitoring, Internet of things and Cloud computing.

1. Machine learning (ML)

The machine learning field, which can be briefly defined as enabling computers make successful predictions using past experiences, has exhibited an impressive development recently with the help of the rapid increase in the storage capacity and processing power of computers. Together with many other disciplines, machine learning methods have been widely employed in bioinformatics. In this section, we first review the definition of machine learning. Finally, we introduce some applications machine learning

1.1.Definition of Machine learning

Machine learning is based on the idea that system can learn from data, identify the patterns and make decision with minimum human intervention. This is the scientific study of algorithms and statistical models with the help of which computer system perform a specific task without using instruction, inference and patterns. Machine learning algorithms build mathematical model based on sample data and then make the decision. [23]

Machine Learning is an application of Artificial Intelligence(AI) and the most popular technique of predicting the future or classifying information to help people in making necessary decisions. Machine Learning algorithms are trained over instances or examples through which they learn from past experiences and also analyses the historical data. Therefore, as it trains over the examples, again and again, it is able to identify patterns in order to make predictions about the future.

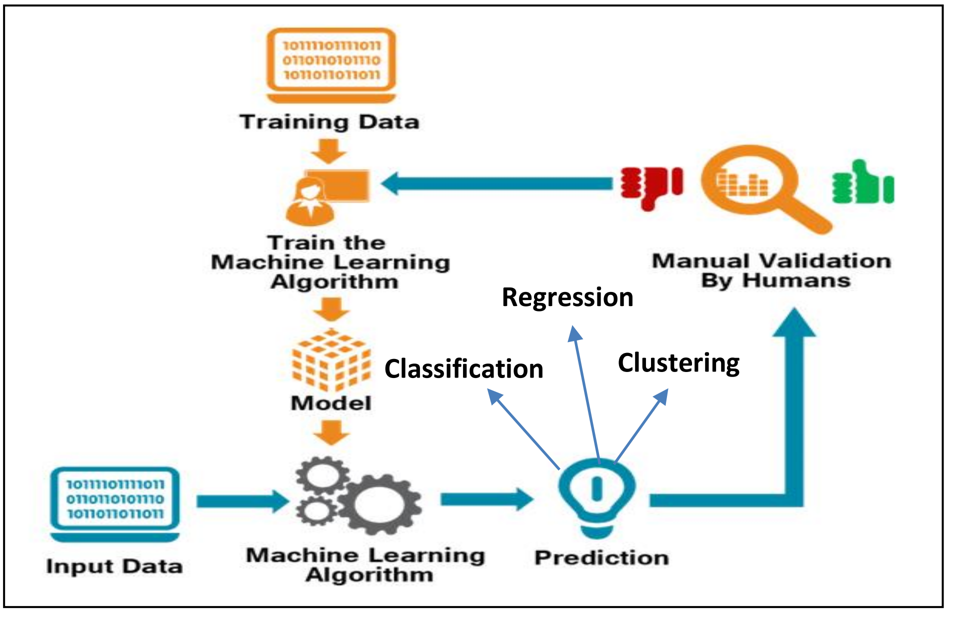

FIGURE 2.1. Machine learning. [23]

1.2. Architecture of machine learning

In this section we illustrate the architecture of machine learning: [24]

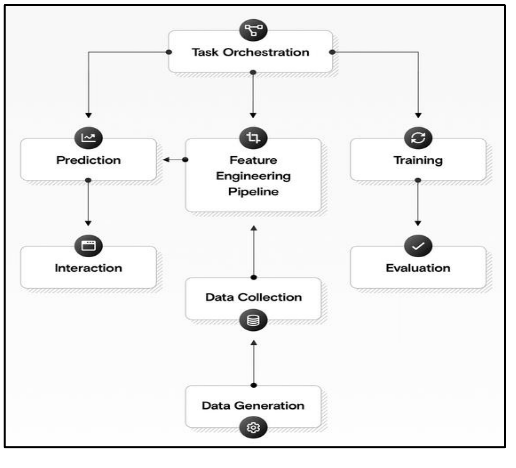

FIGURE 2.2. Architecture of machine learning. [24]

Data Generation

Every machine learning application lives off data. That data has to come from somewhere. Usually it's generated by one of your core business functions.

Data Collection

Data is only useful if it's accessible, so it needs to be stored – ideally in a consistent structure and conveniently in one place.

Feature Engineering Pipeline

Algorithms can't make sense of raw data. We have to select, transform, combine, and otherwise prepare our data so the algorithm can find useful patterns.

Training

This is where the magic happens. We apply algorithms, and they learn patterns from the data. Then they use these patterns to perform particular tasks.

Evaluation

We need to carefully test how well our algorithm performs on data it hasn't seen before (during training). This ensures we don't use prediction models that work well on "seen" data, but not in real-world settings.

Task Orchestration

Feature engineering, training, and prediction all need to be scheduled on our compute infrastructure (such as AWS or Azure) – usually with non-trivial interdependence. So we need to reliably orchestrate our tasks.

Prediction

This is the moneymaker. We use the model we've trained to perform new tasks and solve new problems – which usually means making a prediction.

Infrastructure

Even in the age of the cloud, the solution has to live and be served somewhere. This will require setup and maintenance.

Authentication

This keeps our models secure and makes sure only those who have permission can use them.

Interaction

We need some way to interact with our model and give it problems to solve. Usually this takes the form of an API, a user interface, or a command-line interface.

Monitoring

We need to regularly check our model's performance. This usually involves periodically generating a report or showing performance history in a dashboard.

1.3.Types of Machine learning

ML is about teaching machines to learn patterns from previous data for the purpose of making decisions or predictions. These tasks are widely applicable to many different types of problem, and each problem type requires a different way of learning. There are three types of learning, as shown in FIGURE 2.3. [25]

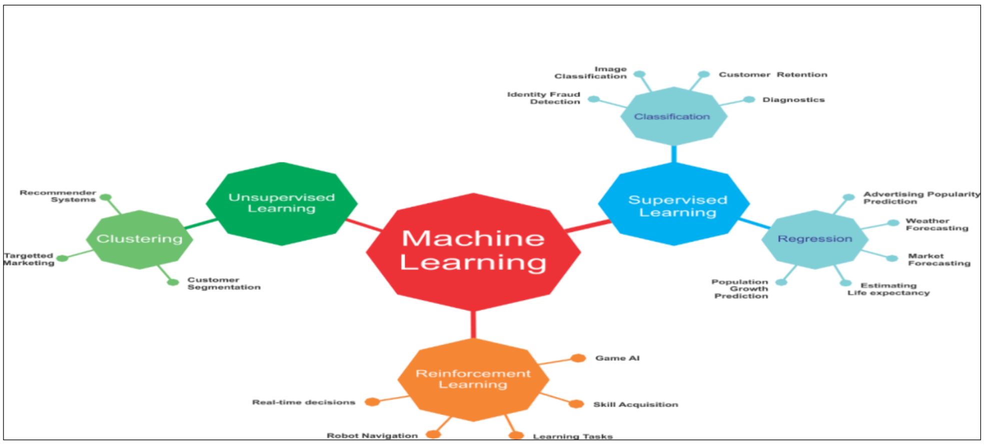

FIGURE 2.3. Types of Machine learning. [25]

1.3.1. Supervised Learning

Supervised learning algorithms can apply what has been learned in the past to new data using labelled examples to predict future events starting from analysis of a known training dataset that learning algorithms produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with correct, intended output and find errors in order to modify the model accordingly**.** [26]

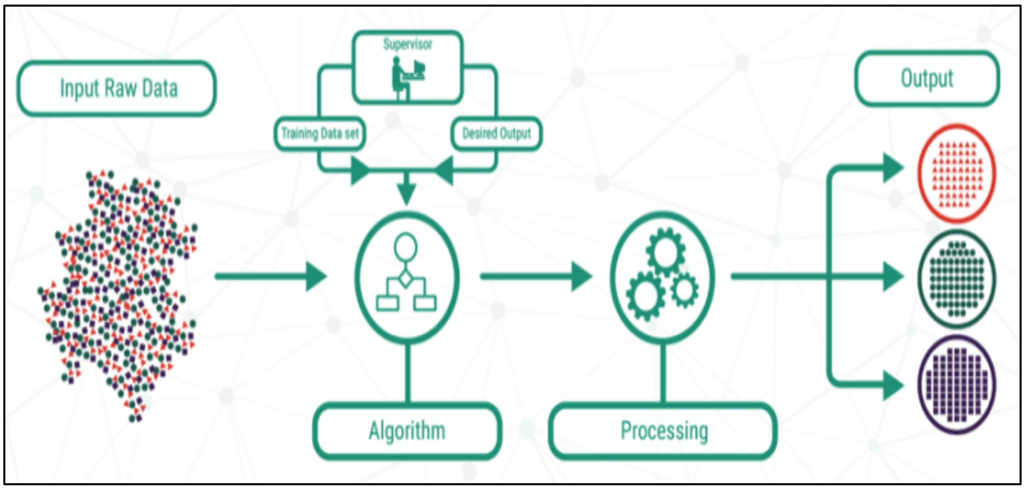

FIGURE 2.4. Supervised learning. [26]

Supervised learning can use two main methods classification and regression.

1.3.1.1.Classification

The concept of categorizing data is based on training with set of the data so that machine can essentially learn boundaries that separate categories of data. Classification is used to predict discrete values as which class is data point of given dataset. The data must be divided into one of two or more classes. **[**27]

1.3.1.2.Regression

Regression means to predict the output values using training data. Regression is used to predict continuous quantity. A regression algorithm may predict discrete values, but the discrete value is in the form of integer quantity. Regression prediction can be evaluated using root mean squared error. Algorithm used into regression are Random forest, Linear regression etc. [27]

1.3.2. Unsupervised Learning

In unsupervised algorithms from unlabeled data to describe a hidden structure. When the system is trying to figure out the output its does not provide the right output, but it explores the data and can inferences can be drawn from data sets to describe the hidden structures from unlabeled data. In this method the user only have input data (X) and there is no labels. The data given for learning model has to be analyzed more. [25]

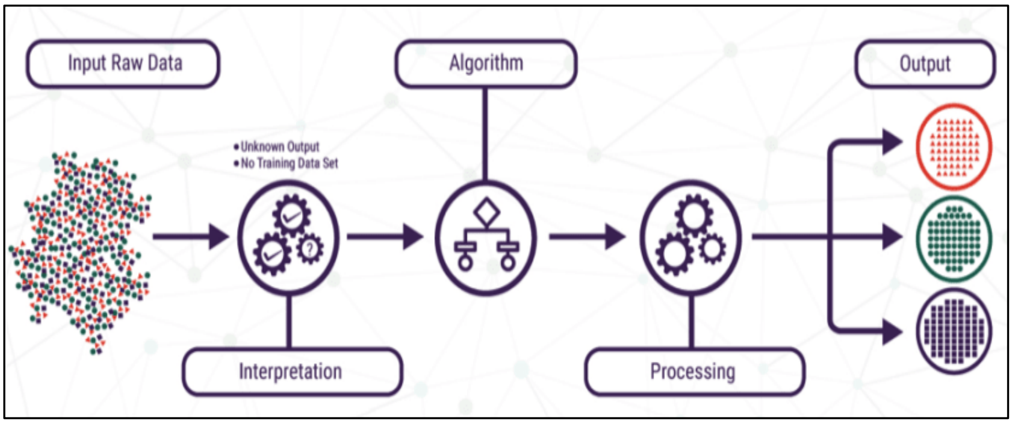

FIGURE 2.5. Unsupervised learning. [25]

Unsupervised algorithm can use method Clustering.

