ALGERIAN DEMOCRATIC AND POPULAR REPUBLIC
Ministry of Higher Education and Scientific Research
Mohamed Khider University – BISKRA
Faculty of Exact Sciences, Natural and Life Sciences
Computer Science Department

Order N°: SIOD36-Startup/M2/2023

Dissertation

Presented to obtain the academic master's degree in

Computer Science

Option: Information Systems, Optimization and Decision (SIOD)

Cervical spine fracture detection using machine learning algorithm

By:
BENBRAHIM Hadjer

Supported on ../06/2023 before the jury composed of:

Berima Salima	MAA	President
Meadi Mohamed Nadjib	MCB	Advisor
Benchaabane Moufida	MAA	Examiner

Academic year 2022-2023

الإهـــداء

بسم الله والصلاة و السلام على رسول الله

﴿ رب أوزعني أن أشكر نعمتك التي أنعمت علي

وعلى والدي وأن أعمل صالحا ترضاه وأدخلني برحمتك في عبادك الصالحين﴾

- سورة النمل الاية 19 –

أبي أمي أهديكما هذه المذكرة بكل حب وتقدير، فأنتما هما ركن الدعم والإلهام في حياتي. لا يمكنني وصف مدى امتناني لكما على كل الحنان والتضحيات التي قدمتماها لي.

لأبي العزيز، أنت رمز للقوة والحكمة. بفضلك، تعلمت أهمية النجاح والعمل الجاد. كان لديك الصبر والتوجيه اللازمين لتشجيعي على تحقيق أحلامي وتجاوز العقبات. أشكرك على كل لحظة قضيتها معي وعلى القيم التي غرستها في داخلي.

وأما لأمي الحنونة، فأنتِ نجمتي الساطعة وحبيبتي الأبدية. بلطفك وعنايتك، زرعتِ في قلبي الحب والرحمة. لقد كنتِ الداعمة المستمرة والصديقة المخلصة، وكنتِ دائمًا هنا لتوجيهي وتشجيعي.

أنتما تجسيدٌ للحب والعائلة، ولا أستطيع أن أتخيل حياتي بدونكما. أدعو الله أن يمنحني القوة والحكمة لأكون فخورًا لكما، تمامًا كما أنا فخورة بكوني ابنتكما.

ابنتكما الممتنة،

هاجر

Acknowledgements

Praise be to Allah, Lord of the Worlds, and may prayers and peace be upon his Messenger. First of all, I thank Allah for his facilitation of this work and for giving me the strength, and knowledge to complete this work.

I want to express my appreciation to my supervisor, Professor ***Meadi Mohamed Nadjib****, for providing me with the courage and support I needed to take the necessary steps toward achieving my goal of graduating. His guidance and valuable contributions have made a significant impact on my work, and I am deeply grateful for that.*

I would also like to express my sincere gratitude to the members of the jury for their evaluation and assessment of this project. I want to take this moment to extend my thanks to everyone in the computer science department, particularly the faculty members.

Thank you dear brothers Munib, Khadija, Adam, and Anfal for your constant encouragement in anything I need. My greetings to everyone from my family ***Benbrahim*** *and* ***Bettayb****.*

My dear friends ***Maria****,* ***Kawthar****, and* ***Ines*** *wouldn't have gotten this far without you.*

Finally, I want to show my appreciation to those who I refer to as my "invisible soldiers" for their help and encouragement throughout this journey.

HADJER BENBRHAHIM

ABSTRACT

Artificial intelligence (AI) has revolutionized various fields of scientific research, including healthcare. In recent years, deep learning (DL) algorithms have been applied to medical imaging and have shown promising results in early disease detection and diagnosis. Cervical spine fractures are common injuries, and their early detection is crucial for successful treatment. With the recent developments in AI and DL, there is potential for automated detection and classification of cervical spine fractures using medical imaging. In this project, we aim to explore the potential of DL algorithms, specifically convolutional neural network (CNN) architecture, for the automated detection and classification of cervical spine fractures using medical imaging datasets. This project uses accessible platforms, such as Kaggle, for model training with various parameters and structures. The results of this project could significantly improve the early detection and treatment of cervical spine fractures, ultimately improving patient outcomes.

Key words: Healthcare, Cervical spine fractures, cervical spine dislocation, Artificial intelligence, Deep learning, Convolutional neural network, Medical imaging.

Résumé

Récemment, l'intelligence artificielle (IA) a été largement utilisée dans tous les domaines de recherche scientifique, y compris la santé, en raison de sa capacité à fournir des solutions efficaces. Les fractures de la colonne cervicale sont une condition médicale qui peut causer de graves problèmes de santé, et leur détection précoce est essentielle pour un traitement réussi. Le développement de l'apprentissage profond en IA fournit des solutions prometteuses pour prédire, prévenir et diagnostiquer différentes maladies, y compris les fractures de la colonne cervicale.Dans ce projet de master, nous avons exploré le potentiel de l'apprentissage profond pour l'analyse d'images de la colonne cervicale. Nous avons utilisé un algorithme de réseau neuronal convolutionnel (CNN) pour construire un modèle de classification multicouche capable de détecter automatiquement les fractures de la colonne cervicale. Nous avons entraîné notre modèle en utilisant Kaggle, une plateforme accessible à tous. Nous avons utilisé différentes bases de données pour notre analyse et appliqué une architecture CNN avec différents paramètres pour obtenir des résultats précis et fiables.

Mots clés: Soins de santé, fractures de la colonne cervicale, Intelligence artificielle, Apprentissage profond, Réseau de neurones convolutionnels, Imagerie médicale.

ملخص

في الآونة الأخيرة، غزى الذكاء الاصطناعي (AI) جميع مجالات البحث العلمي، وذلك بسبب ما يوفره من حلول وإمكانيات في مجال الصحة. تُعَد كسور العمود الفقري العنقي من بين الأمراض الشائعة في العالم وخاصة في الجزائر. حيث يمكن أن يؤدي هذا النوع من الكسور إلى حدوث تشوهات في الحركة والشعور بالتنميل، وفي بعض الحالات يمكن أن تؤدي إلى الشلل. ويمكن أن يساعد الكشف المبكر عن كسور العمود الفقري العنقي كثيرًا في عملية العلاج. وتوفر التطورات الحديثة في مجال الذكاء الاصطناعي، ولا سيما التعلم العميق، حلولاً يمكن استخدامها للتنبؤ والتشخيص المبكر للعديد من الأمراض.

في مشروعنا هذا، قمنا بدراسة إمكانات التعلم العميق (DL) في تحليل صور العمود الفقري العنقي. وقمنا بدراسة مفاهيم التعلم العميق باستخدام خوارزمية الشبكات العصبية التكرارية (CNN) لبناء نموذج تصنيف متعدد الفئات يمكنه الكشف عن كسور العمود الفقري العنقي وتصنيفها تلقائيًا. ولقد استخدمنا منصة Kaggle المتاحة للجميع لتدريب نماذجنا. وفي هذا المشروع، قمنا بتطبيق هندسة شبكة عصبية تكرارية تحتوي على عدة معلمات على مجموعات بيانات تتعلق بكسور العمود الفقري العنقي بأشكال مختلفة.

الكلمات المفتاحية:الرعاية الصحية، كسور العمود الفقري العنقي، الذكاء الاصطناعي، التعلم العميق، الشبكة العصبية التلفيفية، الصور الطبية.

Summary

List of Figures

List of Tables

General introduction

The early and accurate detection of cervical spine fractures is crucial for prompt diagnosis and appropriate medical intervention. These fractures can result from various traumatic events, such as motor vehicle accidents, falls, or sports injuries, and if left untreated, they can lead to severe complications, including paralysis or even death. Therefore, the development of advanced technologies to aid in the detection of cervical spine fractures has become a critical area of research.

Machine learning algorithms have shown tremendous potential in medical imaging applications, including the detection of fractures. By leveraging the power of artificial intelligence and pattern recognition, these algorithms can analyze medical images, such as X-rays or CT scans, to accurately identify and classify cervical spine fractures.

The aim of this graduation note is to explore and implement a machine learning algorithm specifically designed for the detection of cervical spine fractures. This algorithm will leverage a comprehensive dataset of labeled cervical spine images to train and validate its accuracy in identifying fractures. The ultimate goal is to develop a robust and efficient tool that can assist healthcare professionals in making accurate and timely diagnoses.

In this note, we will discuss the key steps involved in the development of the machine learning algorithm for cervical spine fracture detection. These steps include image preprocessing, feature extraction, training the algorithm using labeled data, and evaluating its performance on unseen data. Additionally, we will address the challenges and potential limitations associated with this approach, as well as strategies to overcome them.

Furthermore, we will provide an overview of existing research in the field of cervical spine fracture detection and highlight the advancements made by previous studies. By building upon the existing knowledge and incorporating state-of-the-art machine learning techniques, we aim to contribute to the growing body of research dedicated to improving the accuracy and efficiency of cervical spine fracture detection.

Problematic and motivation:

Cervical spine fractures can have devastating consequences if not identified and treated promptly. However, accurately diagnosing these fractures can be challenging, as they require careful examination of medical images, such as X-rays or CT scans, by skilled radiologists. The interpretation of these images is time-consuming and can be subject to human error, leading to potential misdiagnosis or delayed treatment.

The lack of a reliable and efficient system for cervical spine fracture detection poses a significant problem in the field of radiology. The current reliance on manual interpretation of medical images can lead to delays in diagnosis, increased healthcare costs, and potential patient complications. There is a need for an automated and accurate solution that can assist healthcare professionals in identifying cervical spine fractures in a timely manner, enabling appropriate medical intervention.

The motivation behind this graduation note is to develop a machine learning algorithm that can aid in the early detection of cervical spine fractures. By harnessing the power of artificial intelligence and pattern recognition, this algorithm has the potential to provide a fast and accurate automated system for cervical spine fracture diagnosis.

Dissertation structure

the dissertation is structured as follows:

- **Chapter 1 Theoretical Background:** This chapter discusses cervical spine fracture, as well as the various cervical spine fracture detection techniques, an overview of machine learning and deep learning approaches, as well as a depth look into convolutional neural networks and related works.

- **Chapter 2 System design:** This chapter describes the datasets and outlines the system with all the phases.

- **Chapter 3 Implementation and Results:** This chapter introduces the implementation tools, explains the code, and discusses the end result. also highlights key findings and offers proposals for further consideration.

Chapter 1

Theoretical Background

1.1 Introduction

In recent years, significant advancements in machine learning technology, particularly deep learning techniques, have shown great potential in enhancing the accuracy and efficiency of detecting cervical spine fractures. Cervical spine fractures are a common injury with severe consequences, including paralysis or even death. The conventional diagnostic approach relies on imaging studies such as X-rays, CT scans, or MRI scans, while treatment options range from conservative measures to surgical intervention. This chapter provides an overview of the human body's skeletal structure, focusing on the bones, the structure and function of the vertebral column, and specifically the role of the cervical spine. It also delves into the definitions, causes, types, traditional diagnostic methods, and available treatment options for cervical spine fractures. Additionally, the chapter explores the application of machine learning algorithms in the detection of cervical spine fractures. By leveraging advancements in AI, particularly deep learning, we aim to enhance the detection and management of cervical spine fractures, ultimately leading to improved patient outcomes.

1.2 Bones of human body

Bones are an essential human body element that provides support, structure, and protection. This section presents the definition and type of bones in the human body.

1.2.1 Definition

Bones are solid organs that provide support, protection, and mobility to the body, acting as a framework for attaching muscles and allowing physical activity and overall movement[17].

Bones also play a crucial role in mineral storage, particularly of calcium and phosphate, and in the production of blood cells through the bone marrow. Bone tissue is composed of several types of tissues[18], including bone tissue, cartilage, connective tissue, epithelium, adipose tissue, and nervous tissue. Bones undergo a process of remodeling to maintain their strength and shape, which is regulated by hormones and mechanical stresses. According to Patton, "The remodeling of bone is a lifelong process that occurs as a response to mechanical stress on the skeleton and hormonal regulation of calcium homeostasis" [19].

Bone structure varies by shape and internal composition, with five main categories of bones - long, short, flat, Sesamoid, and irregular - and two main types of bone tissue - spongy and compact.

1.2.2 Type of bones

The human body of the structural system consists of a total of 206 bones. As we can see in figure 1.1 that can be categorized into five primary types as [20]:

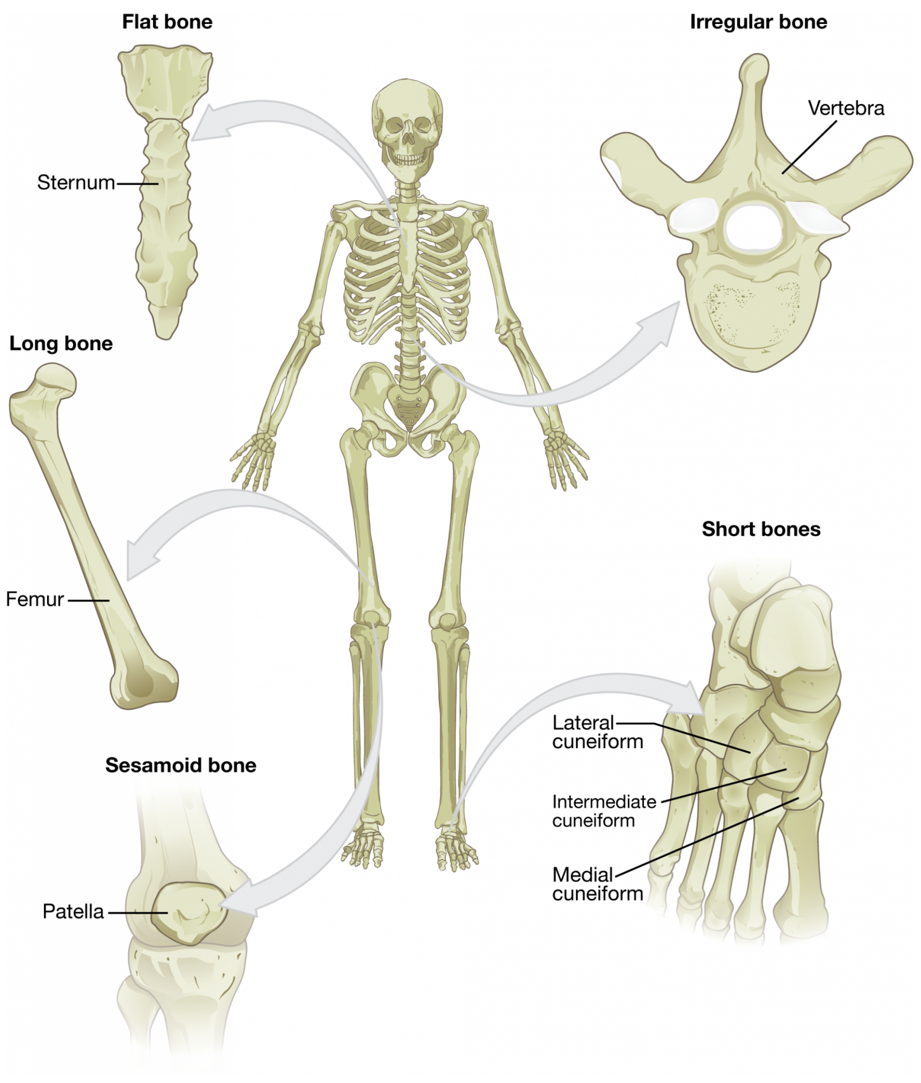


Figure 1.1: Classifications of the Bones[1].

1. Long bones: for instance, the femur and humerus, act as levers to assist body movements and are characterized by being longer in length than width.

2. Short bones: Providing support and stability, short bones, like the ones located in the ankles and wrists, possess similar dimensions in terms of length, width, and thickness.

3. Flat bones: Thin and designed to safeguard internal organs, flat bones, like the ones found in the skull and ribcage, provide protection.

4. Irregular bones: With intricate shapes and versatile roles, irregular bones, for ex-

ample, the vertebrae and hip bones, provide support to the body's weight while also safeguarding the spinal cord.

5. Sesamoid bone: Sesamoid bones are small and round bones found within tendons near joints. They function as pulleys, altering the direction of tendon pull and protecting it from damage.

1.3 Vertebral column of the human body

The vertebral column, commonly known as the spine, is an intricate and essential component of the human body. It is responsible for providing structural support and performing vital functions. Let's delve deeper into its definition, structure, and functions.

1.3.1 Definition

The spine, also known as the vertebral column, is a remarkable and intricate structure composed of bones and cartilage that extends from the skull to the coccyx. It exhibits a remarkable pliancy and is divided into individual segments that work together to provide crucial support and stability to the body. Additionally, it serves as a protective shield for the delicate spinal cord, which runs through a canal that traverses the spine.[21].

Comprising a total of 33 individual vertebrae, the vertebral column demonstrates a remarkable level of intricacy. Each vertebra is separated by intervertebral discs, which act as cushions and facilitate flexibility. The spine is further organized into five distinct regions, each with its own important functions and contributions to the overall structure. These regions not only offer primary support for the body but also ensure the safeguarding of the spinal cord, while simultaneously serving as anchor points for the attachment of muscles and ligaments [22].

The vertebral column can be described as an ingenious arrangement of individual vertebrae and intervertebral discs, working in harmony to uphold and protect the spinal cord. Moreover, this sophisticated structure enables a wide range of bodily movements, allowing for flexibility and mobility. Through its segmentation into multiple regions, each characterized by its unique attributes and roles, the spine accomplishes its diverse functions with remarkable efficiency and adaptability. The understanding of the spine's anatomy and functionality is of utmost importance in various fields, including anatomy, orthopedics, neurology, and physical therapy. Its significance lies in its pivotal role in providing structural support, protecting vital neural pathways, and facilitating the intricate movements required for daily activities and overall well-being [23].

1.3.2 Structure and functions of vertebral column

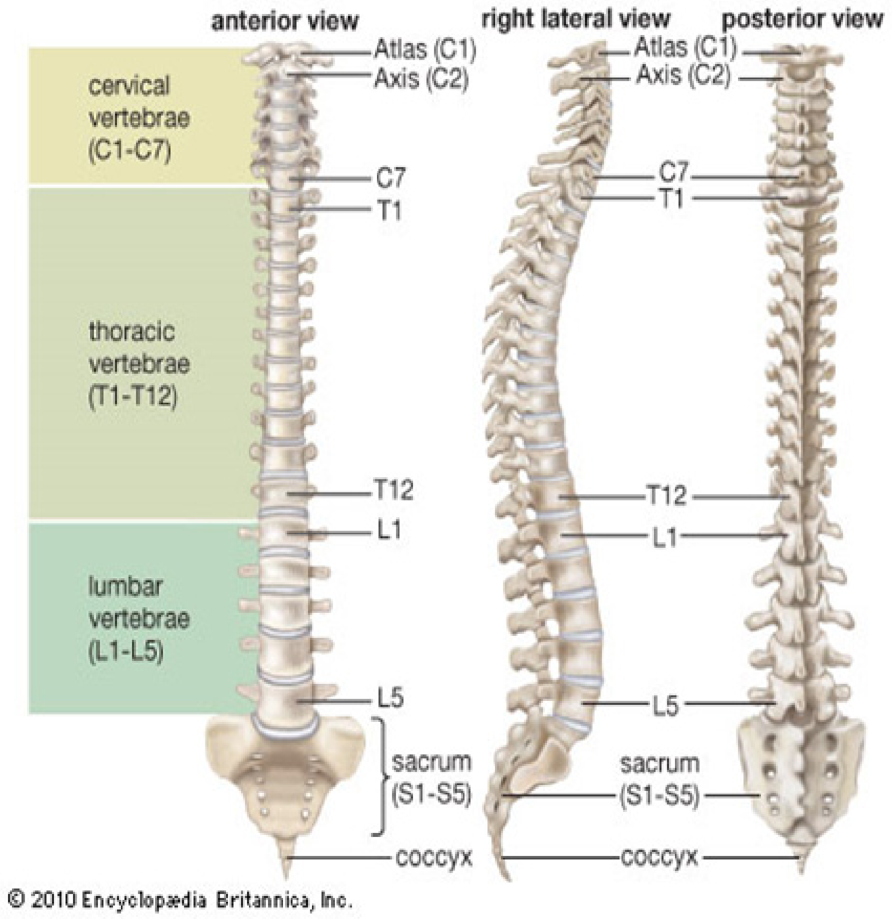


Figure 1.2: The vertebral column of the human [2].

As we can see in figure 1.2 that can be categorized vertebral column into five types as[24]:

1. Cervical spine (neck): This region is located in the neck and consists of 7 vertebrae .it allows for a high degree of mobility and flexibility in the neck. It also provides support for the head and protects the spinal cord.

2. Thoracic spine (upper back): This region is located in the upper back and attaches to the ribcage and consists of 12 vertebrae. It is relatively immobile. It helps to support the upper body's weight and provides attachment points for the ribs.

3. Lumbar spine (lower back): This region is located in the lower back and consists of 5 large vertebrae. It is responsible for bearing the majority of the body's weight. It also provides support for the upper body and allows for limited mobility and flexibility.

4. Sacrum: This triangular bone is located at the base of the spine and consists of

5 vertebrae that are fused together. It provides a strong foundation for the spine and helps to transfer weight from the upper body to the pelvis and legs.

5. Coccyx (tailbone): This small triangular bone is located at the very bottom of the spine and is made up of 3 to 5 small vertebrae that are also fused together. It provides a point of attachment for the muscles and ligaments of the pelvis.

1.4 Cervical spine of human body

In our study, we will focus specifically on the cervical spine, which is located in the upper region of the vertebral column. The cervical spine consists of seven individual vertebrae, labeled C1 to C7, and is situated between the skull and the thoracic spine.

1.4.1 Definition

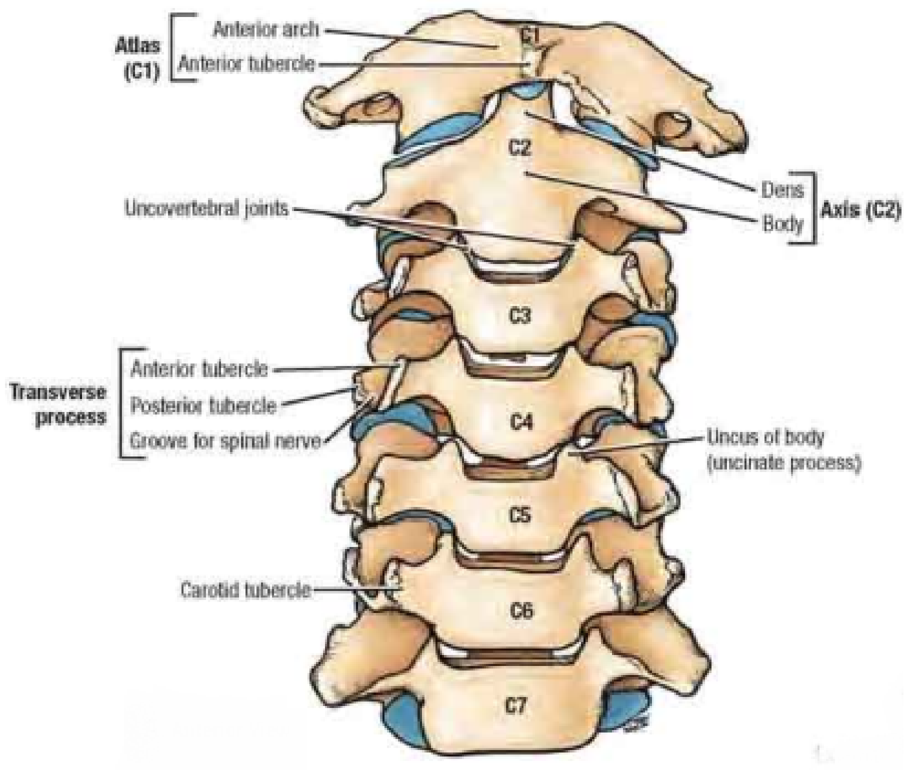


Figure 1.3: Cervical spine of the human [3].

