الجمهورية الجزائرية الديمقراطية الشعبية

The People's Democratic Republic of Algeria

وزارة التعليم العالي والبحث العلمي

Ministry of Higher Education and Scientific Research

جامعة محمد بوضياف بالمسيلة

University Mohamed Boudiaf of M'sila

كلية الرياضيات والإعلام الآلي

Faculty of Mathematics and Computer Science

قسم الإعلام الآلي

Department of Computer Science

Domain: Mathematics and Computer Science

End of Cycle Project Report

Presented to Fulfill the Partial Requirement

for **Master Degree** in Computer Science

Specialty: Information System and Software Engineer

Prepared By: Torki Yasser

Supervised By: Dr.CHALABI Nour elhouda

THEME

Improving Medical Image Classification Using Vision transformers

Academic Year 2024/2025

Dedication

I dedicate this work to my **mother** you were my guiding light, my source of strength, support, and guidance. You have taught me to believe in myself,
and to always persevere , I am truly thankful and honored to have you as my mother.

And to my **father** who have provided me with his encouragement,
love and understanding,

To my brother **Othman** for his support. to all my extended family, to all my friends and teachers at the University of M'sila, to all who were there for me, thank you for your help and encouragement, to all those who have been supportive, caring and patient, I dedicate this simple work.

Acknowledgements

I would like to express my thanks to supervisor Dr.Nour elhouda CHALABI who supported and followed up this work. Also great thanks to the jury president and jury members for agreeing to judge this work. I would like to thank the managers and all the staɣof the computer science Department of M'sila for the facilities they have granted us to complete this work.

Table of contents

List of figures and tables

Chapter 01 Deep Learning

Chapter 2: Vision Transformers (Vits)

Chapter 3: Transformer Architectures: Attention Mechanisms and Vision Applications

Chapter 4: Implementation and experimental results

List of equations

List of Table

Abstract

The integration of Vision Transformers (ViTs) into the field of medical imaging has opened new avenues for accurate, data-driven diagnostics by leveraging self-attention mechanisms to capture global contextual information. in this project we explored the application of ViTs to various medical imaging tasks, including classification, segmentation, and anomaly detection in X-raysour project utilizes the Vision Transformer (ViT) deep learning technique to accurately classify medical images from a chest X-ray dataset, demonstrating its effectiveness in various fields like computer vision and medical applications.

Keywords: Vision Transformer, deep learning, Convolutional Neural Networks, medical image classification, chest X-ray, computer vision, hyperparameters, model efficiency, healthcare improvement.

ملخص

لقد فتح دمج محولات الرؤية (ViTs) في مجال التصوير الطبي آفاقًا جديدة للتشخيص الدقيق القائم على البيانات، وذلك من خلال الاستفادة من آليات الانتباه الذاتي لالتقاط المعلومات السياقية العالمية. في هذا المشروع، استكشفنا تطبيق محولات الرؤية على مهام التصوير الطبي المختلفة، بما في ذلك التصنيف والتجزئة والكشف عن الشذوذ في الأشعة السينية. وقد يستخدم مشروعنا تقنية التعلم العميق لمحول الرؤية (ViT) لتصنيف الصور الطبية بدقة من مجموعة بيانات الأشعة السينية للصدر، مما يُظهر فعاليتها في مجالات مختلفة مثل الرؤية الحاسوبية والتطبيقات الطبية.

الكلمات المفتاحية: محول الرؤية، التعلم العميق، الشبكات العصبية التلافيفية، تصنيف الصور الطبية، الأشعة السينية للصدر، الرؤية الحاسوبية، المعلمات الفائقة، كفاءة النموذج، تحسين الرعاية الصحية.

Résumé

L'intégration des Vision Transformers (ViT) dans le domaine de l'imagerie médicale a ouvert de nouvelles perspectives pour des diagnostics précis et basés sur les données, en exploitant les mécanismes d'auto-attention pour capturer des informations contextuelles globales. Dans ce projet, nous avons exploré l'application des ViT à diverses tâches d'imagerie médicale, notamment la classification, la segmentation et la détection d'anomalies dans les rayons X. Notre projet utilise la technique d'apprentissage profond Vision Transformer (ViT) pour classer avec précision les images médicales à partir d'un ensemble de données de radiographies thoraciques, démontrant ainsi son efficacité dans divers domaines tels que la vision par ordinateur et les applications médicales.

Mots-clés : Vision Transformer, apprentissage profond, réseaux de neurones convolutifs, classification d'images médicales, radiographie thoracique, vision par ordinateur, hyperparamètres, efficacité des modèles, amélioration des soins de santé.

General introduction

Medical Image classification performs a necessary position in cutting-edge healthcare, enabling quicker and greater correct analysis of a number of diseases. Traditional deep studying models, specifically Convolutional Neural Networks (CNNs), have verified full-size success in this domain. However, CNNs have inherent obstacles in shooting long-range dependencies and international context, which are frequently quintessential in inspecting complicated scientific images.

Vision Transformers (ViTs), in the beginning proposed for herbal picture classification, have these days received interest for their capability to mannequin international relationships inside an photograph the use of self-attention mechanisms. Unlike CNNs, ViTs do now not matter on convolutional filters, permitting them to seize greater complete spatial statistics throughout the complete image. This attribute makes them quite promising for enhancing the overall performance of scientific photograph classification tasks, such as detecting lung diseases, tumors, or retinal disorders.

This undertaking targets to discover the effectiveness of Vision Transformers (ViTs) in classifying clinical photographs and examine their overall performance to normal CNN-based models. By leveraging ViTs' international interest capabilities, the venture seeks to beautify classification accuracy and reliability, subsequently contributing to higher medical decision-making.

Manuscript Organization

This document is organized into the following chapters:

Chapter 1: Deep learning.

Chapter 2: Vision Transformers (Vits)

Chapter 3: Transformer Architectures: Attention Mechanisms and Vision Applications

Chapter 4: Implementation and experimental results

Chapter 01

Deep Learning

Introduction:

Artificial intelligence (AI) is the scientific discipline of creating computer programs that perform operations comparable to those of the human mind, such as learning or logical reasoning. Machine learning (ML) goes further, it is this branch of AI that allows machines to learn by themselves, without relying on commands. It refers to the development, analysis and implementation of methods that allow a machine to evolve and perform tasks impossible for a human being through a learning process. Deep learning (DL) is a type of machine learning, but more complex. It is a set of ML methods attempting to model with a high level of data abstraction through articulated architectures of different nonlinear transformations. In this chapter we will first present the basic notions of machine learning and define deep learning, then we will present neural networks and their main architectures.

1. Machine learning:

According to Tom Mitchell **[1]** :"A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E". Machine learning is an area of artificial intelligence concerned with the development of techniques that allow computers to learn. Learning is the ability of the machine to improve its performance based on previous results **[2]**.

1.1.Types of Machine learning:

Machine learning is subdivided according to the nature of the input data and then according to the nature of the output data.

Machine learning according to the nature of the input data

- Supervised learning: This is a form of machine learning that creates artificial intelligence models based on "tagged" learning data.
- Unsupervised learning: It's a machine learning technique that looks for a structure of unlabeled data.
- Reinforcement learning: the algorithm learns a behavior from an observation. The

algorithm's action on the environment produces a return value guiding the learning algorithm

Machine Learning according to the nature of the output data

1) **Classification**: classification is a type of supervised learning, which identifies the category to which a new element belongs on the basis of a data set of data training containing observations of which the category is known **[3]**. There are several types of classification algorithms:

   ● Decision Tree: Classifies the use of a tree structure with if-then rules, executing the input through a series of decisions until it reaches a termination condition. Able to model complex and highly intuitive decision processes, but can easily oversize data –Random forest : a set of decision trees, with automatic selection of the most efficient tree. Provides the strength of the decision tree algorithm without over-learning problems **[4]**.

   ● Naïve Bayes classifier: a probability-based classifier. Calculates the probability that each data point exists in each of the target categories. Simple to implement and precise for a wide range of problems, but sensitive to all the categories selected.**[5]**.

   ● K-Nearest Neighbor: Classifies each data point by analyzing its closest neighbors among learning examples **[6]**. Simple to implement and understand, effective for many problems, especially those with low dimension. Provides lower accuracy than the supervised algorithms, and requires a lot of calculations.

2) **The Regression**: regression is a supervised learning technique, it is the pre- diction of continuous numerical values for a set of data from a learning base, it predicts a continuous output variable (y) based on the value of one or more predictor variables (x). The inputs are called independent values, the output is called the dependent value. There are weights called coefficients, which deter- mine how much each input value contributes to the result, or its importance **[7]**. There are many types of regression :

   ● Linear regression: suitable for dependent values that can be adjusted with a straight line (linear function).

   ● Polynomial regression: suitable for dependent variables that can be adjusted by a curve or series of curves.

   ● Logistic regression: adapted to dependent variables that are binary and therefore not normally distributed.

2. Deep Learning:

The idea of Deep Learning is not a recent one, but it actually dates back to the 1980s, more particularly following the work of multilayer neural networks and the work of some pioneers of Machine Learning and Deep Learning such as the French Yann LeCun **[8]**. Together with two other computer scientists, Kunihiko Fukushima and Geoffrey Hinton, they developed a particular type of algorithm called the Convolutional neural network (CNN)**[9]**. Although this approach gives its results, its progress and evolution is limited by technological advances in microprocessors, computational power, and lack of access to data to train neural networks. However, some researchers have continued to work on this model for about two decades and with the help of technological developments, but especially with the ever-increasing availability of data, have been able to improve this technique. In 2007 the Stanford Vision Lab, with Fei-Fei Li at its head, developed an aggregator of images where several million ImageNet photos are recorded and tagged. In 2010, ImageNet included 15 million images all categorized according to their own characteristics (vehicles, animals, etc.). In 2012 Deep Learning was brought up to date with a resounding success at the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) which is an annual image recognition competition founded by Stanford University, as part of its STANFORD VISION LAB. Several teams of computer scientists compete in this competition every year to award the victory to the program with the lowest failure rate. And while deep learning algorithms are absent from the competition, in 2012 it is indeed a Deep Learning algorithm that will win the 2012 edition to the general surprise **[10]**.

2.1.Definition:

Deep learning is a process that allows computers to learn to execute activities that are inherent in the brain, suc

h as picture identification, as a new branch of machine learning research. Today, the deep learning approach DL is the new trend in machine learning, as it provides far more advanced pattern recognition and picture categorization than the old machine learning approach ML **[11].** Figure.I.1 shows an illustrative diagram of deep learning with multiple layers.

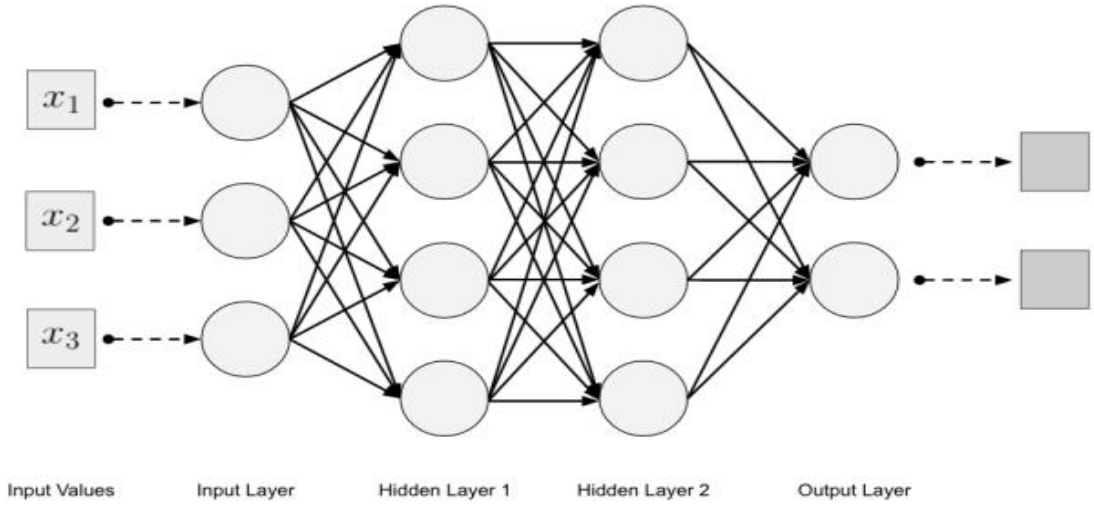

Figure1. 1 multilayer neural network architecture **[7]**.

2.2.Advantages of Deep Learning :

Deep learning is a true revolution in AI and goes well beyond machine learning. It is able to solve a wide variety of AI problems, such as:

- Improve traditional development in AI tasks.
- Exploit a large amount of data such as big data.
- Adapt to several types of problems.
- Extract characteristics automatically.