1.3.2.1.Clustering

Clustering uses to organize unlabeled data points into homogeneous groups where those within a "cluster" have similarities and "clusters" have dissimilarities. Since data is not pre-labeled, there is no training process. Instead, clusters are created using similarity functions that measure the distance between points. Relative to classification, clustering is considered a less complex approach and well suited for large datasets. [28]

1.3.3. Reinforcement Learning

Reinforcement Learning is another part of Machine Learning that is gaining a lot of prestige in how it helps the machine learn from its progress. Readers who have studied psychology in college would be able to relate to this concept on a better level.

Reinforcement Learning spurs off from the concept of Unsupervised Learning, and gives a high sphere of control to software agents and machines to determine what the ideal behavior within a context can be. This link is formed to maximize the performance of the machine in a way that helps it to grow. [29]

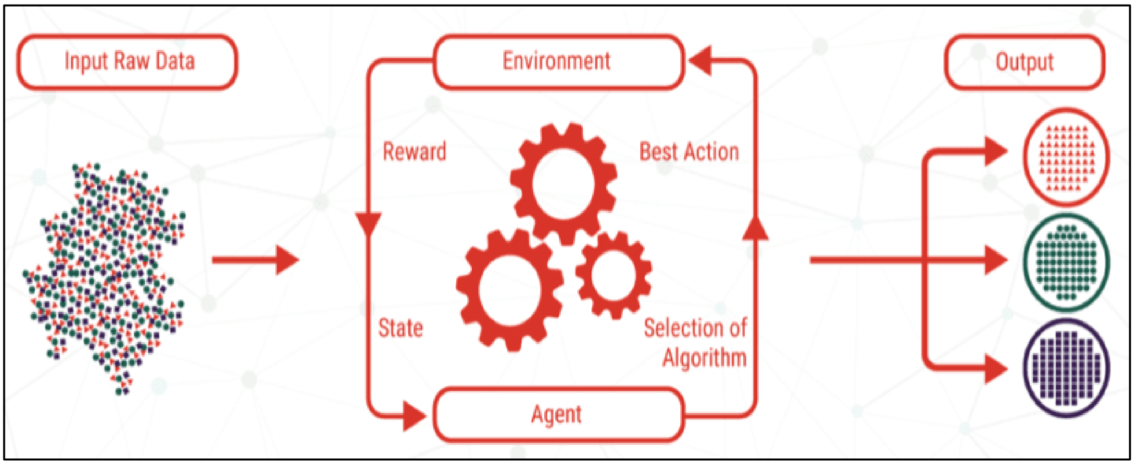

FIGURE 2.6. Reinforcement Learning. [29]

1.4. Machine Learning Algorithms

There are number of machine learning algorithms, such as: gradient boosting, k-nearest neighbor, multi-layer perceptron regression, linear regression, ridge regression, lasso regression, decision tree regressor, random forest regressor, adaboost regressor ;we defined some of them are below: [30]

1.4.1. Gradient Boosting Regressor (GBR)

Gradient Boosting Regressor (GBR) builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage a regression tree is fit on the negative gradient of the given loss function. [31]

1.4.2. K-Nearest Neighbor

K-nearest neighbor (KNN), the training data (which is well-labeled) is fed into the learner. When the test data is introduced to the learner, it compares both the data. K most correlated data is taken from training set. The majority of k is taken which serves as the new class for the test data. [32]

1.4.3. Multi-layer perceptron regression

Multi-layer perceptron regressor (MLPRegressor) is an estimator available as a part of the neural network module of sklearn for performing regression tasks using a multi-layer perceptron. [33]

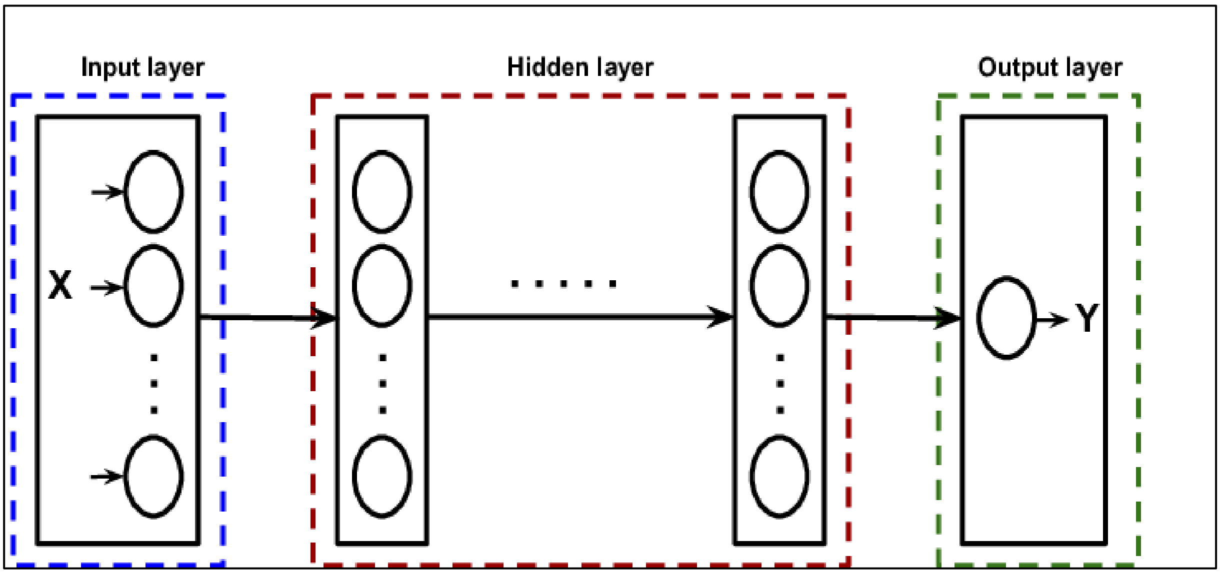

FIGURE 2.7. Multi-layer perceptron regressor (MLPRegressor). [34]

The mathematical equation for a multi-layer perceptron regressor is:

$$Y_i = b_0 + b_1X + e$$

Where:

- $\mathbf{Y_i}$ estimated or predicted value.
- $\mathbf{b_0}$ estimated of the regression intercept.
- $\mathbf{b_1}$ estimate of the regression slope.

- **X** independent variable.
- **e** error term.

1.4.4. Linear regression

Linear regression (LR) is the fundamental regression algorithm where we need to predict the output *y* coordinate from the input x. Imagine the scenario where there are N data points in 1 dimension (i.e., number of features is just one). Each data point has the corresponding *y* coordinate. The task is to predict for a given *x* input, what could be the y coordinate. There are several possible ways in which the given task can be accomplished. [35]

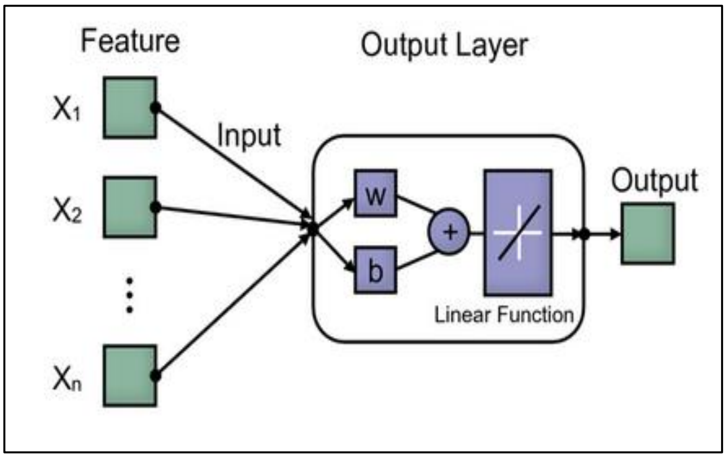

FIGURE 2.8. Linear regression (LR). [36]

The mathematical equation for a linear regression is: $y = ay + b$

Where:

- **Y** dependent variable.
- **X** independent variable.
- **a** coefficient of relationship between “X” & “Y” axes.
- **b** constant intercept. [37]

1.5. Activation function

Activation functions are responsible for calculating the sum of the product of the various weights and inputs with the bias to determine the final output value for the current hidden layer, which would be the input for the next layer. [38]

$$Y = f\left(\sum_{i=1}^{N} w_i \times x_i - w_0\right)$$

There are many possible activation functions, however in practice there are mainly two that are used:

1.5.1. Function of Rectified Linear Unit (Relu)

The Relu function is defined by: [39]

$$F(u) = \max(0, u)$$

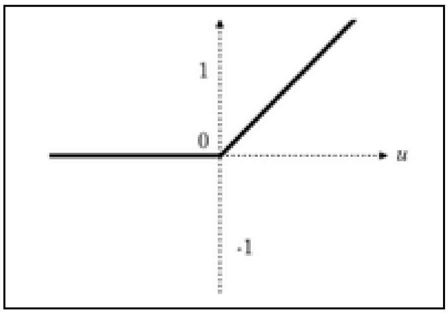

FIGURE 2.9. Function of ReLU activation. [39]

1.5.2. Function of Heaviside

The Heaviside function is defined by:

$$\forall\, x \in R \;\; f(x) = 1 \; si \; x \geq 0 \;\; , else \; 0.$$

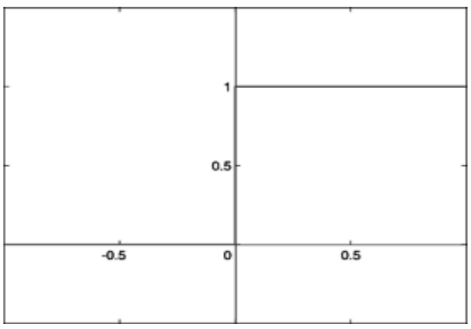

FIGURE 2.10. Function of Heaviside.

1.5.3. Function of sigmoid

The Heaviside sigmoid is defined by:

$$\forall\, x \in R\ f(x) = \frac{1}{1+e^{-x}}$$

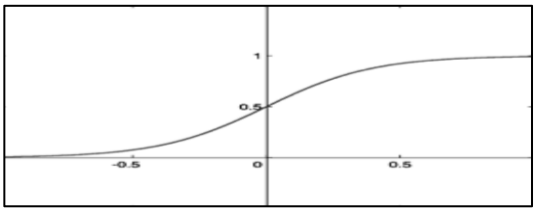

FIGURE 2.11. Function if sigmoid.

1.6. Applications of machine learning

In many applications of machine learning, the goal is to learn a classification/prediction model from the data that can be used to predict the classes/values of new (future or test) cases/instances. One measure of progress in Machine Learning is its significant real-world applications, such as those listed below: [23]

Applications	Description
Financial services	In financial services, machine learning technology is used to identify the important insight in data and to prevent fraud. The insights help to identify investment opportunities or help investors to know when to trade. Data mining concepts also identify high risk profiles of clients or to pinpoint warning signs of fraud.
Health Care	This is the major area in which wearable devices and sensors are used to assess patient's health in real time. Machine learning also helps medical experts to analyze the data to identify trends. This may lead to improve diagnoses and treatment.
Transportation	Machine learning is used to make routes more efficient and to predict the problems to increase profitability. It can be done after analyzing the data to identify patterns and trends. Data

	analysis and modeling aspects are key factors to delivery companies and transportation organizations.