The cervical spine is the uppermost section of the vertebral column, located in the neck region, and is made up of seven cervical vertebrae. As figure 1.3 indicates, these vertebrae are numbered C1 through C7 and are smaller and more delicate than those in other regions of the spine [25].

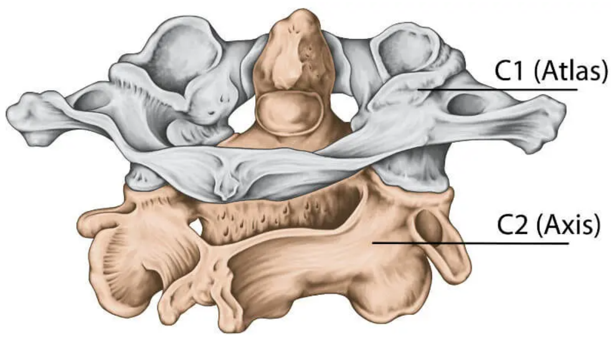


Figure 1.4: C1 and C2 vertebrae [4].

Each cervical vertebra has a unique structure that allows for specific functions. The first cervical vertebra, C1 or the atlas, is unique in that it does not have a body like other vertebrae. Instead, it has two large bony arches that surround the spinal cord and support the weight of the skull. The second cervical vertebra, C2 or the axis, has a distinctive bony protrusion called the odontoid process that extends upward from the body of the vertebra. As figure 1.4 indicates, this structure allows the atlas to pivot around it, providing a wide range of motion for the head[26].

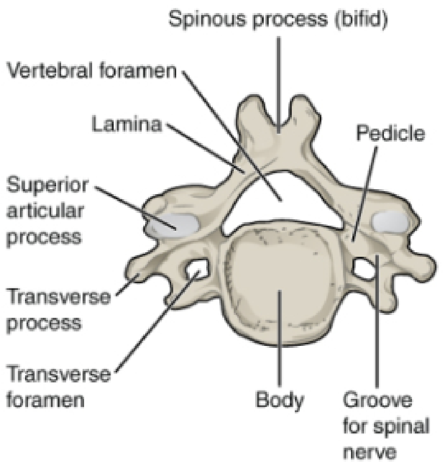


Figure 1.5: Structure of a typical cervical vertebra [5]

The remaining cervical vertebrae, As figure1.4 C3 through C7 are characterized by their smaller size, triangular shape, and presence of transverse foramina. These foramina are openings in the side of the vertebrae that allow for the passage of the vertebral artery and vein, which supply blood to the brain. The cervical vertebrae also have a shallow concave surface on their superior surfaces that form the cervical curve, which helps to absorb shock and distribute weight [27][17].

In addition to each cervical vertebra's unique structure, there are differences in the thickness and length of the spinous processes and the shape and size of the vertebral foramen. These variations allow for differences in the function and mobility of each vertebra [17].

The cervical spine and its components play a critical role in supporting the head and allowing for movement and flexibility in the neck. The unique structures and differences between each cervical vertebra allow for specific functions and contribute to the overall function of the cervical spine as a whole.

1.4.2 Role of cervical spine

The cervical spine has several important roles, including:

1. Protecting the spinal cord: The cervical spine encases and protects the spinal cord, which is a crucial part of the central nervous system[28].

2. Supporting the head: The cervical spine provides strong support for the weight of the head, which can weigh up to 10-13 pounds. This support helps maintain proper posture and prevent strain on the neck muscles [21].

3. Facilitating movement: The cervical spine allows for a wide range of movement, including flexion (forward bending), extension (backward bending), lateral flexion (side bending), and rotation (turning) [29].

4. Enabling sensory input and motor output: The spinal cord passes through the cervical spine, allowing for the transmission of sensory information from the body to the brain and motor commands from the brain to the body. As we can see in figure 1.5, The dermatomes of the cervical spine refer to the areas of skin that are innervated by the sensory nerves exiting from the cervical spinal nerves. There are eight cervical spinal nerves, each of which innervates a specific area of skin [3].

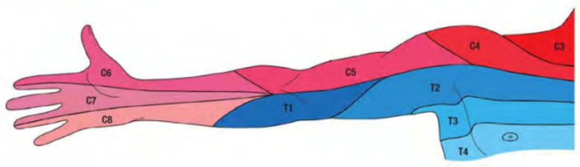


Figure 1.6: Dermatome maps of cervical spine [3].

The cervical spine is not only responsible for supporting the head and facilitating movement but also plays a critical role in protecting the spinal cord and maintaining proper posture. Its health and functionality are paramount for overall well-being and the smooth functioning of the body.

1.5 Cervical spine fracture

The following elements will define cervical spine fracture, including its causes, types, traditional diagnosis, and treatment.

1.5.1 Definitions

A cervical spine fracture refers to a break or cracks in one or more of the seven cervical vertebrae, which form the neck region of the spine. These fractures can vary in severity, ranging from minor cracks to severe damage that affects the spinal cord and surrounding nerves[30]. The severity of a cervical spine fracture depends on factors such as the extent of the break and its effect on the vertebrae.

The severity of cervical spine fractures depends on factors such as the extent of the break and its impact on the vertebrae. Compression fractures occur when a vertebra collapses, burst fractures involve the shattering of the vertebra into multiple pieces, and dislocations result in the displacement of bones from their normal positions [31].

Cervical spine fractures can lead to various complications, including spinal cord injury, nerve damage, and neck instability. Symptoms experienced may vary based on the location and severity of the fracture but can include neck pain, restricted mobility, numbness or weakness in the arms or legs, difficulty breathing, and coordination difficulties [32].

1.5.2 Causes of cervical spine fracture

The most common cause of cervical spine fractures is trauma, which can occur as a result of various types of accidents, including car crashes, falls, and sports injuries. Traumatic cervical spine fractures are most commonly seen in young adults and can range in severity from minor fractures to complete spinal cord injury. Other less common causes of cervical spine fractures include degenerative changes in the spine and underlying medical conditions that weaken the bones, such as osteoporosis and cancer [33].

It is important to note that cervical spine fractures are serious injuries requiring immediate medical attention. Prompt diagnosis and appropriate treatment are crucial for preventing further damage and promoting optimal recovery. The management of cervical spine fractures often involves a multidisciplinary approach, with collaboration between orthopedic surgeons, neurologists, radiologists, and physical therapists to ensure comprehensive care for the patient [32].

1.5.3 Types of cervical spine fractures

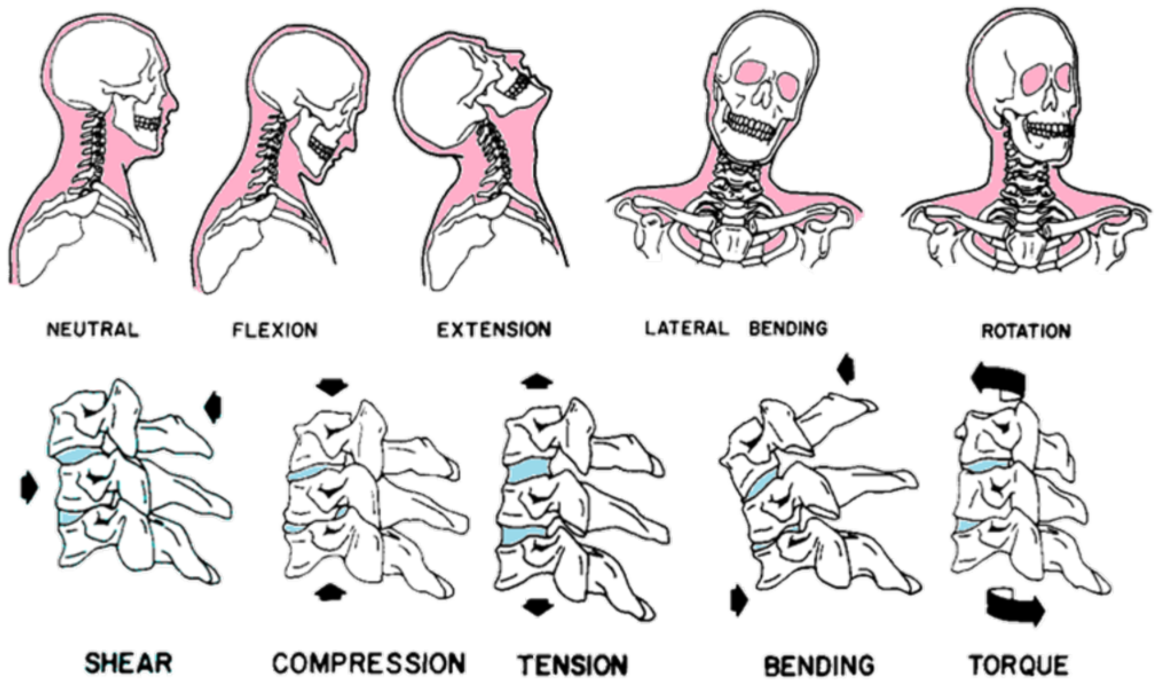


Figure 1.7: types of fractures [6]

There are several types of cervical spine fractures, each with its own unique characteristics and treatment considerations. As we can see in figure 1.7 these include [34][35]:

1. Flexion injuries: These occur when the head is forced forward, causing the spine to bend forward beyond its normal range of motion. This can result in compression fractures or dislocations of the vertebrae.

2. Extension injuries: These occur when the head is forced backward, causing the spine to bend backward beyond its normal range of motion. This can result in fractures or dislocations of the vertebrae.

3. Compression injuries: These occur when the spine is compressed due to a high-impact force, such as a fall or car accident. This can result in compression fractures or burst fractures of the vertebrae.

4. Rotation injuries: These occur when the spine is twisted or rotated beyond its normal range of motion, often resulting in fractures or dislocations of the vertebrae.

5. Shear injuries: These occur when the spine is subjected to a shear force, which can cause a vertebra to slide or shift relative to the adjacent vertebra. This can result in fractures, dislocations, or other types of spinal instability.

Classification of spinal injuries based on the essential traumatic spinal mechanisms is important for determining the appropriate treatment and predicting the potential complications associated with a particular type of injury.

1.5.4 Traditional diagnosis

Cervical spine fractures and dislocations are common injuries that should always be suspected when a patient has experienced trauma or an accident, particularly if they complain of neck pain. However, diagnosis can be challenging, especially when symptoms or physical findings are atypical. In some cases, patients with other fractures or injuries may report pain in other areas but not complain of neck pain, which can complicate diagnosis[30].

In some situations, patients may downplay the severity of their trauma, which could sway clinicians away from ordering cervical X-rays and imaging studies. However, these imaging tests are crucial in accurately diagnosing cervical injuries. Therefore, clinicians must conduct a thorough history and clinical examination, especially inspection and palpation of the spine, before formulating a diagnosis to avoid misdiagnosis. In any patient involved in a severe accident or trauma, especially those with neck pain, it is crucial to carefully evaluate them with x-rays and additional imaging studies, if necessary, to accurately diagnose a cervical injury[30].

Diagnosis of cervical spine fractures typically involves a combination of imaging tests, such as X-rays, CT scans, and MRI scans, along with a physical examination and assessment of neurological function. Treatment options will depend on the type and severity of the fracture, as well as the patient's overall health and other factors. Therefore, a personalized treatment plan must be created based on the patient's specific needs and circumstances [36].

1.6 Limitations of traditional diagnosis cervical spine fractures

Traditional diagnosis of cervical spine fractures involves a physical examination, imaging tests, and sometimes invasive procedures. However, there are some limitations to these traditional methods, such as:

- **Lack of sensitivity:** Physical examination alone may not be sufficient to detect all cervical spine fractures, particularly when there are no apparent external signs of injury. This means that relying solely on a physical examination may result in missed diagnoses [37].

- **Delayed diagnosis:** The process of scheduling and conducting imaging tests can introduce delays in diagnosing and treating cervical spine fractures. This delay can impact patient outcomes, as timely intervention is crucial for optimal recovery [37].

- **Invasive procedures:** Certain imaging tests, such as myelography or discography, require invasive procedures that can be uncomfortable for patients. Moreover, these procedures carry a risk of complications, further adding to the potential drawbacks and limitations of relying solely on invasive imaging techniques [38].

- **Radiation exposure:** Imaging tests like X-rays and CT scans involve exposure to ionizing radiation. While these tests provide valuable diagnostic information, repeated exposure to radiation can have harmful effects on the body, including an increased risk of developing radiation-related health issues [39].

- **Cost:** The cost of imaging tests can be a limiting factor, as some of these procedures can be expensive. This financial barrier may restrict the availability and accessibility of these tests for certain patients, potentially impacting their ability to receive timely and accurate diagnoses [38] [39].

These limitations highlight the need for alternative methods of diagnosis, such as the use of artificial intelligence and machine learning algorithms, which may provide a more accurate and efficient way of detecting cervical spine fractures.

1.7 Machine learning

Machine learning (ML) is a rapidly evolving field that was first described by Arthur Samuel in 1959 as the "field of study that gives computers the ability to learn without being explicitly programmed."

Machine learning refers to a branch of artificial intelligence where algorithms and models are developed to help computers enhance their performance on a given task over time. This is done by inputting substantial amounts of data into the algorithms to allow them to recognize patterns and make predictions or decisions based on the patterns they identify. Applications of machine learning span various fields, including image and speech recognition, natural language processing, and predictive analytics..[40]

ML algorithms can be categorized into four groups, namely, supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning, based on the underlying mappings between input data and anticipated output presented during the learning phase. [41]

1. Supervised learning is a mechanism that identifies input data and a targeted variable subject to prediction. Linear Regression, Logistic Regression, Naïve Bayes, KNN, and deep learning are some of the techniques used in supervised learning.[41]

2. Unsupervised learning algorithms, on the other hand, are designed to discover hidden structures in unlabeled datasets. Apriori, K-means, SVM, Anomaly De-

tection, and Principal Component Analysis (PCA) are some of the unsupervised learning techniques used to identify patterns and relationships in datasets where the desired output is unknown.[41]

3. Semi-supervised learning is a class of machine learning techniques that utilize both labeled and unlabeled examples when learning a model. This methodology can produce considerable improvement in learning and operates between the guidelines of unsupervised and supervised learning.[41]

4. Reinforcement learning is a type of machine learning that involves an agent performing certain actions in an environment to maximize the reward in a particular situation. The learning technique synthesizes an adaptation model by training itself for a given set of experimental actions and observed responses to the state of the environment.[41]

1.8 Deep learning algorithms

Deep learning is a type of machine learning that uses neural networks with multiple layers to learn from data. It is particularly effective for complex problems with large amounts of data, such as image or speech recognition. Deep learning models can automatically learn features from the data, reducing the need for manual feature engineering.

Deep learning algorithms can be classified into various types based on their architecture and functionality. Three of the most popular types of deep learning algorithms are Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Generative Adversarial Networks (GANs).

1. Convolutional Neural Networks (CNNs) are commonly used for image and video analysis tasks. They consist of multiple layers of convolutional and pooling layers that help to identify and extract features from the input data. CNNs are widely used in detecting cervical spinal fractures from medical images.[42]

2. Recurrent Neural Networks (RNNs) are used to process sequential data, such as speech or text. They use a feedback mechanism that allows information to persist through the network, making them useful for tasks such as language translation and speech recognition.[42]

3. Generative Adversarial Networks (GANs) consist of two neural networks: a generator and a discriminator. The generator generates fake data that is then evaluated by the discriminator to determine whether it is real or fake. This feedback loop helps the generator to improve its output and generate more realistic data.[42]

Each of these deep learning algorithms has its own strengths and weaknesses, and they can be combined to create more powerful models. The choice of algorithm depends on the nature of the problem being solved and the type of data being used.

1.8.1 Artificial neural networks

To understand artificial neural networks first, it's essential to comprehend the fundamental principles of biological neural networks and the modifications incorporated into the artificial neural network model.

1.8.1.1 Biological and artificial neuron

This is a biological cell called a neuron (shown in Figure1.8). The neuron has a nucleus and a cell body, which are divided to form dendrites. The dendrites receive information from the outside world and send it to the cell body. The cell body then processes the information and sends it out through an axon to other neurons. The transmission between two neurons happens through a tiny gap called a synapse, which is about 10^{-9} in size [43] [44].

Each artificial neuron (as shown in Figure 1.8) is a simplified version of a biological neuron and serves as a basic processor. It receives input from a set of variables,$X = \{x1, x2, x3, ..., xn\}$, referred to as the input layer, where each input is assigned a weight, w, representing the connection value. The weighted sum of the inputs and their corresponding weights,$\sum_{i=1}^{n}(wixi)$ is transformed by an activation function, f, to produce an output value that is then compared to a threshold value in the output layer to provide a response[43][7].

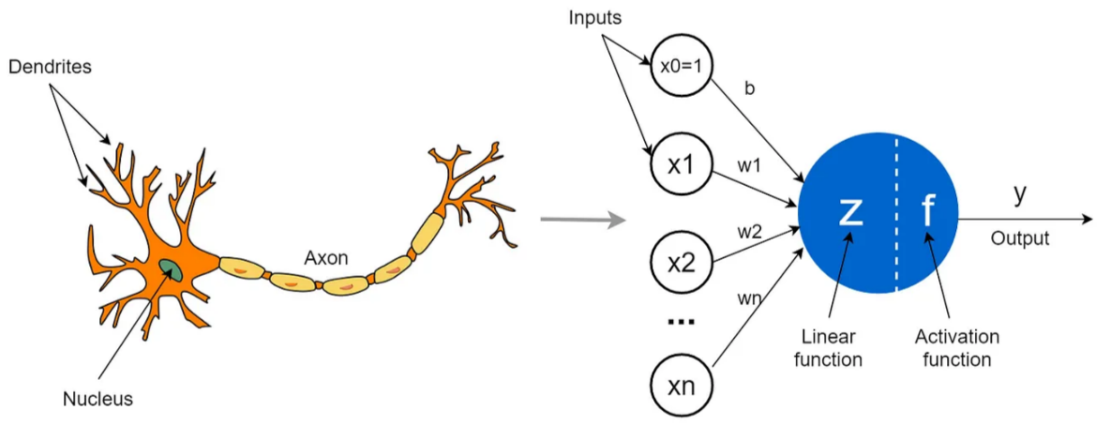


Figure 1.8: Biological and artificial neuron [7].

1.8.1.2 Artificial neural networks

The multi-layer artificial neural network comprises three layers of neurons - an input layer, a hidden layer, and an output layer. This upgraded neural network is designed to tackle more intricate problems than its predecessor. The artificial neural network has 3 types of layers (see figure 1.9).

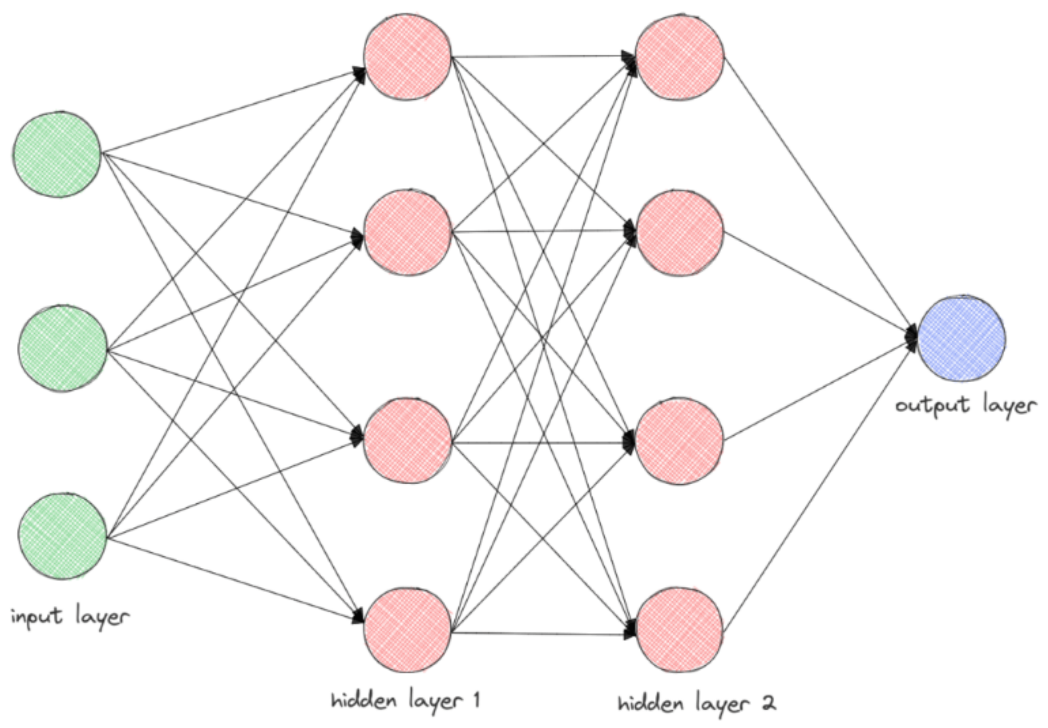


Figure 1.9: artificial neural network [8]

1. **Types of layers:**

   - Input layer: The initial layer of a neural network is known as the input layer, which consists of input neurons that receive data.[8]

   - Hidden layer: The intermediate layer(s) situated between the input and output layers are referred to as hidden layer(s), responsible for solving problems. The number of hidden layers required is determined by the complexity of the problem.[8]

   - Output layer: The final layer of an artificial neural network is the output layer, responsible for generating the program's output. In the case of classification problems, the size of the output layer's neurons will match the number of classes.[8]

2. **Activation function:**

   We mentioned the activation function in the last subsections, in this subsection, we will detail more about the activation function. If a neural network does not have an activation function, it is similar to a basic linear regression model. The activation function changes the input in a non-linear way, allowing the neural network to learn and perform complex tasks.

(a) Sigmoid function:
This function maps any input to a value between 0 and 1, making it useful for binary classification problems. The output of the sigmoid function has a smooth gradient, making it helpful in avoiding vanishing gradient problems. However, its output is not zero-centered, which can make training slower[9].
The curve of the Sigmoid function is in Figure 1.10.