Figure 1.2 below explains the difference between machine and deep learning. Machine learning depends on a human-generated dataset that teaches the algorithm how to define data. On the other hand, deep learning, when a neural network processes an input, it creates its own layers depending on the input and output of the data. To recap the difference between machine learning and deep learning:

- Machine learning uses algorithms to analyze data, learn from it and make informed decisions based on what they have learned.
- Deep learning structures layered algorithms to create an artificial neural network (ANN) capable of learning and making intelligent decisions on its own.
- Deep learning is a subfield of machine learning. Although both belong to the broad category of artificial intelligence, deep learning is what powers the most human-like artificial intelligence.

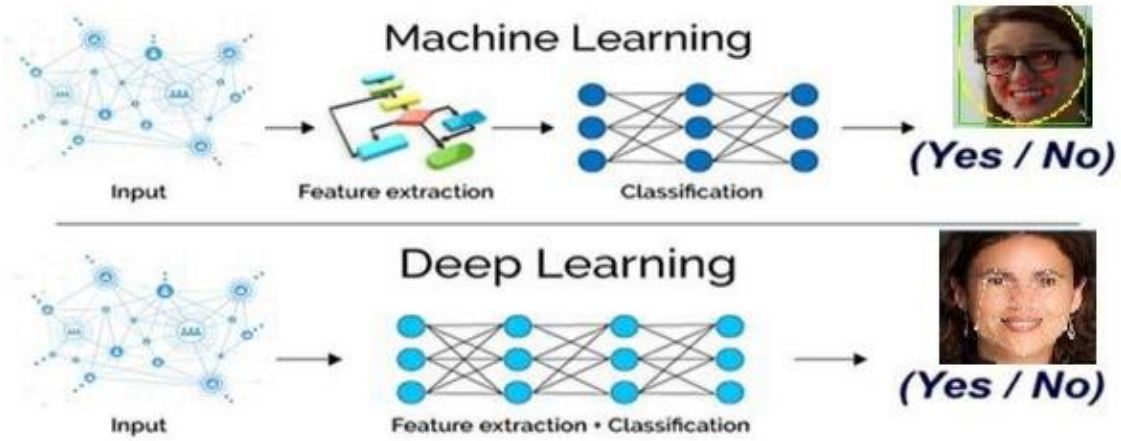

Figure1. 2 : Difference between Machine Learning and Deep Learning**[8]**.

2.3.Application domains of deep learning:

Learning models are developing in the field of computer science applied in robotics, bioinformatics , pattern recognition , cyber security, healthcare, Computer-aided pedagogy and more generally artificial intelligence. Deep learning can, for example, allow a computer to better recognize highly deformable objects and/or analyze for example the emotions revealed by a photograph or filmed face, or analyze the movements and position of the fingers of a hand, which can be useful for translating sign languages, improving the automatic positioning of a camera, etc. They are used for certain forms of medical diagnostic assistance (e.g.: automatic recognition of cancer in medical imaging), or prospective or prediction (e.g., predicting the properties of a soil filmed by a robot).

3. Neural networks:

Neural networks were born in 1943 thanks to W.MCCulloch and W.Pitts **[12]**, they were able to demonstrate that the brain is equivalent to a Turing machine, thought then becomes purely material and logical mechanisms. They stated in 1955 ”The more we learn about organisms, the more we are led to conclude that they are not merely analogous to machines, but what about this machine. McCulloch and Pitts’ demonstration was one of the important players in the creation of cybernetics. In 1949, D. Hebb **[13]** presented in his book “The Organization of Behavior” alearning rule. Many network models today are still inspired by Hebb’s rule. In 1958, F.Rosenblatt [10] developed the Perceptron model. It is a neural network inspired by the visual system. It has two layers of neurons, a perception layer and a decision-making layer. It is the first artificial system capable of learning by experience. In the

same period, the ADALINE model (ADAptive LINear Element) was presented by B.Window **[14]**, an American researcher at Stanford. This model will subsequently be the basic model of multilayer networks. In 1969, M. Minsky and S.Papert published a review of the properties of the Perceptron. This is going to have a big impact on research in this area. It will decrease sharply until 1972, when T.Kohonen **[15]** presented his work on associative memories and proposed applications to pattern recognition. It was in 1982 that J.Hopfield presented his study of a completely looped network, of which he analyzed the dynamics **[16]**

3.1.Definition:

A neural network is the association, in a more or less complex graph, of elementary objects, the formal neurons. The main networks are distinguished by the organization of the graph, that is to say their architecture, its level of complexity (the number of neurons, presence or not of feedback loops in the network), by the type of neurons (their functions of transition or activation) and finally, the goal: supervised or unsupervised learning, optimization, dynamic systems, etc**[17]**.

3.2.Artificial and deep neural networks:

Artificial neural networks are a supervised learning system made up of a large number of simple elements, called neurons or perceptrons. Each neuron can make simple decisions and feed its decisions to other neurons, organized in interconnected layers. Together, the neural network can emulate almost every function and answer virtually every question, with enough samples of training and computing power. A shallow neural network consists of only three layers of neurons as shown in Figure 1.3.

- An input layer that accepts model-independent variables or inputs.
- A hidden layer.
- An output layer that generates predictions.

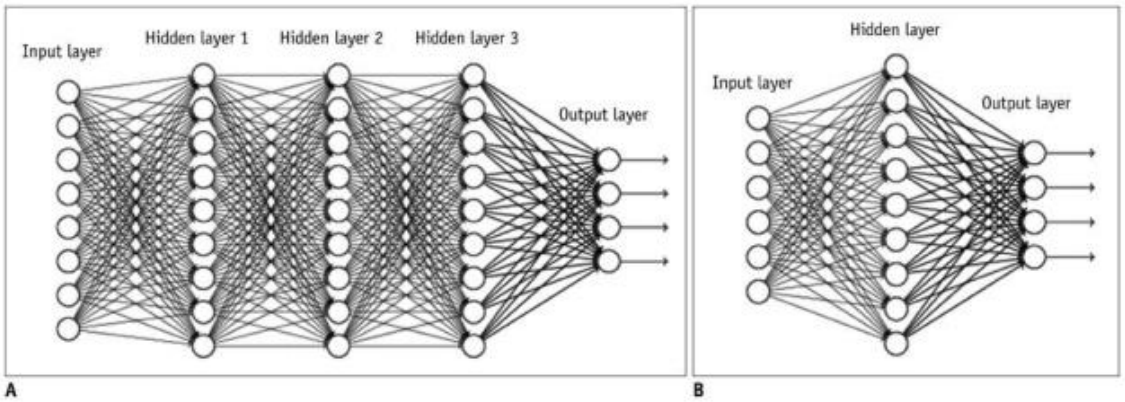

Figure1. 3 the difference between deep and shallow neural networks **[18]**.

A deep neural network has a similar structure, but it has two or more ”hidden layers” of neurons that process inputs (see Figure 1.3.(A)). Goodfellow, Bengio and Courville **[19]** have shown that while shallow neural networks are capable of solving complex problems, deep neural networks are more accurate and improve accuracy as new layers of neurons are added. Additional layers are useful up to a limit of 9 to 10. Today, most neural network models and implementations use a deep network between 3 and 10 layers of neurons.

3.3.Activation functions:

An activation function is a mathematical equation that determines the output of each element (perceptron or neuron) in the neural network. As shown in Fig 1.4, it takes the input of each neuron and transforms it into an output, usually between a 0 or -1 and 1. Classic activation functions used in neural networks include the sigmoid function and tanh. New activation functions, intended to improve the efficiency of neural networks, include ReLu and Swish **[20]**.

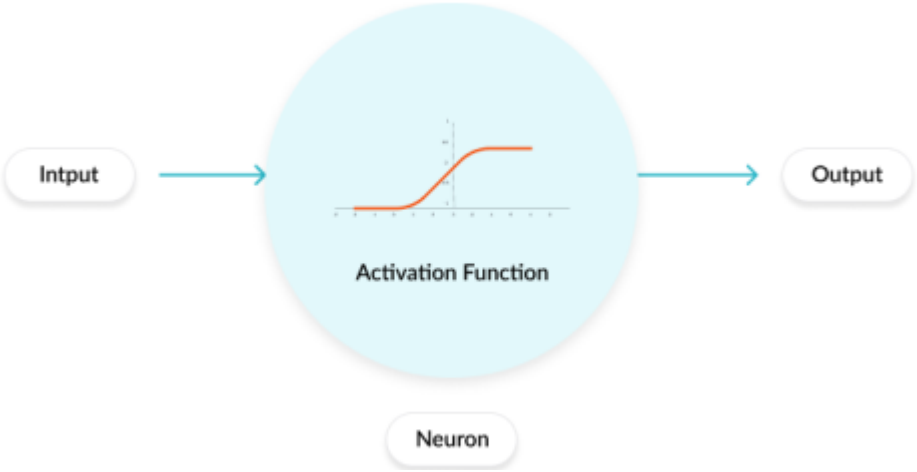

Figure1. 4 Simplified structure of an activation function of an artificial neuron**[21]**.

In a neural network, inputs, which are usually real values, are fed into the neurons in the network. Each neuron has a weight, and the inputs are multiplied by the weight and fed into the activation function. The activation function is the function that makes it possible to process the information that arrives at an artificial neuron in machine learning, as those of the brain do with the electrical signals they receive. The output of each neuron is the input to neurons in the next layer of the network, and so the inputs flow through multiple activation functions until finally, the output layer generates a prediction. Neural networks rely on nonlinear activation functions, the derivative of the activation function helps the network learn through the process of backpropagation. Main existing activation functions:

1. **Sigmoid**: It generates values between zero and one. For very high or low values of input parameters, the network can be very slow to achieve a prediction.

$$f(x) = \frac{1}{[1+e^{-x}]} \quad (1.1)$$

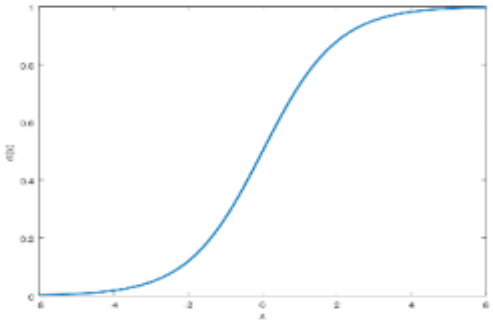

Figure1. 5 Sigmoid function curve.

2. **TanH** : Zero-centered, making it easy to model strong negative, strong positive, or neutral inputs.

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \quad (1.2)$$

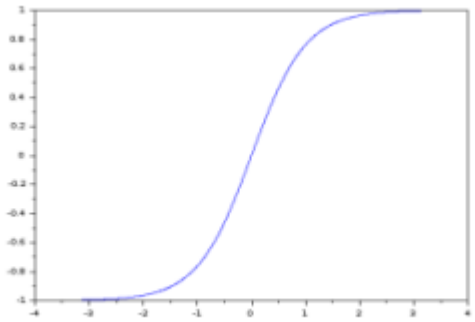

Figure1. 6 TanH function curve.

3. **ReLu**: Very computationally efficient but unable to process inputs that approach zero or negative.

$$f(x) = \mathbf{max}(\mathbf{x}, \mathbf{0}) \quad (1.3)$$

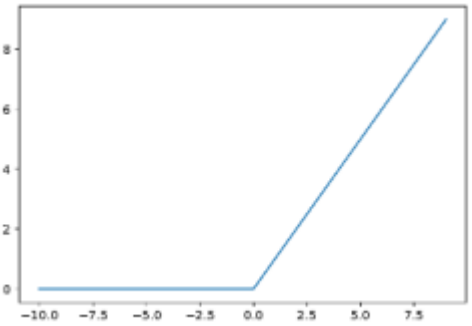

Figure1. 7 ReLu function curve.

3.4. The main architectures of neural networks:

A neural network can take different forms depending on the object of the data it processes and according to its complexity and the method of processing the data. The architectures have their strengths and weaknesses which makes them more suitable for a given task, they can be combined to optimize the results. The choice of architecture is thus crucial, determining the objective to be achieved is the best way to choose the most appropriate approach according to its advantages and especially its constraints. In what follows, we present the main architectures of neural networks.

3.4.1 Convolutional Neural Networks (CNN)

Convolutional neural networks, also called convnet (for ”Convolutional Network”), or CNN (for ”Convolutional Neural Network”), have proven to be very effective for tasks involving closely related data, mainly in the field of vision. by computer. There are two parts in a CNN, a first part which is called the convolutional part of the model and the second part, which we will call the classification part of the model which corresponds to an MLP (Multilayer Perceptron) model**[22]**. It is a multilayer neural network and more precisely it is a deep network composed of four types of layers: the convolution layer, the pooling layer, the ReLU correction layer and the fully connected layer.

● **Convolution layer**: this is the most important layer and the heart of the constituent elements of the convolutional network, and it is also the one that performs the most heavy calculations. Its purpose is to identify the presence of a set of features in the images received as input.

● **Pooling layer**: is often placed between two convolution layers: it receives several feature maps as input, and applies the pooling operation to each of them. The pooling operation consists in reducing the size of the images, while preserving their important characteristics.

● **The ReLU correction layer**: therefore replaces all the negative values received as inputs with zeros. It acts as an activation function.