TABLE 2.1. Applications of Machine learning.

2. Previous studies

In this section we demonstrate some previous studies which concerned field of smart civil engineering.

2.1.Article 01

DONG-SHENG XU and his assistances focus about: Early-Warning System with Quasi-Distributed Fiber Optic Sensor Networks and Cloud Computing for Soil Slopes. [40]

2.1.1. Content

Earthquake and rainfall will result in slope failure, debris flow, cause lots of casualties and property loss, for this reason important to develop an early-warning system based on various sensors, IoT and cloud computing, to ensure safety of human beings and assets.

In recent years, the development of micro-electromechanical systems (MEMS) has gained great interest, cause of limited wear and electromagnetic interference problems for conventional pressure sensors based on electrical signals.

With the development of technology appeared new type of sensors named fiber optic sensing (FOS), has many intrinsic advantages, such as corrosion resistance, resistance to electromagnetic interference (EMI), high resolution and precision, small size, large bandwidth and multiplexing ability.

In this study, an early warning system can be established for a soil slope with fiber optic sensing technologies, the sensor data interpretation, the internet of things (IoT), cloud computing and wireless sensor networks.

Firstly, an early-alarm system for real-time monitoring of soil slopes is presented with four layers, such as a sensing layer, an IoT layer, a cloud-computing layer and an application layer.

2.1.2. The hierarchical structure of early-alarm system

An early-alarm system for real-time monitoring of soil stability is essential to reduce or avoid casualties and property losses for debris flow. (FIGURE 2.12) shows the hierarchical structure of early-alarm system for soil slope.

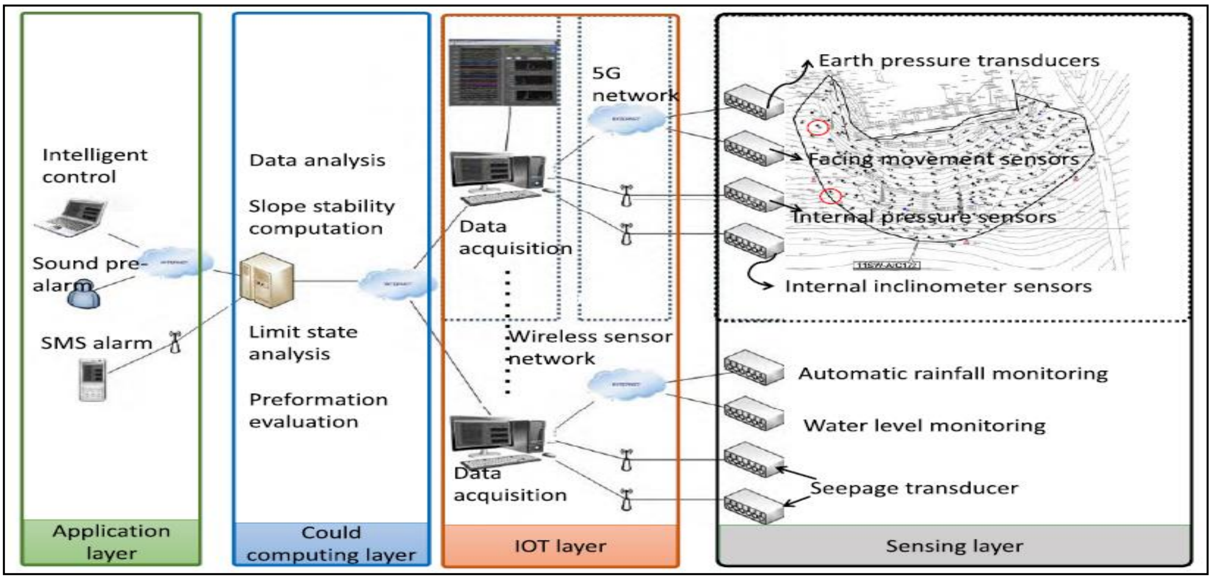

FIGURE 2.12. The hierarchical structure of early-alarm system for soil slope. [40]

2.1.2.1.Sensing layer

Is a most fundament layer in the whole early-warm system, in this layer, they proposed a new type of fiber-optic based pressure sensor (FPS) for pressure measurement of earth materials, fabricated and analyzed, this layer consists also of various fiber optic sensors which can be connected in series with a single fiber.

2.1.2.2.IoT layer

In this layer, fiber optic transducers will be connected with the fiber optic sensing interrogator through an armed optical fiber to obtain the wavelength information in real-time to monitor the surrounding deformations, internal soil stresses, temperatures and rainfall data. These data is recorded in a data acquisition system, then will be uploaded into the cloud-computing platform through wireless network. The instructions for sensing layer can be also downed from the cloud-computing platform. The wireless network can improve the data transmission through sensing layer and cloud computing layer, which is essential for the real-time monitoring and early warm of debris flows.

2.1.2.3.Cloud computing layer

In this layer, the data transmitted from the above layer (IoT layer) will be classified and examined. After this, will be excluded the error data. Then, the transducer data is classified into categories, such as deformation, stress, temperature, seepage and factor of safety (FOS). Finally, the data will be combined in a numerical model to analyze the safety levels. The result is a safety indicator from the cloud-computing layer.

2.1.2.4.Application layer

Is the top layer of the completely early-warning system, is based on the IoT, cloud computing and numerical analysis. The application layer responsible for sending information of slopes to ender users through various methods, such as sound-light alarm system, SMS, emails, and emergency rescue plan. The safety degree of the instrumented soil slope will be divided into three levels, which are safety level, very early warning level and serious warning level.

2.1.3. Result

It proves that the proposed early-warning system has potential to monitor the health condition of the soil slopes.

2.1.4. Conclusions

In this study, numerous fiber optic sensors were put-in in a slope. Through IoT and cloud computing, the health condition of the instrumented slope are often evaluated and therefore the analysis results are sent out through the applying layer. The key findings are listed as follows:

- In the sensing layer, Calibrations results of the FPS with twin diaphragm kind indicated that its accuracy will fulfill the necessity of field slope watching.
- The wireless network and optical fibers can be used to connect sensing layer with the information acquisition system. With the IoT, the data of slope deformation are going to be uploaded to the cloud computing layer.

- Through IoT and cloud computing, it can be found that the rainfall infiltration robust have an effect on the slope stability, especially on the surface of the slope.
- In the cloud computing layer, a finite part simulation demonstrated that the slope are going to be collapse once the rainfall over 0.5mm/h with rainfall length larger than 100 h. The analysis results offer a basis for the early-alarm monitoring system.

2.2. Article 02

Mei Dong and his assistances interest about: Deformation Prediction of Unstable Slopes Based on Real-Time Monitoring and DeepAR Model. [41]

2.2.1. Content

In human activities, have been recorded, numerous failure cases of slope, landslides and rock falls, it pose a risk to traffic and cause high losses every year. For this has become important a processing of slope failure in order to overcome the serious and complex challenges as a result of construction monitoring techniques and early warning techniques.

The deformation and failure mechanism of a landslide is a complex dynamic process, which seriously threatens people’s lives and property. Currently, prediction and early warning of a landslide can be effectively performed by using Internet of Things (IoT) technology to respond to emergencies on a moving slope and an artificial intelligence algorithm to predict the deformation.

At present, research on the prediction and early warning of landslide disasters is mainly divided into regional landslide susceptibility prediction and single landslide deformation prediction.

They used a wireless monitoring system with Micro-electromechanical Sensors (MEMS) to automatically measure the gravity-induced joints, displacement and environmental factors of a slope. High-resolution cameras can obtain images of a slope periodically and then the entire slope stability and displacement of feature points can be analyzed by image recognition technology.

The predictive accuracy of the DeepAR model was verified by the mean absolute error, the root mean square error and the goodness of fit. This study demonstrates that the presented monitoring system and the introduced predictive model had good safety control ability during construction and good prediction accuracy during operation. The proposed approach will be helpful to assess the safety of excavated slopes before constructing new infrastructures.

During the operation period, a DeepAR based data-driven model was adopted to predict displacement early. The prediction of unstable slopes was based on the data collected by the advanced monitoring technology.

2.2.2. Result

The proposed method is an effective approach to perform comprehensive monitoring and to establish an early-warning system for the safety of infrastructure involving steep slopes.

2.2.3. Conclusions

- In this work, an IoT-based monitoring system, combining multiple sensors.
- The monitoring system considered the entire deformation and the local deformation of the high slopes.
- DeepAR based data-driven model was adopted to predict displacement early. The prediction of unstable slopes was based on the data collected by the advanced monitoring technology.

2.3. Article 03

In this article Shitong Hou and Gang Wu interest about: A low-cost IoT-based wireless sensor system for bridge displacement monitoring. [42]

2.3.1. Content

In transportation infrastructure, bridges have an important role. In the past half-century have been constructed A large number of bridges, in the service process of the bridges appear various kinds of problems gradually.

Due to excessive usage, overloading, aging, and environmental impacts, Bridge structures are subjected to deterioration.

Wherefore timely maintenance work needs to be implemented, it is necessary to monitor a bridge to ensure its performance using structural health monitoring (SHM) technology through the interpretation of sensor data.

The traditional SHM system consists of sensors installed in a bridge, and a centralized data acquisition system is equipped to collect the data with cables.

However, the wiring difficulties and high maintenance costs of lines and equipment will be a complex issue when large numbers of bridges need to be monitored. These limitations have prompted researchers to explore the use of wireless sensors for a novel monitoring method.

Pines and Lovell discussed the method of using sensors and wireless communication to monitor large civil structures, and laboratory testing without the loss of a communication signal.

Ayyildiz presented a system using wireless sensor networks to monitor the structural element, and the sensor data was collected and sent to a data center utilizing low-power wireless communication technologies. There are two possible ways to realize wireless SHM: the first way mainly uses a cellular network to send the data from the sensor node to the data center directly, and the other way uses a gateway device that is responsible for collecting data from the sensor node and forwarding the data to the data center.

2.3.2. Results

- low-cost IoT-based sensor system for bridge displacement monitoring, which consists of a sensor transducer, an amplifier circuit, an micro programmed control unit module, an IoT wireless communication module and a cloud server.
- Experiments and field tests show that the system can be successfully implemented in bridge displacement monitoring applications.

2.3.3. Conclusions

- The developed system includes a sensor transducer that excites displacement variation, an amplifier circuit and MCU designed for data processing, and IoT that transmits displacement data wirelessly to the web server.
- It has been shown that the measurement via the wireless sensor system in this work provided an effective and affordable solution for the displacement monitoring of bridges, especially in construction monitoring.
- Furthermore, other bridge monitoring projects can be also conducted to achieve the aim of large-scale monitoring.

Conclusion

In this chapter, we have presented in quiet detailed way, the definition, the architecture, types, algorithms, activation functions and applications of machine learning. A background study in the Literature review on Smart Civil Engineering technologies is carried out. Then, information on Smart civil engineering technology was compiled and the requirement for infrastructure protection is studied using technologies wireless sensor network (WSN) and structural health monitoring (SHM).

The design, architecture of system smart civil engineering and the model of machine learning chooser will be carried in the next chapter.

Chapter 03: Design of system

Introduction

At present, various smart civil engineering technologies are available, yet the knowledge and level of awareness regarding them is still quite low. A contributory cause for the low awareness is that information regarding these technologies is not readily available and not consolidated. There is a need for civil engineers to be made aware of the significant benefits and working principles of these smart civil engineering.