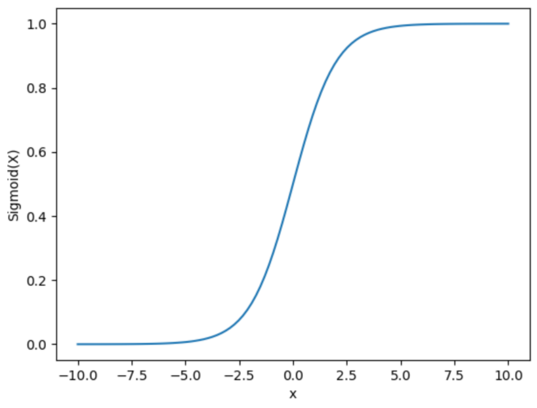


Figure 1.10: Sigmoid function curve [9]

(b) Rectified linear unit (ReLU):
ReLU maps any input to the maximum of 0 and the input itself. It is the most popular activation function due to its simplicity and efficiency in training deep neural networks. ReLU helps in preventing the vanishing gradient problem, and since it is zero for all negative values, it is computationally efficient[9].
The curve of the ReLU function is in Figure 1.11.

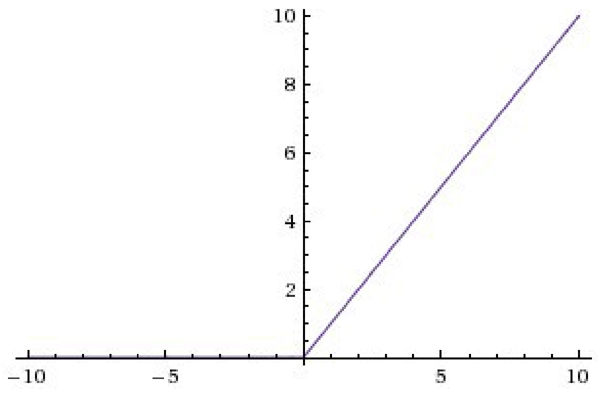


Figure 1.11: ReLU Function curve [9]

(c) Hyperbolic tangent (tanh):
This function maps any input to a value between -1 and 1, making it use-

ful for binary classification problems. It is similar to the sigmoid function, but its output is zero-centered, which helps in faster convergence during training[9].
The curve of the tanh function is in Figure 1.12.

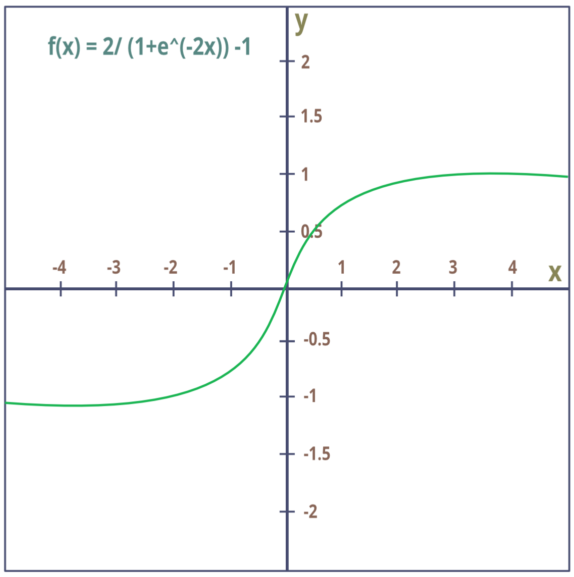


Figure 1.12: tanh function curve [9].

(d) Softmax function:
Softmax maps a vector of real numbers to a probability distribution, which makes it useful for multiclass classification problems. It outputs values between 0 and 1, and the sum of all outputs is equal to 1. The Softmax function is often used in the output layer of neural networks to compute the probabilities of different classes[9].
The curve of the Softmax function is in Figure 1.13.

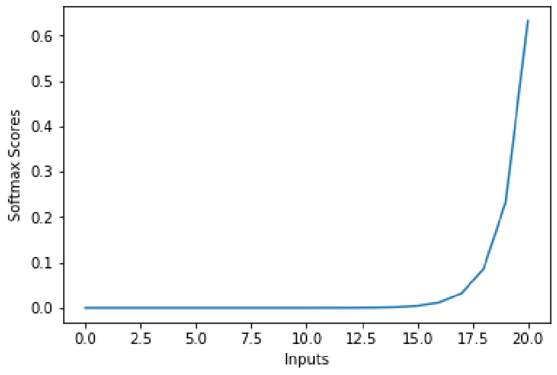


Figure 1.13: Softmax function curve [9].

1.8.2 Convolutional neural network

Convolutional neural networks (CNNs) are a specialized type of neural network primarily used for image processing tasks. While they share some similarities with regular neural networks, CNNs have a specific design tailored for handling images. The architecture of CNNs consists of two main parts: convolutional layers and fully connected layers, each serving different purposes [10]. Figure 1.14 illustrates this architecture.

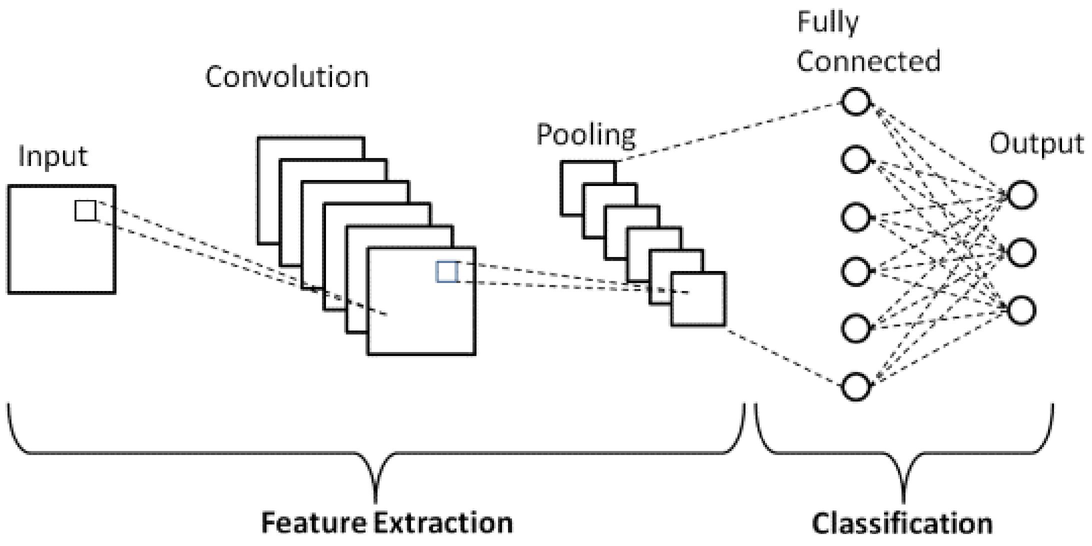


Figure 1.14: Convolutional neural network composition [10].

The following are definitions of different layers shown in the above architecture Figure 1.9:

1. Convolutional layer
   The first part of the neural network is called the "convolutional layer." This layer takes the input data and applies some mathematical operations to it. The results are then sent to the next layer. An example of these operations is shown in Figure 1.15.

   Each neuron in the convolutional layer only looks at a small part of the results from the previous layer. This is done by multiplying the results with a "kernel." The group of results that a neuron sees is called its "receptive field" [11].

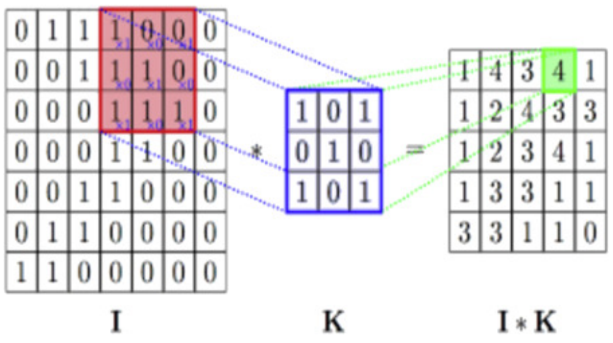


Figure 1.15: Convolutional layer [11]

2. Pooling layer
   The second key part is the "pooling layer". It merges the outputs of the previous layer into a single neuron. Two common types of pooling are average pooling and max pooling as shown in Figure 1.16. Average pooling calculates the average of the input values, while max pooling selects the largest value [11].

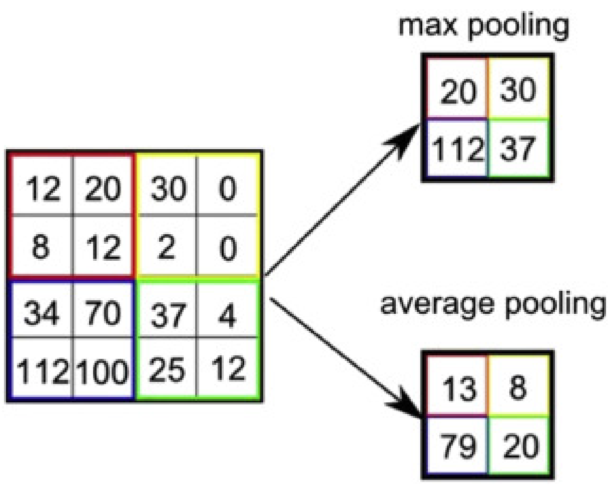


Figure 1.16: Pooling layer[11].

3. Fully connected layer
   The third type of layer in neural networks is called the "fully connected" layer. In these layers, each neuron is connected to all of the neurons in the next layer, which is similar to how traditional artificial neural networks work[11]. You can see an example of fully connected layers in Figure 3.23.

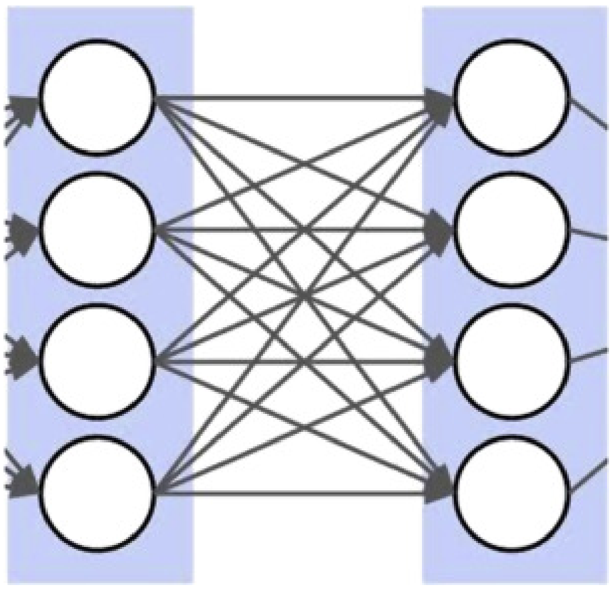


Figure 1.17: Fully connected layer [11].

concepts are important for understanding and configuring CNN models effectively, enabling efficient training and optimizing the performance of the network for specific tasks:

- **The input shape (width, height, channels):** describes an image, where the first two values represent the image's width and height, and the third value represents the number of channels in the image [45].

- **The learning rate:** The learning rate is a changeable setting with a moderate positive value, typically ranging from 0.0 to 1.0, and is employed in training neural networks.
  The learning rate is a factor that decides the speed of the model's adjustment. It is set as a fixed value (usually quite small) to promote a smooth and gradual update of the weights (preventing large jumps and unpredictable actions) [46].

- **Optimizers in CNN:** Optimizers are methods that tweak the properties of neural networks, like weights and learning rate, to minimize losses. The optimizer in a CNN learning model can enhance the model's results and decrease the training time from days to hours or minutes. Some of the various optimizers include Gradient descent optimizer, Stochastic gradient descent, Mini batch gradient descent, RMSprop optimizer, and Adam optimizer [46].

- **Loss function:** In order to minimize the errors in the algorithm, a loss function is employed to evaluate the performance of the model by measuring how accurately it predicts the outcome [45].

- **Epoch:** An epoch refers to a complete cycle of transmitting the entire dataset through the neural network algorithm, both forward and backward. It is considered complete when the algorithm has processed all the samples in the dataset [45].

- **batch size:** a parameter that determines the number of training or validation examples included in each batch during the training process [46].

1.9 Transfer learning

Transfer learning is a machine learning technique that involves taking a pre-trained model and fine-tuning it for a new task. In transfer learning, the knowledge learned by a pre-trained model is transferred to a new task, which can save time and computational resources. Transfer learning is particularly useful in deep learning, where large models are often trained on massive datasets. By fine-tuning a pre-trained model, it is possible to achieve high performance on a new task with a relatively small amount of data. For example, a pre-trained image classification model can be fine-tuned for object detection, where the model learns to detect specific objects in images. Transfer learning has become a popular technique in various domains, including computer vision, natural language processing, and speech recognition [47].

There is an infinite number of architectures, we will describe some of the most popular architectures like VGGNet16 and ResNet50V2.

- **VGG16** is a widely used convolutional neural network for image classification. It is composed of 16 layers, including convolutional and pooling layers, that use small 3x3 filters to extract features from input images. The architecture is straightforward and consistent, with stacked convolutional layers, and it employs max pooling to reduce the spatial dimension of the feature maps. With its depth, VGG16 can capture complex patterns in images and be regarded as state-of-the-art for ImageNet classification in 2014. Nevertheless, it has a high number of parameters, which can increase the computational cost of training. VGG16 is frequently used as a baseline model for comparing other architectures, and it is suitable for transfer learning, where the pre-trained model is adapted for specific tasks [12]. Figure 1.18 displays the architecture of VGG16.

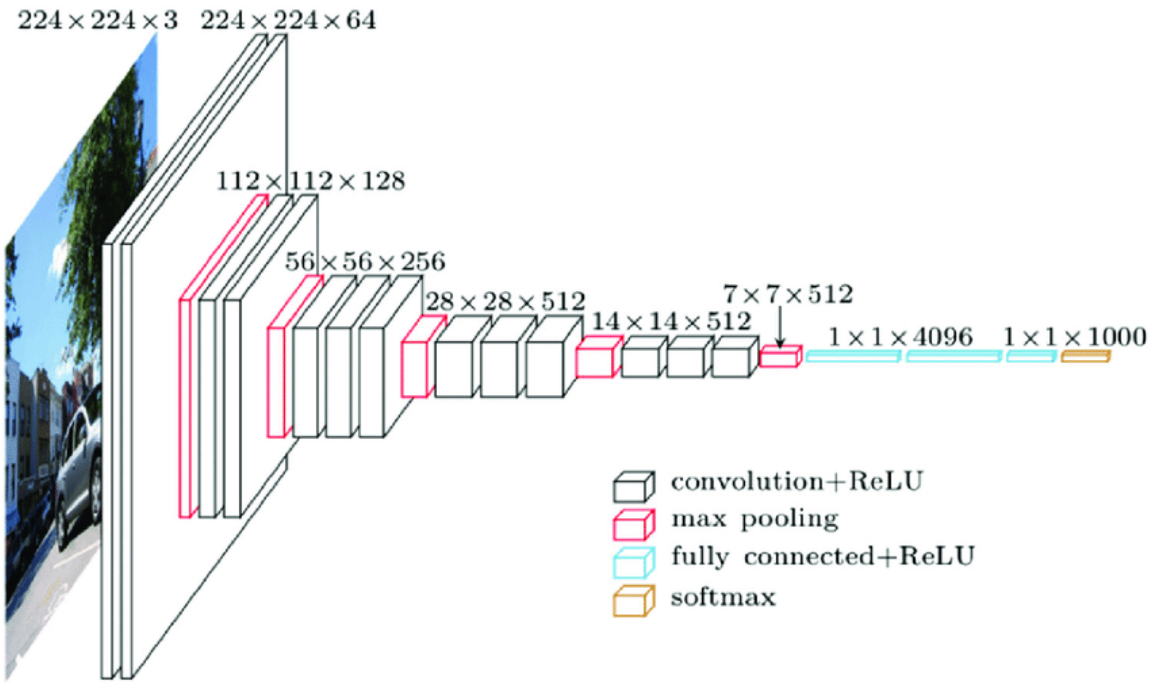


Figure 1.18: Basic architecture of VGG16 [12].

- **ResNet50V2** is a model used for computer vision tasks involving images. It is composed of 50 layers and utilizes residual connections to facilitate information transfer between layers. ResNet50V2 is an improvement over the original ResNet50, and it includes bottleneck blocks that help reduce the computational complexity of the network. The model achieves state-of-the-art performance on various benchmark datasets and is commonly employed as a pre-trained model for transfer learning. ResNet50V2 is a deep neural network that uses layers of neurons to process information, and it is primarily used in research and industry for tasks such as image classification, object detection, and semantic segmentation [13]. Figure 1.19 displays the architecture of ResNet50V2.

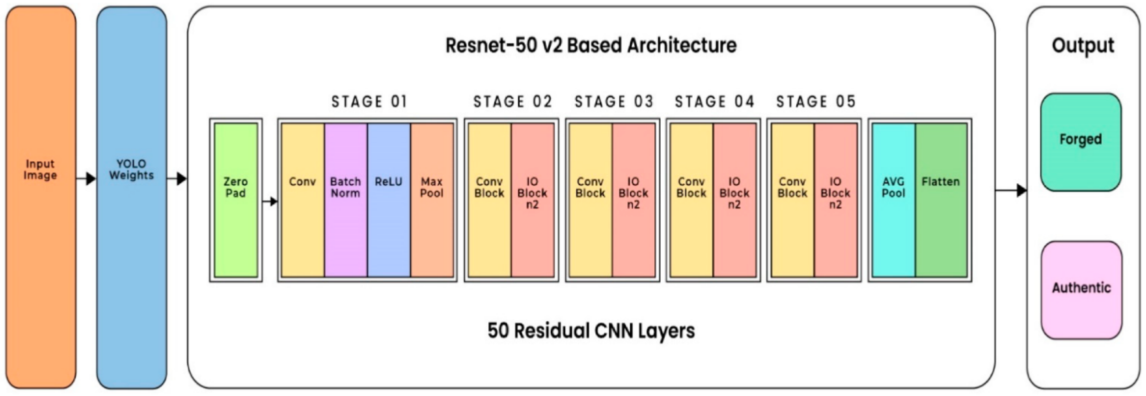


Figure 1.19: Basic architecture of ResNet50V2 [13].

1.10 Related works

In this part, we shall discuss some of the earlier works that have used machine learning algorithms for cervical spine fracture detection.

1.10.1 Classification of cervical spine fracture and dislocation using refined pre-trained deep model and saliency map

Soaad M. et al. [14] proposed this research article. This study focuses on leveraging a refined, pre-trained deep learning model combined with saliency maps to enhance the classification performance of cervical spine fractures and dislocations. By using a pre-trained deep learning model, the authors benefit from the transfer learning capabilities and feature extraction capabilities of the model, which enable effective representation learning from a large dataset. The authors trained the model using a dataset of 2009 X-ray images, consisting of 530 CS dislocation images, 772 CS fracture images, and 707 normal images. The results of the model showed high accuracy, sensitivity, specificity, and precision values. The materials and methods section describes the use of deep neural networks, specifically AlexNet and GoogleNet, for the proposed model (see Figure 1.20). Transfer learning is employed to fine-tune these pre-trained models on the spine dataset. The AlexNet and GoogleNet architectures are briefly explained, highlighting their layer configurations and activation functions.

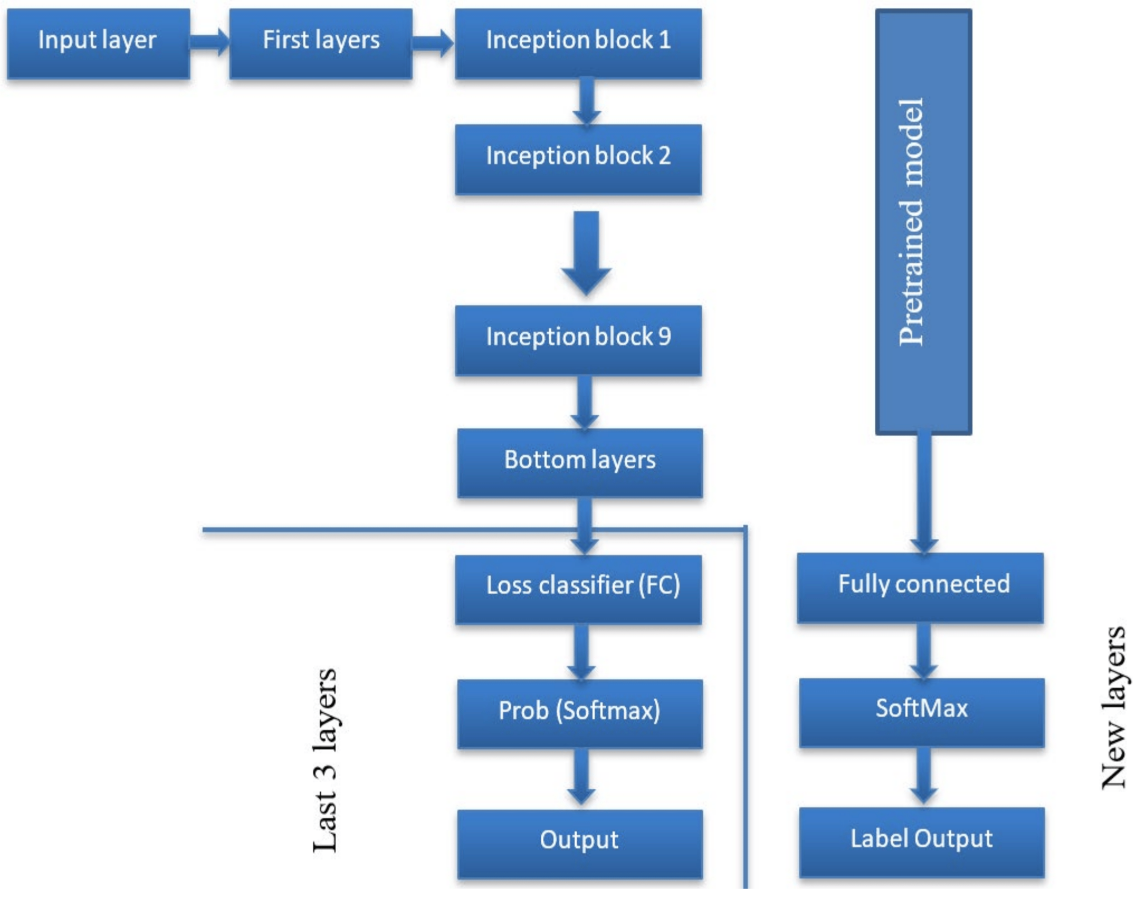


Figure 1.20: The architecture of the proposed method of Soaad M. et al [14]

The proposed deep learning model achieved an impressive accuracy of 99.55%, showcasing its ability to accurately identify cervical spine fractures. The model also demonstrated a high sensitivity of 99.33%, indicating its proficiency in correctly detecting positive cases, and a specificity of 99.66%, reflecting its capability to correctly identify negative cases. With a precision of 99.33%, the model showcased its accuracy in correctly labeling positive predictions. These results highlight the potential of deep learning methods in enhancing the detection of cervical spine fractures. Furthermore, a comparison between the deep learning model and expert radiologists and orthopedic surgeons was conducted. While the model's accuracy of 92.16% was slightly lower than that of the human experts (97.1% for radiologists and 98.5% for orthopedic surgeons). Overall, this related work section provides an overview of the research problem, highlights the novelty of the proposed approach, and briefly explains the deep learning techniques used for the classification of cervical spine injuries.