● **The fully-connected layer**: always constitutes the last layer of a neural network, convolutional or not, it is therefore not characteristic of a CNN. This type of layer receives a vector as input and produces a new vector as output. For this, it applies a linear combination then possibly an activation function to the values received as input .

The following figure shows the architecture of a CNN type neural network:

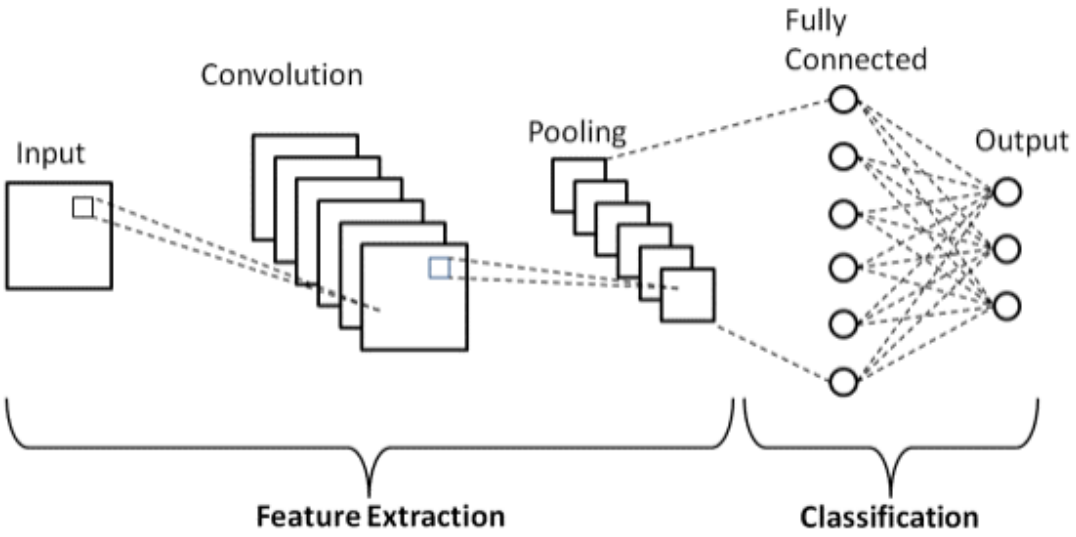

Figure1. 8 How a convolutional neural network works**[23]**.

CNN-type neural networks are used in particular in:

- Facial recognition.
- Identify and classify everyday objects in pictures.
- Powering vision in robots and autonomous vehicles.
- Recognize scenes and suggest relevant captions.

3.4.2 Recurrent Neural Networks (RNN)

A recurrent neural network (RNN) is a type of feedforward neural network that can handle varying sequences. RNN can handle time series because it has a recurrent hidden state whose activation is dependent on the activation of the prior time. Long short-term memory units (LSTMs) are a form of RNN that allows each recurrent unit to adapt to multiple time scale dependencies. Figure that follows shows an unwrapped RNN into a full network. By unwinding, we simply mean that we are writing the network for the complete sequence. For example, if the sequence we are interested in is a 5-word sentence, the network will be unrolled in a 5-layer neural network, one layer for each word.

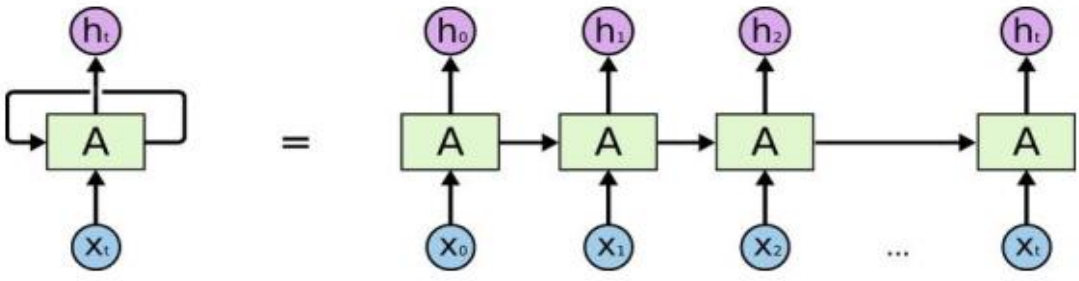

Figure1. 9 Unrolled neural network**[24]**.

An RNN examines a series of inputs over time, X0, X1, X2, up to Xt. For example, it could be a sequence of images in a video or words in a sentence. The neural network has a layer of neurons for each input. When the RNN learns, it backpropagates through time (BPTT), a form of multilayer backpropagation. BPTT uses the chain rule to go back from the last time step (Xt), gradually to each previous step, each time using gradient descent to discover the best weights for each neuron, and also learn the optimal weights that govern the transfer of information between one time step to another. RNNs are used in particular for:

● **Language modeling and text generation**: Models work by predicting the most appropriate next character, next word, or next phrase in a string of text.

● **Automatic translation**: the text in the source language is entered in batches and the model attempts to generate the corresponding text in the target language.

● **Voice recognition**: the input consists of acoustic signals analyzed from an audio recording; the model produces the most probable language phonetic element for each part of the recording.

● **Generation of image captions**: the input is an image and the model identifies elements of the image and generates text that describes it.

● **Time-series anomaly detection**: The input is a series of sequential data, such as a series of events in a potential cybersecurity incident. The model predicts whether the data is an anomaly with respect to the normal behavior.

● **Video markup**: The input is a series of video frames and the model generates a textual description of each frame in the video.

3.4.3 Autoencoder models (AE)

A neural network called an autoencoder can be used to learn a compressed representation of raw input. An autoencoder is made up of two sub-models: an encoder and a decoder. The encoder compresses the input, while the decoder attempts to reconstruct it from the encoder's compressed form**[25]**. The encoder model is saved after training, whereas the decoder is deleted.The encoder can then be used as a data preparation approach to extract features from raw data so that a new machine learning model can be trained. The following figure schematizes a simple autoencoder, whose encoder processes images (inputs), in order to represent them as points in a two-dimensional space (encoded representation), then decodes this representation, in order to find the starting data (output).

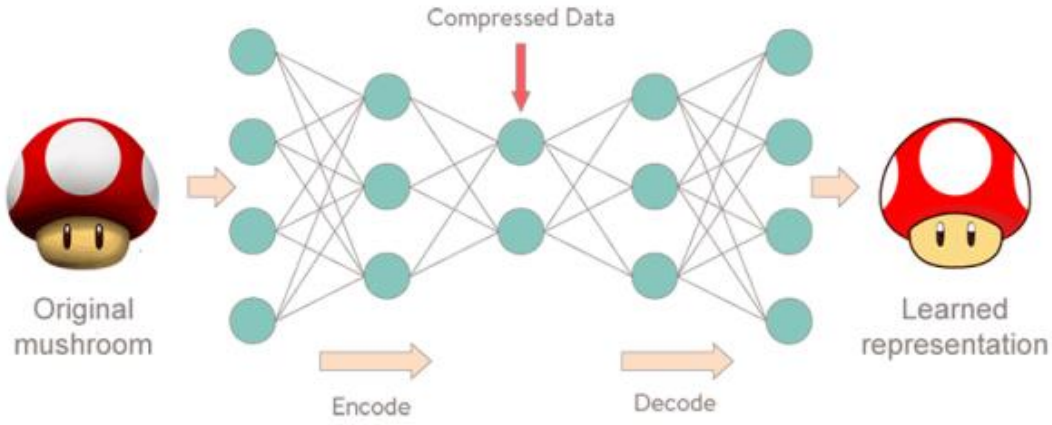

Figure1. 10 Autoencoder**[26]**.

3.5 Reinforcement learning (RL):

Reinforcement Learning is a type of Machine Learning approach in which the agent is given a delayed reward in the following time step in order to evaluate its prior activity. It was largely employed in video games (e.g., Atari, Mario), with human-like or even superior performance. As the method advances with the use of Neural Networks, it is now capable of tackling more complicated problems**[27]**.

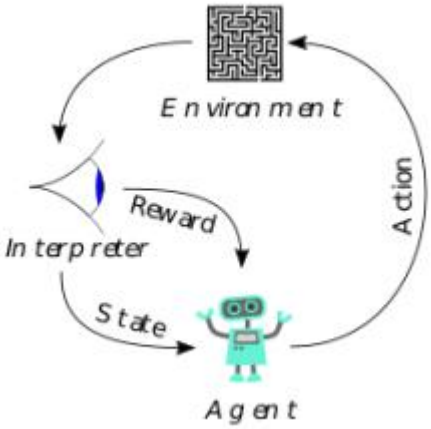

Figure1. 11 principle of reinforcement learning.

3.6 Advantages and disadvantages:

Neural networks have been widely used in recent years in many fields because they have more advantages than disadvantages. We list its advantages and disadvantages below.

Advantages:

- Often high precision.

- Robust, works when the learning set contains noise.
- Allows complex problems to be solved by machine learning (no need to manually define AI rules as in the classic method).
- The output can be a discrete, continuous value, or a vector of several discrete or continuous values.
- Adapts well to problems that require high levels of abstraction that are difficult to describe explicitly with the classical method (code all treatments and conditions).

Disadvantages:

- Long learning curve (due to high number of parameters).
- Difficult to understand the model.
- Non-evidence of the choice of hyperparameters.

Conclusion:

Machine learning and deep learning are the two most important concepts that make artificial intelligence possible. Machine learning that learns based on a large set of structured data. Deep learning does not need structured data, but systems that work from several layers of neural networks without being guided by humans. We have seen in this chapter what deep learning is and its fields of application, with the important notions that are related to (definition, architectures, etc.), artificial neural networks and deep neural networks . With some platforms among the fields of application of deep learning are recommender systems, we describe this area in the third chapter, because it is the context of our work.

Chapter 02

Vision transformers (Vits)

Introduction:

Nowadays, a giant variety of deep mastering fashions are used in quite a number fields, specifically in the clinical domain. The principal goal of these fashions is to help in the evaluation and classification of scientific images, aiming to enhance diagnostic accuracy and efficiency. Among these models, Vision Transformers (ViTs) have currently emerged as a effective choice to ordinary convolutional neural networks. ViTs system snap shots as sequences of patches, permitting them to seize long-range dependencies in scientific data. In this project, a Vision Transformer is skilled from scratch to classify chest X-ray pix into everyday and pneumonia classes. This strategy lets in for an in-depth find out about of the ViT's performance, mainly when skilled on confined clinical datasets.

Vision Transformer History:

The roots of Vision Transformers (ViTs) can be traced returned to the success of Transformer fashions in herbal language processing (NLP), mainly after the introduction of the authentic Transformer structure with the aid of Vaswani et al. in 2017. This architecture, based totally absolutely on interest mechanisms, confirmed sturdy overall performance in sequential facts processing, which stimulated researchers to adapt it to pc imaginative and prescient tasks. In 2020, Dosovitskiy et al. delivered the Vision Transformer (ViT), marking a giant shift in the layout of photo classification fashions by way of changing convolutional layers with self-attention mechanisms. Instead of processing complete photos as grids of pixels, ViTs divide photos into fixed-size patches and deal with them as tokens, comparable to phrases in NLP. Since their emergence, ViTs have proven aggressive or choicest overall performance in contrast to regular CNNs, specifically when educated on massive datasets. In the clinical imaging domain, researchers have begun making use of ViTs for duties such as ailment detection, segmentation, and classification. Their capacity to seize world context makes them specifically treasured in inspecting complicated clinical photographs such as chest X-rays, MRIs, and retinal scans. Today, Vision Transformers are regarded a promising course in the improvement of AI-driven diagnostic equipment in healthcare.

1. Definition:

Vision Transformers (ViTs) symbolize a current category of deep mastering models that remember on the Transformer structure to method visible facts as a substitute of the use of ordinary architectures like Convolutional Neural Networks (CNNs). They can be described as: “Models that divide an picture into small patches and manner them as a sequence, mastering significant representations to operate duties such as classification, prediction, or segmentation with excessive accuracy and flexibility.”

In the context of clinical picture classification, ViTs can be seen as clever structures succesful of extracting fine-grained facets from clinical scans (such as chest X-rays or MRI images) and inspecting them to produce diagnostic outputs or predictions primarily based on visible patterns that may also now not be effortlessly observable through traditional means.

Two primary elements concerned in scientific photograph classification the use of Vision Transformers are the photo and the user. The "image" refers to the clinical facts (e.g., X-rays), whilst the "user" is generally a scientific specialist or radiologist who receives the prediction or analysis from the model.

To operate classification, the gadget makes use of education records to examine associations between photo facets and diagnostic labels. This discovered illustration is then used to predict labels for new, unseen images. Over time, with perfect comments and facts updates, the gadget can enhance its accuracy—similarly to how recommender structures refine person profiles primarily based on interactions.

2. Objectives of Vision Transformers for Medical Image Classification:

Previously, we described Vision Transformers (ViTs) as fashions succesful of examining visible facts by means of dividing photographs into patches and processing them sequentially the use of transformer-based architectures. In this section, we purpose to furnish a extra specified view via highlighting the doable functions and dreams of the use of Vision Transformers in the scientific domain.