This short study will consolidate information on them. Besides that, this study will also interest to connect civil engineering buildings to internet using technology internet of things, build a database to aid practicing engineers to monitor these buildings and structures, finally, predict and diagnosis results using machine learning.

In this chapter, we present the objective of this study, illustrate the architecture proposed of the system to solve the problems and we demonstrate the operations of our system using smart device Raspberry Pi.

1. Problematics

Today, using the concept of IoT we make sensors to communicate each other and to transmit the data to edge computing and cloud computing that carry inherent data processing capabilities, monitor the parameters and ensures maintenance of the civil engineering. IoT provides solutions for various problems and it allows things to be sensed or controlled remotely in network infrastructure.

- ✓ How to connect civil engineering buildings to internet with IoT technology?
- ✓ How we do the diagnosis of civil engineering buildings?

2. Objectives

Generally, Main objective of this study is to obtain an effective low-cost and flexible solution for condition monitoring and infrastructure management of civil engineering, by connecting the buildings with the internet using the technology IoT.

The objectives of this study are as follows:

- To identify information that must be gathered to effectively monitor a civil engineering structure, such as humidity, temperature, etc.

- An effective implementation for Internet of Things used for monitoring civil engineering.
- Reducing loss of life and property due to the collapse of the slope and the infiltration of rainfall into the debris.
- The stability of smart civil engineering.
- Sensors are used to monitor the status of the civil engineering.
- Predictions about Landslides and rock falls to limit a risk to traffic that cause high losses every year.
- The civil engineering monitoring uses sensors, to detect the dangers. For this has become important a processing of slope failure in order to overcome the serious and complex challenges as a result of construction monitoring techniques and early warning techniques.
- Main objective is to obtain an effective low-cost and flexible solution for condition monitoring and infrastructure management of civil engineering, by connecting the buildings with the internet using the technology IoT.

3. Significance of Study

The study will provide consolidated information on smart civil engineering where details of smart buildings technologies are compiled and analyzed to help increase knowledge and awareness about them.

A user friendly database system in this study will be useful for engineers to identify, evaluate smart civil engineering for their structural health monitoring.

The database will serve as a starting point for the development of a more comprehensive information system on smart civil engineering technologies.

4. Architecture of system

The simplified architecture of the proposed system is demonstrated in FIGURE 3.1, which contains two parts: connectivity part consist internet of things (IoT) devices connect to buildings and prediction part composed machine learning to diagnosis the result of our system.

In the system, we chose the technique of wireless network to avoid risks of wired network, lowering and reduction cost, the system consist 3 layers: IoT layer, edge computing layer and cloud computing layer.

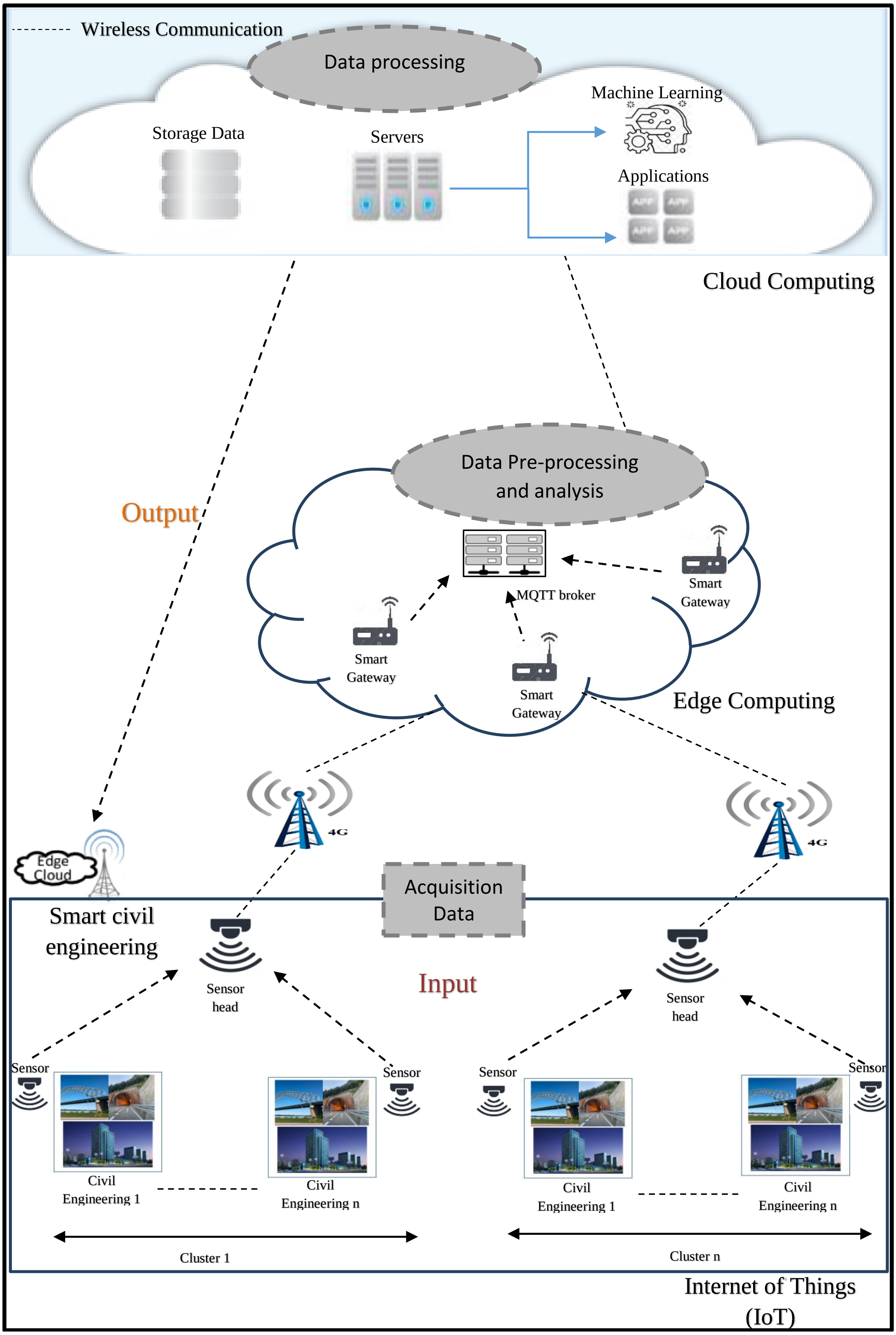

FIGURE 3.1. Architecture of system proposed.

4.1. IoT layer

Is the basis layer of system, contains intelligent nodes tied with hierarchical topology in order to increase the stability of the system, where the network was divided into groups called "clusters". Each group consists of nodes send the data into head node, these nodes responsible to sense, collect and acquisition data in real time to monitor temperature, humidity and distance, these data will be upload into base station (smart gateway) in edge computing platform through wireless network using the IoT protocol MQTT.

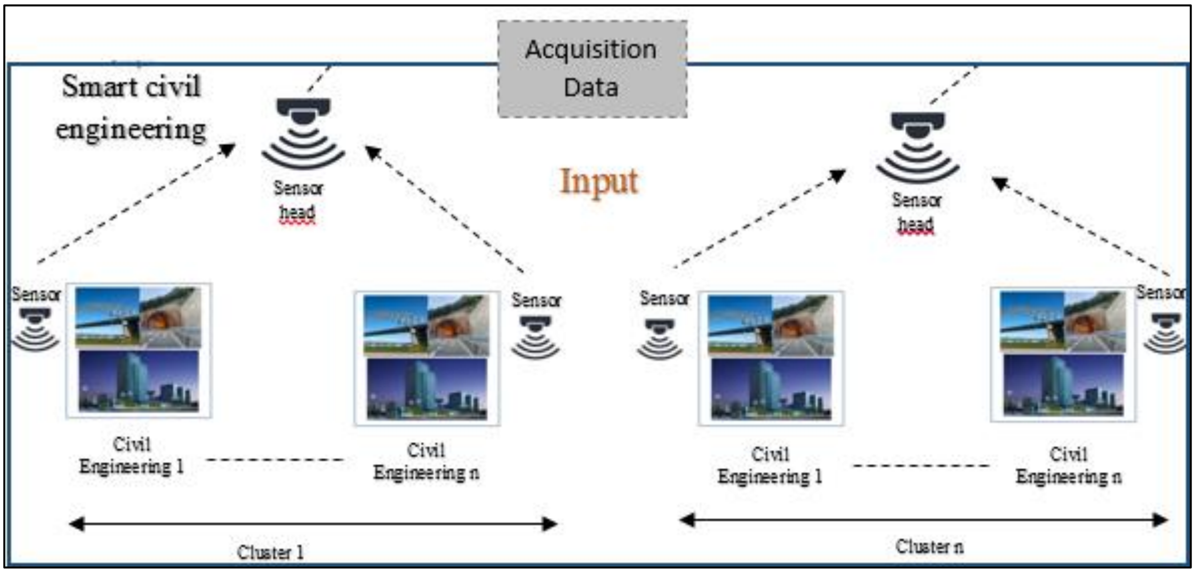

FIGURE 3.2. Internet of things (IoT) Layer.

4.2. Edge computing layer

In this layer, the data collected and transmitted from the bottom layer (IoT layer), will be preprocessing, analysis and performs filtering, to reconstructing it into more useful form in base station, after that, send these data to broker MQTT to be saved temporary in database of the broker. In the end, will be uploading only necessary and ready-to-send data to the cloud computing layer.

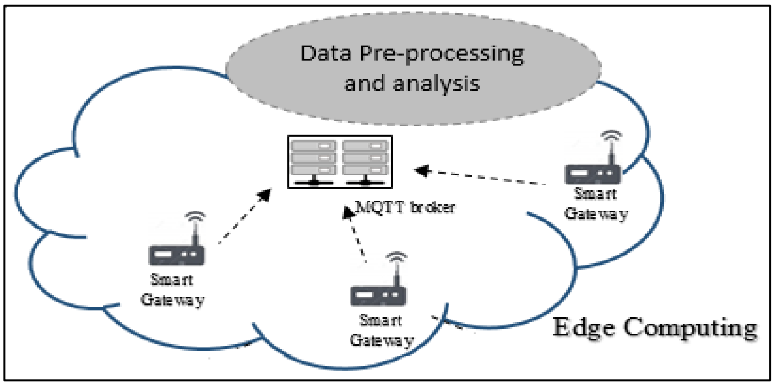

FIGURE 3.3. Edge computing Layer.

4.3. Cloud computing layer

Is the top layer of Smart civil engineering system, contains servers, application and database, where this layer responsible for analysis, processing and storage data. Firstly, the data sent from edge computing layer is saved and record in database for diagnosis and predict of the risks before they occur using machine learning. The result is a safety system for smart civil engineering.