1.10.2 Artificial intelligence accurately detects traumatic thoracolumbar fractures on sagittal radiographs

In their research article, GS Rosenberg et al. [15] addressed the clinical and financial impact of missed fractures on plain radiographs, particularly in low- and middle-income countries where advanced imaging techniques such as CT and MRI may be limited. They utilized a dataset comprising imaging studies of 151 patients, including radiographs, CT scans, and/or MRI scans, all confirmed to have fractures by a group of expert spinal surgeons. From the sagittal radiographs, 630 single vertebra images were extracted, with 302 exhibiting fractures and 328 showing no fractures.

To develop a deep learning model for detecting traumatic fractures on sagittal radiographs of the thoracolumbar (TL) spine, the authors selected two models, ResNet18 and VGG16 (see Figure 1.21), based on their established performance in computer vision tasks. Transfer learning techniques were employed to adapt these models for the specific task of classifying vertebral fractures. The dataset was split into training and testing sets, and reinforcement techniques were applied to enhance the model's generalization capabilities. The evaluation of the models encompassed multiple performance metrics, including accuracy, sensitivity, and specificity.

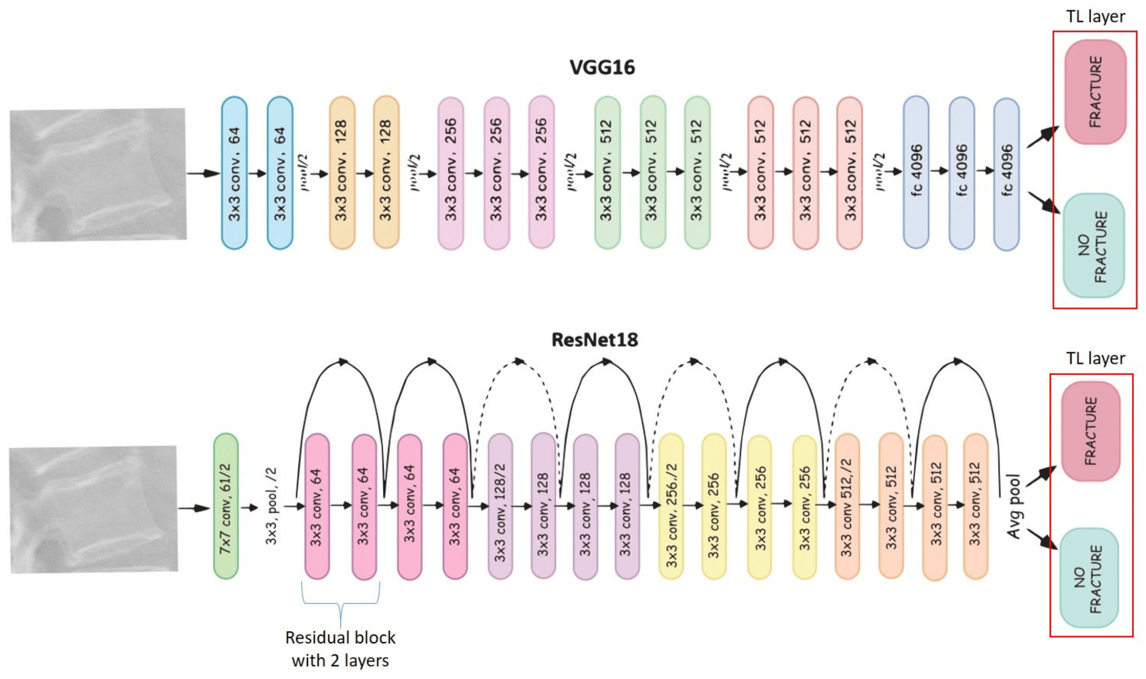


Figure 1.21: model architectures VGG16 and ResNet18 of GS rosenberg et al [15].

The study yielded promising results, with the ResNet18 model outperforming the VGG16 model in terms of sensitivity, specificity, and accuracy. Achieving a sensitivity of 91%, specificity of 89%, and accuracy of 88%, the ResNet18 model demonstrated its capacity to accurately identify traumatic vertebral fractures on sagittal radiographs. These findings showcase the potential of deep learning models in improving fracture detection, which could have significant implications in clinical practice, particularly in resource-constrained settings where access to advanced imaging modalities is limited.

The study highlights the potential of deep learning models in improving the detection of vertebral fractures on plain radiographs, which could have significant clinical and financial implications, particularly in low- and middle-income countries where advanced imaging modalities may not be readily available.

1.10.3 CT cervical spine fracture detection using a convolutional neural network

In a study conducted by J.E. Small et al. [48], the diagnostic accuracy and agreement between a Convolutional Neural Network (CNN) and radiologist ratings were compared for the detection of cervical spine fractures. The study utilized 665 CT images of the cervical spine, retrospectively examining fractures identified on CT scans, MR imaging, and CNN output data, which served as the ground truth. Sensitivity, specificity, positive and negative predictive values, as well as k coefficients with 95% confidence intervals, were determined to evaluate the performance of the CNN model.

The detection model for cervical spine fractures consisted of two stages: region proposal and false-positive reduction. In the first stage, a 3D fully convolutional deep neural network with a Residual Network architecture was used to create a 3D segmentation map from segmented scans. The second stage employed a combination of unlearned designed characteristics from conventional image-processing techniques and learned features from a multilayered classification head to classify each region proposal as positive or negative. These characteristics were merged through a second neural network to determine the presence of a fracture. Importantly, the model was trained from scratch without utilizing pretraining from other datasets.

The study revealed that the CNN achieved an accuracy rate of 92% in identifying cervical spine fractures, with a sensitivity of 76% and specificity of 97%. In comparison, radiologists achieved an accuracy of 95%, with sensitivity of 93% and specificity of 96%. Both the CNN and radiologists missed fractures in similar positions and levels, including fractures of anterior osteophytes, transverse processes, spinous processes, and lower cervical spine fractures that can be challenging to detect due to CT beam attenuation. These findings suggest that while the CNN demonstrated high accuracy, there is still room for improvement to match the performance of experienced radiologists in certain complex fracture cases.

1.10.4 Artifcial intelligence for the detection of vertebral fractures on plain spinal radiography

In their study, Murata et al. [16] aimed to diagnose vertebral fractures (VF) using patient images collected in accordance with ethical guidelines. The dataset comprised 300 patients, with 150 patients having VF and 150 without VF. VF diagnosis was conducted using portable thoracolumbar radiographs (PTLR), and magnetic resonance imaging (MRI) images were obtained within one month after symptom onset. Before inputting the images into a Deep Convolutional Neural Network (DCNN) for diagnosis (see Figure 1.21), they were labeled and reviewed by spine surgeons. The DCNN utilized techniques such as convolution and pooling, adjusting neural network weights based on the discrepancy between output and true label. The input images were resized to 512x512 pixels with 8-bit grayscale color.

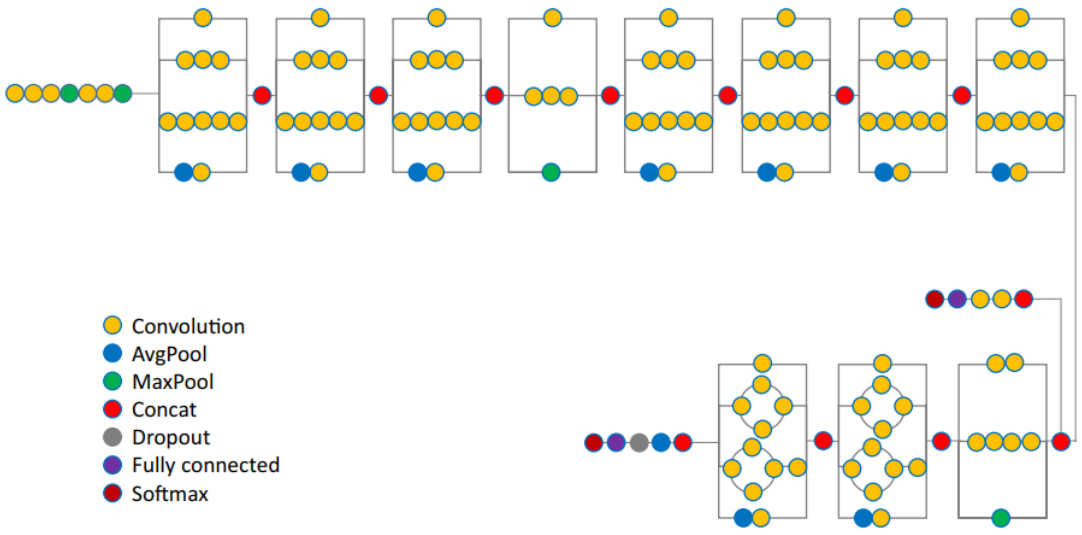


Figure 1.22: Representative of visual recognition V3 model [16].

The study assessed the performance of the DCNN, orthopedic residents, orthopedic surgeons, and spine surgeons in detecting VF on spine radiographs. The DCNN exhibited an area under the curve (AUC) of 0.91, displaying higher sensitivity compared to orthopedic residents but without statistical significance in accuracy and sensitivity compared to orthopedic surgeons. In all aspects, the performance of spine surgeons was significantly superior. The DCNN's kappa coefficient was 0.36 for orthopedic residents, 0.48 for orthopedic surgeons, and 0.66 for spine surgeons. The DCNN successfully diagnosed VF in all cases except for one patient, while there was only one misdiagnosis of VF by both the DCNN and physicians.

The study's findings indicate that the DCNN, trained with a focus on PTLR, achieved high accuracy and sensitivity in identifying VFs. This suggests its potential as a screening tool to aid physicians in VF detection. However, further research involving a diverse patient cohort, including individuals with osteoporosis and those without fractures, is necessary to enhance the model's performance.

1.11 Conclusion

In conclusion, cervical spine fractures are a serious medical condition that traditional methods of diagnosis and treatment have limitations in addressing. Machine learning algorithms, particularly convolutional neural networks, offer the potential to revolutionize the detection of cervical spine fractures, providing faster and more accurate diagnoses for better patient outcomes. Understanding the anatomy and function of the cervical spine and the various types of fractures and their causes is crucial in developing and implementing these algorithms. Further research and development in this area have the potential to greatly benefit both patients and healthcare providers. The next chapter will focus on the system's design, the dataset, its preparation, and the possible models to detect cervical spine fractures in a smart grid.

Chapter 2

System design

2.1 Introduction

In recent years, there has been a significant focus on the detection of cervical spine fractures, and this has become possible with the utilization of machine learning and deep learning algorithms, particularly convolutional neural networks. The possibility of creating our own convolutional neural network to detect and classify cervical spine fractures is being explored. To achieve this, image processing methods are employed to analyze the X-ray images of the cervical spine.

In this chapter, the methodology employed for the detection of cervical spine fractures using a machine learning algorithm is presented. A detailed description of the dataset utilized in the study is provided. Preprocessing techniques, including data augmentation, normalization, and resizing, are applied to enhance the quality and consistency of the input data. The dataset is then appropriately divided into training, testing, and validation subsets. Finally, various CNN models are explored for the prediction of cervical spine fractures, and their performance and potential in the detection task are evaluated.

2.2 Methodology

The overall system architecture can be summarized in three main steps, as illustrated in Figure 2.1:

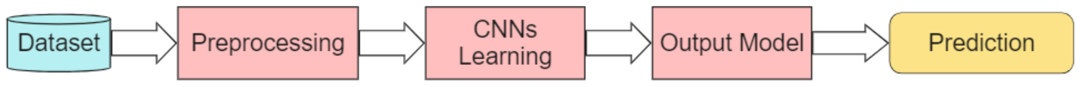


Figure 2.1: Global architecture of the system

The detailed architecture of the system is based on several key steps, This architecture is illustrated in Figure 2.2:

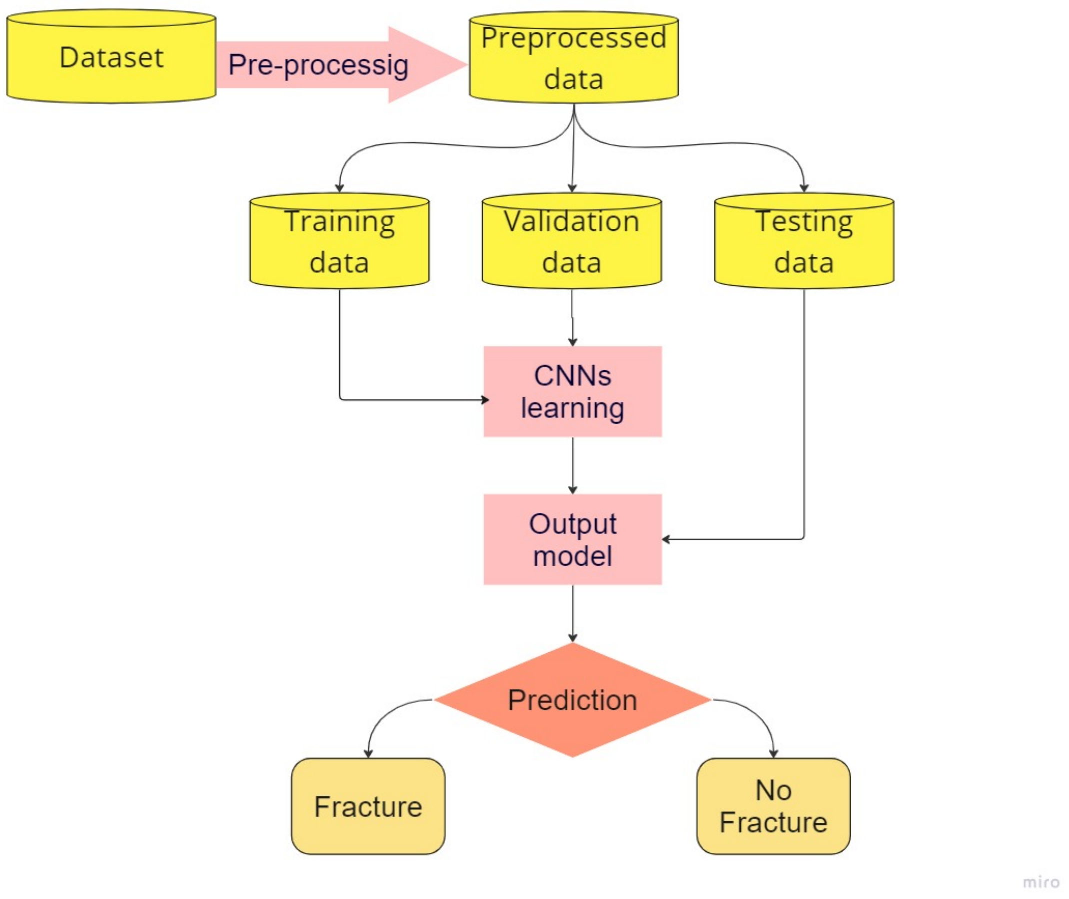


Figure 2.2: Detailed architecture of the system

The dataset is obtained by the system. After this, preprocessing is performed on the dataset, which involves augmentation, normalization, and resizing the database to make it suitable for training. A newly processed database will be divided into 3 sections: training and validation data, which will be used to train our models, and testing data, which will be used to determine the validity of our models. In the next stage, a training dataset is inputted into our learning models, which are CNN models, so that they can learn and construct a new predictive model. Once the models are ready, they will be applied to the test dataset.

2.3 Detailed architecture

2.3.1 Dataset description

The dataset has been obtained from Kaggle [14]. It contains 2009 images. The images are organized into three groups. The first group includes 530 CS dislocation images Figure 2.3. The second group includes 772 CS fracture images Figure 2.4. Finally, the third group includes 707 normal images Figure 2.9.

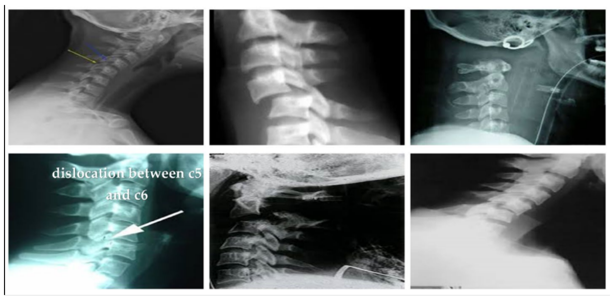


Figure 2.3: Samples from the first group (CS dislocation images).

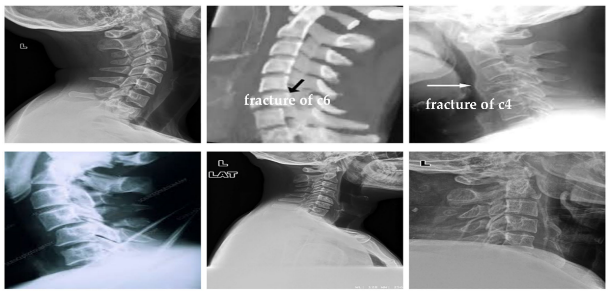


Figure 2.4: Samples from the second group (CS fracture images).

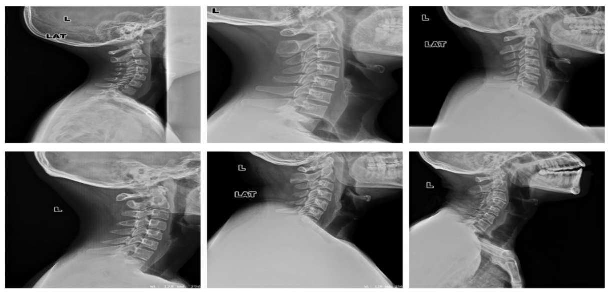


Figure 2.5: Samples from the third group (CS normal images).

2.3.2 Preprocessing

Preprocessing is an essential step in data preparation for machine learning models. In the context of image data, several common preprocessing techniques are employed, including data augmentation, data normalization, and data resizing. Here's a Descriptive model of each technique:

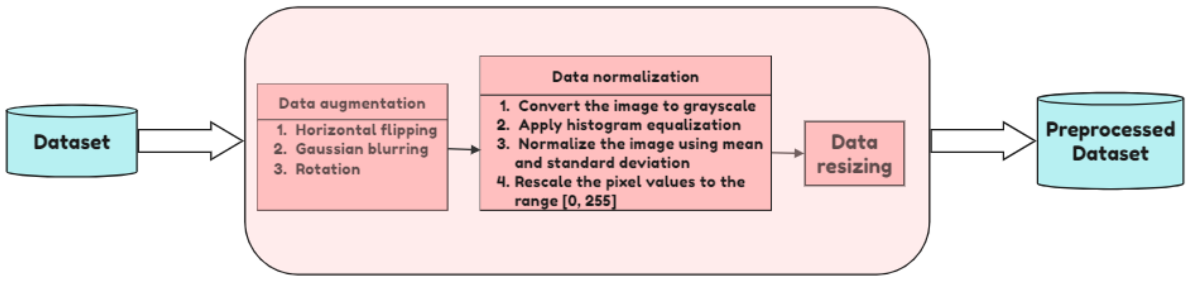

Dataset
Data augmentation
1. Horizontal flipping
2. Gaussian blurring
3. Rotation
Data normalization
1. Convert the image to grayscale
2. Apply histogram equalization
3. Normalize the image using mean and standard deviation
4. Rescale the pixel values to the range [0, 255]
Data resizing
Preprocessed Dataset

Figure 2.6: Preprocessing phase.

2.3.2.1 Data augmentation

To improve our dataset and obtain a wide range of training data, the assistance of augmentation data technologies is utilized to enhance the diversity of our dataset. By incorporating these augmentation techniques, the understanding of various examples by our CNN model is facilitated, resulting in a larger number of learning instances for the model. Through this approach, it is anticipated that our model will achieve better performance.

In our thesis, we will talk about three specific augmentation techniques we used to make our images better. These techniques are horizontal flipping, Gaosi jamming, and rotation. We will explain each of these technologies in more detail

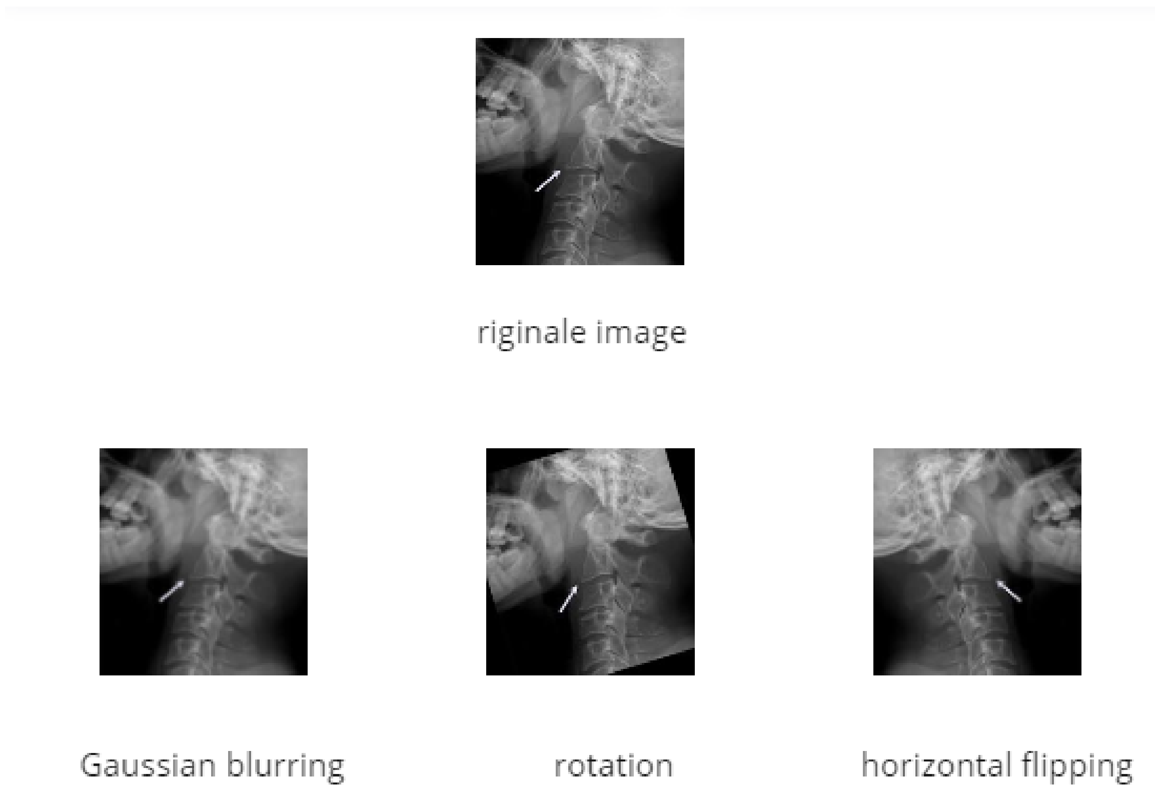


Figure 2.7: Example of images after applying augmentation techniques

1. **Horizontal flipping:** The cervical spine image is horizontally flipped along its vertical axis, generating a mirrored version of the original image. This technique introduces variations to the training data and enhances the CNN model's ability to handle diverse orientations of the cervical spine.