To commence with, it is integral to distinguish between the two important actors worried in such systems: the clinical practitioner (user) and the healthcare company or lookup group (service provider). For example, a health center that deploys a ViT-based diagnostic device is involved in enhancing diagnostic accuracy, decreasing workload on scientific staff, and optimizing healthcare delivery. On the different hand, a radiologist or clinician makes use of

such structures to aid in detecting ailments extra efficiently, making sure quicker and extra correct decision-making.

There are a number of targets at the back of the use of Vision Transformers in scientific photograph classification:

- .**Improve diagnostic accuracy**: One of the important desires of the use of ViTs is to decorate the accuracy of sickness detection. By shooting complicated spatial patterns in scientific images, ViTs purpose to decrease misdiagnosis and extend the reliability of medical decisions. The gadget recommends diagnostic labels that are anticipated to align carefully with specialist interpretation.
- **Support early disease detection:** Vision Transformers are succesful of figuring out delicate abnormalities in clinical snap shots that may additionally be left out in early levels of disease. This allows well timed intervention, which is especially imperative in prerequisites such as cancer, tuberculosis, or pneumonia.
- **Enhance interpretability and trust:** Modern ViT fashions can be geared up with interest maps that spotlight areas of activity inside the image. This interpretability function approves clinicians to apprehend why the mannequin made a specific decision, growing have faith and adoption in medical environments.
- **Reduce workload and time:** By automating components of the diagnostic pipeline, ViTs assist decrease the burden on radiologists. They can pre-screen snap shots or prioritize high-risk cases, permitting healthcare vendors to allocate assets greater efficiently.

Now we define the functionalities that clinical experts would possibly are seeking for when the use of Vision Transformer-based systems:

- **Prioritize and rank diagnostic cases:** ViTs can rank affected person pictures primarily based on the possibility of pathology, supporting clinicians focal point first on the most essential cases. This mirrors the rating characteristic discovered in recommender systems.
- **Sequential diagnosis support:** Some ViT structures can be tailored for non-stop monitoring, helping a collection of diagnostic assessments over time. For example, in follow-up scans, the gadget can song development or regression of a sickness.
- **Enhanced navigation through large image datasets:** In scientific settings the place lots of snap shots may also want review, ViTs can help in filtering and highlighting

applicable images, enhancing the searching and retrieval system for scientific professionals.

3. Phases of Visions transformers:

- **Image Splitting into Patches:**

  The enter picture is divided into small fixed-size patches (e.g., 16x16 pixels), comparable to reducing the photo into small tiles.

- **Patch Embedding:**

  Each photo patch is flattened and handed via a linear projection layer to convert it into a vector (embedding).

- **Position Embedding:**

  Since Transformers lack a built-in grasp of spatial relationships, positional embeddings are delivered to every patch embedding to maintain the positional information.

- **Transformer Encoder:**

  The sequence of embedded patches (with positional information) is handed thru more than one Transformer encoder layers. These layers use self-attention and feed-forward networks to mannequin the relationships between patches.

- **Classification Head:**

  A one of a kind token (often referred to as [CLS]) is used to mixture the statistics from all patches, and its output is handed to a classification layer (e.g., softmax) to predict the last classification label.

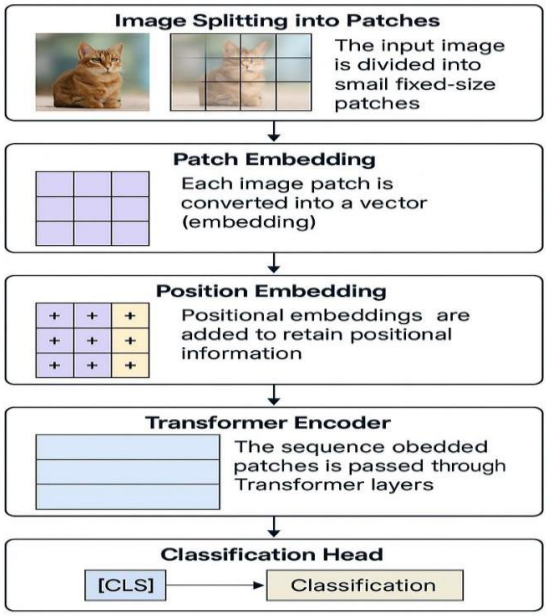

Figure 2.1 Phases of vision transformer process

4. Transformer in medical imaging:

4.1 Datasets:

Transformers are generally known to perform better in large training than small sized training due to the absence of inductive bias that bolsters few shot learning. In an attempt to solve this problem, researchers have proposed several hybrid architectures that seeks to incorporate strengths of the convolutional neural network into the transformer. The availability of public medical datasets of diverse modalities have been a major deterrent to the training and re-training of state-ofthe-art CNN architectures like ResNet **[28]** and EfficientNet **[29]**for the development of domain specific weights to serve as a feature extractor layer in transfer learning for both CNNs and Transformers. A few researchers have proven that transformers benefit more from transfer learning than CNNs **[30]**. However, standard weights like ImageNet do not serve as efficient feature extractors for medical imaging tasks across architectures **[31]** hence the need for a detailed outline of publicly available medical datasets. The detailed compilation of most of the publicly available datasets of variousmedical image modalities with their **[32]**

4.2 Compilation of published available dataset of medical image modalities:

1 Detection :

- **Histopathology Images:**

Cancer Genome Atlas **[33]**: The dataset represents heat maps of 33 tumor types, and 3 distinct expressions; reverse-phase protein array, gene expression and miRN

- **CT-scans**

COVID-19 CT-2A **[34]**: A benchmark dataset containing 3 classes, COV19 pneumonia,non-COV19 pneumonia and normal. Comprises data collated from 15 countries and contains about 4,500 samples

- **X-rays**

COVIDx **[34]:** A dataset of 3 classes: COVID positive, viral pneumonia and normal images.

COVIDGR-E **[35]:** Contains about 430 images of COVID-19 pneumonia.

- **Fundus Images**

DRiD **[36]**: A dataset comprising 81 images for Micro-aneurysm detection.

2 Classification :

- **Histopathological Images**

Cancer Histology Dataset **[37]**: These is a dataset comprising whole-slide images of colorectal cancer cells.

- **CT-scan**

COVID-CT-set **[38]**: A database contain a large number of lung scans for covid-19 classification.

COVID-19-CTDB **[39]**: An annotated dataset of chest scans.

- **MRI-scan**

MRNet **[40]**: A dataset collated at the Stanford medical center, containing about 1400 knee MRIs.

- **X-ray**

COV19 chest x-ray **[41]**: A dataset contain chest x-rays for classifying covid-19 frombacterial pneumonia.

Extensive-XR-CT **[42]**: A dataset of CT and X-rays for patients with and without covid.

Posterior-Anterior Chest Radiography COV19 **[43]**: This is a combination dataset of about 15 smaller chest X-ray datasets.

- **Fundus Images**

Color Fundus **[44]**: This Dataset contains fundus images in DR-grading.

3 Segmentation :

- **CT-scans**

Kits19 **[45]**: A dataset annotated for the segmentation of renal tumor.

- **MRI-scans**

M&MS-21 **[46]**: A dataset containing 375 annotated cardiac magnetic resonance images.

- **X-ray**

OAC **[47]**: A dataset containing annotated knee

images.DICRLN **[48]**: A database of annotated chest images for detecting lung nodule.

IN-breast **[49]**: A database containing annotated breast mammograms.

4.3 Classification and Segmentation:

Classification of medical images is a vital task in healthcare because of the increasing need for diagnosis, identification and distinction of healthcare images and relevance in intended applications. Varieties of transformer architectures have been employed in modelling with the aim of achieving higher efficiency than previously obtained. Researchers have employed pure vision transformers an d hybrid models of disparate architectures for classification of medical images in literature, hence the extensive review of the various transformerbased methods developed for the classification of medical images, with a focus on the image modality, as presented in the writeup.

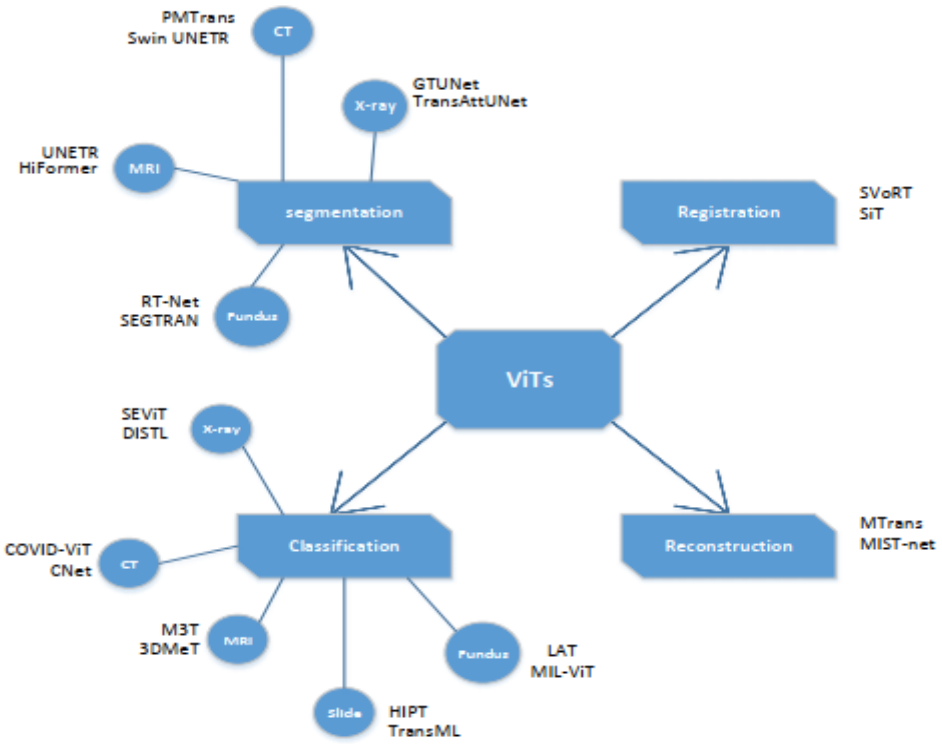

Figure2.2: A taxonomy on the application of vision transformers in medical imaging detailing prominent models of varying modalities and disparate tasks.

5. Leading Models Based on Vision Transformers (ViTs) for Medical Image Classification:

- **TransMIL [50]:**

TransMIL integrates transformers into a multiple instance learning (MIL) framework to introduce correlations between various instances. MIL is achieved by pooling operations performed on learning instances extracted by a pre-trained CNN (He et al., 2016). These instances, after squaring, are passed into a block containing multi-head self-attention (MSA), positional encoding, and an MLP for weakly supervised classification. The resultant models achieved AUC scores of 93.09% for classifying binary tumors and 96.03% for cancer subtypes.

- **CTransPath [51]:**

Wang et al. (2022) introduced a contrastive learning strategy based on self-supervised learning (SSL), which aims to improve the annotation of large-scale data by using an unlabeled

dataset. This process helps in extracting useful representations that generalize well to multiple downstream tasks. Unlike traditional contrastive learning, semantically relevant contrastive learning (SRCL) seeks to acquire more informative representations by aligning closely related positive pairs. CTransPath integrates CNNs with multiple Swin transformers and is pre-trained with a large amount of unlabeled data. This model then serves as a feature extractor for SRCL in downstream tasks.

- **GasHis-Transformer [52]:**

GasHis-Transformer is designed for gastric histopathological image classification. It captures both local and global information. Local information is captured by a CNN architecture, while global information is captured through attention blocks coupled with CNNs.

- **Self-Supervised ViTs for Histopathology [53]:**

Chen & Krishnan (2022) proposed a novel vision transformer (ViT) based on the hierarchical nature of whole slide images (WSIs). The Hierarchical Image Pyramid Transformer (HIPT) employs a two-level self-supervised learning framework that exploits both the hierarchical nature of WSIs and the large sequence length of WSI tokens due to their pixel nature. It aggregates visual tokens at three levels (cell, patch, and region) to form the slide representation, using self-attention as a permutation-equivariant aggregation layer. HIPT, with hierarchical pre-training, showed better performance than state-of-the-art methods for survival prediction and cancer subtyping.

- **CTNet [54]:**

CTNet is a hybrid model comprising a CNN feature extractor and ViT for detecting COVID-19 from 3D chest scans. After training on the COV19-CT dataset, it achieved 88% F1 score.

- **ScopeFormer [55]:**

ScopeFormer is a hybrid model designed for intracranial hemorrhage classification from CT images. It stacks multiple Xception CNN blocks, aggregating their feature maps to create a feature-rich map. This map serves as the input to a ViT, improving model performance. The model achieved a test accuracy of 98.04% on this classification task.

