PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
الجمهـــوريـــــة الجزائـريـــــة الديـمقـراطيـــــة الشـعـبيـــــة
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
وزارة التعليـم العالـــي والبحـــث العلمـي
HIGHER SCHOOL OF COMPUTER SCIENCE AND DIGITAL TECHNOLOGIES-BEJAIA
المـــدرسة العـــليا في علـــوم وتكنـولوجيـات الاعـــلام الالـــي بجـــاية

Dissertation Submitted to the Department Of Computer Science in Partial Fulfillment of the Requirements for Engineer's Degree in Computer Science

Specialty: Artificial Intelligence and Data Sciences

Submitted By: **Anis MOHAMMEDI**

Seismic Image Segmentation based on Deep Learning for the Characterization of Hydrocarbon Reservoirs in the Oil & Gas Industry : Comparative Study

Supervised by: **Dr.** Abderrazek SEBAA (ESTIN)

Dr. Kamel SOUADIH (SONATRACH RTC-BEJAIA)

Members of jury:

▪ **Dr.** Sylia ZENADJI	President	ESTIN
▪ **Dr.** Chemseddine BERBAGUE	Examiner	ESTIN
▪ **PhD Student.** Amine BECHAR	Examiner	ESTIN
▪ **PhD Student.** Chawki ABBES	Examiner	ESTIN

Academic year: 2023/2024

Abstract

Accurate identification and delineation of Salt Domes from seismic images play a critical role in geological studies and resource exploration. However, traditional methods often struggle with complex geological structures and require extensive manual intervention. This work addresses these challenges by proposing a deep learning-based approach for semantic segmentation of Salt Domes. We introduce novel techniques including Transformers for spatial context aggregation, U-Net for precise feature extraction, and VAE-Liquid layers for enhanced representation learning. Through rigorous experimentation and evaluation on real-world datasets, we demonstrate the effectiveness of our approach in automating and improving the accuracy of Salt Dome identification. This work contributes to advancing automated geological analysis, offering insights into subsurface structures vital for both exploration and hazard assessment.

Keywords— Semantic Segmentation, Seismic Images, Salt Domes, Resource Exploration, Deep Learning, Transformers, U-Net, VAE-Liquid Layers

As when a fog disperseth gradually
Our vision traces what the mist involves
Condens'd in air; so piercing through the gross
And gloomy atmosphere, as more and more
We near'd toward the brink, mine error fled ...

The Divine Comedy, Vol. 1 (Inferno) - Canto XXXI , Dante Alighieri

Acknowledgment

I would like to extend my sincere gratitude to everyone who has supported and guided me throughout the completion of my engineering degree thesis.

First and foremost, I am deeply grateful to my advisor, Dr. SOUADIH Kamel, for his exceptional guidance, insightful advice, and unwavering support. His expertise and encouragement have been crucial in the development and completion of this thesis.

I would also like to thank the faculty and staff of the School for providing a conducive learning environment and the resources necessary for my research. I also extend my sincere thanks to Dr. SEBAA Abderrazek for his valuable notices and advices. His critical insights and constructive feedback have significantly improved the quality of this work. His attention to detail and willingness to share his knowledge have been greatly appreciated.

I am particularly thankful to my classmates and friends who have been a constant source of support and motivation. Their constructive feedback and camaraderie have significantly enriched my academic experience.

I would like to acknowledge the financial and logistical support provided by the library stuff especially Mr. ADOUANE Nacer. Your contributions have been instrumental in enabling this research.

A heartfelt thank you goes to my family for their endless support, patience, and love. To my parents and my dear sister, your belief in my abilities and your constant encouragement have been my greatest strength.

Finally, I would like to thank everyone who has contributed, directly or indirectly, to the successful completion of this thesis. Your support and encouragement have been greatly appreciated.

Contents

List of Figures

List of Tables

General Introduction

Salt domes are critical geological formations with substantial subsurface salt concentrations, playing key roles in various industrial applications, including hydrocarbon reservoirs and potential nuclear waste storage sites. Found in sedimentary basins, these structures significantly influence geological history and resource extraction processes.

They also impact geohazard assessment and environmental management due to risks like subsidence and groundwater contamination. Understanding salt domes' complex structure is essential, yet challenging, due to the limitations of traditional imaging techniques.

Seismic surveying is a primary method for imaging salt domes, utilizing seismic waves to create detailed subsurface images. However, salt's unique properties complicate this process, requiring advanced imaging approaches.

With the rise of deep learning, especially in semantic segmentation, these techniques are now being applied to seismic imaging. In 2018, TGS's Kaggle competition (*https://www.kaggle.com/c/tgs-salt-identification-challenge*) showcased the potential of deep learning for segmenting salt domes in seismic images, achieving an IoU score of 87%, despite the absence of advanced models like transformers at the time. This highlighted the promising intersection of deep learning and seismic imaging for better salt dome characterization.

Imaging salt domes with seismic surveying is challenging due to the rapid seismic wave velocities in salt, which create complex interference patterns and obscure underlying geological features. Traditional methods, such as U-Net [1], struggle with these complexities because seismic waves in salt act like lenses, leading to unpredictable focusing and refraction, and generating numerical artifacts like multiples and diffractions that distort seismic

data. Additionally, the scarcity of annotated data limits the development and training of robust segmentation models, resulting in less effective and poorly generalized solutions.

Goals

The main aim of this dissertation is to tackle the current challenges in segmenting salt domes and to develop a new, efficient model that can overcome these obstacles. The objectives include:

- Developing an innovative segmentation model tailored specifically for salt dome imaging, leveraging state-of-the-art deep learning techniques.
- Enhancing segmentation accuracy and robustness by incorporating advanced methodologies, such as mixed-transformer architectures and novel loss functions.
- Mitigating the impact of data scarcity by implementing effective data augmentation strategies and transfer learning techniques.
- Validating the proposed model through comprehensive quantitative and qualitative evaluations, comparing its performance against existing methodologies.

Work Plan

The dissertation is structured into four chapters:

1. **Chapter 1:Salt Domes and Seismic Survey** : This chapter provides an in-depth overview of seismic surveying, salt dome imaging challenges, and the state of the art in segmentation methodologies.
2. **Chapter 2: Machine Learning and Deep Learning** : Here, we detail the development of our proposed segmentation model, outlining the architectural design, training procedures, and data augmentation strategies.
3. **Chapter 3 & Chapter 4 : Experimental Evaluation and Results** : This chapter presents the comparative evaluation results of our model

against existing techniques, emphasizing its superior performance and implications.

Chapter 1

Salt Domes and Seismic Survey

1.1 Introduction

Salt Domes and Seismic Surveys are vital in hydrocarbon exploration. Salt Domes trap oil and gas, while Seismic Surveys visualize subsurface structures. This chapter outlines Salt Dome significance and detection methods, followed by an exploration of Seismic Surveys, their purpose, applications, and role in Salt Dome detection.