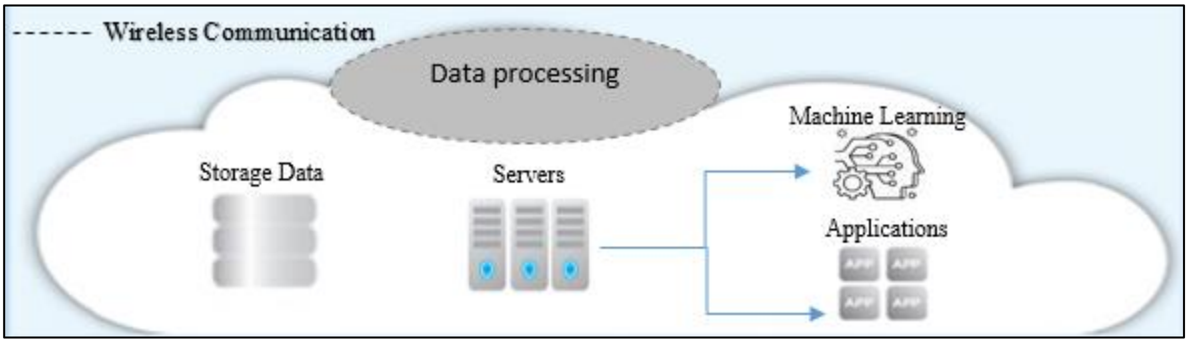

FIGURE 3.4. Cloud Computing Layer.

5. System hardware design

Our system is demonstrated in FIGURE 3.5, consists of two principal part (publisher and subscriber) connect each other with Wi-Fi (based on IEEE 802.11).

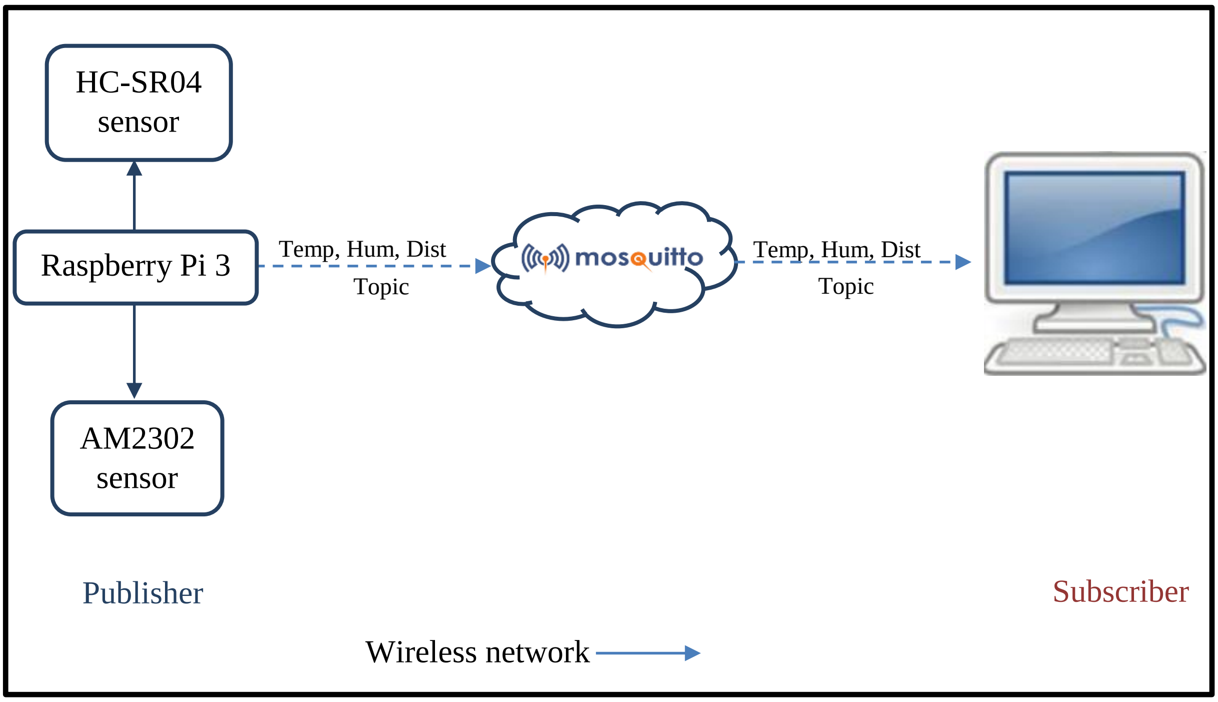

FIGURE 3.5. System hardware design.

Description

publisher part contains raspberry pi as a sensor node controller tied to sensor nodes AM2302 and HC-SR04, these sensor nodes is primarily responsible for sensing, collection and acquisition of information or sensor data. AM2302 sensor responsible to measure humidity and temperature. HC-SR04 sensor is the administrator to measure the distance between this sensor and the building. After that, these data will send it to broker with the topic "test" using port 1883. We chose the broker (mosquitto) because is an open source, free to use, lightweight, support for TLS with client certificate based authentication, support for authorization using a database and scalable horizontally and vertically (clustering, multithreaded...), the broker mosquitto send the data to subscriber in order to save these data in database using SQLite Browser, the subscriber contain also machine learning, for monitoring and processing the result of system.

In this system architecture, we have combined the sensors into a single board computer (Raspberry Pi) using female threads, then we connected the computer to the Raspberry Pi though USB cable for the alimentation.

The data sensed by the sensors are continuously transmitted through Raspberry pi to the cloud over the internet because of its good network connectivity.

5.1. Components of publisher part

Below we explain the components of publisher part: raspberry pi, sensor nodes AM2302 and HC-SR04.

5.1.1. Raspberry Pi

This device works well as a multi-processor. It has a graphics card, a volatile memory, RAM, device interfaces and other external wireless device interfaces. This raspberry Pi is consuming very less power, but it is still cheap and powerful. It requires a keyboard to provide commands, display unit and power supplies as a standard PC. Here, Raspberry Pi used the SD card as a hard disk. Raspberry Pi able to connect via a LAN / Ethernet or via a USB modem or via wireless. Raspberry Pi is supposed to support for various home and business applications. Raspberry Pi runs on a Linux-based OS and which operated by the Raspbian OS. Python is a programming language used to implement the Raspberry-Pi. It is capable of communicating with other external devices using wireless communication technologies, cellular networks, NFC, ZigBee, Bluetooth etc. [43]

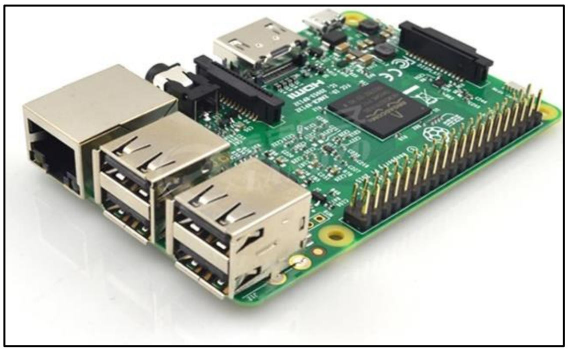

FIGURE 3.6. Raspberry Pi model 3. [43]

5.1.2. AM2302 sensor

Low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed). It's fairly simple to use, but requires careful timing to grab data. Contain 3 pins VCC - wire Connect to 3.3 - 5V power. Sometime 3.3V power isn't enough in which case try 5V power, Data out and Ground. [44]

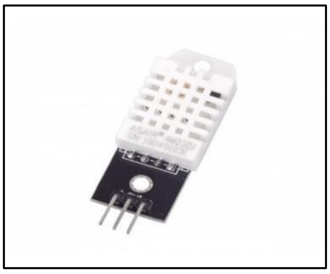

FIGURE 3.7. AM2302 sensor. [44]

5.1.3. HC-SR04 sensor

HC-SR04 is an ultrasonic sensor mainly used to determine the distance of the target object contain 4 pins Vcc, Trigger, Echo, and Ground. It measures accurate distance using a non-contact technology - A technology that involves no physical contact between sensor and object. Transmitter and receiver are two main parts of the sensor where former converts an electrical signal to ultrasonic waves while later converts that ultrasonic signals back to electrical signals. These ultrasonic waves are nothing but sound signals that can be measured and displayed at the receiving end. Test distance = (high level time * velocity of sound (340M/S) / 2. [45]

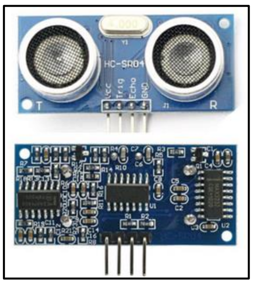

FIGURE 3.8. HC-SR04 sensor. [45]

5.2. WI-FI Standard

Wi-Fi is the name given to the IEEE 802.11 standard suite by the Wi-Fi Alliance. 802.11 defined the initial Wireless Local Area Networks (WLAN) standard. Basically, Wi-Fi means the transmission of radio signals. Wi-Fi, like Ethernet, is a physical link layer interface. [46]

5.3.Mosquitto

Mosquitto is an open source (EPL/EDL licensed) message broker that implements the MQTT protocol versions 5.0, 3.1.1 and 3.1. Mosquitto is lightweight and is suitable for use on all devices from low power single board computers to full servers.

The Mosquitto project provides a C library for implementing MQTT clients, and the very popular mosquitto_pub and mosquitto_sub command line MQTT clients. [47]

5.4.Components of subscriber part

Below we explain the components of subscriber part: SQLite Browser and machine learning.

5.4.1. SQLite Browser

DB Browser for SQLite (DB4S) is a high quality, visual, open source tool to create, design, and edit database files compatible with SQLite.

DB4S is for users and developers who want to create, search, and edit databases. DB4S uses a familiar spreadsheet-like interface, and complicated SQL commands do not have to be learned. [48]

In our system, we used two databases, are Ka Kp are respectively active and passive earth pressure coefficients due to the soil weight.

5.4.2. Machine learning

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. [49]

In order to implement our model, we split the dataset into two tasks: training and testing.

Training

The process of training an ML model involves providing an ML algorithm (that is, the learning algorithm) with training data to learn from. The term ML model refers to the model artifact that is created by the training process. We use the ML model to get predictions on new data for which you do not know the target.

Testing

The process where the performance of a fully trained model is evaluated on a testing set. it should be follow the same probability distribution as the training set.

Machine learning has many algorithms of supervised regression: Random Forest Regressor, Linear Regression and MLP Regressor, …, etc. In this work we run numerous algorithms in both databases FIGURE 3.9

Model
Linear Regression
Ridge Regression
Lasso Regression
K Neighbors Regressor
Decision Tree Regressor
Random Forest Regressor
Gradient Boosting Regressor
Adaboost Regressor
MLP Regressor

FIGURE 3.9. Regression models.

5.4.2.1.Linear Regression (LR)

Linear Regression is a supervised learning algorithm used for a statistical method to regress the data with dependent variable having continuous values whereas independent variables can have either continuous or categorical values. In other words "Linear Regression" is a method to predict dependent variable (Ka) based on values of independent variables (j, $\frac{d}{j}$, $\frac{b}{j}$). [36]

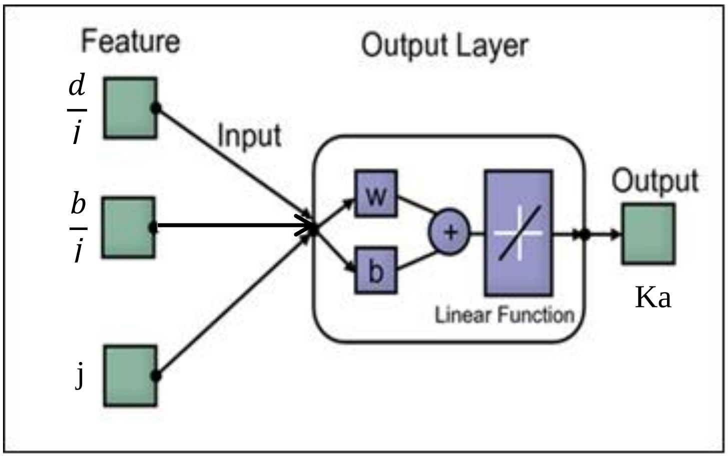

FIGURE 3.10. Schema of linear regression (LR).