- **Pancreatic Cancer Diagnosis [56]:**

Xia et al. (2021) demonstrated the effective diagnosis of pancreatic cancer from 2D CT images. A region-of-interest (ROI) feature map is developed from annotated training data using

the U-net segmentation algorithm. This ROI feature map is then fed into a transformer built upon the U-net for segmentation. The model was trained on a 3-class labeled dataset with 1321 patients and achieved specificity and sensitivity of 95.2% and 95.8%, respectively.

- **Swin UNETR [57]:**

Swin UNETR is a prominent model for 3D CT image classification. It employs a hierarchical encoder for self-supervised pre-training and is fine-tuned for classification and segmentation tasks in the Beyond the Cranial Vault (BTCV) challenge. This model outperformed all other models submitted for the challenge.

6. Transformer and CNNs:

Considering the properties of both architectures and optimizations performed to mitigate their limitations, a comparative analysis on how both model types perform on different domains, considering: training from initialized weights, transfer learning, self-supervision, and inference.

7.1 Training From Randomly Initialized Weights :

Random weight initialization requires the model to learn representations from scratch and only from the data fed into the model for training. The performance of transformers trained from scratch and CNNs are compared in a variety of literature **[58]**. On smaller datasets, CNNs seem to have performed better because of the inductive bias inherent within their architecture. However, as the data size increases, transformers are seen to gradually outperform their CNN counterparts. Training from scratch is seldom practiced in medical imaging due to the small size of most medical data.

7.2 ImageNet and Same Domain Pre-training :

To augment for the size of most domain-specific data in computer vision and medical imaging, ImageNet pre-trained weights are often employed for model initialization. This usually results in improved performance **[59]**. Researchers have investigated the benefits and extent of transfer learning to ViTs. They discovered that both CNNs and ViTs seem to benefit from ImageNet pre-trained weights and same-domain pre-training. It is also recorded that transformers benefit the most from transfer learning **[31] , [60]**.

7.3 Self-Supervision :

Self-supervision is the most effective solution to the problem of lack of large-sized, well-annotated data for modeling in computer vision and medical imaging. The most utilized self-supervised learning schemes, BYOL and DINO, have been found to achieve performances comparable to supervised learning schemes. This has prompted their utilization, alongside supervised fine-tuning, in computer vision, achieving state-of-the-art performance **[61]**. This concept has been attempted on a variety of medical image modalities **[62]**, also recording state-of-the-art performances. Additionally, ViTs are found to benefit slightly more from self-supervised pretraining than CNNs.

8. Inference without Fine-Tuning:

In cases of extremely scarce data, features extracted from a pre-trained network can be utilized directly for classification and clustering operations. This is most optimal when the pre-trained features are closely related to the target features. One research assessed whether ViTs performed better than CNNs in inference without fine-tuning by applying k-NN evaluation on the penultimate layer of a CNN and the CLS token of a ViT. They evaluated for both in-domain and out-domain pre-trained weights and recorded that ViTs performed better in both scenarios **[58]**.

Conclusion:

In this chapter, we review the importance of using Vision Transformers (ViTs) models in medical image classification, especially in light of the rapid development of deep learning in the healthcare field. ViTs are an advanced alternative to traditional convolutional neural networks (CNNs), which are characterized by their ability to process images by dividing them into batches and analyzing them as series, allowing for the discovery of long-range relationships within the data. The study covered the stages of ViT development, as well as the most prominent challenges associated with the scarcity of medical data. We also focused on adding a set of available databases and a set of leading models that have adopted ViTs in medical applications. We also compared the performance of ViTs with CNNs in cases of training from scratch, transfer learning, self-learning, and inference without retraining. ViTs demonstrated competitive performance, especially when large datasets are available or advanced learning strategies are used. This chapter highlights the growing role of these models in improving the accuracy of medical diagnoses and reducing the workload for physicians

Chapter 03

Transformer Architectures: Attention Mechanisms and Vision Applications

TRANSFORMERS:

1.1 Attention in Transformers:

The fundamental transformer architecture as proposed by Vaswani **[63]** is a sequence-tosequence model that is composed of self-attention and point-wise feed-forward network (FFN) block that extracts global dependencies between tokens (words). The complete architecture includes layer normalization after each attention block, a linear transformation function and a softmax function. This architecture constitutes the basis for more complex and efficient, supervised or self-supervised models today, a part of this success can be attributed to the concept of Attention. Attention mechanism is the primary way humans sort relevant from irrelevant data by unintentionally paying attention to some part of data set, while discarding other parts. Few scientists have attempted to build neural networks that model this behavior, initially for use in language processing tasks **[64].** A typical attention, regarded as the "Bahdanau attention" computes a weighted sum of each feature while highlighting the most relevant
features from a ***feature matrix. Self-attention*** was designed to emphasize relationships between data regardless of their position in the sequence. It is mathematically expressed by a map function of queries, keys and values such that for each input $X \in \mathbb{R}c$, $i = 1, \dots, n$ there exist a query $Q \in \mathbb{R}n\times d$, a key $K \in \mathbb{R}n\times d$, and a value $V \in \mathbb{R}n\times d$, which are utilized in generating learning parameters W_q, W_k, W_v respectively.

$$\begin{aligned} Q &= X \times W_q,\ W_q \in \mathbb{R}c\times d, \\ K &= X \times W_k,\ W_q \in \mathbb{R}c\times d, \\ V &= X \times W_v,\ W_q \in \mathbb{R}c\times d, \end{aligned} \qquad (3.1)$$

The output is a probability that requires normalization, usually achieved by a softmax function to attain an output distribution represented by the equation (2).

$$\text{Attention}(\text{Q}, \text{K}, \text{V}) = \text{softm ax}\left(\frac{QK^T}{\sqrt{\text{Dк}}}\right) * \text{V} = \text{AV} \qquad (3.2)$$

The output of a self-attention block is the sum of element V and the matrix A equipping this attention block with the ability to capture global dependencies within a data. **Multi-head self-attention** can be applied to better capture hierarchical features. These are computed in parallel after the final output is obtained by concatenating each individual attention block. This operation is similar to the use of multiple kernels within convolution operations, mathematically expressed by:

$$Z_i = Attention(Q \times W_{iq}, K \times W_{iq}, V \times W_{iv})$$
$$MSA(Q, K, V) = concat[Z_1, \dots, Z_h] \times W_o \quad (3.3)$$

Here $\boldsymbol{h}$ represents the total number of heads $\boldsymbol{W_o}$ represents an output matrix of theconcatenated projection of all selfattention $\boldsymbol{W_{iq}}$, $\boldsymbol{W_{iq}}$ $\boldsymbol{W_{iv}}$ of the i_{th} attention.

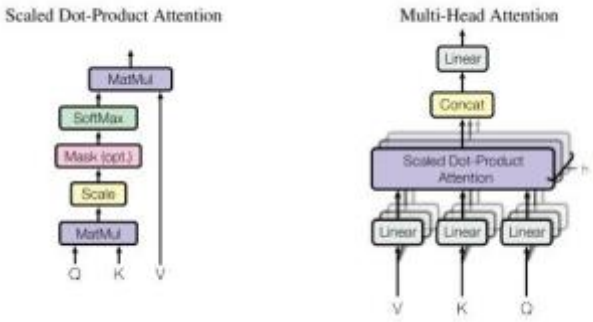

Figure3.1: Multiple attention layers arranged in parallel (left) Self-Attention. (Right) Multi-Head Self-Attention (Vaswani et al., 2017)

1.2 Point-Wise Feed-Forward Network:

The output from the multi-head self-attention block is fed into a feed-forward network comprising two linear activation function and a rectified linear unit (RELU) activation, as expressed in equation (4).

$$FFN(X) = ReLU(XW_a + B_a)W_b + B_b \quad (3.4)$$

Here $\boldsymbol{X}$ represents the output from the previous layer and $\boldsymbol{W_a, W_b, B_a, B_b}$ are trainable parameters of dimensions $\boldsymbol{D_c}$ and $\boldsymbol{D_n}$, represented as $\boldsymbol{W_i} \in \mathbb{R}^{\boldsymbol{D_c \times D_n}}$ and $\boldsymbol{B_i} \in \mathbb{R}^{\boldsymbol{D_c \times D_n}}$ where $\boldsymbol{i = a, b}$. It is to be noted that n should always be larger than c.

1.3 Positional Encoding:

Learnable parameters are typically employed to aid the network retain positional information, this could be achieved by recurrence or convolutions however, in the transformer architecture this is achieved by inputting information about the position of tokens (words or patches) into the sequence. **[63]** experimentally utilized sine and cosine functions of varying frequencies.

$$PE_{(pos,i)} = \begin{cases} \sin(pos * w_n), if \ \ i = 2n \\ \cos(pos * w_n), if \ \ i = 2n + 1 \end{cases} \quad (3.5)$$

$$w_n = \frac{1}{10000^{\frac{2n}{k}}} \quad , \ \ n = 1, \dots, \frac{k}{2}$$

Here $\boldsymbol{pos}$ represents the initial position of the vector while $\boldsymbol{n}$ represents the length of the vector, $\boldsymbol{i}$ represents the particular instance. In the first pure vision transformer by Dosovitskiy **[65]** the learned position is outputted as a vector and serves as input into the encoder. This vector is a sequence of n-dimension patches where $\boldsymbol{n = 1, \dots, k}$. While a lower n-value will store better positional information; it will also be computationally expensive.

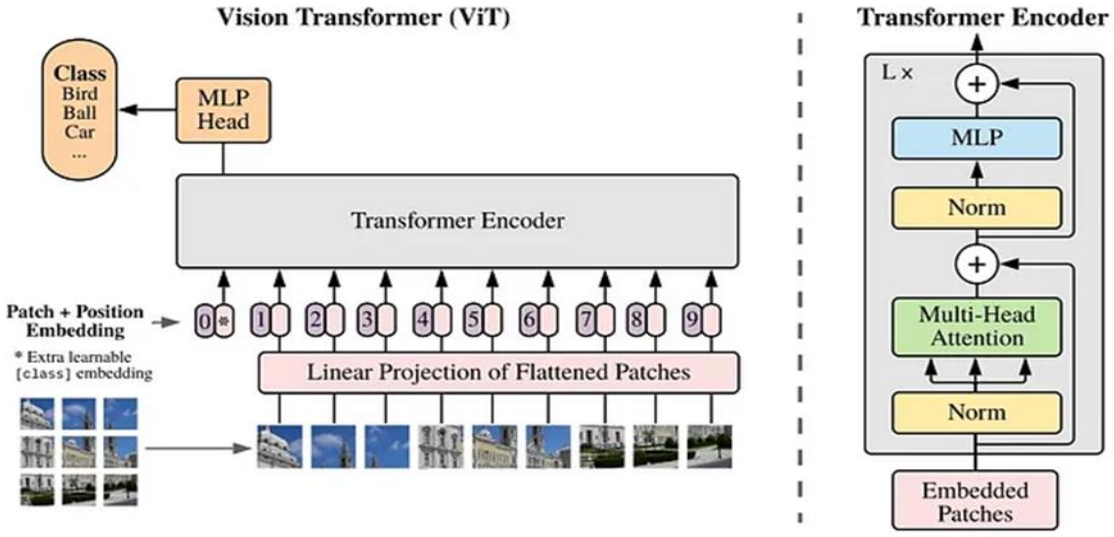

Figure 3.2:Vision Transformer model overview: Based on original transformer architecture (Dosovitskiy et al., 2020)

1.4 Vision Transformers:

A variety of transformer based vision models exists, a few of the prominent ones are ,Vision Transformer (ViT) **[65]**, Detection Transformer (DETR) **[66]**, dataefficient image transformer (DeiT) **[67]** and Swin-Transformer **[68]**. The ViT was the first pure adaptation of the vanilla transformer in the field of computer vision. It entails an encoder and a task specific decoder structure where input images are broken into a sequence of non-overlapping patches of size ($C \times P \times P$). Here C represents the number of channel of the image and P represents its length and width. The information about the position of patches in the image is converted into a vector. This positional information, along with the sequence of non overlapping patches are fed into the encoder block of the transformer containing multi-head selfattention, layer normalization and a multi-layer perceptron (FFN), this is depicted in Figure (2). DETR is a hybrid that attaches a transformer encoder to a CNN architecture; it was the first attempt at partially conveying attention from the vanilla transformer to a vision task of object detection. The swin-transformer was designed to reduce the computational cost of the ViT and was tested majorly on segmentation tasks. **[68]** attempted varying patch sizes so as to reduce the requirement of full attention due to the input sequence of non-overlapping patches; they introduced the shifted window attention that creates patches of various sizes in a hierarchical order, and further assists in preserving spatial information. The efficiency of the ViT could only be observed during large scale training as it performs poorly on small datasets, and **[67]** proposed the DeiT in an attempt at solving this problem. DeiT adopts a CNNs teacher and a transformer student structure in a knowledge distillation framework in which a distillation token is added for the purpose of learning from the teacher model. This knowledge is then inherited by the student model while imparting inductive bias.