2. **Gaussian blurring:** Gaussian blurring involves applying a Gaussian blur filter to the cervical spine image, resulting in image smoothing and noise reduction. This technique proves beneficial when dealing with noisy cervical spine images or those containing excessive high-frequency details. The blurring effect aids in increasing the CNN model's resilience to noise and other forms of image distortion.

3. **Rotation:** This technique entails rotating the cervical spine image by a specified angle. By employing this augmentation technique, a broader range of training data is generated, thus augmenting the dataset's diversity. Moreover, the CNN model is trained to recognize cervical spine images captured from various angles or perspectives, thereby enabling effective diagnosis of different cervical spine conditions.

2.3.2.2 Data normalization

The data normalization process is considered a crucial step in the preparation of data for machine learning algorithms, including those utilized for cervical spine fracture detection. The input data is transformed to a standard scale or range during this process to ensure effective learning from the data by the algorithm.

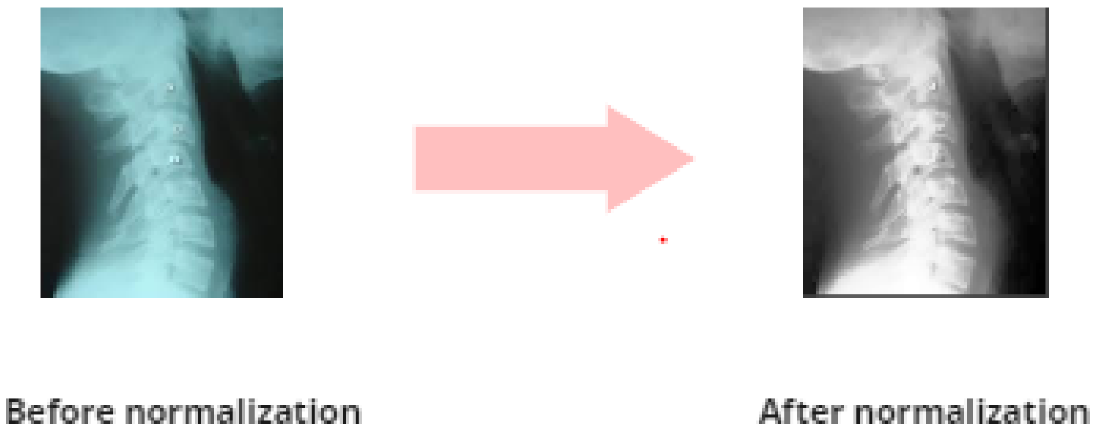


Figure 2.8: Example of the image after applying normalization techniques

In the context of image processing for cervical spine fracture detection, here are the elements of data normalization:

- **Convert the image to grayscale:** Grayscale conversion simplifies the image by reducing the number of color channels from three (RGB) to a single channel (gray). This step helps to remove any color information that may be relevant for fracture detection.

- **Apply histogram equalization:** Histogram equalization is a technique used to enhance the contrast of an image. The pixel intensities are redistributed in such a way that the entire range of intensities is effectively utilized. By applying histogram equalization, the visibility of important features and details in the image can be improved.

- **Normalize the image using mean and standard deviation:** Normalization based on mean and standard deviation helps to standardize the pixel values of the image. The mean pixel value is subtracted from each pixel, and then it is divided by the standard deviation. This process ensures that the pixel values have zero mean and unit variance, which can be beneficial for training machine learning algorithms.

- **Rescale the pixel values to the range [0, 255]:** After normalization, the pixel values are rescaled to the range [0, 255] to ensure that they fall within the valid range for image representation. This step involves linearly scaling the pixel values using the minimum and maximum pixel values in the image. The minimum value is mapped to 0, and the maximum value is mapped to 255, with the other pixel values scaled accordingly.

By applying these elements of data normalization, the images in our cervical spinal fracture detection dataset can be processed in advance, rendering them suitable for training and enhancing the performance of our machine learning algorithm.

2.3.2.3 Data resizing

The initial dataset contained images of varying sizes, but we resized all images to a standardized dimension of (300x300x3), where the first two values correspond to the width and height of the image, and the third value denotes the three color channels of the RGB format.

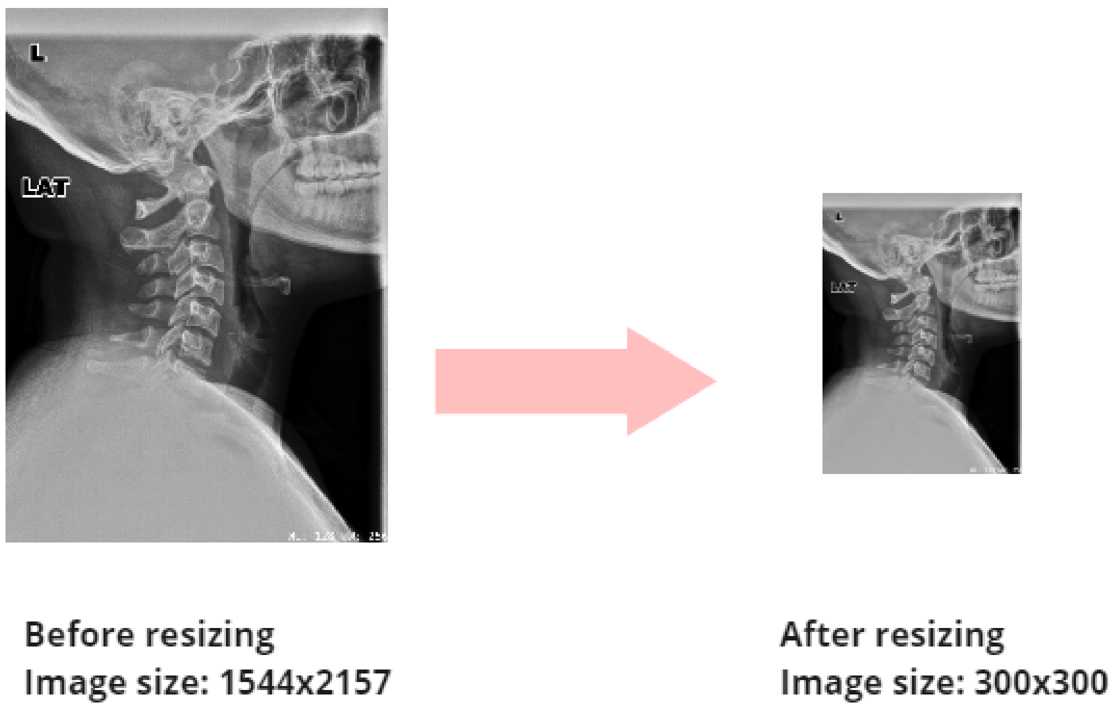


Figure 2.9: Example of the image after applying to resize

The reason for resizing the images is to ensure uniformity in the size of input images to the CNN model. Using large-sized images directly would require a larger input shape for the CNN model, which would in turn increase the number of parameters in the model and prolong the training phase. Resizing the images to a fixed size reduces the complexity of the CNN model and speeds up the training process.

2.3.3 Splitting dataset

In deep learning, the dataset is divided into three parts: the training subset, the validation subset, and the testing subset. We split our dataset into these three subsets using a distribution of 70% images per class for training, 15% for validation, and 15% for testing.

2.3.4 CNNs models

The architecture designed for the purpose of detecting cervical spine fractures through the utilization of a deep learning algorithm involved the implementation of various types of convolutional neural network (CNN) models. These models consisted of a custom CNN architecture tailored specifically for this task, as well as the utilization of two popular pre-trained models: VGG16 and ResNet50V2, employing the technique of transfer learning.

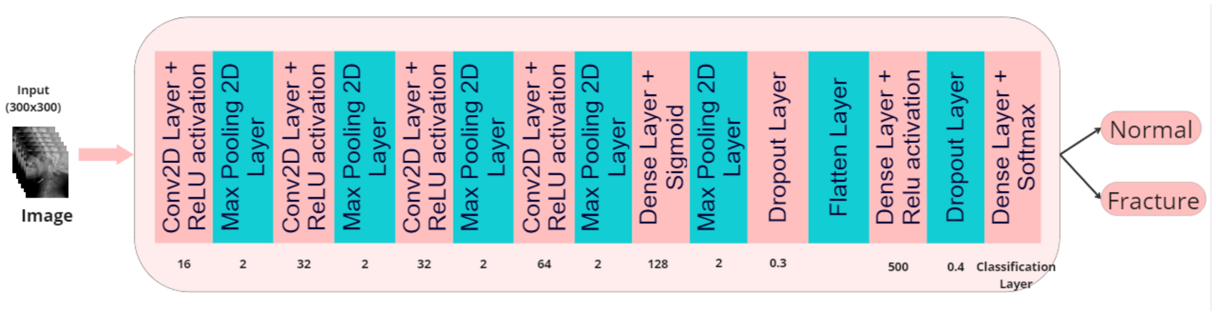


Figure 2.10: Our CNN model.

The custom CNN model (as shown in Figure 2.10) is composed of several convolutional layers with increasing numbers of filters (16, 32, 32, 64, and 128) and kernel sizes of 3. Each convolutional layer is succeeded by a max-pooling layer with a pool size of 2. Dropout layers with a dropout rate of 0.3 were also incorporated into the model to prevent overfitting. The flattened output is then connected to fully connected layers consisting of 500 units and ReLU activation. Another dropout layer with a dropout rate of 0.4 is introduced prior to the final dense layer, which contained 2 units and employs softmax activation for binary classification.

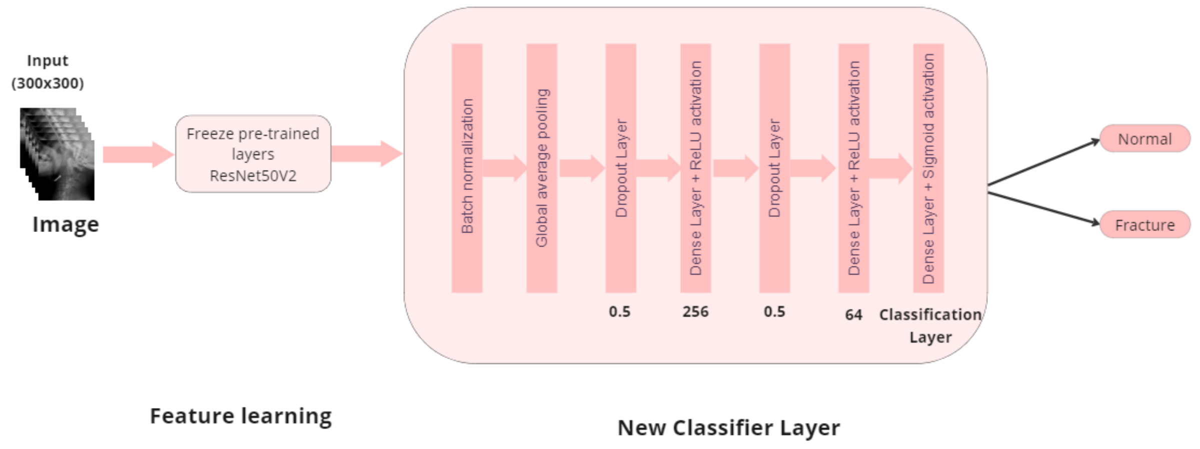


Figure 2.11: The architecture of transfer learning using ResNet50V2.

The ResNet50V2 model (shown in Figure 2.11) is utilized as the base model. The pre-trained layers were frozen to retain their learned features. Batch normalization is applied to the output of the base model, followed by global average pooling to reduce spatial dimensions. The model is regularized using two dropouts with a rate of 0.5. Two dense layers with 256 and 64 units and ReLU activation were subsequently added, preceding the final dense layer with 2 units and sigmoid activation for binary classification.

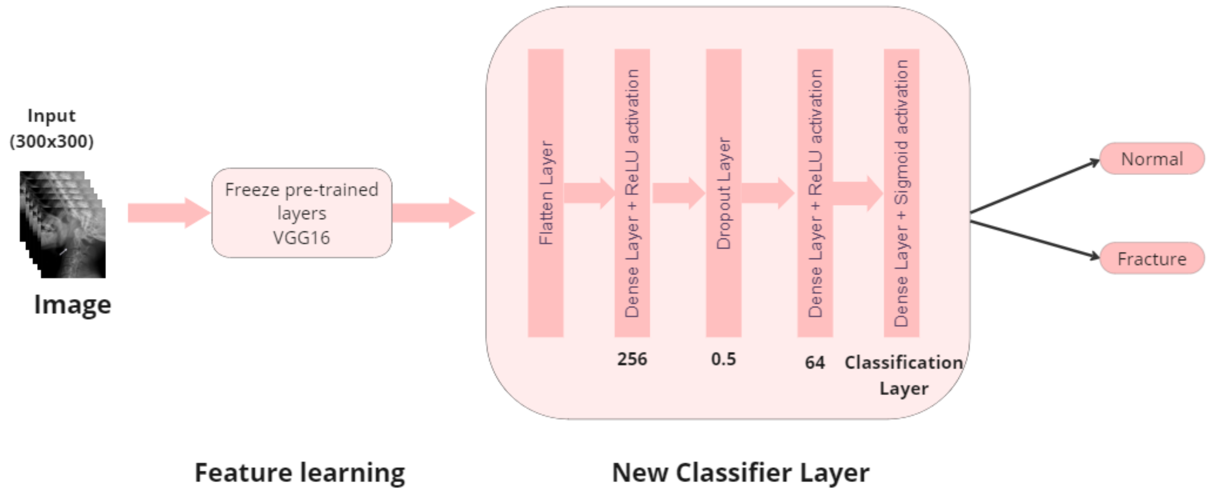


Figure 2.12: The architecture of transfer learning using VGG16.

The base model used in the study is the VGG16 model (shown in Figure 2.12). In order to preserve the learned representations, the pre-trained layers were frozen. Integration of the VGG16 base model into a sequential model is followed by the addition of a flattened layer. Two dense layers, each consisting of 256 and 64 units and employing ReLU activation, were subsequently added. For the purpose of regularization, a dropout layer with a dropout rate of 0.5 was included. Finally, binary classification is performed using a dense layer consisting of 2 units and employing sigmoid activation.

These three different CNN models offer various architectural designs for cervical spine fracture detection, providing flexibility in choosing the most suitable model for the task at hand.

2.3.5 Prediction

After the training phase, the prediction phase begins, where the CNN model is utilized to classify the data and measure the accuracy. The second subset of the dataset is used for testing the CNN model. The steps involved in feeding the test data into the models and obtaining the predictions are elaborated upon. Evaluation metrics and performance analysis are also discussed to assess the accuracy and robustness of the fracture detection algorithm.

Furthermore, the best trained model is incorporated into the interface, where an easy-to-use interface is provided to doctors for utilizing the cervical spinal fracture detection algorithm. With this integration, the model can be comfortably and effectively used in a real-world clinical environment.

2.4 Deployment

In this section, we use UML's use case to explain how to exploit the model 2.13:

The doctor uploads the X-ray image of the cervical spine to the interface. The image is processed in advance by changing its size and normalizing pixel values. Next, the specially designed trained model is loaded to classify cervical spinal fractures. After making predictions on the pre-treated image, the result of the classification is then presented to the doctor. Finally, the doctor sees the result of the classification on the interface.

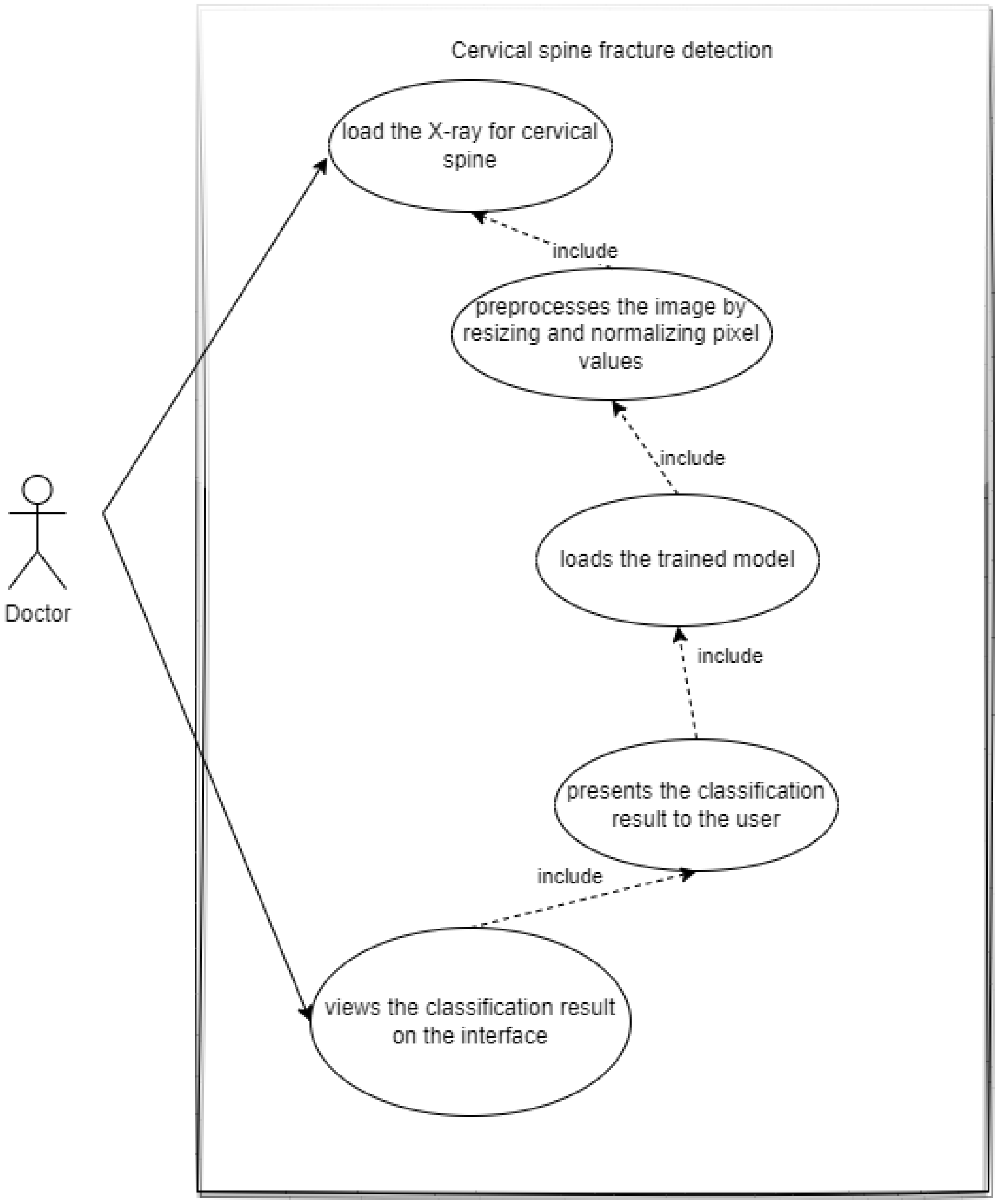

Cervical spine fracture detection
load the X-ray for cervical spine
include
preprocesses the image by resizing and normalizing pixel values
include
loads the trained model
include
presents the classification result to the user
include
views the classification result on the interface
Doctor

Figure 2.13: Use case diagram.

2.5 Conclusion

This chapter's objective is to provide a comprehensive and in-depth review of our system architecture, which encompasses our datasets, the preprocessing that was performed, the CNN models that were utilized, as well as our preprocessing activities. In the subsequent chapter, the various frameworks, tools, and libraries that were employed will be examined, along with an explanation of how our cervical spine detection models were constructed. Furthermore, the results obtained will be discussed, and the most effective model based on our data will be identified.

Chapter 3

Implementation and results

3.1 Introduction

In the previous chapter, we explained how our system works and showed how we prepared the dataset. We also gave an overview of the models we're using. In the next chapter, we'll talk about the tools and frameworks we used to build the system and show you how we wrote the code. Then, we'll discuss the different models we tested and the outcomes we observed.

3.2 Implementation frameworks and tools

Various free online tools are available for implementing deep learning, and we utilized Python as the programming language. Alongside Python, we employed several libraries such as Tensorflow, Keras, Matplotlib, Numpy, and Gradio to achieve our objectives. Tensorflow aims to simplify the process of building and training deep learning models. Keras aims to enable fast experimentation with deep neural networks, allowing for easy and rapid prototyping. Matplotlib's objective is to provide a simple and effective way to visualize and analyze data, which is crucial in deep learning. Numpy's objective is to provide a fast and efficient way to perform complex mathematical operations, which are a critical component of deep learning. Gradio provides a simple and intuitive interface for creating and deploying customizable web interfaces for machine learning models, allowing users to interact with models using various input types and receiving real-time predictions as outputs. To execute our code, we used Kaggle as our programming environment.

3.3 Evaluation metrics

3.3.1 Confusion Matrix

A confusion matrix is a way to see how well a classification algorithm is doing. It looks at whether the model is correctly predicting positive or negative results, and if it's making any mistakes:

- **True Positive:** This is when the model correctly predicts a positive result.
- **False Positive:** This is when the model predicts a positive result, but it's actually negative.
- **True Negative:** This is when the model correctly predicts a negative result.
- **False Negative:** This is when the model predicts a negative result, but it's actually positive.

3.3.2 Accuracy

Accuracy refers to how often a classification model is correct in its predictions. It's calculated by dividing the number of correct predictions by the total number of predictions made, which results in the following equation 3.1:

$$Accuracy = \frac{TN + TP}{TN + TP + FN + FP} \tag{3.1}$$

3.3.3 Precision

Precision refers to the proportion of true positives (correctly predicted positive results) out of all the positive predictions made by the model. In other words, it measures how precise the model's positive predictions are, which results in the following equation 3.2:

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

3.3.4 Recall

Recall refers to the proportion of true positives out of all the actual positive instances in the data. It measures how well the model can identify positive instances, which results in the following equation 3.3:

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

3.3.5 AUC

The term "AUC" stands for the area under the ROC curve. AUC is a statistical measure used to evaluate the performance of a binary classifier. It provides a quantitative assessment of model predictions in a probabilistic setting. In simple terms, the ROC curve illustrates the relationship between the false-positive rate and true-positive rate at different levels of model prediction probabilities. On the other hand, the AUC represents the entire two-dimensional area under the complete ROC curve, where:

$$TPR = \frac{TP}{TP + FN} \quad FPR = \frac{FP}{FP + TN} \tag{3.4}$$

The AUC or ROC curve plots the ratio of true positives to false positives. Sensitivity is another term for True Positive Rate, while the rate of false positives is also known as (1-Specificity). The Y-axis represents sensitivity, and the X-axis represents (1-Specificity), as shown in Figure 3.1.