1.2 Salt Domes

1.2.1 Definition

Salt Domes, also known as salt diapirs, are geological structures that form when underground salt layers or beds rise toward the Earth's surface. These salt layers, composed primarily of halite (rock salt), are less dense than the surrounding sedimentary rocks. As a result, they can migrate vertically over geological time, pushing their way through the layers of sedimentary rock above them (see Figure 1.1). Over millions of years, the salt can intrude and deform the overlying rock layers, causing the formation of a dome-like structure. These Salt Domes can vary in size from a few meters to several kilometers in height and width [2].

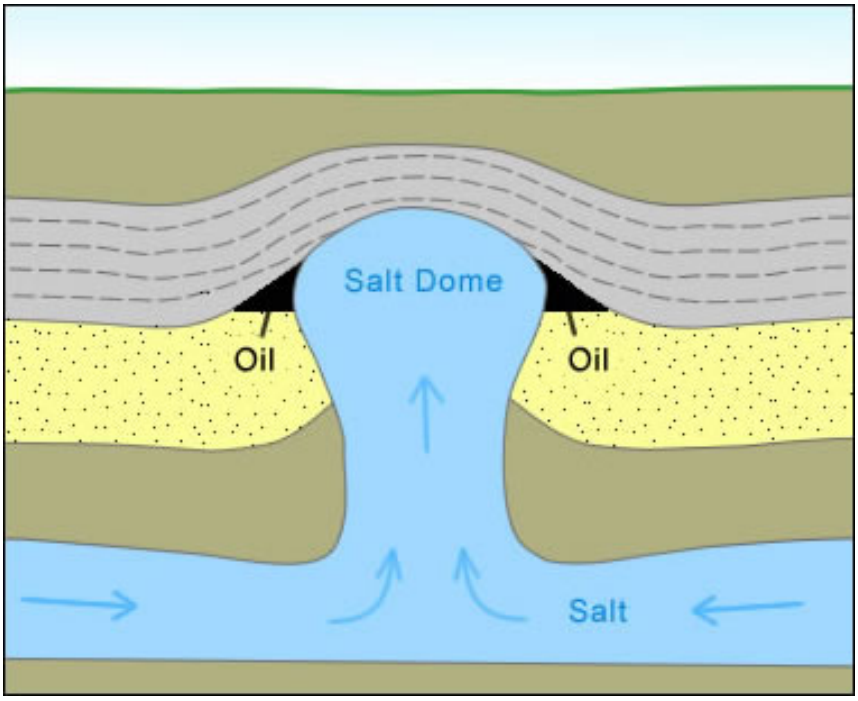


Figure 1.1: Formation of a Salt Dome Through Sedimentary Rock Layers [3].

1.2.2 Importance of Salt Domes

Salt Domes hold significant relevance in hydrocarbon exploration, serving as potential reservoirs for oil and gas. They are a focal point of interest due to their role as geological structures that can trap hydrocarbons, making them crucial in the energy industry. Understanding and detecting Salt Domes are essential for efficient hydrocarbon resource exploration.

Salt deposits are found all over the world, but they are most concentrated in certain regions. The following is a list of some of the countries with the largest salt deposits (see Figure 1.2) :

- **North America**: United States, Canada, Mexico
- **South America**: Brazil, Argentina, Chile
- **Europe**: Russia, Germany, United Kingdom, France, Poland
- **Asia**: China, India, Iran, Pakistan, Saudi Arabia
- **Africa**: Algeria, Angola, Egypt, Libya

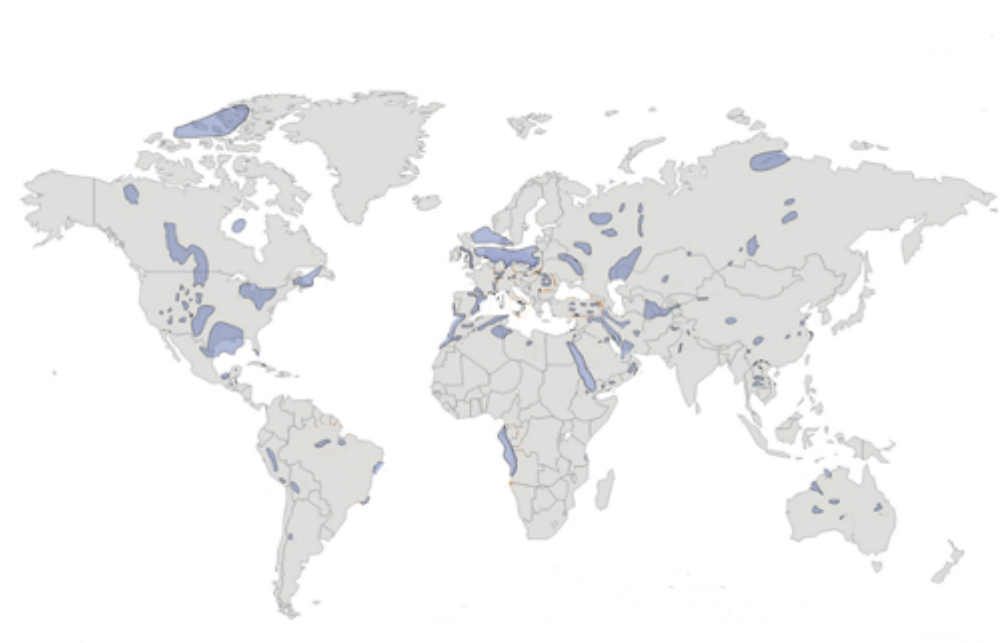


Figure 1.2: Salt deposits world wide [4].

- **Australia**

That is, the importance of these structures derives from these reasons [3]:

- **Hydrocarbon Traps** : Salt Domes can serve as traps for hydrocarbons, such as oil and natural gas. Over millions of years, salt can create structural deformations in the overlying sedimentary rock layers, forming pockets where hydrocarbons can accumulate. This makes Salt Domes important targets for oil and gas exploration.

- **A Source of Sulfur** : Sulfur recovery from Salt Domes with sulfur-rich cap rock involves drilling a well and injecting superheated water and air to melt and bring the sulfur to the surface. However, this method is generally less cost-effective than obtaining sulfur as a byproduct of crude oil refining and natural gas processing, which is the primary source of sulfur production today.

- **Salt Production** : Certain Salt Domes have been utilized for underground mining operations, extracting salt used as a raw material in the chemical industry and for de-icing snowy highways. In some instances, Salt Domes are mined through a solution mining technique, where hot water is injected down a well to dissolve the salt. The saline solution

is then brought to the surface through production wells, where the salt is recovered through evaporation or employed in chemical processes.