Conclusion:

In this chapter, we discuss the basic architecture of Transformer models, as first introduced by Vaswani et al. in 2017. These models rely on self-attention and point-wise convolutional networks (FFNs) to extract global relationships between elements in a sequence. "

Attention" is used to identify the most important parts of data and was initially applied to language processing and later adopted in computer vision. The architecture includes several key components, such as positional encoding to preserve data order and multi-head attention to extend the model's ability to learn hierarchical features. We also highlight Transformer-based vision models: the Vision Transformer (ViT), DETR, DeiT, and Swin Transformer. ViT is the first explicit implementation of a Transformer in image classification, while Swin Transformer was developed to reduce computational complexity by using overlapping attention windows. DeiT is presented as a solution for training ViTs on small datasets using cognitive distillation techniques. This development marks a fundamental shift in computer vision paradigms from traditional architectures to architectures that rely primarily on attention.

Chapter 04

Implementation and experimental results

Introduction:

In this chapter, we will talk about the proposed model, which aims to improve medical images using a Vision Transformer-based classification system. Then, we will explain the implementation of this system and detail the steps followed to preprocess the medical image dataset. After that, we will present how each component of our project works. We will then describe the model built to train on the dataset. Finally, we will discuss the results obtained and evaluate the performance of the proposed model in enhancing medical image classification.

1. Development environment ,programming language, Libraries and Frameworks:

In this project we have used many tools, frameworks, libraries and new software. We will define all of them briefly.

Python:

Figure 4. 1 Python Logo

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse.

The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms and can be freely distributed**[69].**

NumPy:

Figure 4. 2 NumPy logo

NumPy is a scientific package designed for scientific computing with Python language, and it allows using code of some other languages like C/C++ and Fortran, it has N-dimensional array objects, broadcasting functions, and other complex mathematical functions **[70].**

PyTorch:

Figure 4. 3 PyTorch logo

PyTorch is an optimized tensor library for deep learning using GPUs and CPUs.**[71]**.

Torchvision:

Figure 4. 4 Torchvision logo

This library is part of the PyTorch project. PyTorch is an open source machine learning framework. The torchvision package consists of popular datasets, model architectures, and common image transformations for computer vision.**[72]**.

Matplotlib:

Figure 4. 5 Matplotlib logo

Matplotlib has extensive text support, including support for mathematical expressions, TrueType support for raster and vector outputs, newline separated text with arbitrary rotations, and Unicode support. These tutorials cover the basics of working with text in Matplotlib **[73]**.

TQDM :

Figure 4. 6 TQDM logo

tqdm means "progress" in Arabic (*taqadum*, تقدّم) and is an abbreviation for "I love you so much" in Spanish (*te quiero demasiado*).Instantly make your loops show a smart progress meter - just wrap any iterable with tqdm(iterable), and you're done**[74]**.

Google Colab :

Figure 4. 7 Google Colab logo

Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. Google Colab's major differentiator from Jupyter Notebook is that it is cloud-based and Jupyter is not. This means that if you work in Google Colab, you do not have to worry about downloading and installing anything to your hardware **[75].**

Google Drive :

Figure 4. 8 Google Drive logo

Google Drive is a cloud-based storage service that enables users to store, access, and share files and folders online. The web service provides a centralized cloud storage hub where changes to content are automatically saved and synchronized across all connected devices. This allows multiple users to collaborate on the same content in real time and always see the latest version of each file**[76]**.

1. Machine specs:

Since we are using google colab we can get plenty of performant hardware we cite below the free plan.

Table 4.1: Machine specs.

CPU	ntel(R) Xeon(R) CPU @ 2.20GHz
RAM	13GB
Hard disk space	40GB
GPU	NVIDIA Tesla T4 (16GB)

2. Project description:

In this project, we aimed to boost an untrained AI mannequin based totally on the Vision Transformer (ViT) structure to classify chest X-ray image into two categories: pneumonia or not. This mannequin depends on segmenting pictures into patches, reworking them into expression vectors, and then processing them the usage of transformer encoders, besides the want to depend on usual convolutional neural networks (CNNs).

The mannequin is skilled on a scientific dataset containing photographs of contaminated and non-infected individuals, the use of present day techniques such as pre-normalization, AdamW optimization, and records augmentation to make sure top generalization and decrease overfitting. This work ambitions to make contributions to clinical selection aid via offering an high-quality and correct device for early detection of pneumonia.

3. Our work:

To achieve this goal, we created the model and then trained it on a chest X-ray dataset to achieve the highest possible accuracy.

3.1 Dataset:

The dataset is organized into 3 folders (train, test, val) and contains subfolders for each image category (Pneumonia/Normal). There are 5,863 X-Ray images (JPEG) and 2 categories (Pneumonia/Normal).

Chest X-ray images (anterior-posterior) were selected from retrospective cohorts of pediatric patients of one to five years old from Guangzhou Women and Children's Medical Center, Guangzhou. All chest X-ray imaging was performed as part of patients' routine clinical care.

For the analysis of chest x-ray images, all chest radiographs were initially screened for quality control by removing all low quality or unreadable scans. The diagnoses for the images were then graded by two expert physicians before being cleared for training the AI system. In order to account for any grading errors, the evaluation set was also checked by a third expert **[77]**.

Figure 4. 9 Representative Chest X-ray Images Showing Normal Case, Bacterial Pneumonia, and Viral Pneumonia

3.2 Data preparation:

First ,we connect Google Colab with our Google Drive account so that we can directly access the dataset from the Colab envirmont after uploading it to Drive.

```
from google.colab import drive
drive.mount('/content/drive')
```

Then we specified the device on which the computations will be performed in PyTroch.

```
[ ] device = torch.device('cuda') if torch.cuda.is_available() else torch.device('cpu')
```

We need to import libraries and dataset, since we are using google colab most of the python libraries came preinstalled, so we just need to import them.

```
import torch
import torch.nn as nn
from torchvision import transforms, datasets
from torch.utils.data import DataLoader
from tqdm import tqdm
import numpy as np
import matplotlib.pyplot as plt
```

Here, we defined the VIT model and identified its basic components. We also created a layer called PatchEmbedding, which is responsible for transforming images into sets of small patches and converting each patch into a digital representation suitable for input into the model. This layer also adds a Cls token, which will later be used to identify the image's class. It also adds positional embedding, which helps the model understand the order of patches within the image.

```
class ViT(nn.Module):
    def __init__(self, num_patches, img_size, num_classes, patch_size,
                 embed_dim, num_encoders, num_heads, hidden_dim,
                 dropout, activation, in_channels):
        super().__init__()
        self.embeddings_block = PatchEmbedding(
            embed_dim=embed_dim,
            patch_size=patch_size,
            num_patches=num_patches,
            dropout=dropout,
            in_channels=in_channels
        )
```

In this section, we created transformer layers responsible for understanding the relationships between image components. First, we defined a single layer, the Transformer Encoder Layer, which contains the attention mechanism and a feedforward link. This layer was then duplicated multiple times using the Transformer Encoder to form the complete encoder, which in turn represents the analytic core of the model.

```
encoder_layer = nn.TransformerEncoderLayer(
            d_model=embed_dim,
            nhead=num_heads,
            dim_feedforward=hidden_dim,  # Added missing parameter
            dropout=dropout,
            activation=activation,
            batch_first=True,
            norm_first=True
        )
        self.encoder_blocks = nn.TransformerEncoder(encoder_layer, num_layers=num_encoders)
```

We have defined the Classification Head, which represents the last part of the model and gives the final result.

```
self.mlp_head = nn.Sequential(
            nn.LayerNorm(embed_dim),
            nn.Linear(embed_dim, num_classes)
        )
```

In this part, the image goes through stages called embedding and analysis, after which CLS is used to determine the medical class of the image.

```
def forward(self, x):
        x = self.embeddings_block(x)
        x = self.encoder_blocks(x)
        return self.mlp_head(x[:, 0])  # Use CLS token for classification
```

This part of the code segments the image into patches and then converts them to a numerical representation.

```
class PatchEmbedding(nn.Module):
    def __init__(self, embed_dim, patch_size, num_patches, dropout, in_channels):
        super().__init__()
        self.patcher = nn.Sequential(
            nn.Conv2d(in_channels, embed_dim, patch_size, patch_size),
            nn.Flatten(2))
```

Here we have prepared the sequence resulting from the image segmentation ,this is done by :

- Adding a classification token (cls_token).
- Adding positional embeddings (pos_embed).
- Preparing a Dropout layer.

```
        self.cls_token = nn.Parameter(torch.randn(1, 1, embed_dim))
        self.pos_embed = nn.Parameter(torch.randn(1, num_patches + 1, embed_dim))
        self.dropout = nn.Dropout(dropout)
```

Here we represent the forwarder function in the VIT model. This part prepares a sequential image representation (similar to sentences in natural language processing) for later input into the transformer layers. It adds a CLS token to the beginning of the image representation after transforming it into a series of small patches. This token is learned during training and is used to represent the image as a whole. After this sequence is passed through the transformer layers, this CLS token is used to determine the image's class in the classification.

```
def forward(self, x):
      batch_size = x.shape[0]
      x = self.patcher(x)  # Output shape: [B, E, N_patches]
      x = x.permute(0, 2, 1)  # [B, N_patches, E]

      cls_tokens = self.cls_token.expand(batch_size, -1, -1)
      x = torch.cat((cls_tokens, x), dim=1)

      x = x + self.pos_embed
      return self.dropout(x)
```

The CFG is a dictionary that contains the basic parameters for training the VIT model. We used it to organize the hyperparameters, which are divided into three section :

- Model Settings.
- Training Settings
- Augmentation and Regularization

```
# @title Default title text
CFG = {
    'img_size': 224,
    'patch_size': 16,
    'num_classes': 2,
    'embed_dim': 768,
    'num_encoders': 6,
    'num_heads': 8,
    'hidden_dim': 2048,
    'dropout': 0.3,
    'activation': 'gelu',
    'in_channels': 3,
    'lr': 3e-4,
    'weight_decay': 0.1,
    'batch_size': 32,
    'num_workers': 2,
    'epochs': 20,
    'device': 'cuda' if torch.cuda.is_available() else 'cpu',


    'mixup_alpha': 0.2,  # For MixUp augmentation
    'cutmix_alpha': 1.0,  # For CutMix augmentation
    'smoothing': 0.1,     # Label smoothing
    'grad_clip': 1.0
}
```

We calculate the number of patches the image is divided into before entering it into the Vit model.

```
CFG['num_patches'] = (CFG['img_size'] // CFG['patch_size']) ** 2
```

3.3 The model:

Then we created a Vit model and configured it using defined stting in the CFG dictionary .

```
model = ViT(
    num_patches=CFG['num_patches'],
    img_size=CFG['img_size'],
    num_classes=CFG['num_classes'],
    patch_size=CFG['patch_size'],
    embed_dim=CFG['embed_dim'],
    num_encoders=CFG['num_encoders'],
    num_heads=CFG['num_heads'],
    hidden_dim=CFG['hidden_dim'],
    dropout=CFG['dropout'],
    activation=CFG['activation'],
    in_channels=CFG['in_channels']
).to(CFG['device'])
```

In this section, we defined the transformations applied to the image during loading. The goal of this is to improve the performance of the Vit model, especially since we are dealing with medical image data, which is represented by x-rays. It also reduces overfitting.

```
train_transform = transforms.Compose([
    transforms.Grayscale(num_output_channels=3),
    transforms.Resize((CFG['img_size'], CFG['img_size'])),
    transforms.RandomRotation(20),
    transforms.RandomHorizontalFlip(),
    transforms.RandomAdjustSharpness(1.2),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406],
                         std=[0.229, 0.224, 0.225])
])

val_transform = transforms.Compose([
    transforms.Grayscale(num_output_channels=3),
    transforms.Resize((CFG['img_size'], CFG['img_size'])),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406],
                         std=[0.229, 0.224, 0.225])
])
```

This part of the code allows us to load images from folders organized by category.

- **Root** : represents the path to the data folder.
- **Transform** : represents the path to the image.

```
class ChestXRayDataset(datasets.ImageFolder):
    def __init__(self, root, transform=None):
        super().__init__(root=root, transform=transform)

    def __getitem__(self, index):
        image, label = super().__getitem__(index)
        return image, label
```

Here, we loaded data from the folders (train,val,test) and applied appropriate augmentations using train_transform and val_transform. We then created a dataLoader for aggregation and passed it to the model in batches to facilitate model training.

```
train_dataset = ChestXRayDataset(root='/content/drive/My Drive/chest_xray/train', transform=train_transform)
val_dataset = ChestXRayDataset(root='/content/drive/My Drive/chest_xray/val', transform=val_transform)
test_dataset = ChestXRayDataset(root='/content/drive/My Drive/chest_xray/test', transform=val_transform)

train_loader = DataLoader(train_dataset, batch_size=CFG['batch_size'], shuffle=True,
                          num_workers=CFG['num_workers'], pin_memory=True)
val_loader = DataLoader(val_dataset, batch_size=CFG['batch_size'], shuffle=False,
                        num_workers=CFG['num_workers'], pin_memory=True)
test_loader = DataLoader(test_dataset, batch_size=CFG['batch_size'], shuffle=False,
                         num_workers=CFG['num_workers'], pin_memory=True)
```

This part of the code defines the following:

1. Loss function:

Helps reduce overfitting and improve regularization.