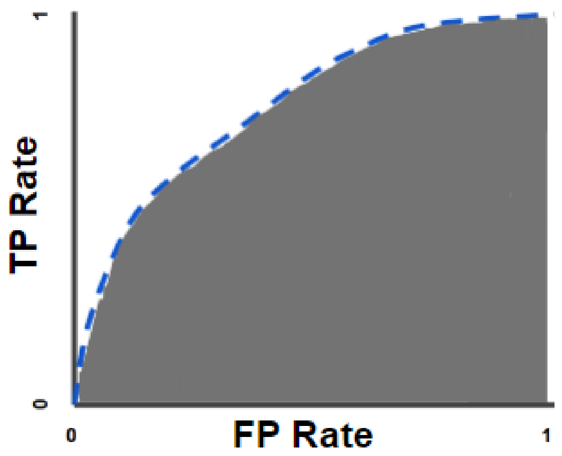


Figure 3.1: Area under the ROC Curve.

3.3.6 F1-Score

F1-score is a metric that combines precision and recall to give an overall measure of the model's performance. It's the harmonic mean of precision and recall, and it ranges from 0 to 1, where a score of 1 means perfect precision and recall, which results in the following equation 3.5:

$$F1 - Score = \frac{TP}{TP + \frac{1}{2}(FN + FP)} \tag{3.5}$$

3.4 Implementation phases

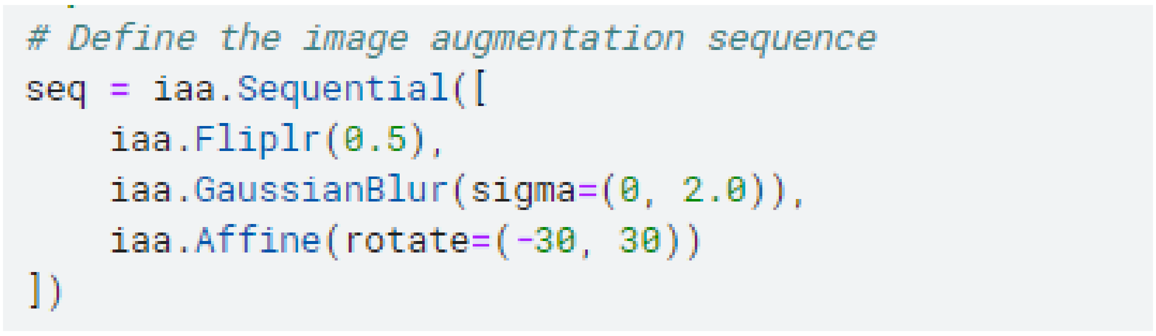
In this section, the stages of implementing our system and how it was constructed and evaluated using Python code and the Kaggle platform will be discussed.

3.4.1 Loading and preprocessing the datasets

- **Preparing the environment**
  First, we need to create an account on Kaggle. Then, we create the new Notebook. Second, we need to modify the accelerator to GPU P100.

- **Data augmentation**

  Data augmentation techniques have been applied to enhance the dataset. The mentioned augmentation methods include horizontally flipping images, applying Gaussian blur for noise reduction, and performing affine transformations such as rotation within a certain range:



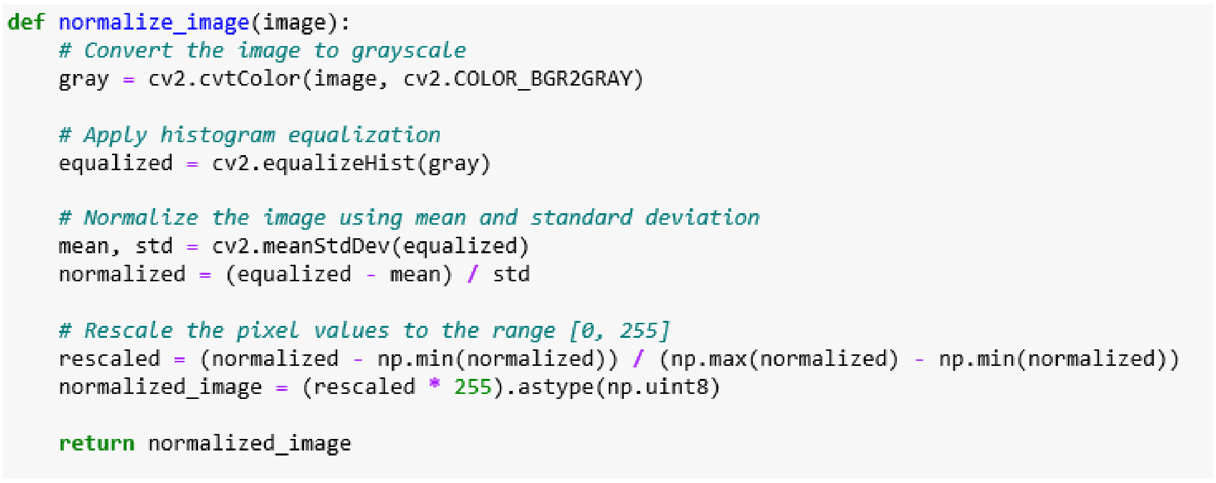
```
# Define the image augmentation sequence
seq = iaa.Sequential([
    iaa.Fliplr(0.5),
    iaa.GaussianBlur(sigma=(0, 2.0)),
    iaa.Affine(rotate=(-30, 30))
])
```

Figure 3.2: the method of augmentation data.

Augmentation function	**Description**
Fliplr(0.5)	Images are randomly flipped horizontally with a 50% probability by this operation. The image is mirrored along a vertical axis, effectively creating a mirror image.
GaussianBlur(sigma=(0, 2.0))	Gaussian blurring is applied to the image by this operation. Gaussian blurring is a smoothing technique that reduces image noise and sharpens edges. The amount of blurring is controlled by the sigma parameter, with values randomly selected between 0 and 2.0.
Affine(rotate=(-30, 30))	Affine transformations on the image are performed by this operation, specifically rotation. The image is randomly rotated within a range of -30 to 30 degrees. Variations in the orientation of the image can be introduced by this operation.

Table 3.1: Description of augmentation data.

- **Data normalization**



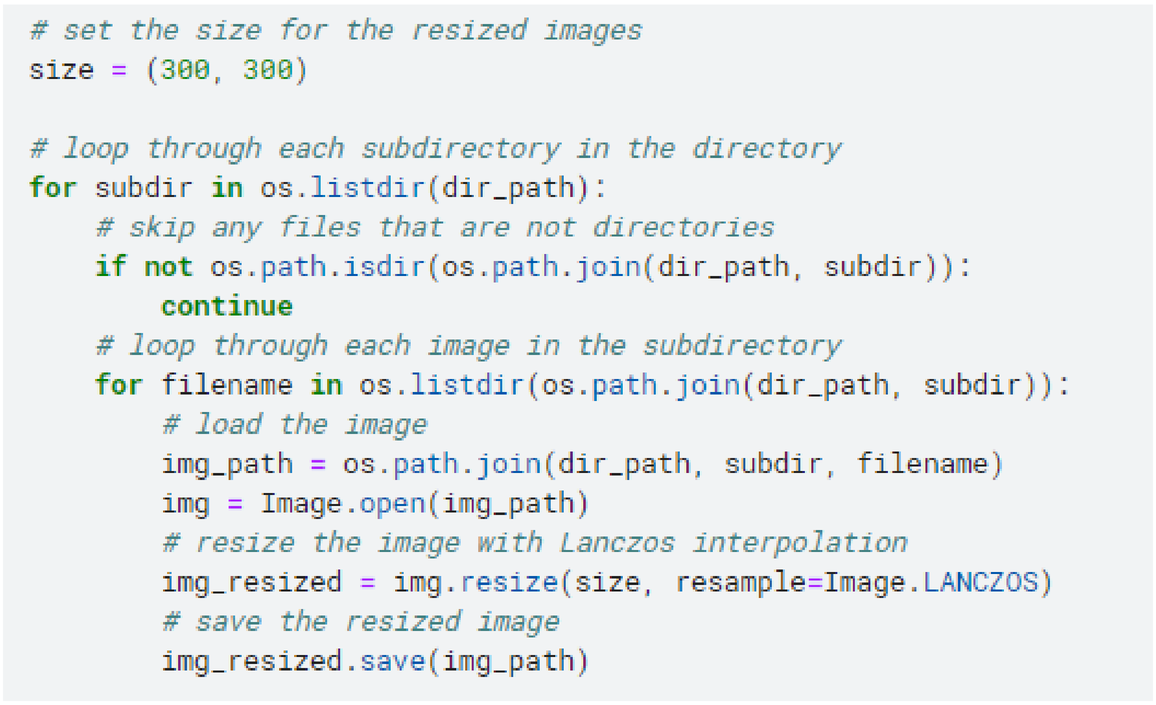
```
def normalize_image(image):
    # Convert the image to grayscale
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Apply histogram equalization
    equalized = cv2.equalizeHist(gray)

    # Normalize the image using mean and standard deviation
    mean, std = cv2.meanStdDev(equalized)
    normalized = (equalized - mean) / std

    # Rescale the pixel values to the range [0, 255]
    rescaled = (normalized - np.min(normalized)) / (np.max(normalized) - np.min(normalized))
    normalized_image = (rescaled * 255).astype(np.uint8)

    return normalized_image
```

Figure 3.3: Data normalisation.

A series of image normalization operations have been performed on an input image. First, the image is turned into grayscale using the OpenCV cv2.cvtColor function. Thereafter, the equation of the program is applied to promote gray image variation. The image is then normalized using the average and calculated standard deviation of the equivalent image. Pixel values are adjusted to fall within the range [0, 255] to ensure that they are within the correct intensity range of an 8-bit gray image. Finally, the resulting natural image is converted into the data type np.uint8.

- **Data resizing**

If the size of the image is not equal to the target size, all images will be resized to (300, 300). For us, the images have already been resized in the preprocessing phase.

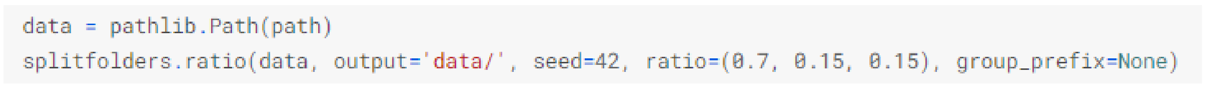


```
# set the size for the resized images
size = (300, 300)

# loop through each subdirectory in the directory
for subdir in os.listdir(dir_path):
    # skip any files that are not directories
    if not os.path.isdir(os.path.join(dir_path, subdir)):
        continue
    # loop through each image in the subdirectory
    for filename in os.listdir(os.path.join(dir_path, subdir)):
        # load the image
        img_path = os.path.join(dir_path, subdir, filename)
        img = Image.open(img_path)
        # resize the image with Lanczos interpolation
        img_resized = img.resize(size, resample=Image.LANCZOS)
        # save the resized image
        img_resized.save(img_path)
```

Figure 3.4: resize images.

3.4.2 Data splitting

The split_folder.ratio has been used to split our dataset into 3 subsets, as mentioned before. The split_folder.ratio parameters are as follows:
splitfolders.ratio(input_folder,output,seed,ratio,group_prefix)



```
data = pathlib.Path(path)
splitfolders.ratio(data, output='data/', seed=42, ratio=(0.7, 0.15, 0.15), group_prefix=None)
```

Figure 3.5: Splitting Dataset.

Following the pretreatment of the data that was indicated in the previous subsection, the dataset should be split. Splitting data is a crucial part of data science. Splitting refers to the process of dividing data into two or more parts. Our dataset is split into three parts, with 70% of the data being used to train the model, and the remaining two parts being utilized to evaluate and test the data, with 15% each.

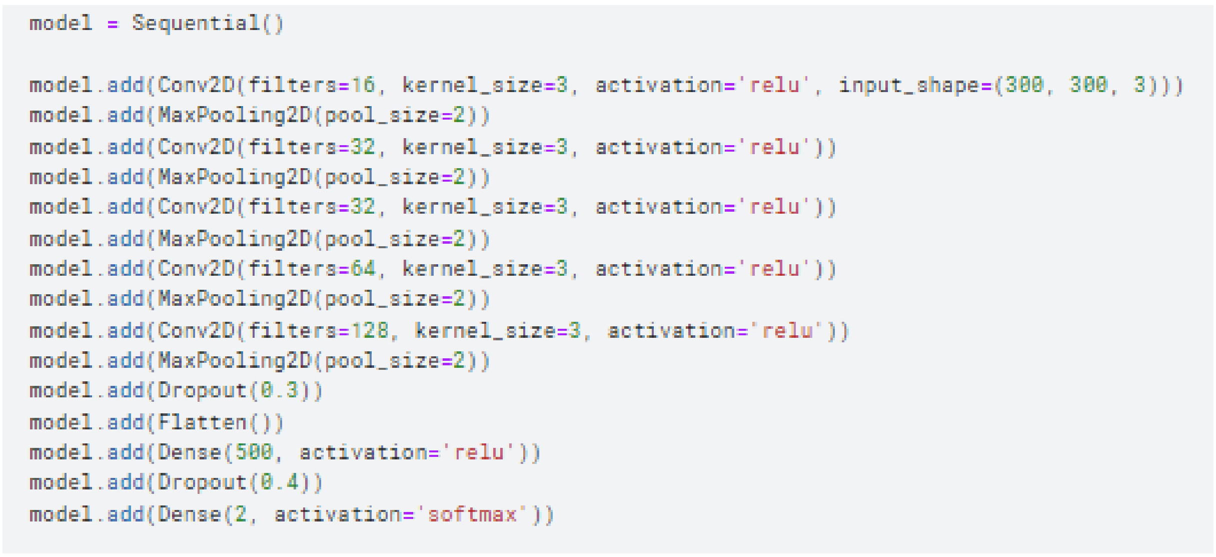
	Training	**Validation**	**Test**	**Total**
fracture	1106	237	237	1580
normal	923	198	199	1320
Total	2029	435	436	2900

Table 3.2: Dataset structure.

3.4.3 Building CNN model

(a) **proposed CNN model**

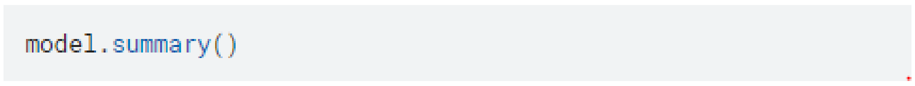
After trying many CNN configurations, the one that will be described had the best results.
We used the sequential model to create deep learning models where an instance of the sequential class is created and model layers are added.



```
model = Sequential()

model.add(Conv2D(filters=16, kernel_size=3, activation='relu', input_shape=(300, 300, 3)))
model.add(MaxPooling2D(pool_size=2))
model.add(Conv2D(filters=32, kernel_size=3, activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Conv2D(filters=32, kernel_size=3, activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Conv2D(filters=64, kernel_size=3, activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Conv2D(filters=128, kernel_size=3, activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Dropout(0.3))
model.add(Flatten())
model.add(Dense(500, activation='relu'))
model.add(Dropout(0.4))
model.add(Dense(2, activation='softmax'))
```

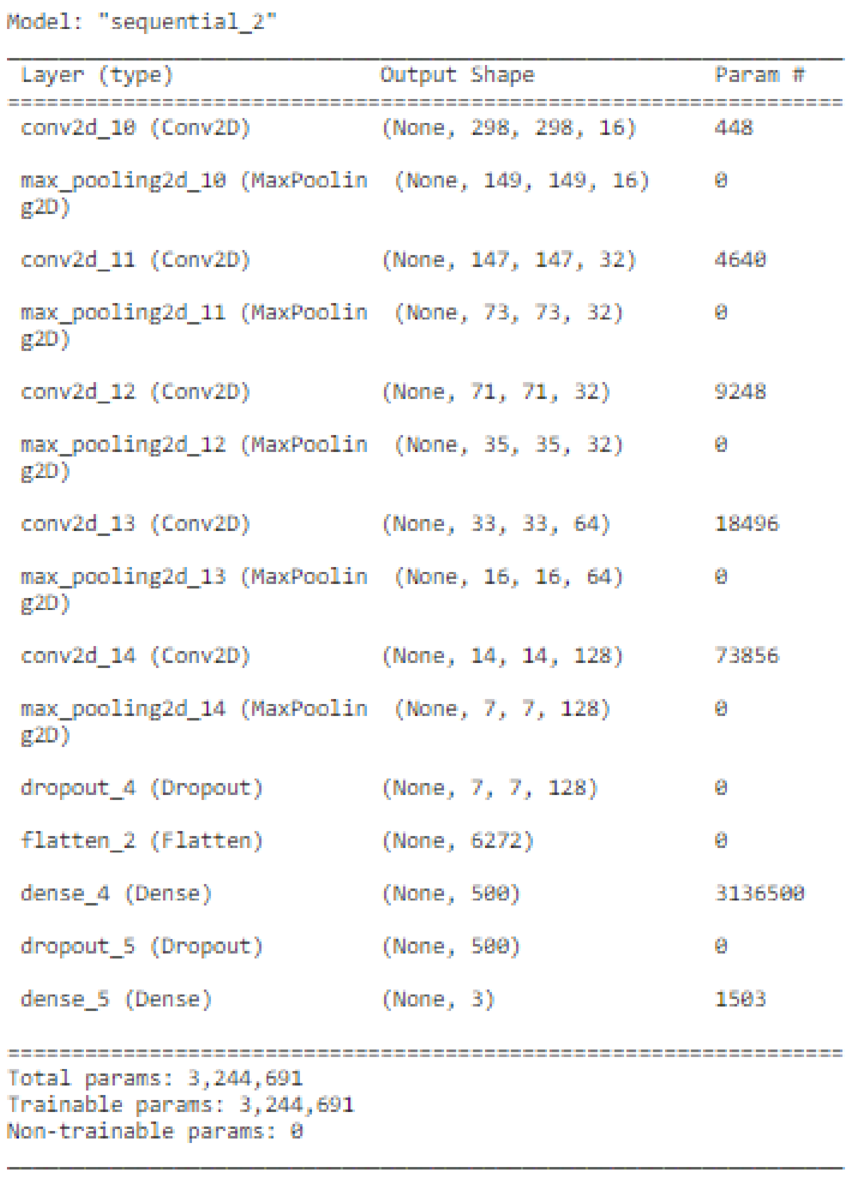
Figure 3.6: cervical spine fracture Detection CNN Model.

Once our model is ready, we can call the summary() method to display its contents.



```
model.summary()
```

Figure 3.7: Model summary.

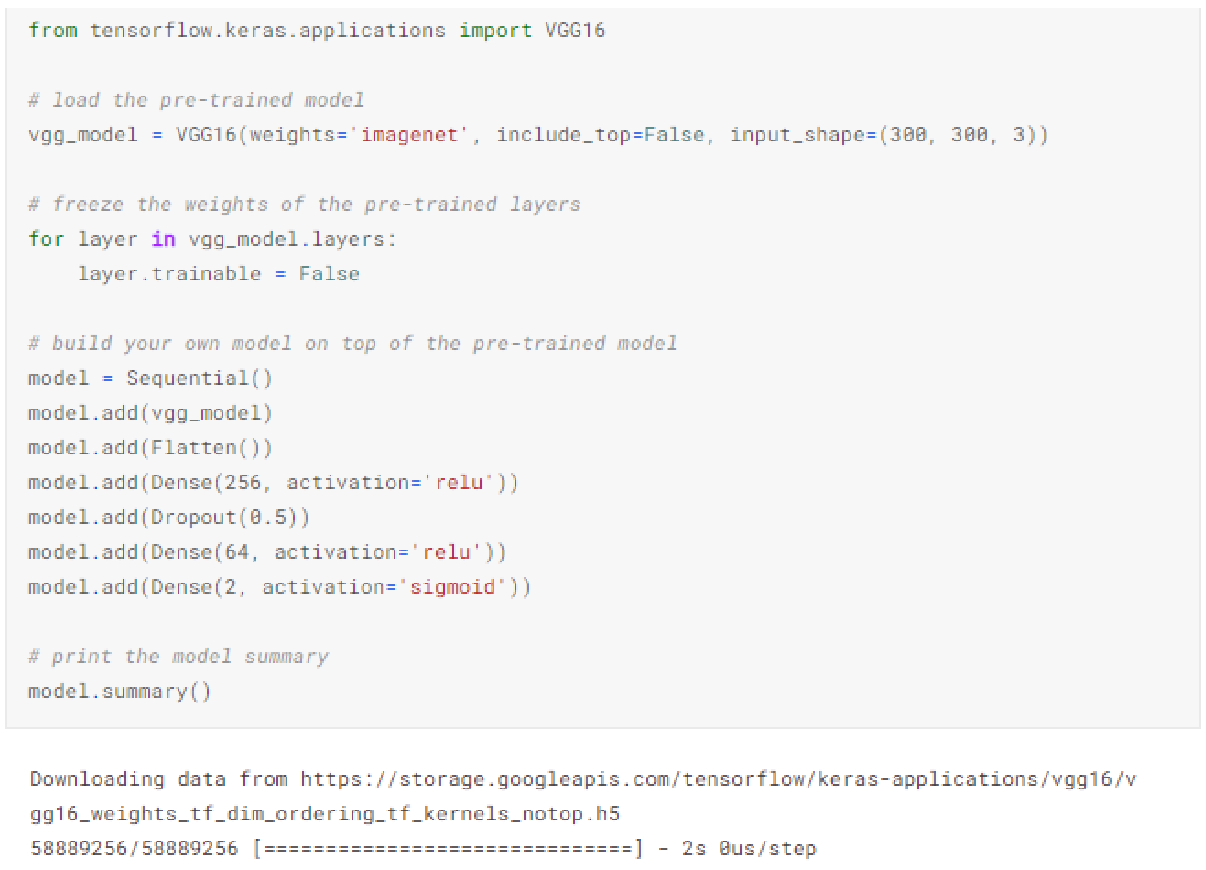


```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_10 (Conv2D)          (None, 298, 298, 16)      448

 max_pooling2d_10 (MaxPoolin  (None, 149, 149, 16)     0
 g2D)

 conv2d_11 (Conv2D)          (None, 147, 147, 32)      4640

 max_pooling2d_11 (MaxPoolin  (None, 73, 73, 32)       0
 g2D)

 conv2d_12 (Conv2D)          (None, 71, 71, 32)        9248

 max_pooling2d_12 (MaxPoolin  (None, 35, 35, 32)       0
 g2D)

 conv2d_13 (Conv2D)          (None, 33, 33, 64)        18496

 max_pooling2d_13 (MaxPoolin  (None, 16, 16, 64)       0
 g2D)

 conv2d_14 (Conv2D)          (None, 14, 14, 128)       73856

 max_pooling2d_14 (MaxPoolin  (None, 7, 7, 128)        0
 g2D)

 dropout_4 (Dropout)         (None, 7, 7, 128)         0

 flatten_2 (Flatten)         (None, 6272)              0

 dense_4 (Dense)             (None, 500)               3136500

 dropout_5 (Dropout)         (None, 500)               0

 dense_5 (Dense)             (None, 3)                 1503

=================================================================
Total params: 3,244,691
Trainable params: 3,244,691
Non-trainable params: 0
_________________________________________________________________
```

Figure 3.8: Model summary.