- **Underground Storage Reservoirs** : Certain Salt Dome mines have been deliberately sealed and repurposed as storage facilities for oil, natural gas, and hydrogen. In both the United States and Russia, Salt Domes are utilized as national repositories for government reserves of helium gas. The unique quality of salt as a rock with exceptionally low permeability [1] enables it to securely retain even the smallest helium atoms.

- **Waste Disposal** : Salt's impermeable nature and ability to self-seal fractures make Salt Domes suitable for hazardous waste disposal, including oil field drilling waste, while also being considered for high-level nuclear waste storage in the United States.

1.2.3 The Detection of Salt Domes

There are variety of techniques used for detecting Salt Domes, including:

- **Seismic surveys** : Seismic surveys are the most common method for detecting Salt Domes. Seismic waves are generated at the surface and travel through the subsurface, reflecting off of geological interfaces. The reflected waves are recorded at receivers on the surface and then processed to create images of the subsurface. Salt Domes can be identified on seismic images by their characteristic diapiric shape and high acoustic impedance [5].

- **Gravity surveys** : Gravity surveys measure the variations in the Earth's gravity field. Salt Domes have a high density, so they cause a positive anomaly in the gravity field. This anomaly can be used to identify the location and extent of Salt Domes [6].

- **Magnetic surveys** : Magnetic surveys measure the variations in the Earth's magnetic field. Salt Domes can sometimes cause magnetic anomalies, but these anomalies can be difficult to interpret [6].

[1] Permeability is how easily a liquid or gas can move through a material with tiny holes, like water soaking through a sponge.

- **Borehole data** : Borehole data, such as well logs and core samples, can be used to identify Salt Domes. Salt Domes typically have a distinctive well log response and lithology [7].

- **Machine learning** : In addition to these traditional methods,some of ML techniques are developed recently for detecting Salt Domes [8].

1.3 Seismic Exploration

1.3.1 Definition

Seismic Surveying is the most important geophysical exploration method because it can detect subsurface features of all sizes. Seismic methods involve sending sound waves into the Earth and measuring their reflections to determine the shapes and properties of subsurface layers. Early oil explorers found oil by drilling at natural oil seeps and large folds in exposed rocks. Once these easy targets were depleted, geologists turned to seismic surveying to find more elusive oil and gas traps. Seismic technology has been used to measure water depths and detect icebergs since the early 1900s. In 1924, seismic data was first used to discover an oil field in Texas [9].

Seismic exploration is used in a wide variety of industries, including:

- **Oil and gas**: Seismic exploration is used to find and map oil and gas deposits [10].

- **Mining**: Seismic exploration is used to find and map mineral deposits, such as gold, copper, and iron [10].

- **Geotechnical engineering**: Seismic exploration is used to assess the stability of soil and rock formations for construction projects [11].

- **Environmental science**: Seismic exploration is used to study the subsurface environment and to identify potential hazards, such as groundwater contamination and seismic faults [12].

- **Archaeology**: Seismic exploration is used to map buried archaeological sites[13].

1.3.2 Basic Principles

Seismology is based on the transmission of sound waves by the rocks of the crust. Strong earthquakes create pressure waves (natural sources of seismic waves) that are transmitted through the entire Earth and detected by seismographs (receivers) on the other side (see Figure 1.4) [9].

Seismic exploration, however, as employed by the petroleum geologist, makes use of artificially generated pulses (seismic sources).

An impulse source sends acoustic energy into the Earth. This energy propagates in many directions and is reflected and refracted when it encounters boundaries between two layers. Sensors (seismic receivers) placed on the surface measure the reflected or refracted acoustic energy.

Different seismic sources are used in land and marine seismic acquisitions (see Figure 1.3). In marine environments, arrays of air guns are used to generate seismic energy. In land seismic, explosives or vibrators are used (see Table 1.1).

Table 1.1: Different seismic sources used in land and marine seismic acquisitions

Seismic source	Environment	Advantages	Disadvantages
Air gun	Marine	Relatively inexpensive and environmentally friendly	Can be affected by weather conditions
Vibrator	Land	Less expensive and disruptive than explosives	Not as powerful as explosives
Dynamite	Land	Most powerful seismic source	Most expensive and disruptive seismic source

Seismic receivers are devices that detect seismic energy and convert it into an electrical signal. The two most common types of seismic receivers are hydrophones and geophones (see Table 1.2).

Table 1.2: Different seismic receivers used in land and marine seismic acquisitions

Seismic receiver	Type	Application
Hydrophone	Underwater	Marine seismic surveys
Geophone	Surface	Surface seismic surveys (land and offshore)
3C geophone	Surface	Seafloor seismic surveys
4C detector unit	Surface	Seafloor seismic acquisition

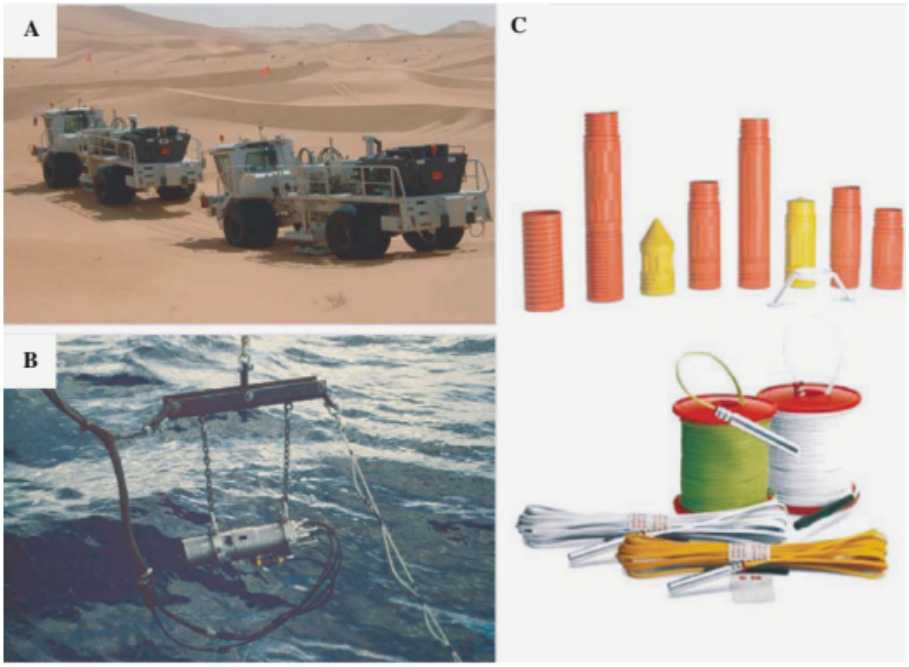


Figure 1.3: (a) Seismic acquisition in a desert where vibrator is the most common source of seismic energy. (b) An air-gun before deployment in the water. It releases compressed air into the water during marine seismic survey. (c) Explosives (top) [9].