5.4.2.2.Multi-layer perceptron Regressor (MLPRegressor)

MLPRegressor is an estimator available as a part of the neural network module of sklearn for performing regression tasks using a multi-layer perceptron.

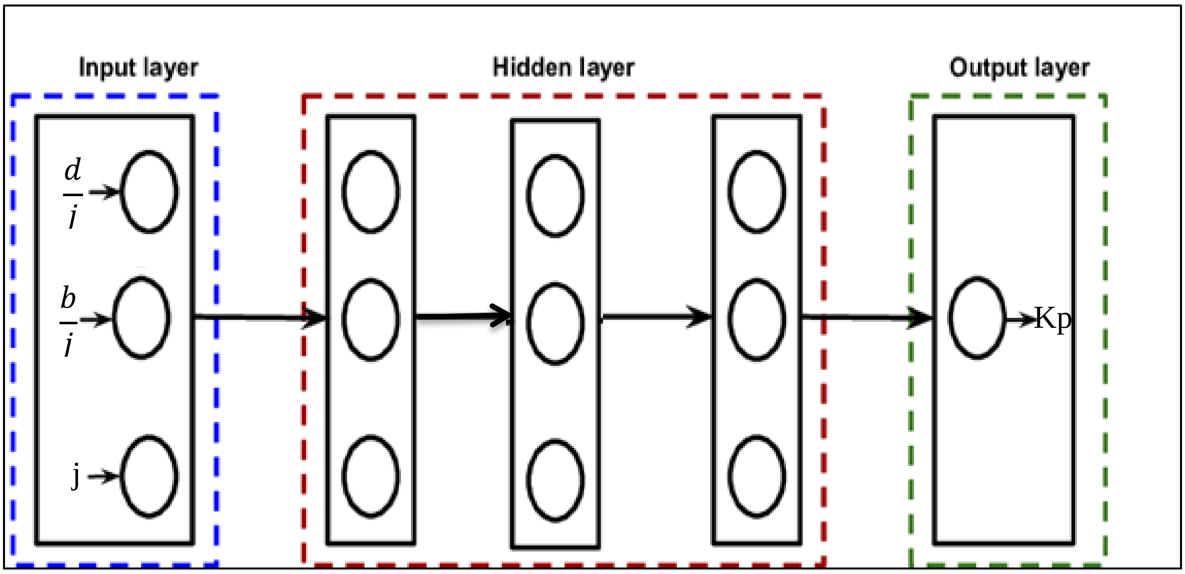

FIGURE 3.11. Schema of multi-layer perceptron regressor (MLPRegressor).

To judge the efficiency of a model, it is necessary to calculate the error of the learning carried out. For this we took into account two metrics to analyze the quality of prediction of the models:

- **Root mean square error (RMSE)**

RMSE is the measure of dispersion of predicted values from actual values. The formula for calculating RMSE:

$$\boldsymbol{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(S_i - O_i)^2}$$

Where $\mathbf{O_i}$ are the observations, $\mathbf{S_i}$ predicted values of a variable, and **n** the number of observations available for analysis.

RMSE is a good measure of accuracy, but only to compare prediction errors of different models or model configurations for a particular variable and not between variables, as it is scale-dependent. [50]

- **Mean Absolute Error (MAE)**

MAE is another useful measure widely used in model evaluations that here also we take the difference between the predicted and the actual value and divide it with the number of values, however, instead of squaring this difference, we take an absolute value of it The formula for calculating MAE: [51]

$$\boldsymbol{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

Where:

y_j: Actual output value.

$\hat{y}_j$: Predicted output value.

- **R-squared (R^2)**

R-squared (R^2) is an important statistical measure which is a regression model that represents the proportion of the difference or variance in statistical terms for a dependent variable which can be explained by an independent variable or variables. In short, it determines how well data will fit the regression model. [52]

$$R^2 = 1 - \frac{SSE}{SS_{yy}}$$

Where:

$$SSE = \sum(y - \hat{y})^2$$

$$SS_{yy} = \sum(y - \bar{y})^2$$

y is the actual value.

$\hat{y}$ is the predicted value of y.

$\bar{y}$ is the mean of the y values.

Conclusion

The proposed system provides accurate, low power and low cost system for remote monitoring of civil engineering with the dedicated sensors. where wireless sensor system that uses raspberry pi as a sensor node and Wi-Fi as a networking protocol. The IoT system use MQTT protocol for remote monitoring of weather parameters such as temperature and humidity. Then, will store the data in cloud computing through SQL-Browser, which be used for predict the risks using supervised regression algorithms of machine learning linear regression and multi-layer perceptron Regressor. This basic design can be extended and modified suitably to realize other IoT applications as well.

In the next chapter, we will do the implementation of the system using the programming language python.

Chapter 04: Implementation and experimental results

Introduction

This chapter is devoted to presenting the implementation using software and hardware to realize application and the obtained results of our system of IoT based solution for the prediction of Civil engineering building. The first part consists of the presentation of connectivity (Internet of things part) and the second part is about the prediction (Machine learning part) using the high performance model.

To ameliorate the performances of models, we utilize some simple techniques and effects like Dropout.

1. Environments and tools

For the development of our system, different environments and tools are used, to build the system from the programming language, libraries, packages, and IDE etc.

1.1.Hardware configuration

Our hardware configuration of an HP laptop and Lenovo of the following characteristics:

	Hp	Lenovo
Processor	Intel(R) i5-4300U CPU	Intel(R) i5-3230M CPU
Processor frequency	2.50 GHz	2.60 GHz
RAM	4 Go	8 Go

TABLE 4.1. Hardware used**.**

1.2.Development environment

Tools Logo	**Descriptions**
	MobaXterm is ultimate toolbox for remote computing. In a single Windows application, it provides loads of functions that are tailored for programmers, webmasters, IT administrators and pretty much all users who need to handle their remote jobs in a more simple

	fashion. MobaXterm provides all the important remote network tools (SSH, RDP, X11, SFTP, FTP, Telnet, Rlogin ...) to Windows desktop, in a single portable exe file which works out of the box. Some plugins can be used to add functions to MobaXterm such as UNIX commands (bash, ls, cat, sed, grep, awk, rsync ...). When developing MobaXterm, they focused on a simple aim: proposing an intuitive user interface in order for you to efficiently access remote servers through different networks or systems. Screenshots MobaXterm is being actively developed and frequently updated by Mobatek. [53]
Google colab	Google Colab or Colaboratory is a cloud service, offered by Google (free), based on Jupyter Notebook and intended for training and research in machine learning. This platform allows to drive Machine Learning models directly into the cloud. So without having to install anything on our computer except a browser. [54]
Th	Thonny is a free Python Integrated Development Environment (IDE) that was especially designed with the beginner Pythonista in mind. Specifically, it has a built-in debugger that can help when you run into nasty bugs, and it offers the ability to do step through expression evaluation, among other really awesome features. [55]

TABLE 4.2. Developing tools.

1.3.Programming language and packages

During the development of our application, we used Python as programming language and the different packages which provides for the machine learning.

Tools Logo	Descriptions
	Python is an interpreter, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant white space. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. [56]
pandas	Pandas is a Python package providing fast, flexible, and expressive data structures designed to make it easier to work with relational or labelled data. [57]
NumPy	Numpy is the fundamental package needed for scientific computing with Python. Besides its obvious scientific uses. It can also be used as an efficient multi-dimensional container of generic data. Arbitrary data types can be defined. This allows Numpy to seamlessly and speedily integrate with a wide variety of databases. Numpy is a successor for two earlier scientific Python libraries: Numeric and Numarray. [58]
Matplotlib	Matplotlib is a 2D graphics package used for Python for application development, interactive scripting, and publication-quality image generation across user interfaces and operating systems. [59]
TensorFlow	TensorFlow is an open-source software library for numerical computation using data ow graphs. TensorFlow was created and is maintained by the Google Brain team within Google's Machine Intelligence research organization for ML and DL. It is currently released under the Apache 2.0 open source license. [60]

TABLE 4.3. Programming language and packages.

1.4.The implementation of system

In this section, we provide the implementation functions used for each part in our system.

1.4.1. Connectivity part (IoT)

This part consist the connection link between the internet and buildings using technology internet of things (IoT) utilize the device raspberry Pi 3 and sensors for collect and sense the data to save it in dataset.

Libraries

The libraries is important to reduce the time required to code, FIGURE 4.1 demonstrated the libraries used.

```
#Libraries
import RPi.GPIO as GPIO
import time
import sqlite3
import datetime
import sys
import Adafruit_DHT
import paho.mqtt.client as paho #mqtt library
import os
```

FIGURE 4.1. Libraries.

1.4.1.1. Publisher part

These lines of code in FIGURE 4.2 is about publisher part located in raspberry pi to sense data using sensors and send these data to subscribe part in the other computer using broker “mosquitto” topic “test” and port 1883 for storage data in SQLite browser.

```python
publisher.py
port=1883 #MQTT data listening port
if __name__ == '__main__':
    while True:
        while temperature is not None and humidity is not None:
            dist = distance()
            print('Time:        %s'    % timestamp)
            print('Humidity:    {0:0.1f}%'.format(humidity))
            print('Temperature: {0:0.1f}°'.format(temperature))
            print ("Measured Distance = %.1f cm" % dist)
            client1.on_publish = on_publish #assign function to callback
            client1.connect("test.mosquitto.org",port,keepalive=60) #establishing connection
            payload= timestamp + " , " + str(humidity ) + " , " + str(temperature ) + " , " + str(dist )
            ret= client1.publish(topic,payload) #topic name is test
```

FIGURE 4.2. Publisher part.

1.4.1.2. Subscriber part

FIGURE 4.3 illustrate the code of subscribe part in order to save the sent data from raspberry pi (publisher).

```python
def on_message(client, userdata, message):
  sql='create table if not exists ' + data_table_name + ' (timestamp
text, humidity real, temperature real, distance real)'
  c.execute(sql)
  list = str(message.payload.decode("utf-8")).split(" , ")
  print("received data is :")
  print (list)
  if not "I am going offline" in list:
   c.execute("insert into datatry
(timestamp,humidity,temperature,distance) values (?,?,?,?)", list )
   conn.commit()
   print("        ****data saved succefully**** ")
   print("")
   workbook = Workbook(xl_path + xl_name)
   worksheet = workbook.add_worksheet()
   c.execute("select * from datatry")
      Python ▾   Largeur des tabulations : 8 ▾      Lig 29, Col 39   ▾   INS
```

FIGURE 4.3. Subscriber part.

Result of execution

To save the data in SQLite Browser and convert it to form Excel, we must run the code of subscriber (FIGURE 4.3) and publisher (FIGURE 4.2) respectively. The results are shown in FIGURE 4.4 and FIGURE 4.5.

```
Python 3.7.3 (/usr/bin/python3)
>>> %Run publisher.py

 Time:        2021-07-01 01:35:53
 Humidity:    28.1%
 Temperature: 28.1°
 Measured Distance = 157.7 cm
 published data is :
 2021-07-01 01:35:53 , 28.100000381469727 , 28.100000381469727 , 157.73667097091675
 Please check data on your Subscriber Code
```

FIGURE 4.4. Results of publisher.