(b) **Transfer learning using VGG16**

The code snippet provided demonstrates how to create a new model by adding custom layers on top of a pre-trained VGG16 model using the TensorFlow-Keras API.all layers of the basic model are frozen, preventing updating their weights during training. By freezing pre-trained layers, we can retain their acquired features while training only newly added layers. Here's a breakdown of the code:



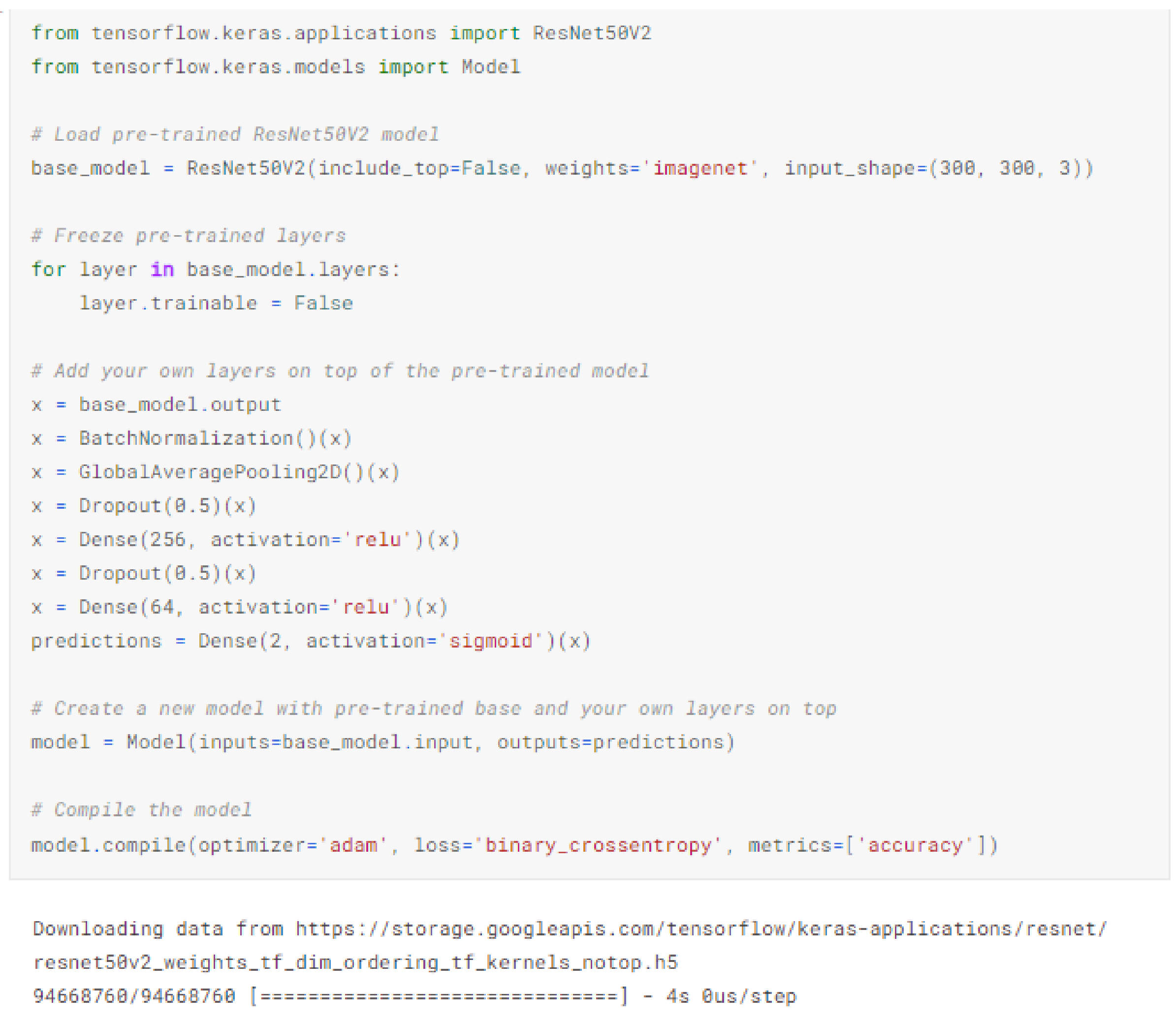
```python
from tensorflow.keras.applications import VGG16

# load the pre-trained model
vgg_model = VGG16(weights='imagenet', include_top=False, input_shape=(300, 300, 3))

# freeze the weights of the pre-trained layers
for layer in vgg_model.layers:
    layer.trainable = False

# build your own model on top of the pre-trained model
model = Sequential()
model.add(vgg_model)
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dense(2, activation='sigmoid'))

# print the model summary
model.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/v
gg16_weights_tf_dim_ordering_tf_kernels_notop.h5
58889256/58889256 [==============================] - 2s 0us/step
```

Figure 3.9: Import transfer learning model of VGG16.

(c) **Transfer learning using ResNet50V2**

In the same way, we have a new model that combines the basic model of pre-trained ResNet50V2 with additional layers:



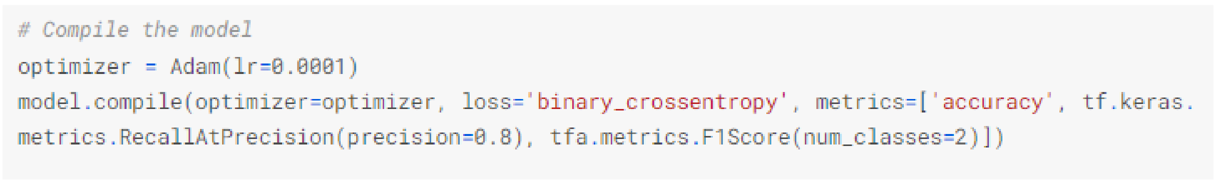
```
from tensorflow.keras.applications import ResNet50V2
from tensorflow.keras.models import Model

# Load pre-trained ResNet50V2 model
base_model = ResNet50V2(include_top=False, weights='imagenet', input_shape=(300, 300, 3))

# Freeze pre-trained layers
for layer in base_model.layers:
    layer.trainable = False

# Add your own layers on top of the pre-trained model
x = base_model.output
x = BatchNormalization()(x)
x = GlobalAveragePooling2D()(x)
x = Dropout(0.5)(x)
x = Dense(256, activation='relu')(x)
x = Dropout(0.5)(x)
x = Dense(64, activation='relu')(x)
predictions = Dense(2, activation='sigmoid')(x)

# Create a new model with pre-trained base and your own layers on top
model = Model(inputs=base_model.input, outputs=predictions)

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/
resnet50v2_weights_tf_dim_ordering_tf_kernels_notop.h5
94668760/94668760 [==============================] - 4s 0us/step
```

Figure 3.10: import transfer learning model of ResNet50V2.

3.4.4 Compile the model

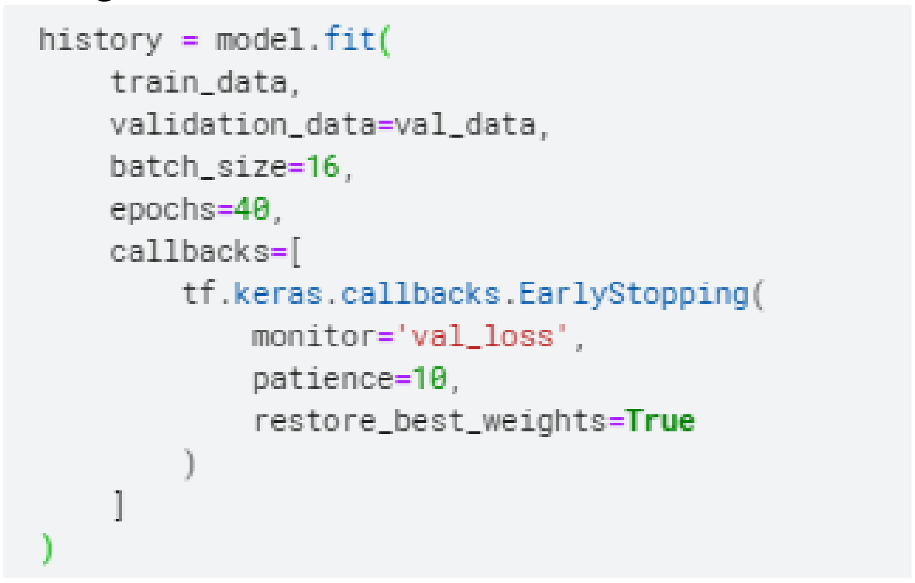
The model has been compiled with the Adam optimizer, a binary cross-entropy loss function, and several evaluation metrics such as accuracy, recall at a precision 0.8, and F1 score.



```
# Compile the model
optimizer = Adam(lr=0.0001)
model.compile(optimizer=optimizer, loss='binary_crossentropy', metrics=['accuracy', tf.keras.
metrics.RecallAtPrecision(precision=0.8), tfa.metrics.F1Score(num_classes=2)])
```

Figure 3.11: Compile the model.

3.4.5 Training CNN model



```
history = model.fit(
    train_data,
    validation_data=val_data,
    batch_size=16,
    epochs=40,
    callbacks=[
        tf.keras.callbacks.EarlyStopping(
            monitor='val_loss',
            patience=10,
            restore_best_weights=True
        )
    ]
)
```

Figure 3.12: Fitting model.

The model.fit() function is used to train a machine learning model. It takes several parameters:

- **train_data:** This represents the training dataset, which is used to train the model. It is assumed to be already prepared and ready for training.
- **validation_data:** This represents the validation dataset, which is used to evaluate the model's performance during training. It helps in monitoring the model's generalization and prevents overfitting. It is assumed to be prepared and ready for evaluation.
- **batch_size:** This parameter determines the number of samples that will be propagated through the model at once during each training iteration. In this case, the batch size is set to 16, meaning that 16 samples will be processed in parallel before updating the model's weights.
- **epochs:** This parameter defines the number of times the entire training dataset will be passed through the model during training. In this case, the model will be trained for 40 epochs.
- **callbacks:** This parameter is a list of callback functions that are applied during training to perform certain actions based on specific conditions. In this case, a single callback is used:
  tf.keras.callbacks.EarlyStopping. It monitors the validation loss and stops training if the loss does not improve for a certain number of epochs (defined by the pa-

tience parameter). Additionally, it restores the best weights of the model, which were achieved during training.

3.5 Experiments and results

Specification of parameters initialization for this structure: In this first structure, we propose the following set of parameters in Table 3.3.

Parameters	**Informations**
input shape	(300,300,3)
Number of Epochs	40
batch size	16
Patience	10
Last output Layer	Dense(2)
loss	binary_crossentropy

Table 3.3: Table of parameters for the proposed structure.

To enhance the results of our experiment, we performed training on an augmented dataset. The performance of the model was evaluated using the confusion matrix and visualizations. Specifically, the following components were mentioned:

1. **visualize the results:** We evaluated the model using training accuracy, validation accuracy, training loss, and validation loss. To visualize the performance of the model, we present the following plots:
   • A plot of accuracy and validation accuracy over epochs:

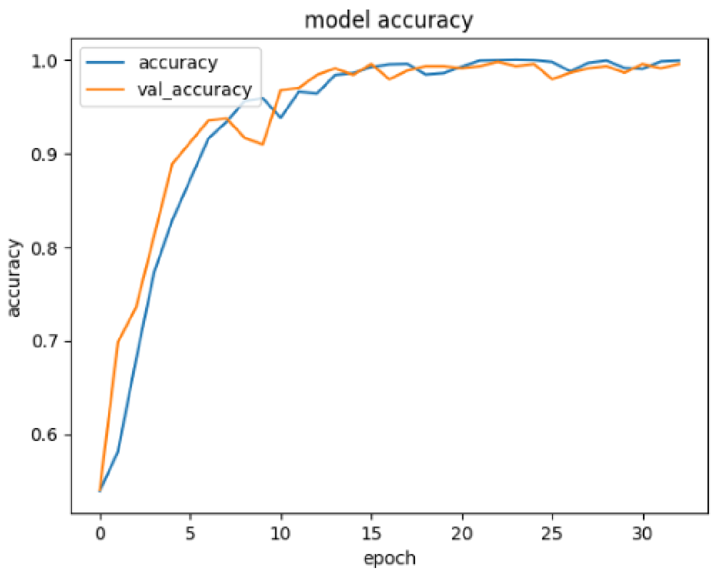


Figure 3.13: Model accuracy.

The model training process yielded good results without overfitting. This can be observed from the learning curves plotted using the training and validation data. The blue line represents the model's accuracy, showing its improvement with each epoch. Similarly, the orange line represents the validation accuracy, indicating how well the model generalizes to unseen data. Both lines show positive trends without significant divergence, suggesting that the model performs well and maintains its generalization capability throughout the training process. This indicates a successful training outcome with satisfactory performance.
• A plot of loss and validation loss over epochs:

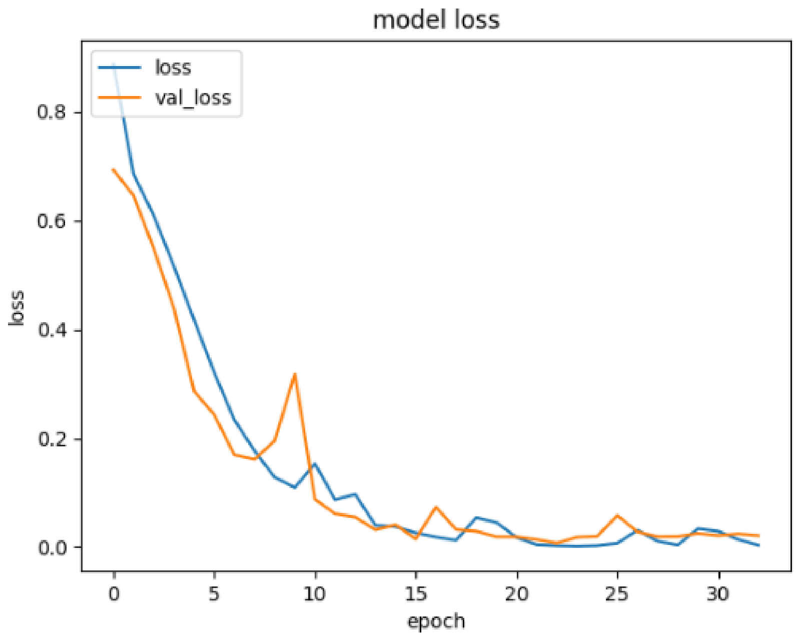


Figure 3.14: Model loss.

Furthermore, the obtained model loss serves as additional evidence to support the model's strong performance without overfitting. The consistent decrease in the loss for both the training and validation sets throughout the training process is indicative of the model's ability to generalize well and effectively learn from the data. This reinforces the notion that the model has been trained successfully and is performing well without exhibiting overfitting issues.
•plot ROC curve for each class:

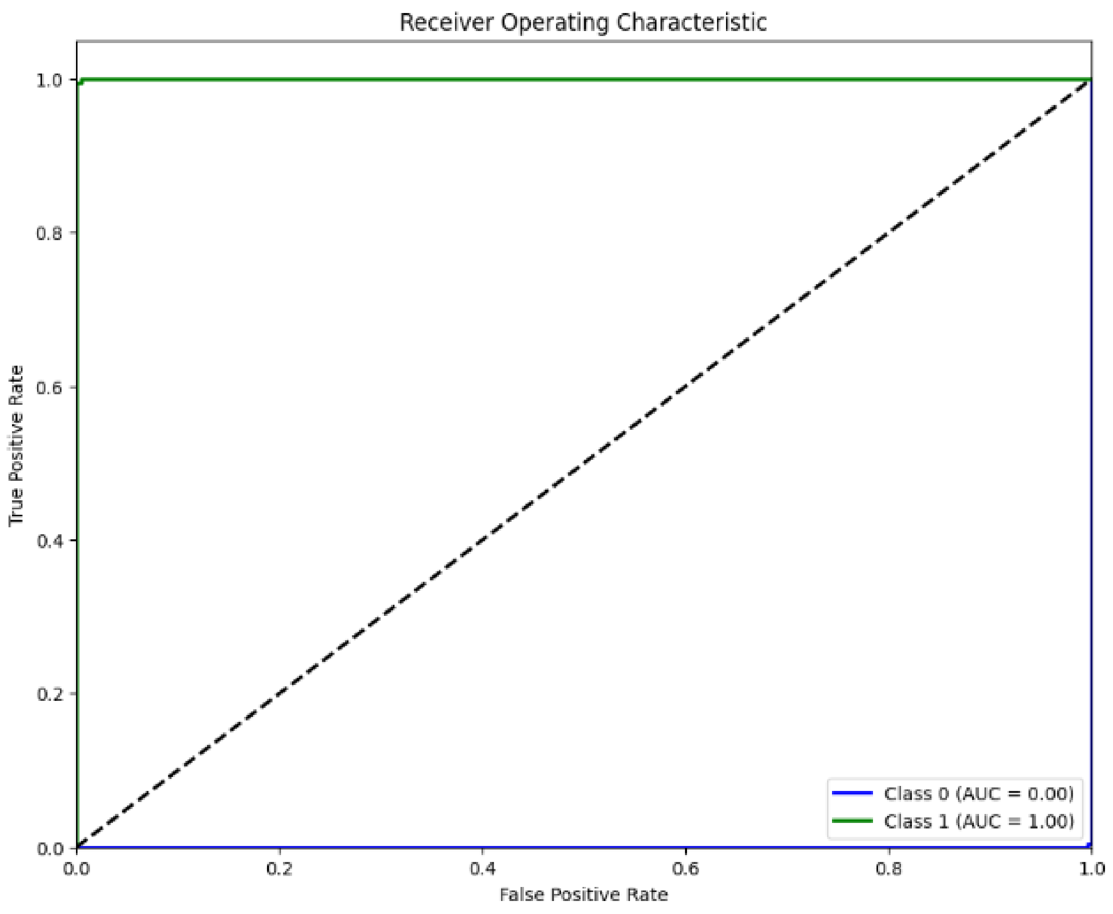


Figure 3.15: ROC curve for each class.

The model's performance was evaluated by computing the area under the receiver operating characteristic (ROC) curve for each class. The resulting AUC (Area Under the Curve) value is a measure of the model's predictive accuracy, with a higher value indicating better performance. These findings validate the effectiveness of the model and its ability to make accurate predictions across multiple classes.

2. **Regarding the confusion matrix:**

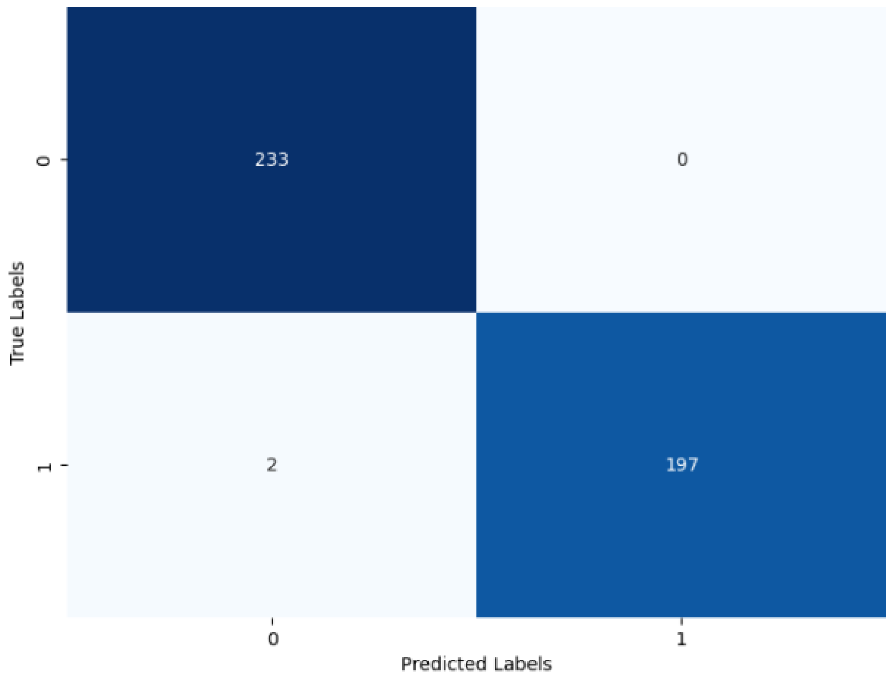
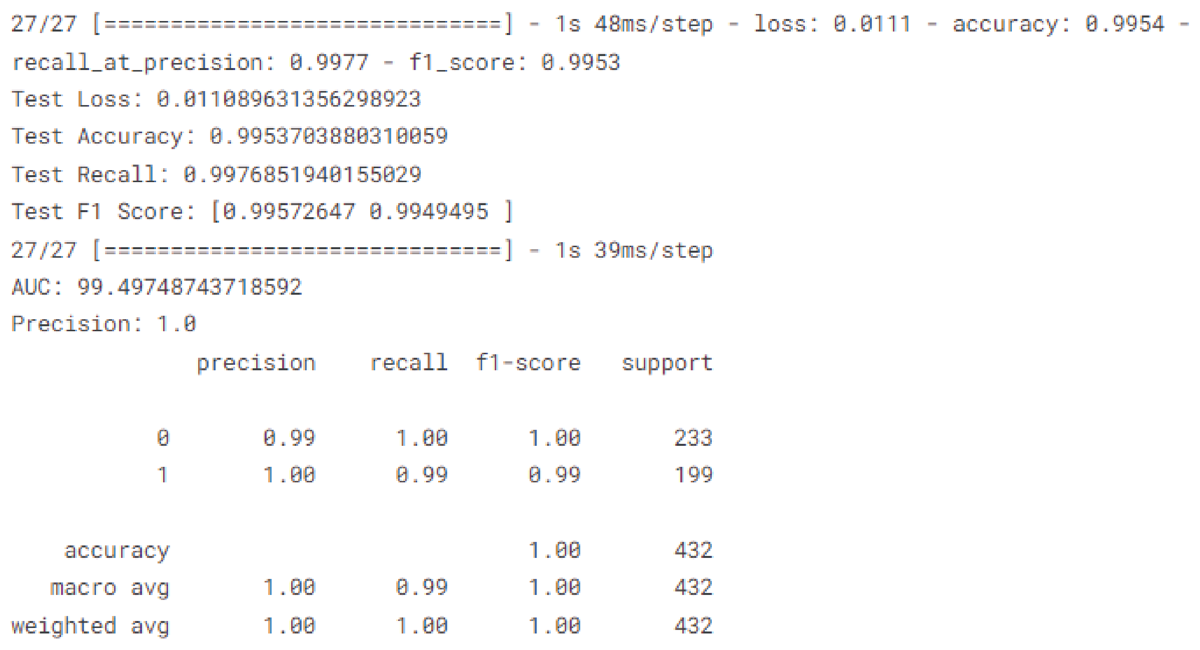


Figure 3.16: The confusion matrix.