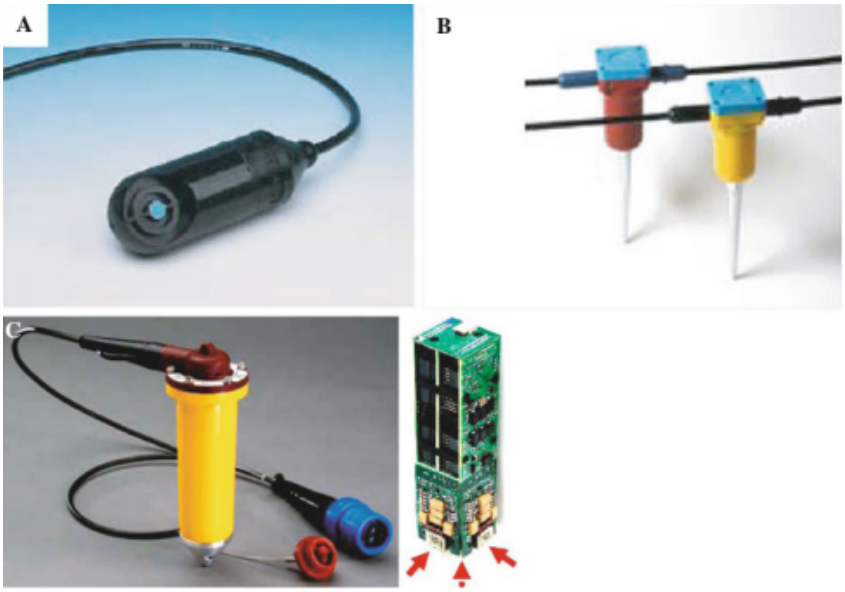


Figure 1.4: (a) Hydrophone, (b) geophone and (c) multi-component geophone (Courtesy ION Geophysical) [9].

1.3.3 The Process of Detecting Salt Domes

The process of using a geophysical method to estimate subsoil parameters typically involves the following steps [14]:

1. **Collect data**: This typically involves deploying sensors on the surface of the ground and recording the response of the subsoil to a controlled source of energy, such as a seismic vibrator or electrical transmitter.

2. **Process the data**: Once the data has been collected, it must be processed to remove noise and artifacts. This can be done using a variety of software programs according to the following steps :

   (a) **Preprocessing**:data are converted to a convenient format for the next processing steps. Preprocessing also involves trace editing, where noisy traces could be eliminated. Gaining also could be performed at this stage, which involves amplitude compensation due to geometric dispersion.

   (b) **Deconvolution**: Deconvolution is a process that is used to improve the resolution of the seismic data. It does this by compressing the seismic wavelet, which is the signature of the seismic source.

   (c) **Stacking**:Stacking is the process of combining multiple traces of seismic data to improve the signal-to-noise ratio.

   (d) **Migration**:Migration is a process that is used to image the subsurface in its true spatial position. This is done by correcting for the effects of wave propagation.

3. **Interpret the data**: Once the data has been processed, it must be interpreted to determine the subsoil parameters of interest. This typically involves using a combination of forward modeling and inverse modeling techniques.

1.3.4 Scientific Problems

Interpreting salt structures in the context of geology and geophysics can be particularly challenging due to several factors. In fact ,The density of Salt Domes and the surrounding rocks can vary significantly. Salt is typi-

cally less dense than sandstone and limestone, but more dense than shale (2.16 g/cm³ for Halite, whereas 2.71g/cm³for sandstone). This can lead to significant acoustic impedance contrasts between Salt Domes and the surrounding rocks. Acoustic impedance is a measure of how much a material resists the passage of sound waves.therefore, Seismic waves travel through salt at much higher velocities than through other rocks. This is because salt is more elastic than other rocks (4.5km/s for salt structures whereas 3.5km/s for the surronding structures) [15, 14].

Here are some problems which make salt interpretation difficult:

- Interpret seismic data using a prior model based on the geological knowledge of the region.
- Steeply inclined flanks are difficult to map because seismic waves graze on them, making it hard to define their boundaries. This can be improved by increasing the distance between the source and receiver.
- High impedance contrast between salt and sediments traps seismic waves inside salt structures, generating multiples that interfere with primary events, making sub-saline sediments difficult to image.
- Seismic waves follow complex paths in the vicinity of Salt Domes, resulting in weak correlation between real and predicted data, lower image quality, and prism waves. Prism waves are generated by two reflections along the travel path from source to receiver, creating spurious artifacts. They are common in regions with steeply sloping flanks, such as Salt Domes.

This is why it can take weeks and demand higher quality experts to detect Salt Domes in seismic images as shown in Figure 1.5 [16], efforts have been made to expedite the process. In response to this challenge, scientists have started exploring the application of deep learning techniques in the domain like [8], which has shown interesting and promising results, underscoring the potential transformative impact these technologies hold in rapidly and accurately detecting Salt Domes within seismic imagery, we will explore more in the next chapter.

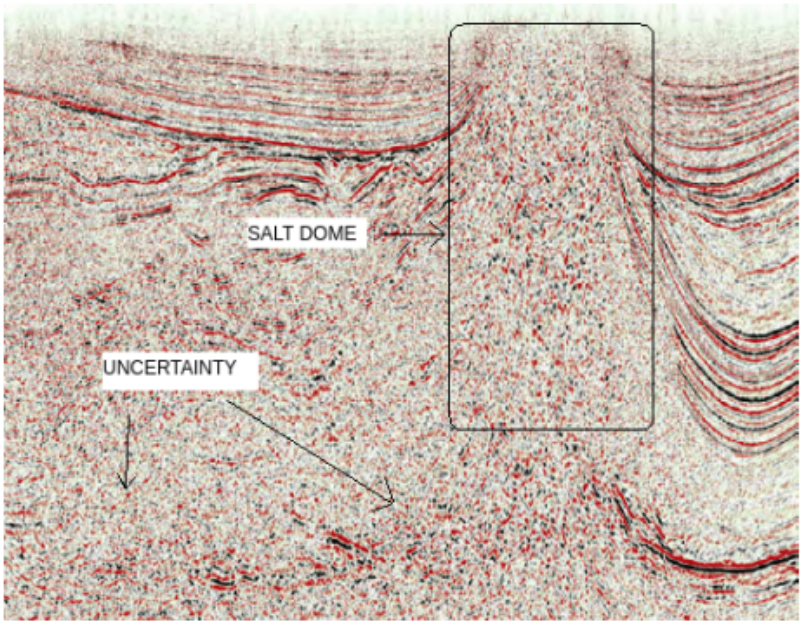


Figure 1.5: Example of a seismic image of a Salt Dome [14].

1.4 State of The Art

In this section ,we present some of works related to seismic data processing techniques used as a means for salt delineation.