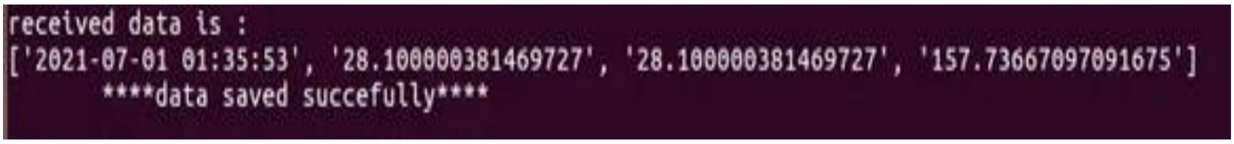

FIGURE 4.5. Result of subscriber.

Dataset

It is obvious of using a model in machine learning we must have a dataset.

In our system we used two private datasets for the prediction, Ka and Kp are respectively the active and passive earth pressure coefficients due to the soil weight, these datasets include three inputs:

j: Internal friction angle.

d/j: Ratios of soil–wall friction angle to internal friction angle.

b/j: Ratios of backfill inclination angle to internal friction angle.

1.4.2. Prediction part of machine learning (ML)

This part responsible to do prediction of the data send from raspberry pi (publisher) to the computer (subscriber), using the model regression which has in test case the value of RMSE is little.

Models used

In our system we choose the model at the expense of the RMSE value in testing case, we have the best model in dataset of active earth pressure coefficients due to the soil weight (Ka) is linear regression FIGURE 4.6, but regarding the dataset of passive earth pressure coefficients due to the soil weight (Kp), is Multi-layer perceptron Regressor FIGURE 4.7.

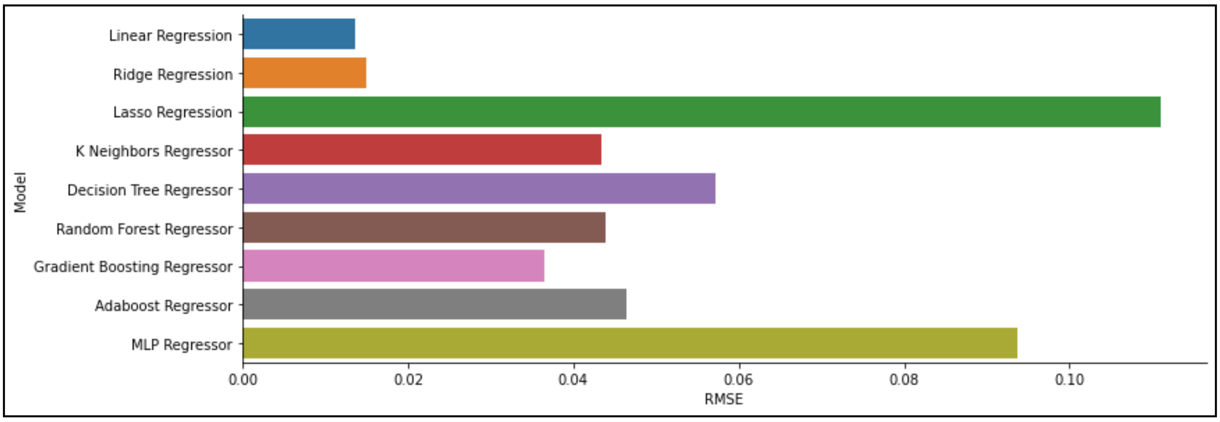

FIGURE 4.6. Result of testing score Ka.

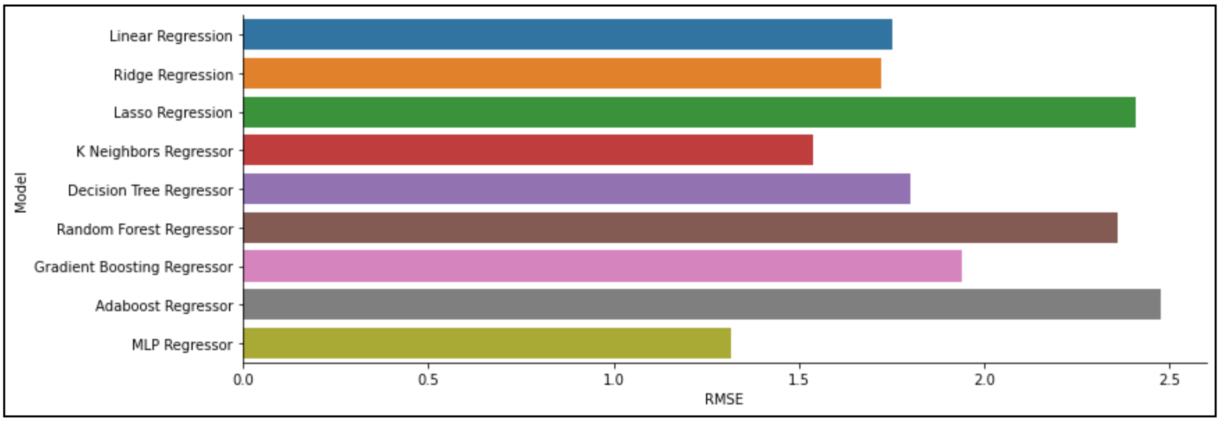

FIGURE 4.7. Result of testing score Kp.

1.4.2.1.Results obtained and discussion

After preparation dataset Ka and Kp (training and testing), we apply machine learning to them. In order to show the results obtained for two models, we illustrate below the results in terms of precision and error.

Import libraries and modules

For building machine learning models with Keras, we first import the various libraries and modules as they are presented in FIGURE 4.8 and described in the following.

```
import pandas as pd
import seaborn as sns
from fractions import Fraction
from keras.layers import Dense, Dropout
from keras.layers.core import Activation
from keras.models import Sequential
from keras.callbacks import EarlyStopping, ModelCheckpoint
from sklearn.model_selection import train_test_split
import keras
import keras.backend as K
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from math import sqrt
from keras import Input, Model
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from keras.models import load_model
from IPython.display import Image
from sklearn.neural_network import MLPRegressor
from sklearn.neural_network import MLPRegressor
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import (RandomForestRegressor, GradientBoostingRegressor,
                              AdaBoostRegressor)
```

FIGURE 4.8. Libraries of model.

a. Results obtained for model LR in Ka dataset

Firstly, we uploading dataset form excel in our model.

```
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
orig_data = pd.read_excel("/content/drive/MyDrive/Ka.xlsx")
orig_data.head()
```

FIGURE 4.9. Import DataSet Ka.

Split the dataset

The most frequent splitting strategy of the dataset, in machine learning, is to divide the dataset in training and test sets, where, in our system we split 25% testing and 75% training,

test_size: variable is where we actually specify the proportion of test set.

Random_state: controls the shuffling applied to the data before applying the split.

Shuffle method: takes a sequence, like a list, and reorganize the order of the items.

```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(data[data.columns[:-1]],
                                                    data[[data.columns[-1]]],
                                                    test_size = 0.25,
                                                    random_state = 1,
                                                    shuffle=True)
print ("TRAIN")
print(x_train)
print ("TEST")
print(x_test)
```

FIGURE 4.10. Splitting LR model.

Training and predict model

We have split our data into training and testing sets, fitting our data and now is finally the time to train our algorithm with following commands:

```
lin_reg = LinearRegression()
lin_reg.fit(x_train,y_train)
```

FIGURE 4.11. Fitting LR model.

Score(): Function return the coefficient of determination root squared of the prediction.

```
R_Squared = lin_reg.score(x_train,y_train)
```

FIGURE 4.12. Coefficient of determination.

Data frame: is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns.

GridSearchCV(): select the best parameters from the listed hyper parameters.

Predict(): function which use in the data to be tested.

```
x_test_scaled = pd.DataFrame(scaler.transform(x_test),
                             columns = x_test.columns)
clf = GridSearchCV( estimator=LinearRegression(),param_grid=param_grid,scoring='neg_mean_squared_log_error')
clf.fit(x_train_scaled,y_train)
pred=clf.predict(x_test)
print("Test RMSE: ", np.sqrt(mean_squared_error(y_test, clf.predict(x_test_scaled))))
print("Test R^2: ", r2_score(y_test, clf.predict(x_test_scaled)))
```

FIGURE 4.13. Scaled and predict LR.

Results

In order to assess the performance of our model and after training and testing our model, we show the results of RMSE and R squared, in train case we have lower RMSE value than in the test case, TABLE 4.4.

LR	Train	Test
RMSE	0.0171	0.022
R^2	0.973	0.959

TABLE 4.4. Results of LR model.

b. Results obtained for model MLP in Kp dataset

After import the important libraries, we uploading and print dataset.

```
data = pd.read_excel('/content/drive/MyDrive/Kp.xlsx')
print(data)
print (data.dtypes)
```

FIGURE 4.14. Import DataSet Kp.

Split the dataset

Split is a function in Sklearn model selection for splitting data arrays into two subsets: for training data and for testing data.

```
train_X, val_X, train_Y, val_Y = train_test_split(X, Y, test_size= .2, shuffle=True)
print(train_X.shape)
print(train_Y.shape)
print(val_X.shape)
print(val_Y.shape)
```

FIGURE 4.15. Splitting MLP model.

Creating the model

After loading the datasets and initializing the hyper-parameters, we have use TensorFlow to design our architecture. According the command:

model = Sequential()

Keras core layers

Dense: Is used to instantiate a dense layer.

Activation: Is used to add an activation function to the sequence of layers.

Compile: Once the model is created, we can config the model with model.compile() method , which takes three arguments: optimizer, loss and metrics. The optimizer controls the learning rate. We have used:

keras.optimizers.RMSprop(): it is recommended to leave the parameters of this optimizer at their default values.

mean_absolute_error: for our loss function, is a model evaluation metric used with regression models.

Metrics mae: mean absolute error is a measure of errors between paired observations expressing the same phenomenon.

root_mean_squared_error: measures the average magnitude of the error.

Once a model is "built", we call *summary()* method to display its contents:

```
model.add(Dense(10, input_dim=3 , activation='relu'))
#model.add(Dropout(0.25))
model.add(Dense(15, activation='relu'))
model.add(Dense(1, activation=K.exp))
# Configure the model and start training
model.compile(loss='mean_absolute_error',  optimizer=keras.optimizers.RMSprop(), metrics=['mae',r2_keras,
                                                   root_mean_squared_error,'mean_squared_error'])

print(model.summary())
```

FIGURE 4.16. Layers of MLP model.

Fitting the model

In order to train our model, the function *model.fit()* was used from Keras package. This function allows us to fitting literally our data to the model. It takes as input:

- **The validation data**: or testing data.
- **Number of epochs**: which represents the number of rounds that our model is passing through the dataset, in our system is 200.
- **Model checkpoint**: callback that is used to save the model, the weights as it shows the metrics during the training.
- **Shuffle method**: takes a sequence, like a list, and reorganize the order of the items.
- **Callbacks**: is used in conjunction with training using *model.fit()* to save a model or weights (in a checkpoint file) at some interval, so the model or weights can be loaded later to continue the training from the state saved.

```
# Callbacks
early_stopping = EarlyStopping(monitor='val_loss',
                               min_delta=0,
                               patience=100,
                               verbose=0,
                               mode='min')
checkpoint = ModelCheckpoint(baseline_path,
                             monitor='val_loss',
                             save_best_only=True,
                             mode='min',
                             verbose=0)

history =model.fit(train_X, train_Y, epochs=200, batch_size=16, validation_data=(val_X, val_Y), shuffle=True, verbose=2,
          callbacks = [early_stopping, checkpoint])
print(history.history.keys())
```

FIGURE 4.17. Fitting MLP model.