The model's performance was assessed using a confusion matrix, which provides an overview of the model's predictions compared to the true labels. In this case, the confusion matrix shows that the model achieved good results. Out of 233 instances labeled as "fracture," the model correctly classified all of them. Similarly, out of 199 instances labeled as "normal," the model accurately predicted 197 of them. The small number of misclassifications demonstrates the model's effectiveness in accurately distinguishing between "fracture" and "normal" cases.
Compute the confusion matrix:

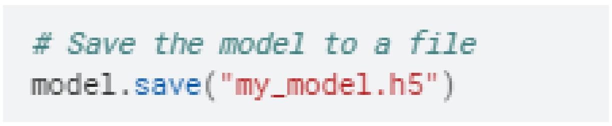


```
27/27 [==============================] - 1s 48ms/step - loss: 0.0111 - accuracy: 0.9954 -
recall_at_precision: 0.9977 - f1_score: 0.9953
Test Loss: 0.011089631356298923
Test Accuracy: 0.9953703880310059
Test Recall: 0.9976851940155029
Test F1 Score: [0.99572647 0.9949495 ]
27/27 [==============================] - 1s 39ms/step
AUC: 99.49748743718592
Precision: 1.0
              precision    recall  f1-score   support

           0       0.99      1.00      1.00       233
           1       1.00      0.99      0.99       199

    accuracy                           1.00       432
   macro avg       1.00      0.99      1.00       432
weighted avg       1.00      1.00      1.00       432
```

Figure 3.17: Compute the confusion matrix.

In experiment two, the model achieved a significantly higher accuracy of 99.54% on the test dataset, with a recall of 99.77% and an F1 score of 99.53%. The precision was measured at a perfect 100%. The confusion matrix showed that the model correctly predicted 233 out of 233 fracture images and 197 out of 197 normal images.The AUC was exceptionally high at 99.50%. The model demonstrated excellent performance in accurately classifying both classes, with a balanced precision, recall, and F1 score for both classes. The weighted average of the precision, recall, and F1 score was 100%, indicating outstanding overall performance.

In summary, the results of the experiments demonstrate that the model achieved high accuracy and recall, indicating its effectiveness in predicting fractures and normal cases. The second experiment outperformed the first one, suggesting that the enhanced dataset led to superior results.
Saving Model:
The best model was saved in epoch number 15. The performance metrics of this model are in Table 4.3.



```
# Save the model to a file
model.save("my_model.h5")
```

Figure 3.18: save model.

3.6 Our CNN Vs transfer learning

Model	**Recall**	**Precision**	**Accuracy**	**ROC**	**F1 score**
ResNet50V2	1.0	0.9704	0.9814	0.9814	0.9820
VGG16	1.0	1.0	0.9930	0.9924	0.9936
Proposed CNN	0.9976	1.0	0.9953	0.9949	0.9953

Table 3.4: Comparison of our work.

Based on the comparison of different models for cervical spine fracture detection, the results in Table 3.4 demonstrate the performance of each model in terms of recall, precision, accuracy, ROC, and F1 score. Based on these results, it can be concluded that all models achieved high performance, with the VGG16 model demonstrating the highest precision 100% and the proposed CNN model achieving the highest recall 100%. Therefore, we will use the proposed VGG16 or CNN for integration into our web interface, with each model exhibiting slightly different strengths and areas of improvement. Therefore, we will use the proposed VGG16 or CNN for integration into our interface.

3.7 Comparisons our Work and the previous works

In this part, we will compare the various results achieved from our proposed CNN madel with the previous study of Soaad M. et al.[14]. To make a comparison, we choose precision and precision.

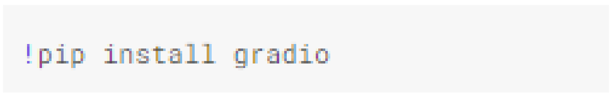
Model	**Precision**	**Accuracy**
Soaad M. et al.(2023)[14]	0.9955	0.9933
Our proposed CNN	0.9976	1.0

Table 3.5: Accuracy and precision comparisons.

Based on the accuracy and precision comparisons shown in Table 3.5, the proposed system achieved a precision of 0.9955 and an accuracy of 0.9933 when compared to previous works. On the other hand, our CNN proposed model demonstrated a higher precision of 0.9976 and a perfect accuracy of 1.0. These results indicate that the proposed CNN model outperforms both the previous works and the proposed system in terms of precision and accuracy. This suggests that our CNN model is more effective in accurately identifying and classifying cervical spine fractures.

3.8 Application deployment

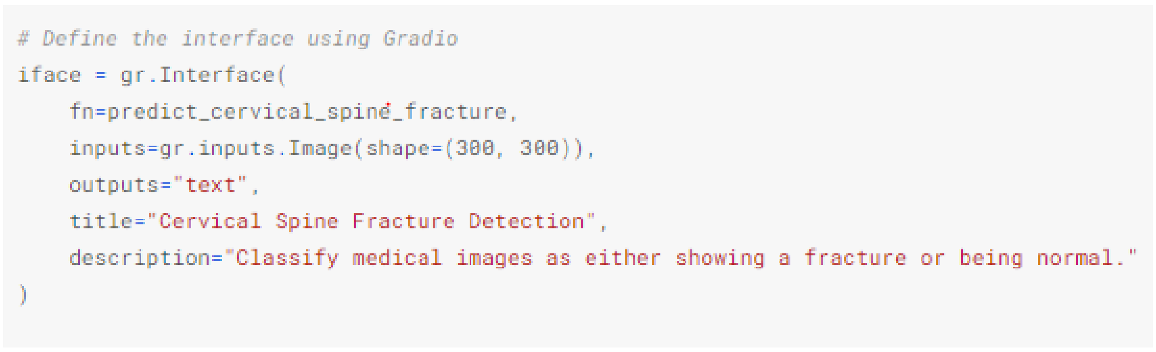
First, you need to ensure that you have the gradio library installed.



```
!pip install gradio
```

Figure 3.19: install a library of gradio.

After that, you create the iface object using gr. Interface, specifying the prediction function, the input type (an image with shape 300x300), the output type (text), and providing a title and description for the interface.



```
# Define the interface using Gradio
iface = gr.Interface(
    fn=predict_cervical_spine_fracture,
    inputs=gr.inputs.Image(shape=(300, 300)),
    outputs="text",
    title="Cervical Spine Fracture Detection",
    description="Classify medical images as either showing a fracture or being normal."
)
```

Figure 3.20: Define the interface.

The function predict_cervical_spine_fracture is defined to predict whether an input image of a cervical spine shows a fracture or is normal. Here is a description of the function:

The function takes an image as input and performs the following steps:

1. Convert the input image array to a PIL image using Image.fromarray function. The image is assumed to have RGB channels.

2. Resize the image to a fixed size of 300x300 pixels using the resize method.

3. Normalize the pixel values of the resized image by dividing each value by 255.0, which scales the pixel values between 0 and 1.

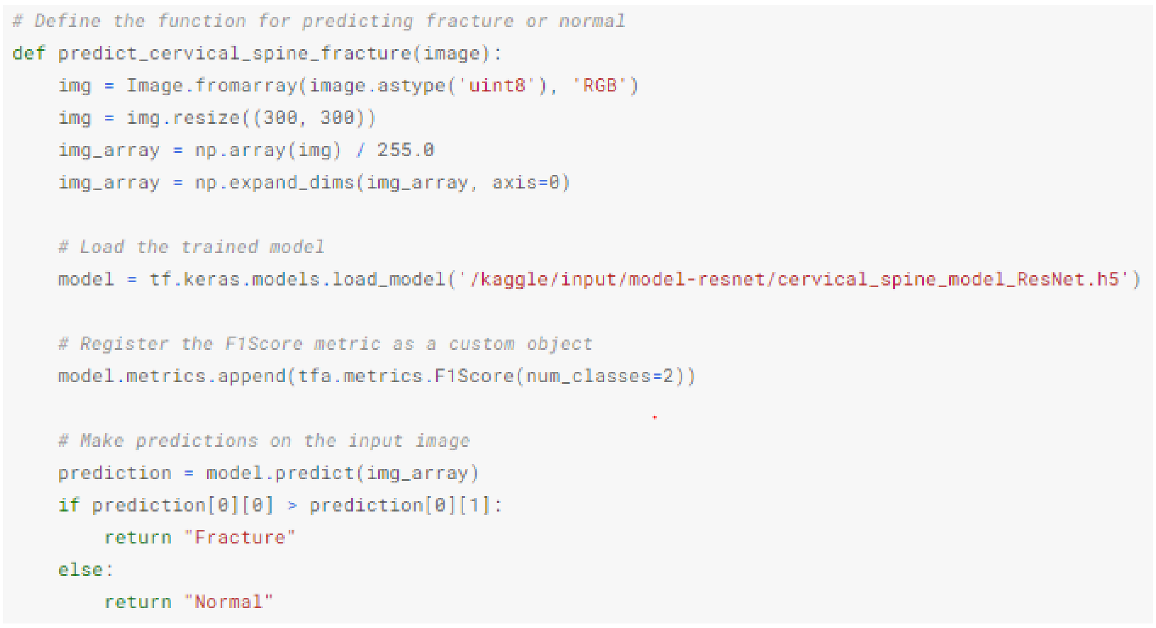
4. Expand the dimensions of the image array to match the expected input shape of the model. This is done by adding an extra dimension at the beginning, resulting in a shape of (1, 300, 300, 3).

5. Load the trained model for cervical spine fracture detection from the specified file path (/kaggle/input/model-resnet/cervical_spine_model_ResNet.h5).

6. Register the F1Score metric from the TensorFlow Addons (tfa) library as a custom object. This ensures that the metric is recognized when evaluating the model's performance.

7. Use the loaded model to make predictions on the input image. The predict method returns a prediction array, where prediction[0][0] represents the predicted probability of the image being classified as a fracture, and prediction[0][1] represents the predicted probability of the image being classified as normal.

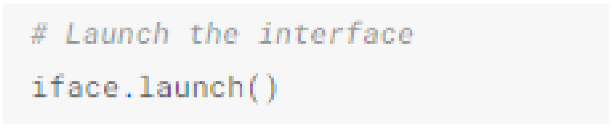
8. Compare the predicted probabilities and determine the final prediction. If prediction[0][0] is greater than prediction[0][1], the function returns "Fracture". Otherwise, it returns "Normal".



```
# Define the function for predicting fracture or normal
def predict_cervical_spine_fracture(image):
    img = Image.fromarray(image.astype('uint8'), 'RGB')
    img = img.resize((300, 300))
    img_array = np.array(img) / 255.0
    img_array = np.expand_dims(img_array, axis=0)

    # Load the trained model
    model = tf.keras.models.load_model('/kaggle/input/model-resnet/cervical_spine_model_ResNet.h5')

    # Register the F1Score metric as a custom object
    model.metrics.append(tfa.metrics.F1Score(num_classes=2))

    # Make predictions on the input image
    prediction = model.predict(img_array)
    if prediction[0][0] > prediction[0][1]:
        return "Fracture"
    else:
        return "Normal"
```

Figure 3.21: Function for predicting fracture or normal.

Finally, you can launch the interface using iface.launch(), and it will start a local web server where you can interact with the interface and test your cervical spine fracture detection model.



```
# Launch the interface
iface.launch()
```

Figure 3.22: Launch the interface.

example for testing the model in a web interface.

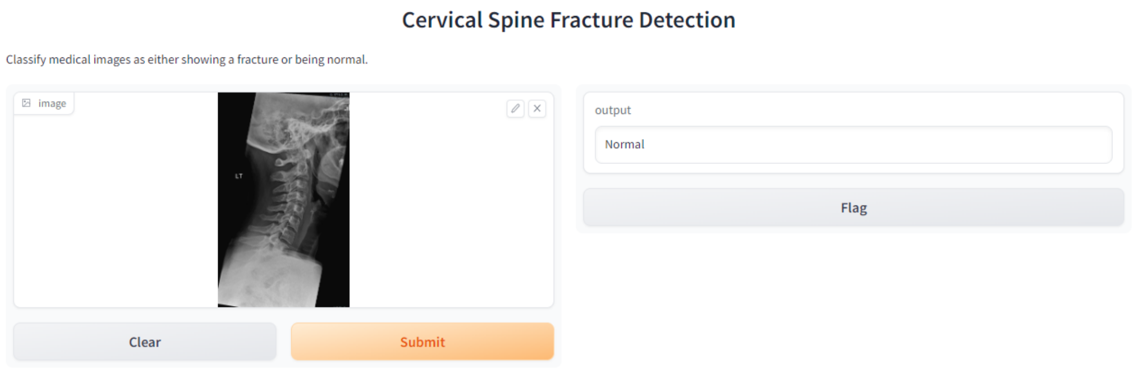

Cervical Spine Fracture Detection
Classify medical images as either showing a fracture or being normal.
image
output
Normal
Flag
Clear
Submit

Figure 3.23: web interface.

3.9 Conclusion

This chapter included the theme's primary findings, with the implementation and experiments conducted being elucidated. Annotated tables, plots, and the confusion matrix for the test phase are presented, showcasing the impressive performance of our model. The obtained results are compared, demonstrating the remarkable performance achieved by our model. These findings serve as a strong motivation to further enhance our model design, while also highlighting the development of the web interface.

Conclusion and future work

In conclusion, the application of deep learning (DL) in healthcare has garnered significant attention from researchers due to its suitability for various applications. Cervical spine fracture, a severe condition affecting a substantial portion of the global population, has been a focus of investigation in this study.

Throughout this dissertation, we proposed a modified deep learning architecture specifically designed for the detection and classification of cervical spine fracture levels. Our research involved an exploration of multiple projects and architectures aimed at image classification and cervical spine fracture detection. Building upon previous work, we introduced a variant architecture to enhance the existing results. We utilized a dataset for model training and validation purposes.

The obtained results demonstrate the efficacy of Convolutional Neural Networks (CNN) models in medical diagnosis. Our model exhibited favorable performance in terms of classifying the dataset. However, limitations in hardware and time constraints prevented us from extensively exploring additional preprocessing methods, datasets, and other deep learning techniques.

In summary, this research paves the way for numerous future endeavors, including model refinement, exploration of alternative deep learning techniques, and the investigation of different preprocessing methods.

Bibliography

[1] Parajuli A. Mechanical adaptation of diabetic and perlecan deficient skeletons. University of Delaware; 2020.

[2] Britannica. Vertebral Column;. Refer to 2023/05/30/19:52. Available from: https://www.britannica.com/science/vertebral-column.

[3] Agur AM, Dalley AF. Grant's atlas of anatomy. Lippincott Williams & Wilkins; 2009.

[4] ATLAS CORRECTION: SMALL VERTEBRA, LARGE INFLUENCE;. Refer to 30/05/2023/8:33. Available from: https://www.liebscher-bracht.com/en/encyclopedia-of-pain/atlas-correction/.

[5] Waxenbaum JA, Reddy V, Futterman B. In: Anatomy, Back, Cervical Vertebrae. Treasure Island (FL): StatPearls Publishing;. Refer to 30/05/2023/8:33. Available from: http://www.ncbi.nlm.nih.gov/books/NBK459200/.

[6] Dowling-Medley J. Biomechanics of the lower cervical spine during shear loading. University of British Columbia; 2018.

[7] The Concept of Artificial Neurons (Perceptrons) in Neural Networks;. Refer to 30/04/2023/4:20. Available from: https://towardsdatascience.com/the-concept-of-artificial-neurons-perceptrons-in-neural-networks-fab22249cbfc.

[8] Hidden Layers in a Neural Network;. Refer to 28/05/2023/2:35. Available from: https://www.baeldung.com/cs/hidden-layers-neural-network.

[9] Activation functions in Neural Networks;. Refer to 30/05/2023/19:45. Available from: https://www.geeksforgeeks.org/

activation-functions-neural-networks/.

[10] Basic CNN Architecture: Explaining 5 Layers of Convolutional Neural Network;. Refer to 30/05/2023/10:05. Available from: https://www.upgrad.com/blog/basic-cnn-architecture/.

[11] Convolutional Neural Network;. Refer to 30/05/2023/19:40. Available from: https://www.sciencedirect.com/topics/mathematics/convolutional-neural-network.

[12] Everything you need to know about VGG16;. Refer to 30/05/2023/10:20. Available from: https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918#:~:text=with%20transfer%20learning.-,VGG16%20Architecture,layers%20i.e.%2C%20learnable%20parameters%20layer.

[13] Qazi EUH, Zia T, Almorjan A. Deep Learning-Based Digital Image Forgery Detection System. Applied Sciences. 2022;12(6):2851.

[14] Naguib SM, Hamza HM, Hosny KM, Saleh MK, Kassem MA. Classification of Cervical Spine Fracture and Dislocation Using Refined Pre-Trained Deep Model and Saliency Map. Diagnostics. 2023;13(7):1273.

[15] Rosenberg GS, Cina A, Schiró GR, Giorgi PD, Gueorguiev B, Alini M, et al. Artificial Intelligence Accurately Detects Traumatic Thoracolumbar Fractures on Sagittal Radiographs. Medicina. 2022;58(8):998.

[16] Murata K, Endo K, Aihara T, Suzuki H, Sawaji Y, Matsuoka Y, et al. Artificial intelligence for the detection of vertebral fractures on plain spinal radiography. Scientific Reports. 2020;10(1):1-8.

[17] Marieb EN, Hoehn K. Human anatomy & physiology. Pearson education; 2007.

[18] Tortora GJ, Derrickson BH. Principles of anatomy and physiology. John Wiley & Sons; 2018.

[19] Odya E, Norris MA. Anatomy & Physiology for Dummies. John Wiley & Sons; 2017.

[20] Patton KT, Thibodeau GA, Hutton A. Anatomy and Physiology Adapted International Edition E-Book. Elsevier Health Sciences; 2019.

[21] Standring S, Ellis H, Healy J, Johnson D, Williams A, Collins P, et al. Gray's anatomy: the anatomical basis of clinical practice. American journal of neuroradiology. 2005;26(10):2703.

[22] Cramer GD, Darby SA. Clinical anatomy of the spine, spinal cord, and ANS. 2013.

[23] Thibodeau GA, Patton KT. Anatomy & physiology. Mosby; 2007.

[24] Moore KL, Dalley AF, Agur AM. Clinically oriented anatomy. Lippincott Williams & Wilkins; 2013.

[25] Kaiser JT, Reddy V, Lugo-Pico JG. Anatomy, head and neck, cervical vertebrae. 2019.

[26] Betts JG, Young KA, Wise JA, Johnson E, Poe B, Kruse DH, et al. Anatomy and physiology; 2013.

[27] Netter FH. Atlas of human anatomy, Professional Edition E-Book: including NetterReference. com Access with full downloadable image Bank. Elsevier health sciences; 2014.

[28] Drake RL, Vogl AW, Mitchell A. Gray's anatomy for students. ed. Churchill Livingstone Elsevier, Philadelphia, Pa[ua]. 2015.

[29] Faulkner R, Lockwood P. Could posterior-anterior projection cervical spine radiographs improve image quality and dose reduction. Radiography Open. 2022;8(1):38-50.

[30] Mark J Spoonamore MD. Cervical Spine Fractures Dislocations;. Refer to 30/05/2023/10:30. Available from: `https://www.uscspine.com/conditions-treated/neck-disorders/cervical-spine-fractures-dislocations/`.

[31] McMordie JH, Viswanathan VK, Gillis CC. Cervical spine fractures overview. 2017.

[32] Hutton MJ, McGuire RA, Dunn R, Williams R, Robertson P, Twaddle B, et al. Catastrophic cervical spine injuries in contact sports. Global spine journal. 2016;6(7):721-34.

[33] Torlincasi AM, Waseem M. Cervical injury. 2017.

[34] Holdsworth F. Fractures, dislocations, and fracture-dislocations of the spine. The Journal of Bone and Joint Surgery British Volume. 1963;45(1):6-20.

[35] William Schecter M. Spine and Spinal Cord Injuries;. Refer to 30/05/2023/10:35. Available from: https://docplayer.net/13692897-Spine-and-spinal-cord-injuries-william-schecter-md.html.

[36] DiPompeo CM, Das JM. Subaxial cervical spine fractures. In: StatPearls [Internet]. StatPearls Publishing; 2022. .

[37] Beeharry MW, Moqeem K, Rohilla MU. Management of cervical spine fractures: a literature review. Cureus. 2021;13(4).

[38] Jain AK, Sahu D. Indian Journal of Orthopaedics: The journey so far. Indian Journal of Orthopaedics. 2010;44(1):1.

[39] Kwon BK, Vaccaro AR, Grauer JN, Fisher CG, Dvorak MF. Subaxial cervical spine trauma. JAAOS-Journal of the American Academy of Orthopaedic Surgeons. 2006;14(2):78-89.

[40] Alpaydin E. Introduction to machine learning. MIT press; 2020.

[41] Awad M, Khanna R. Efficient learning machines: theories, concepts, and applications for engineers and system designers. Springer nature; 2015.

[42] Top 10 Deep Learning Algorithms You Should Know in 2023;. Refer to 29/05/2023/7:20. Available from: https://www.simplilearn.com/tutorials/deep-learning-tutorial/deep-learning-algorithm.

[43] Mhatre S. What Is The Relation Between Artificial And Biological Neuron?;. Refer to 29/05/2023/7:40. Available from: https://smhatre59.medium.com/what-is-the-relation-between-artificial-and-biological-neuron-18b05831036.

[44] Overview of neuron structure and function;. Refer to 28/05/2023/2:30. Available from: https://www.khanacademy.org/science/biology/human-biology/neuron-nervous-system/a/overview-of-neuron-structure-and-function.

[45] Millstein F. Convolutional neural networks in Python: beginner's guide to convolutional neural networks in Python. Frank Millstein; 2020.

[46] Zaki SZM, Zulkifley MA, Stofa MM, Kamari NAM, Mohamed NA. Classification of tomato leaf diseases using MobileNet v2. IAES International Journal of

Artificial Intelligence. 2020;9(2):290.

[47] Brownlee J. A gentle introduction to transfer learning for deep learning. Machine Learning Mastery. 2017;20.

[48] Small J, Osler P, Paul A, Kunst M. Ct cervical spine fracture detection using a convolutional neural network. American Journal of Neuroradiology. 2021;42(7):1341-7.