1.4.1 Traditional Methods (Classical Signal-Image Processing methods)

Below, we present notable works that have employed traditional image processing techniques :

N. KESKES & al. (1982) - APPLICATION OF IMAGE ANALYSIS TECHNIQUES TO SEISMIC DATA [17]: It's the first paper that mention the using of basics of digital image processing to segment cross-sections seismical images either in 3D or in 2D.The article focuses on applying digital image analysis techniques to seismic cross-sections, which are 2D vertical slices through 3D seismic data volumes used in oil/gas exploration. It presents an edge detection algorithm to automatically detect seismic horizons/rock interfaces, and employs statistical texture analysis using first and second-order statistics to characterize and classify different seismic facies regions based on textural patterns. The article also describes using dynamic clustering with texture/depth features to classify points into classes, and a split-and-merge

segmentation technique utilizing local edge orientation to identify geological features like faults and channels. Overall, it demonstrates how adapting image processing methods can aid seismic interpretation by automating horizon/facies detection and delineating structures.

I. PITAS & al. (1987) - AGIS [18]: The paper presents an expert system called AGIS for automated interpretation of seismic images in oil exploration. It employs techniques like edge detection, texture analysis, and line extraction for low-level image processing. At the high-level, it uses a semantic network with frames to represent geological knowledge about seismic patterns like horizons, anticlines, faults, and Salt Domes. Hypothesis verification with certainty factors is used for pattern matching and recognition. The system combines image processing routines with expert knowledge to automate seismic interpretation tasks like detecting traps, faults, and geological formations.

Adam D. Halpert & al. (2013) - Salt delineation via interpreter-guided 3D seismic image segmentation [19] : This paper presents a semi-automatic method for delineating salt bodies in 3D seismic images using a graph-based image segmentation algorithm. The technique modifies the efficient pairwise region comparison (PRC) algorithm by using larger stencils and edge weighting schemes tailored for seismic data. In areas where salt boundaries are poorly imaged, limited manual interpretations on 2D slices can guide the 3D segmentation by modifying the input image amplitudes. The method combines computational efficiency, accurate automatic segmentation, and the ability to incorporate interpreter insight for challenging areas, making it a promising tool for salt interpretation workflows.

M.DERICHE & al. (2015) - A Novel Approach for Salt Dome Detection using A Dictionary-based Classifier [20] The paper presents a novel dictionary-based classification approach for detecting Salt Domes in seismic data using texture attributes. It combines Gray Level Co-occurrence Matrix (GLCM) attributes and Gradient of Texture (GoT) attributes with a dictionary-based learning model to classify salt boundary and non-salt boundary patches. The proposed algorithm is shown to work well even with weak reflectors along the salt boundary and outperforms existing gradient-based and texture-based techniques when used separately, while using a minimal set of features.

M.DERICHE & al. (2015) - A new approach for Salt Dome detection using a 3D multidirectional edge detector [21] The paper presents a new approach for accurate Salt Dome detection from 3D seismic data using a novel 3D multidirectional edge detector. It combines gradient maps computed along diagonal directions with those computed in the x, y, and z directions using modified 3D Sobel operators. This overcomes the limitations of existing techniques that consider edges only along the inline, crossline, and time directions. The proposed algorithm first normalizes the data, computes the multidirectional gradients, combines them, and applies thresholding and morphological operations to delineate the Salt Dome boundaries. Results on the Netherlands offshore F3 block data show superior accuracy compared to existing edge-based and texture-based techniques.

1.4.2 Machine Learning and Deep Learning methods

Below, we exhibit some of works that used machine learning and deep learning methode to overcome the probleme :

Haibin Di & Al. (2017) - Seismic Multi-attribute Classification for Salt Boundary Detection:A Comparison [22] The paper compares the performance of various machine learning algorithms including logistic regression, decision trees, random forests, support vector machines, artificial neural networks, and k-means clustering for detecting salt boundaries in 3D seismic data using multi-attribute analysis. The proposed workflow involves attribute selection (using 12 different seismic attributes including a novel saliency attribute), training sample picking, model training with the different algorithms, and volumetric processing to generate salt probability volumes and extract the salt surface. The results indicate similar performance across most algorithms except k-means clustering, with the potential for further accuracy improvement using more training samples and advanced techniques like convolutional neural networks.

M.Karchevskiy & al. (2018) - Automatic salt deposits segmentation: A deep learning approach [23]: Deep learning methods were employed for salt deposits segmentation in seismic images, utilizing a U-Net architecture with SE-ResNeXt-50 encoder and attention gates. Training involved Adam optimizer, cyclic learning rate, and snapshot ensembling techniques. Data augmentation,

including horizontal flips, was used to increase the training dataset. The approach achieved the 27th place in a Kaggle competition, showcasing the effectiveness of the implemented techniques. The code for the solution is available as an open-source project for further exploration.

Y. Babakhin & AL. (2019) - Semi-Supervised Segmentation of Salt Bodies in Seismic Images using an Ensemble of Convolutional Neural Networks [24]: The methodology employed in this study involves an iterative self-training process for semantic segmentation, where pseudo-labels are generated for unlabeled data to improve model accuracy. By utilizing an ensemble of U-ResNet34 and U-ResNeXt50 architectures and incorporating Hypercolumns for segmentation mask prediction, the model achieves top performance in salt body delineation. Leveraging 18000 unlabeled images for self-training, the approach demonstrates state-of-the-art results in this domain, showcasing the effectiveness of the proposed methodology.

A. Milosavljevi (2019) - Identification of Salt Deposits on Seismic Images Using Deep Learning Method for Semantic Segmentation [25]: The research presented a novel CNN architecture for seismic salt-body delineation, originating from U-Net. The implementation utilized Python and the Keras library with TensorFlow backend. Training strategy involved ensemble learning with 5 networks and 5-fold cross-validation to ensure robustness. Results highlighted the FPN model's superior public performance, while the proposed model without dropout excelled in private scoring. A comparison of diverse segmentation models showcased subtle score disparities. The final output was generated using a sigmoid activation function for binary mask prediction, ensuring accurate delineation of salt bodies in seismic images.

M. DERICHE (2022) & al. - Robust Concurrent Detection of Salt Domes and Faults in Seismic Surveys Using an Improved UNet Architecture [26] : The methodology employed in this study focused on leveraging deep learning techniques for seismic interpretation, specifically utilizing pre-trained convolutional neural network models such as UNet and ResNet. The approach involved a hybrid architecture that combined these models to enhance the detection of Salt Domes and faults in seismic data. Evaluation of the methodology included both qualitative and quantitative assessments, with the training conducted on the F3 block seismic dataset. By comparing the performance

of the hybrid architecture with a basic UNet model, the study demonstrated improved fault detection capabilities through the integration of pre-trained networks, highlighting the efficacy of this approach in seismic event detection.

1.4.3 Softwares for Seismic Data Analysis

There are numerous softwares used for seismic processing and analysis where the images which are stored ad datasets in SEGY format for each can be loaded into structures or images that can be loaded in Python , but the used ones by algerian engineers are ProMAX and SeisSpace [27].