The prediction

This phase is used to do predictions on new test, in order to estimate and evaluate the model if performs and learned right. The used functions toward the prediction phase are presented as follow:

Load the model: we load the best trained model with the saved it in the h5 format.

Predict: calling *model.predict()* where it takes as input the test set or the data that we wanted to predict on them , the epochs and the exact label that you want to predict on as well.

The FIGURE 4.18 illustrates the different functions used on the prediction step.

```
baseline_path="MLP.h5"
model = load_model(baseline_path, custom_objects={"exp": K.exp,'r2_keras': r2_keras,'root_mean_squared_error':root_mean_squared_error})

y_val_pred = model.predict(val_X)
y_train_pred = model.predict(train_X)
print(len(y_train_pred))
print(len(y_val_pred))
```

FIGURE 4.18. Prediction steps MLP model.

Results Visualization

In our experimental results, the results to be demonstrated using curves plotting that represent the development of metrics during the learning process for both the training and testing steps (see FIGURE 4.19), example of plotting loss.

```
# summarize history for Loss
fig_acc = plt.figure(figsize=(10, 10))
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
fig_acc.savefig("model_regression_loss.png")
```

FIGURE 4.19. Plotting loss MLP model.

Experimental Results

In this section, we present the obtained results for our model of the used dataset.

When designing and configuring our machine learning model (MLP), we have attempted various configurations. The proposed architecture for MLP model, it illustrate in FIGURE 4.20.

```
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_18 (Dense)             (None, 10)                40
_________________________________________________________________
dense_19 (Dense)             (None, 15)                165
_________________________________________________________________
dense_20 (Dense)             (None, 1)                 16
=================================================================
Total params: 221
Trainable params: 221
Non-trainable params: 0
```

FIGURE 4.20. MLP configuration.

Tuning the batch size

In this first experiment, we have look at tuning the batch size on 200 epochs when training the machine learning MLP, to show its sensitivity for this parameter.

The batch size in iterative is the number of patterns shown to the network before the weights are updated. It is also an optimization in the training of the network, defining how many patterns to read at a time and keep in memory. The number of epochs is the number of times that the entire training dataset is shown to the network during training.

To analyze the comportment of the machine learning MLP with the set of hyper parameters for the MLP layers presented in table, concerning the batch size parameter, we have evaluated its performance, using the training/ testing root mean squared error RMSE (RMSE and Val RMSE respectively), training/ testing mean squared error (MAE and Val MAE respectively) and the training/ testing loss (loss and Val loss respectively) at batch size: 16. The results for the performance loss are shown on plot: FIGURE 4.21.

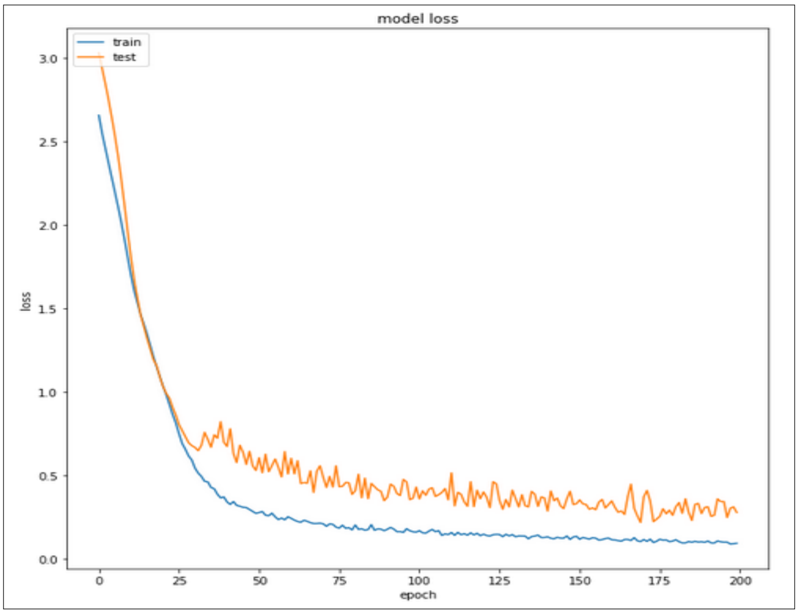

FIGURE 4.21. Graph of loss MLP.

According to FIGURE 4.21, loss decrease in interval of the number of epochs [0, 50], and in interval [50, 200] test loss stabilizes in level (0.4) and train loss still at level (0.1).

In this section, we illustrate the results of train and test model, FIGURE 4.22.

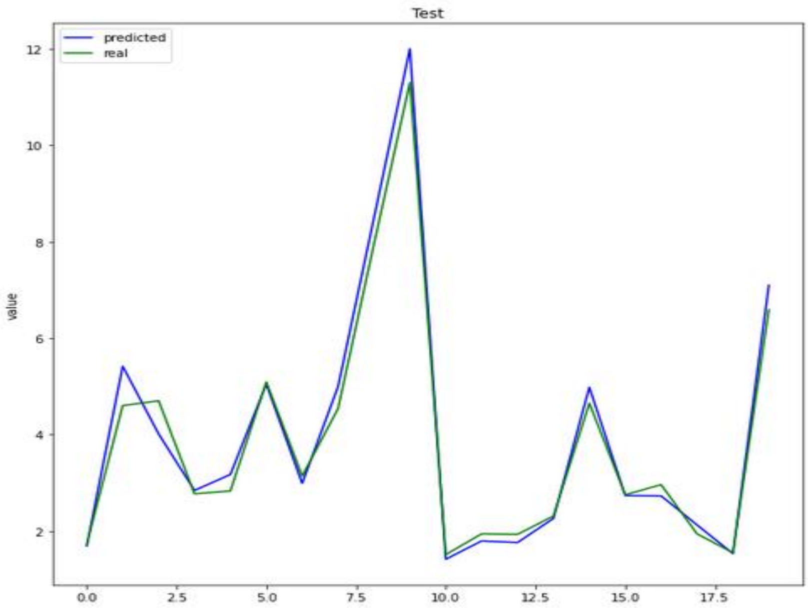

FIGURE 4.22. Results of train and test MLP.

In order to train and predict our MLP model, we have appeared the predicted and real graph, the values are wobbly; where the highest value in the predicted case is 12 and in real case takes 11, but the lowest value is 1 in both cases.

Discussion

As shown above, our model has been trained over different subjects and modalities, we have used the exam ids to identify which subject to use for both training and testing.

During the training and testing we see that the exams were mixed between healthy tissues and with abnormalities.

There are a relationship between RMSE and accuracy of model, whenever the value of RMSE is small, the accuracy of the model is high.

Conclusion

In this chapter, we have described the detailed design for our models system with the used tools for developing of this system. Then, we have also presented the different function required in our implementation. The experimental results were evaluated considering different performance metrics including: value of RMSE and R squared. The obtained results were very promising.

General conclusion

General conclusion

In this work, we presented what smart civil engineering and which components are necessary to make a Smart civil engineering.

According to the huge development of the civil engineering field and the technologies used; smart civil engineering is a thinking building, able to receive, process, save and diagnose the data. To apply this concept, should be connect buildings with internet using technology internet of things (IoT) to collect sensors information use a publish/subscribe; we have leveraged Edge Computing paradigm which works in tandem with Cloud Computing to deliver real-time applications for end devices with minimum latency and save the essential data in cloud computing, the large amount of data stored in the cloud computing is an input for machine learning algorithms which are able to provide embedded intelligence of IoT data.

We have explored the potential of IoT for monitoring and sensing the information of buildings in civil engineering, and ML for the prediction of IoT data.

In our system, contain two parts:

The first part, is the connectivity part, we connect civil engineering to internet using Internet of things (IoT) technology, with the protocol MQTT using broker mosquitto to receive the messages sent by the publishers and to forward them to the subscribers.

The second part, is the prediction part, for the prediction of results using machine learning (ML), we tried many methods of regression. Moreover, Linear regression and MLPRegressor have shown better results and performance than the other methods in testing score. We used Linear regression to predict the active earth pressure coefficients due to the soil weight, but MLPRegressor for the passive earth pressure coefficients due to the soil weight. The prediction RMSE of the proposed model LR is about 0.013 with R2 value 0.978, but about the second model MLPRegressor the value of RMSE is 1.31 with R2 is 0.60. In addition, different learning samples were used, the results showed that the proposed models required concrete training beforehand in order to give an acceptable predictive accuracy.

In conclusion, our approach provides the optimal solution based on IoT for the prediction of Civil engineering building systems using MQTT protocol and supervised regression of machine learning.

Perspectives

In future more studies on many other open questions involving investigations protocols and algorithms based on smart civil engineering should be investigated.

More development and more implementation can be possible to our system, with applying:

- Trying new broker, which supports more scalability and interoperability with the web
- New intelligent algorithms such as deep neural networks to improve accuracy of diagnosis, analyze big data efficiently and maximize utilization.

General conclusion

In this work, we presented what smart civil engineering and which components are necessary to make a Smart civil engineering.

According to the huge development of the civil engineering field and the technologies used; smart civil engineering is a thinking building, able to receive, process, save and diagnose the data. To apply this concept, should be connect buildings with internet using technology internet of things (IoT) to collect sensors information use a publish/subscribe; we have leveraged Edge Computing paradigm which works in tandem with Cloud Computing to deliver real-time applications for end devices with minimum latency and save the essential data in cloud computing, the large amount of data stored in the cloud computing is an input for machine learning algorithms which are able to provide embedded intelligence of IoT data.

We have explored the potential of IoT for monitoring and sensing the information of buildings in civil engineering, and ML for the prediction of IoT data.

In our system, contain two parts:

The first part, is the connectivity part, we connect civil engineering to internet using Internet of things (IoT) technology, with the protocol MQTT using broker mosquitto to receive the messages sent by the publishers and to forward them to the subscribers.

The second part, is the prediction part, for the prediction of results using machine learning (ML), we tried many methods of regression. Moreover, Linear regression and MLPRegressor have shown better results and performance than the other methods in testing score. We used Linear regression to predict the active earth pressure coefficients due to the soil weight, but MLPRegressor for the passive earth pressure coefficients due to the soil weight. The prediction RMSE of the proposed model LR is about 0.013 with R2 value 0.978, but about the second model MLPRegressor the value of RMSE is 1.31 with R2 is 0.60. In addition, different learning samples were used, the results showed that the proposed models required concrete training beforehand in order to give an acceptable predictive accuracy.

In conclusion, our approach provides the optimal solution based on IoT for the prediction of Civil engineering building systems using MQTT protocol and supervised regression of machine learning.

Perspectives

In future more studies on many other open questions involving investigations protocols and algorithms based on smart civil engineering should be investigated.

More development and more implementation can be possible to our system, with applying:

- Trying new broker, which supports more scalability and interoperability with the web
- New intelligent algorithms such as deep neural networks to improve accuracy of diagnosis, analyze big data efficiently and maximize utilization.