2. Optimizer:

- **AdamW** : is an improved version of Adam that better handles weight regularization.
- **Lr** :represents the learning rate.
- **Weight_decay**: is used for regularization to prevent the model from memorizing excessive details.

3. Scheduler:

Cosine: gradually adjusts the learning rate.

- **T_max=CFG['epochs']** : represents the number of cycles in which the adjustment is made.

```
criterion = nn.CrossEntropyLoss(label_smoothing=0.1)  # Regularization
optimizer = torch.optim.AdamW(model.parameters(), lr=CFG['lr'], weight_decay=CFG['weight_decay'])
scheduler = torch.optim.lr_scheduler.CosineAnnealingLR(optimizer, T_max=CFG['epochs'])
```

3.4 Training:

This section represents the basic training loop, where the deep learning model is trained over multiple epochs. Several basic steps are performed in each epoch.

- Activate the model's training mode using **model.train()**
- Initialize the **train_loss** variable to aggregate the loss values across all batches
- Use **tadm** to display a progress bar showing the training progress
- Transfer images and labels to the **GPU**
- Zero previous gradients using **optimizer.zero_gard()** to avoid accumulation
- Pass images to the model
- Calculate the loss between predictions using the criterion loss function
- Implement **loss.backward()** to calculate the gradients
- Update the model's weights based on the gradients using **optimizer.step()**
- Update the **train_loss** loss set by calculating the loss and multiplying it by its size

```
best_val_acc = 0.0
train_losses = []
val_losses = []
val_accuracies = []



for epoch in range(CFG['epochs']):
    model.train()
    train_loss = 0.0
    progress_bar = tqdm(train_loader, desc=f'Epoch {epoch+1}/{CFG["epochs"]}')

    for images, labels in progress_bar:
        images = images.to(CFG['device'])
        labels = labels.to(CFG['device'])

        optimizer.zero_grad()
        outputs = model(images)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

        train_loss += loss.item() * images.size(0)
        progress_bar.set_postfix({'loss': loss.item()})
```

In this section, we evaluate the model's performance on validation data after Each epoch where we execute the following:

- Switch the model to evaluation mode using model.eval(), which disables layers that behave differently in training mode, such as Dropout and BatchNorm.
- Initialize the val_loss variables to calculate the total loss, correct, and total to calculate the model's accuracy.
- Use with troch.no_gard() to disable gradient calculations during validation to save memory and speed up the process.
- Calculate the loss between the outputs and add it to val_loss.
- Determine the final predictions by taking the class with the highest value from the outputs using troch.max.
- Update the total number of samples and the number of correct predictions.
- After completing validation on all data, calculate the total accuracy (val_acc), the percentage of correct predictions.
- Finally, update the learning rate scheduler (scheduler.step()).

```
model.eval()
    val_loss = 0.0
    correct = 0
    total = 0

    with torch.no_grad():
        for images, labels in val_loader:
            images = images.to(CFG['device'])
            labels = labels.to(CFG['device'])

            outputs = model(images)
            loss = criterion(outputs, labels)
            val_loss += loss.item() * images.size(0)

            _, predicted = torch.max(outputs.data, 1)
            total += labels.size(0)
            correct += (predicted == labels).sum().item()

    val_acc = correct / total
    scheduler.step()
```

In this section, we save the best model trained based on the validation accuracy. The following steps are performed at this level:

- The current validation accuracy (val_acc) is compared with the best validation accuracy achieved so far (best_val_acc).
- The model that achieved the best performance is saved using troch.save(model.state_dict() , 'best_vit_model.pth')
- The performance summary is printed including:
    a) Epoch number.
    b) Average training loss (train loss) divided by the number of training samples.
    c) Average validation loss (val loss) divided by the number of validation samples.
    d) Verification accuracy.

```
if val_acc > best_val_acc:
        best_val_acc = val_acc
        torch.save(model.state_dict(), 'best_vit_model.pth')

    train_losses.append(train_loss / len(train_loader.dataset))
    val_losses.append(val_loss / len(val_loader.dataset))
    val_accuracies.append(val_acc)

    print(f'Epoch {epoch+1} | '
          f'Train Loss: {train_loss/len(train_loader.dataset):.4f} | '
          f'Val Loss: {val_loss/len(val_loader.dataset):.4f} | '
          f'Val Acc: {val_acc:.4f}')
```

Table4.2 Performance Metrics of the Model Across Training Epochs

Epoch	Train Loss	Val Loss	Val Acc
1	0.5775	0.7206	0.6875
2	0.4344	0.6626	0.7500
3	0.4192	1.2628	0.5000
4	0.3868	1.3140	0.5000
5	0.3792	1.0224	0.5625
6	0.3692	0.9078	0.5625
7	0.3447	1.0621	0.5625
8	0.3476	0.5896	0.6875
9	0.3387	1.1179	0.5625
10	0.3251	1.2155	0.6250
11	0.3295	0.8235	0.5625
12	0.3239	1.0545	0.5000
13	0.3192	0.7688	0.6875
14	0.2992	0.9949	0.5000
15	0.2989	0.9787	0.5625
16	0.2956	0.9633	0.6250
17	0.2965	0.8143	0.6250
18	0.2887	0.9387	0.6250
19	0.2888	0.8181	0.6250
20	0.2865	0.8493	0.6250

The code evaluates the Vision Transformer model trained on the test dataset. The model's saved weights are loaded from the file 'best_vit_model.pth' and set to evaluation mode using model.eval(). The with torch.no_grad() block disables gradient computation to reduce memory usage and speed up execution. The model on the GPU computes the output using torch.max function. The final predictions are compared to true labels to count the number of correct predictions. The accuracy is calculated by dividing the number of correct predictions by the total number of samples.

```
model.load_state_dict(torch.load('best_vit_model.pth'))
model.eval()
test_correct = 0
test_total = 0

with torch.no_grad():
    for images, labels in test_loader:
        images = images.to(CFG['device'])
        labels = labels.to(CFG['device'])

        outputs = model(images)
        _, predicted = torch.max(outputs.data, 1)
        test_total += labels.size(0)
        test_correct += (predicted == labels).sum().item()

print(f'Test Accuracy: {test_correct / test_total:.4f}')
```

```
Test Accuracy: 0.8013
```

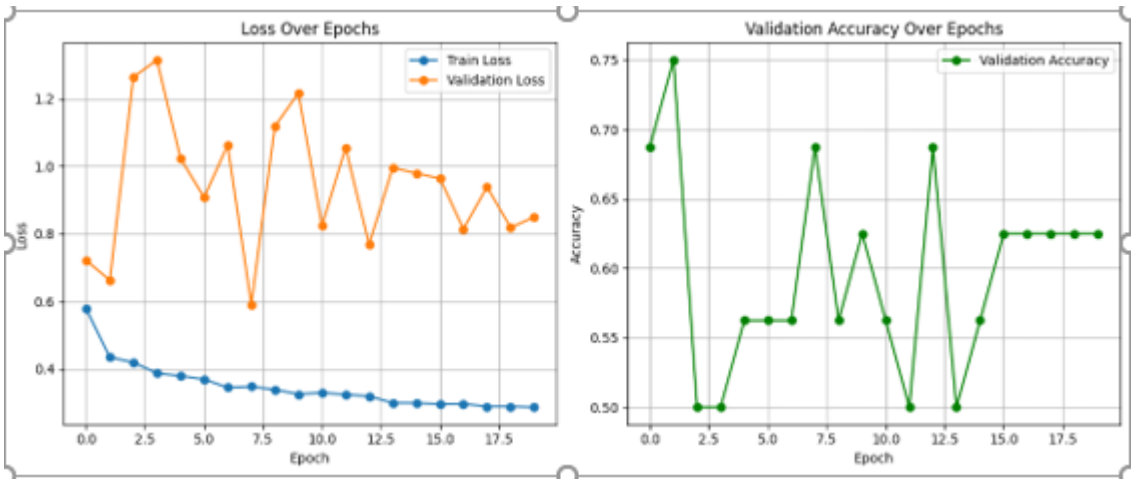

Figure 4. 10 Training vs. Validation Loss and Accuracy Curves for Vision Transformer on Chest X-ray Classification Task

3.5 Results Analysis :

The blue curve represents the model's train loss on the training data, the orange curve represents the model's validation loss on the validation data, and the second green curve on the right represents the model's validation accuracy on the validation data.

- We observed a steady decrease in the training loss, which means that the model is learning well from the training data.
- Validation accuracy improvement, despite significant fluctuations. Validation accuracy ultimately stabilized at around 62.5% over the last five epochs, marking some improvement in performance on the validation data.
- For the test data, the model achieved a good accuracy of 80.13% on the test set. This means our model can generalize to data it didn't see during training.

3.6 Disadvantages of the model:

The validation loss is unstable and displays significant fluctuations, which is an sign of overfitting. We also observed instability in performance on validation data, with accuracy dropping sharply to 0.50 for some epochs and then rising again, which shows us that our model suffers from instability in generalization. Furthermore, the gap between the training and

validation losses widens over time, meaning that the model learns the training data very well but does not generalize as efficiently to new data.

3.7 Some future suggestions to improve our model:

We can increase the model's efficiency using methods like:

- Dropout or L2 Regularization
- Experimenting with data augmentation to increase data diversity to help the model generalize better and reduce overfitting
- Implementing early stopping to prevent the model from continuing training past a certain point when the model begins to deteriorate on validation.

General conclusion

The work presented in this thesis falls within the framework of leveraging deep learning techniques in medical image processing , specifically in the classification of chest X-ray images using the Vision Transformer (ViT) model. In our case, we tested a ViT model that was not pre-trained, focusing on studying its performance when applied to a dataset specialized in chest X-ray images. The results showed that the model has a good learning capacity; however, we observed challenges related to overfitting, which we analyzed through a series of experiments and adjustments to the hyperparameters.

Through the evaluation process and the use of appropriate metrics, we were able to observe the effect of data size, number of layers, and learning rate on the model's performance. These experiments allowed us to gain a deeper understanding of the ViT's capabilities in the medical classification domain, while highlighting the importance of carefully tuning the model to achieve optimal performance.

In fact, we faced technical difficulties during implementation, particularly in adapting the model to work with real-world data beyond classical training datasets such as MNIST. The experience was a valuable exploration of deep learning's potential in the medical field.

As future perspectives for this work, The model's effectiveness in medical images can be evaluated by comparing it with other models like CNN or ResNet.
The model's performance could be enhanced by utilizing a larger, diverse dataset or local data from hospitals or medical centers through research partnerships. Additionally, a decision-support tool for doctors in preliminary diagnosis could be developed as a user interface, thereby enhancing the practical value of this research.

Bibliography

[1] AZOUZI, A. (2022). *DEBBI Aimad-Eddin University of M9sila President BOUZAROURA Ahlem University of M9sila Reporter YAGOUBI Rached University of M9sila Examiner* (Doctoral dissertation, MINISTRY OF HIGHER EDUCATION).

[2] Galindo, J. (Ed.). (2008). *Handbook of research on fuzzy information processing in databases*. IGI Global.

[3] Maglogiannis, I. G. (Ed.). (2007). *Emerging artificial intelligence applications in computer engineering: real word ai systems with applications in ehealth, hci, information retrieval and pervasive technologies* (Vol. 160). Ios Press.

[4] Dahan, H., Cohen, S., Rokach, L., Maimon, O., Dahan, H., Cohen, S., ... & Maimon, O. (2014). *Proactive data mining using decision trees* (pp. 21-33). Springer New York.

[5] Surhone, L. M., Timpledon, M. T., & Marseken, S. F. (2010). Naive Bayes classifier: classifier (mathematics), Bayes' theorem, probability theory, Bayesian inference, Bayesian probability, empirical Bayes method, statistics, conditional probability. *(No Title)*.

[6] Avila, J., & Hauck, T. (2017). *Scikit-learn cookbook: over 80 recipes for machine learning in Python with scikit-learn*. Packt Publishing Ltd.

[7] Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction. *(No Title)*.

[8] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, *1*(4), 541-551.

[9] Fukushima, K. (2021). Artificial vision by deep CNN neocognitron. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *51*(1), 76-90.

[10] AZOUZI, A. (2022). *DEBBI Aimad-Eddin University of M9sila President BOUZAROURA Ahlem University of M9sila Reporter YAGOUBI Rached University of M9sila Examiner* (Doctoral dissertation, MINISTRY OF HIGHER EDUCATION).

[11] Khelil, M. I., & Ladjal, M. (2021, June). Hybrid Predictive Models for Water Quality Assessment Based on Water Quality Index Using ANN, LSSVM and multivariate statistical Methods. In *9th (Online) International Conference on Applied Analysis and Mathematical Modeling (ICAAMM21) June 11-13, 2021, Istanbul-Turkey* (p. 82).

[12] McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, *5*, 115-133.