ProMAX : is a knowledge-based seismic data processing system that provides interactive analysis tools, effective algorithms, and data management capabilities for maximizing the value of seismic data investments. Key features include 3D visualization, velocity analysis, noise reduction, migration, reservoir characterization workflows, and efficient management of large 3D surveys through parallelization and batch processing. The system is designed to leverage user expertise, optimize processing sequences, increase productivity, and reduce project cycle times for better drilling decisions.

SeisSpace : is seismic processing suite enables easily and efficiently process large volumes of seismic data to develop an insightful and accurate understanding of the subsurface even in the most challenging and complex environments, equiped with intuitive analysis tools, state-of-the-art geophysical algorithms and an optimized parallel infrastructure, the software helps teams get the most out of seismic data, increase productivity and reduce project cycle times (in reality it was built on ProMax, so it is referred as SeisSpace/Promax instead).

1.5 Conclusion

The significance of Salt Domes and the indispensable role of Seismic Surveys in hydrocarbon exploration have been highlighted in this chapter. Salt Domes, as natural reservoir traps, and Seismic Surveys, as imaging tools, are foundational in the quest for valuable oil and gas resources beneath

the Earth's surface,where we addressed the inherent challenges of applying these classical methods. As we move forward into the next chapter, we step into the realm of cutting-edge technology: Deep Learning and Machine Learning. These powerful techniques hold the promise of revolutionizing the way we approach exploration, detection, and extraction processes in the hydrocarbon industry

Chapter 2

Machine Learning and Deep Learning

2.1 Introduction

Machine Learning and Deep Learning have revolutionized Artificial Intelligence by enabling computers to learn from data and make informed decisions. These techniques have found widespread applications in areas such as image recognition, speech processing, and autonomous systems. By leveraging sophisticated algorithms and Neural Networks, Machine Learning and Deep Learning have fundamentally transformed the way computers process information. In this chapter, we delve into their theoretical underpinnings and explore the intricate relationship between these influential methodologies.

2.2 Machine Learning

2.2.1 Definition

Many academics and researchers perceive "Machine Learning" (ML) as an integral component of Artificial Intelligence, although it is often used interchangeably with the broader term. This distinction will be further explored later in this discussion [28]. In the literature, researchers have proposed numerous definitions for Machine Learning. One notable definition (Koza et al., 1996) offers insights into the nature of this field and its underlying principles:

"ML describes a set of methods commonly used to solve a variety of real-world problems with the help of computer systems, which can learn to solve a problem instead of being explicitly programmed to do so."

2.2.2 Machine Learning and Artificial Intelligence

Artificial Intelligence is defined as systems that display intelligent behaviour by analysing their environment and taking actions – with some degree of autonomy – to achieve specific goals [29].

Artificial intelligence (AI) is frequently associated with cutting-edge technologies, and in recent years, it has gained significant momentum largely due to advancements in Machine Learning. Consequently, the terms AI and Machine Learning are often used interchangeably. However, it is important to note that while Machine Learning is a part of AI, the two terms are not synonymous (see Figure 2.1).

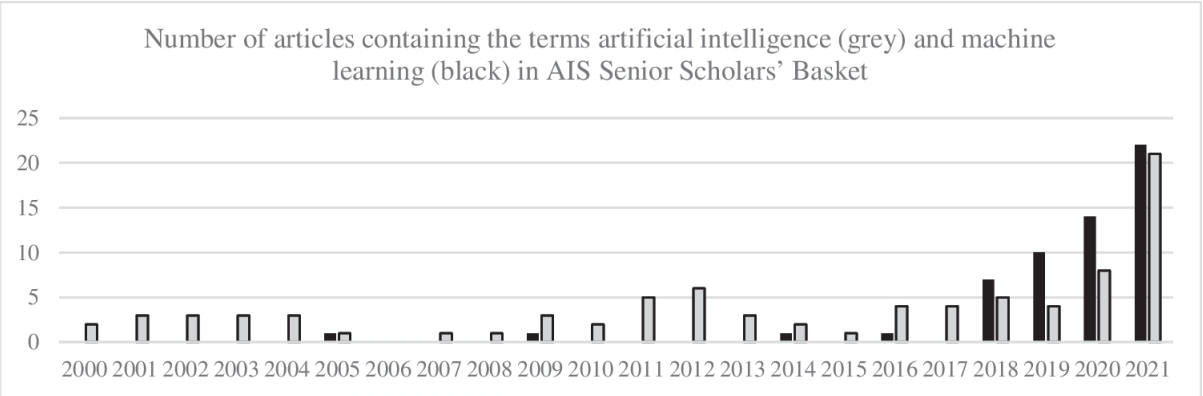


Figure 2.1: Appearance of the terms "artifcial intelligence" and "Machine Learning" and in AIS Senior Scholars [28].

Machine Learning is a technology that enables computers to learn directly from data and examples, distinguishing it from traditional programming methods that rely on predefined rules. In Machine Learning, systems are provided with a specific task and a substantial amount of data to learn from, either as examples or patterns. Through this process, the system learns how to achieve the desired output by analyzing and extracting insights from the given data. Machine Learning can be considered as a form of Narrow Artificial Intelligence (AI), as it focuses on training intelligent systems to learn

specific functions based on provided data.

On the other hand, AI encompasses a broader spectrum of technologies, ranging from traditional AI, such as Good Old Fashion AI (GOFAI), to more advanced connectionist architectures like Deep Learning. Machine Learning is a subfield of AI that specifically deals with learning algorithms and their application to data (see Figure 2.2). It encompasses various techniques (we will demonstrate them later) [29].

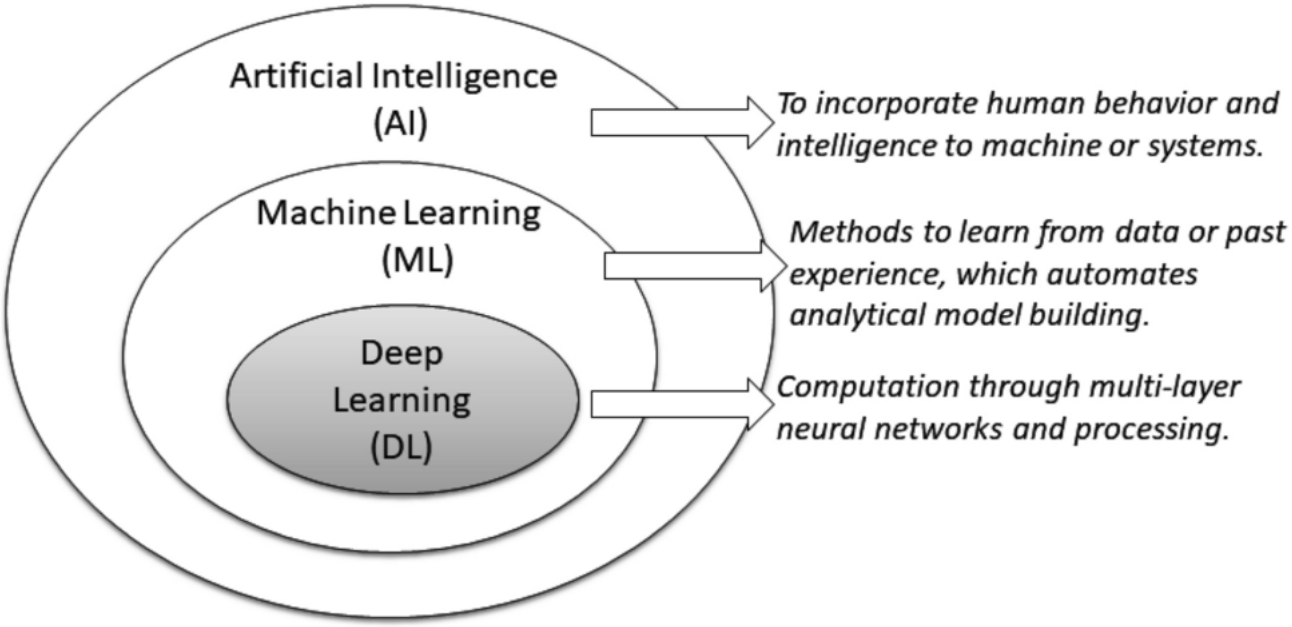


Figure 2.2: An illustration depicting the relationship between Deep Learning (DL), Machine Learning (ML), and Artificial Intelligence (AI) [30].

2.2.3 Types of Machine Learning

Machine Learning algorithms can be broadly categorized into four main types: Supervised Learning, unsupervised learning, Semi-Supervised Learning, and Reinforcement Learning as shown in Figure 2.3 [31]. Each of these techniques has its own scope and applicability in solving real-world problems [32]:

- **Supervised** : Supervised learning is a core aspect of Machine Learning, wherein the objective is to acquire knowledge about a function that can map input data to corresponding output values, leveraging labeled training instances [33]. This approach involves utilizing a set of input-output pairs to deduce the underlying function. By adopting a

task-oriented perspective [34], Supervised Learning focuses on achieving specific goals given a defined input space. The most prevalent supervised tasks encompass classification, which involves categorizing data into distinct classes, and regression, which entails fitting data to a continuous output. For instance, an exemplification of Supervised Learning is the prediction of sentiment or classification labels for textual data like tweets or product reviews.

- **Unsupervised** : Unsupervised learning, in contrast to Supervised Learning, involves analyzing unlabeled datasets without the need for human intervention, making it a data-centric process [33]. This approach is commonly employed to extract generative features, identify significant trends and structures, uncover groupings in results, and facilitate exploratory analysis. Various tasks fall under the domain of unsupervised learning, including clustering, density estimation, feature learning, dimensionality reduction, finding association rules, and anomaly detection, among others. These techniques enable the algorithm to uncover hidden patterns and structures within the data without prior knowledge of the desired output, allowing for a more holistic understanding of the underlying information.

- **Semi-supervised** : Semi-supervised learning can be described as a hybrid approach that combines elements of both supervised and unsupervised methods, leveraging both labeled and unlabeled data [33, 34]. It lies between the realms of "learning without supervision" and "learning with supervision". In many real-world scenarios, labeled data may be scarce while unlabeled data are abundant, making Semi-Supervised Learning particularly valuable [31]. The primary objective of a Semi-Supervised Learning model is to yield better prediction outcomes than what can be achieved using solely labeled data. This approach finds applications in various domains, including machine translation, fraud detection, data labeling, and text classification. By harnessing the advantages of both labeled and unlabeled data, Semi-Supervised Learning offers a powerful framework for addressing challenges in these areas.

- **Reinforcement** : Reinforcement learning is a Machine Learning algorithm that empowers software agents and machines to autonomously evaluate optimal behavior within a specific context or environment,

with the aim of improving efficiency [35]. It follows an environment-driven approach, where the learning process is driven by the interactions between the agent and its environment. This type of learning relies on the concept of rewards and penalties, with the ultimate goal of using insights gained from environmental feedback to take actions that maximize rewards or minimize risks [31]. Reinforcement learning serves as a powerful tool for training AI models that can enhance automation and optimize operational efficiency in complex systems such as robotics, autonomous driving, manufacturing, and supply chain logistics. However, it may not be the preferred choice for addressing basic or straightforward problems, as its full potential is best realized in more intricate and challenging scenarios.

2.2.4 Machine Learning Tasks and Algorithms

In this part, we delve more to the most important tasks in Machine Learning and the most used algorithms for each task [32]:

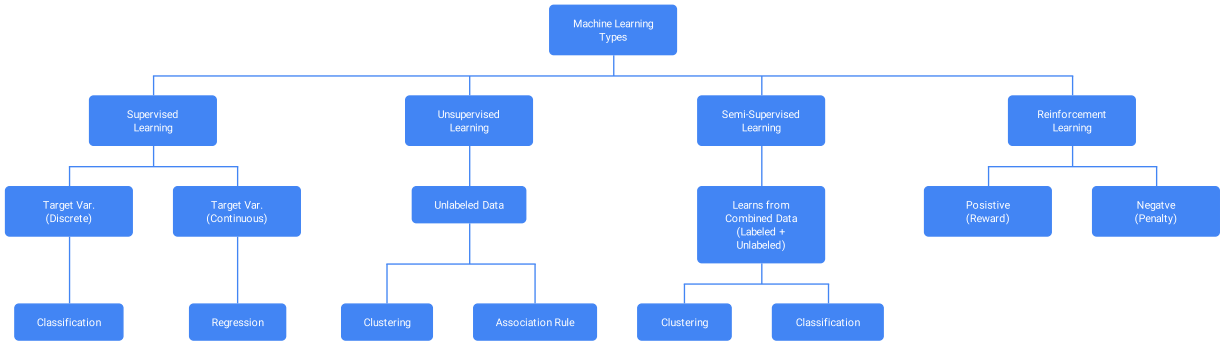


Figure 2.3: Various types of Machine Learning techniques [32].

I Classifcation Analysis

Classification is a Supervised Learning technique in Machine Learning that involves predicting a class label for a given input example [33]. It mathematically maps a function (f) from input variables (X) to output variables (Y), representing target labels or categories. Classification can be applied to structured or unstructured data to predict the class of data points. An example of classification is spam detection in email service providers, where the problem involves classifying emails as "spam" or "not spam".the most common algorithms are :

- **Naive Bayes (NB)** : The naive Bayes algorithm, assuming feature independence based on Bayes' theorem, is effective for both binary and multi-class categorization tasks [36]. It finds applications in various real-world scenarios, including document classification and spam filtering.