[13] Hebb, D. O. (1949). Organization of behavior. new york: Wiley. *J. Clin. Psychol*, *6*(3), 335-307.

[14] B. Widrow and M. Hoff, "Adaptive switching circuits," 1960 IRE WESCON Convention Record, New York: IRE, pp. 96-104, 1960.

[15] Kohonen, Teuvo. (2007). Description of Input Patterns by Linear Mixtures of SOM Models.

[16] Hopfield, John. (1984). Neurons With Graded Response Have Collective Computational Properties Like Those of Two-State Neurons. Proceedings of the National Academy of Sciences of the United States of America. 81. 3088-92. 10.1073/pnas.81.10.3088. Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the national academy of sciences*, *81*(10), 3088-3092.

[17] AZOUZI, A. (2022). *DEBBI Aimad-Eddin University of M9sila President BOUZAROURA Ahlem University of M9sila Reporter YAGOUBI Rached University of M9sila Examiner* (Doctoral dissertation, MINISTRY OF HIGHER EDUCATION).

[18] Lee, J. G., Jun, S., Cho, Y. W., Lee, H., Kim, G. B., Seo, J. B., & Kim, N. (2017). Deep learning in medical imaging: general overview. *Korean journal of radiology*, *18*(4), 570-584..

[19] Heaton, J. (2018). Ian goodfellow, yoshua bengio, and aaron courville: Deep learning: The mit press, 2016, 800 pp, isbn: 0262035618. *Genetic programming and evolvable machines*, *19*(1), 305-307.

[20] Szandała, T. (2020). Review and comparison of commonly used activation functions for deep neural networks. In *Bio-inspired neurocomputing* (pp. 203-224). Singapore: Springer Singapore.

[21] AZOUZI, A. (2022). *DEBBI Aimad-Eddin University of M9sila President BOUZAROURA Ahlem University of M9sila Reporter YAGOUBI Rached University of M9sila Examiner* (Doctoral dissertation, MINISTRY OF HIGHER EDUCATION).

[22] Zhang, Y., & Wallace, B. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*.

[23] Yann LeCun, Léon Bottou, Yoshua Bengio, Ptrick Haffner, Proc of the IEEE nov 1998. !!!!!!!!!!!!

[24] Kull, M., & Kuhaupt, N. (2019). *Application and Evaluation of LSTM Architectures for Energy Time-Series Forecasting* (Doctoral dissertation, Master's Thesis, University of Tartu, Tartu, Estonia, 2019.[Google Scholar]). [25] dataanalytics(website), june 2022, url : https://dataanalyticspost.com/Lexique/auto-enco

[26] (website), june 2022 url:

https://community.canvaslms.com/t5/Canvas-Developers-Group/Canvas-LMS-Cheat-DetectionSystem-In-Python/m-p/118134.

[27] Lebigdata website, june 2022,url: https://www.lebigdata.fr/reinforcement-learning-definition.

[28] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

[29] Tan, M., & Le, Q. (2019, May). Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning* (pp. 6105-6114). PMLR.

[30] Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., & Joulin, A. (2021). Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 9650-9660).

[31] Hosseinzadeh Taher, M. R., Haghighi, F., Feng, R., Gotway, M. B., & Liang, J. (2021). A systematic benchmarking analysis of transfer learning for medical image analysis. In *Domain Adaptation and Representation Transfer, and Affordable Healthcare and AI for Resource Diverse Global Health: Third MICCAI Workshop, DART 2021, and First MICCAI Workshop, FAIR 2021, Held in Conjunction with MICCAI 2021, Strasbourg, France, September 27 and October 1, 2021, Proceedings 3* (pp. 3-13). Springer International Publishing.

[32] Parvaiz, A., Khalid, M. A., Zafar, R., Ameer, H., Ali, M., & Fraz, M. M. (2023). Vision Transformers in medical computer vision—A contemplative retrospection. *Engineering Applications of Artificial Intelligence*, *122*, 106126.

[33] Weinstein, J. N., Collisson, E. A., Mills, G. B., Shaw, K. R., Ozenberger, B. A., Ellrott, K., ... & Stuart, J. M. (2013). The cancer genome atlas pan-cancer analysis project. *Nature genetics*, *45*(10), 1113-1120.

[34] Gunraj, H., Sabri, A., Koff, D., & Wong, A. (2022). COVID-Net CT-2: Enhanced deep neural networks for detection of COVID-19 from chest CT images through bigger, more diverse learning. *Frontiers in Medicine*, *8*, 729287.

[35] Tabik, S., Gómez-Ríos, A., Martín-Rodríguez, J. L., Sevillano-García, I., Rey-Area, M., Charte, D., ... & Herrera, F. (2020). COVIDGR dataset and COVID-SDNet methodology for predicting COVID-19 based on chest X-ray images. *IEEE journal of biomedical and health informatics*, *24*(12), 3595-3605.

[36] Saeed, F., Hussain, M., Aboalsamh, H. A., Al Adel, F., & Al Owaifeer, A. M. (2023). Designing the architecture of a convolutional neural network automatically for diabetic retinopathy diagnosis. *Mathematics*, *11*(2), 307.

[37] Kather, J. N., Zöllner, F. G., Bianconi, F., Melchers, S. M., Schad, L. R., Gaiser, T., ... & Weis, C. A. (2016). Collection of textures in colorectal cancer histology. *Zenodo https://doi. org/10*, *5281*.

[38] Karacop, E., & Ozdemir, R. (2020). Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website. *Journal of Electrocardiology*, *62*, 59-64.

[39] Kollias, D., Arsenos, A., Soukissian, L., & Kollias, S. (2021). Mia-cov19d: Covid-19 detection through 3-d chest ct image analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 537-544).

[40] Soares, E., Angelov, P., Biaso, S., Froes, M. H., & Abe, D. K. (2020). SARS-CoV-2 CT-scan dataset: A large dataset of real patients CT scans for SARS-CoV-2 identification. *MedRxiv*, 2020-04.

[41] Maguolo, G., & Nanni, L. (2021). A critic evaluation of methods for COVID-19 automatic detection from X-ray images. *Information fusion*, *76*, 1-7.

[42] Yang, X., He, X., Zhao, J., Zhang, Y., Zhang, S., & Xie, P. (2020). Covid-ct-dataset: a ct scan dataset about covid-19. *arXiv preprint arXiv:2003.13865*.

[43] Haghanifar, A., Majdabadi, M. M., Choi, Y., Deivalakshmi, S., & Ko, S. (2022). Covid-cxnet: Detecting covid-19 in frontal chest x-ray images using deep learning. *Multimedia tools and applications*, *81*(21), 30615-30645.

[44] Hajeb Mohammad Alipour, S., Rabbani, H., & Akhlaghi, M. R. (2012). Diabetic retinopathy grading by digital curvelet transform. *Computational and mathematical methods in medicine*, *2012*(1), 761901.

[45] Heller, N., Isensee, F., Maier-hein, K. H., Hou, X., Xie, C., Li, F., Nan, Y., Mu, G., Lin, Z., Han, M., Yao, G., Zhang, Y., Wang, Y., Hou, F., Yang, J., Xiong, G., Tian, J., Zhong, C., Rickman, J., … States, U. (2022). HHS Public Access. 1– 39. https://doi.org/10.1016/j.media.2020.101821.The

[46] Campello, V. M., Gkontra, P., Izquierdo, C., Martin-Isla, C., Sojoudi, A., Full, P. M., ... & Lekadir, K. (2021). Multi-centre, multi-vendor and multi-disease cardiac segmentation: the M&Ms challenge. *IEEE Transactions on Medical Imaging*, *40*(12), 3543-3554.

[47] Peterfy, C. G., Schneider, E., & Nevitt, M. (2008). The osteoarthritis initiative: report on the design rationale for the magnetic resonance imaging protocol for the knee. *Osteoarthritis and cartilage*, *16*(12), 1433-1441.

[48] Shiraishi, J., Katsuragawa, S., Ikezoe, J., Matsumoto, T., Kobayashi, T., Komatsu, K. I., ... & Doi, K. (2000). Development of a digital image database for chest radiographs with and without a lung nodule: receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. *American journal of roentgenology*, *174*(1), 71-74.

[49] Moreira, I. C., Amaral, I., Domingues, I., Cardoso, A., Cardoso, M. J., & Cardoso, J. S. (2012). Inbreast: toward a full-field digital mammographic database. *Academic radiology*, *19*(2), 236-248.

[50] Shao, Z., Bian, H., Chen, Y., Wang, Y., Zhang, J., & Ji, X. (2021). Transmil: Transformer based correlated multiple instance learning for whole slide image classification. *Advances in neural information processing systems*, *34*, 2136-2147.

[51] Wang, X., Yang, S., Zhang, J., Wang, M., Zhang, J., Yang, W., ... & Han, X. (2022). Transformer-based unsupervised contrastive learning for histopathological image classification. *Medical image analysis*, *81*, 102559.

[52] Chen, H., Li, C., Wang, G., Li, X., Rahaman, M. M., Sun, H., ... & Grzegorzek, M. (2022). GasHis-Transformer: A multi-scale visual transformer approach for gastric histopathological image detection. *Pattern Recognition*, *130*, 108827.

[53] Chen, R. J., & Krishnan, R. G. (2022). Self-supervised vision transformers learn visual concepts in histopathology. *arXiv preprint arXiv:2203.00585*.

[54] Liang, S., Zhang, W., & Gu, Y. (2021). A hybrid and fast deep learning framework for Covid-19 detection via 3D Chest CT Images. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 508-512).

[55] Barhoumi, Y., & Rasool, G. (2021). Scopeformer: n-CNN-ViT hybrid model for intracranial hemorrhage classification. *arXiv preprint arXiv:2107.04575*.

[56] Xia, Y., Yao, J., Lu, L., Huang, L., Xie, G., Xiao, J., ... & Zhang, L. (2021). Effective pancreatic cancer screening on non-contrast CT scans via anatomy-aware transformers. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part V 24* (pp. 259-269). Springer International Publishing.

[57] Tang, Y., Yang, D., Li, W., Roth, H. R., Landman, B., Xu, D., ... & Hatamizadeh, A. (2022). Self-supervised pre-training of swin transformers for 3d medical image analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 20730-20740).

[58] Fauw. (2022). *Can Transformers Replace Cnns for Medical Image Classification?* 1–15

Park, N., & Kim, S. (2022). *How Do Vision Transformers Work?* http://arxiv.org/abs/2202.06709

Shuang Yu, K. M. (2021). *MIL-VT: Multiple Instance Learning Enhanced Vision Transformer for Fundus Image Classification*

[59] Morid, M. A., Borjali, A., & Del Fiol, G. (2021). A scoping review of transfer learning research on medical image analysis using ImageNet. *Computers in biology and medicine*, *128*, 104115.

[60] Raghu, M., Zhang, C., Kleinberg, J., & Bengio, S. (2019). Transfusion: Understanding transfer learning for medical imaging. *Advances in neural information processing systems*, *32*.

[61] - Afouras, T., Owens, A., Chung, J. S., & Zisserman, A. (2020). Self-supervised learning of audio-visual objects from video. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16* (pp. 208-224). Springer International Publishing.

- Hendrycks, D., Mazeika, M., Kadavath, S., & Song, D. (2019). Using self-supervised learning can improve model robustness and uncertainty. *Advances in neural information processing systems*, *32*.

- Kolesnikov, A., Zhai, X., & Beyer, L. (2019). Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 1920-1929).

- Xu, J., Xiao, L., & López, A. M. (2019). Self-supervised domain adaptation for computer vision tasks. *IEEE Access*, *7*, 156694-156706.

[62] Wang, X., Yang, S., Zhang, J., Wang, M., Zhang, J., Yang, W., ... & Han, X. (2022). Transformer-based unsupervised contrastive learning for histopathological image classification. *Medical image analysis*, *81*, 102559.

[63] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

[64] Bahdanau, D., Cho, K. H., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 1–15.

[65] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

[66] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020, August). End-to-end object detection with transformers. In *European conference on computer vision* (pp. 213-229). Cham: Springer International Publishing.

[67] Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., & Jégou, H. (2020). *Training data-efficient image transformers & distillation through attention*. 1–22. http://arxiv.org/abs/2012.12877

[68] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., ... & Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 10012-10022).

[69] Python. What is Python? June 1, 2022. url: https://www.python.org/doc/essays/blurb/

[70] "Numpy official website," [Online]. Available: https://www.numpy.org/

[71] https://docs.pytorch.org/docs/stable/index.html

[72] https://docs.pytorch.org/vision/stable/index.html

[73] https://matplotlib.org/stable/users/explain/text/index.html

[74] https://tqdm.github.io/

[75] Google Colab official website, url: https://colab.research.google.com/

[76] https://www.techtarget.com/searchmobilecomputing/definition/Google-Drive

[77] Data: https://data.mendeley.com/datasets/rscbjbr9sj/2

License: CC BY 4.0 Citation: http://www.cell.com/cell/fulltext/S0092-8674(18)30154-5