- **Logistic Regression (LR)** : Logistic Regression (LR) is a widely used statistical model in Machine Learning for classification tasks [37]. It estimates probabilities using a logistic function (sigmoid function) and performs well when data can be linearly separated, although it can overfit high-dimensional datasets. Regularization techniques (L1 and L2) can mitigate overfitting [38]. The assumption of linearity between dependent and independent variables is a limitation of Logistic Regression.

- **K-Nearest Neighbors (KNN)** : K-Nearest Neighbors (KNN) is an instance-based learning algorithm that stores training data instances in an n-dimensional space and classifies new data points based on similarity measures [39]. It relies on a simple majority vote of the k nearest neighbors and is robust to noisy training data, with accuracy dependent on data quality. Choosing the optimal number of neighbors is a key consideration, and KNN can be used for both classification and regression tasks [32].

- **Support Vector Machine (SVM)** : Support Vector Machine (SVM) is a commonly used technique in Machine Learning for classification, regression, and other tasks [40]. It constructs hyperplanes in high- or infinite-dimensional space, aiming to achieve strong separation between classes by maximizing the margin from the nearest training data points. SVM is particularly effective in high-dimensional spaces and can employ different mathematical functions, known as kernels, such as linear, polynomial, radial basis function (RBF), and sigmoid [38]. However, SVM may not perform well when dealing with noisy datasets that contain overlapping target classes[32].

- **Decision Tree (DT)** : Decision Tree (DT) is a widely used non-parametric Supervised Learning method that can be applied to both classification and regression tasks . DT classifies instances by traversing the tree

from the root to leaf nodes, evaluating attributes at each node and moving down the corresponding tree branch based on the attribute value. Popular splitting criteria include "gini" for Gini impurity and "entropy" for information gain [38].

- **Random Forest (RF)** : Random Forest (Figure 2.4) classifier is a widely recognized ensemble classification technique used in Machine Learning and data science across various applications [41]. This method employs parallel classifiers on different sub-samples of the dataset, combining their predictions through majority voting or averaging [38]. By minimizing overfitting and enhancing prediction accuracy and control, Random Forest outperforms single decision tree models [42]. It constructs a series of decision trees with controlled variation by combining bootstrap aggregation (bagging) and random feature selection [43][44]. Random Forest is adaptable to both classification and regression problems and performs well with categorical and continuous values.

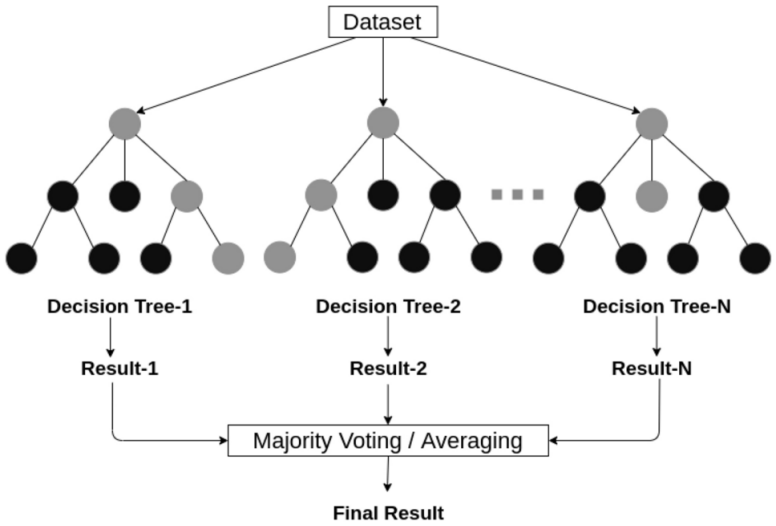


Figure 2.4: An example of a random forest structure considering multiple decision trees [32].

II Regression Analysis

Regression analysis is a Machine Learning method that predicts a continuous outcome variable (y) based on one or more predictor variables (x). Unlike classification, which predicts class labels, regression focuses on estimating continuous quantities. Regression models find applications in various fields, such as financial forecasting, cost estimation, trend analysis, and drug re-

sponse modeling. Common types of regression algorithms include linear regression, polynomial regression, lasso regression, and ridge regression [33]. The most common algorithms are :

- **Simple and Multiple Linear Regression** : Linear Regression is a widely used and popular Machine Learning and regression technique. It models the relationship between a continuous dependent variable (Y) and one or more independent variables (X) using a straight line (regression line) that provides the best fit [33]. The independent variables can be continuous or discrete, while the dependent variable is continuous.

- **Polynomial Regression** : Polynomial Regression is a regression technique that extends linear regression by using polynomial functions to model the relationship between the dependent variable and the independent variables. It allows for capturing non-linear patterns in the data by fitting higher-degree polynomial curves to the data points [33]. This flexibility makes polynomial regression useful when the relationship between variables is not strictly linear.

III Cluster Analysis

Cluster Analysis is a technique used in unsupervised Machine Learning to identify groups or clusters within a dataset based on the similarity of data points. The goal is to group similar data points together while maximizing the dissimilarity between different clusters. Cluster Analysis has various applications such as customer segmentation, anomaly detection, and image segmentation [33]. K-Means is a widely used clustering algorithm that partitions data into K distinct clusters based on their proximity to centroid points. It is iteratively performed by assigning data points to their closest cluster center and updating the centroids. While efficient and scalable, it requires specifying the number of clusters in advance and assumes spherical and similar-sized clusters[45].

IV Dimensionality Reduction

Dimensionality Reduction is a technique used to reduce the number of input features in a dataset while preserving its essential information. It is beneficial for several reasons, including reducing computational complexity, removing redundant or irrelevant features, and visualizing high-dimensional

data . the most Popular dimensionality reduction method is known as Principal Component Analysis (PCA) [32].

V Association Rule Learning

Association Rule Learning is a data mining technique used to discover interesting relationships or associations between items in large datasets. It involves identifying frequent itemsets, which are sets of items that often occur together, and generating association rules based on these itemsets. The support and confidence measures are used to evaluate the strength and significance of the rules, we mentioned as most popular algorithms : A-PRIORI and FP-GROWTH [32].

VI Neural Network

Deep Learning is a subset of Artificial Neural Networks (ANN) that involves multiple layers of processing units to learn from data. It offers improved performance, especially with large datasets, due to its computational architecture and representation learning capabilities. Deep Learning's advantages over traditional Machine Learning methods are evident in various domains, as shown in its performance graph when considering increasing data amounts. However, performance may vary based on data characteristics and experimental setups [33, 34, 46].

2.3 Deep Learning

The term "Deep Learning" made its initial appearance in the field of Machine Learning during a conference by Dechter in 1986. It was further associated with Artificial Neural Networks in a publication by Aizenberg et al. in 2000 [20]. The introduction of this term marked a significant milestone in the advancement of both fields, paving the way for the development and exploration of Deep Learning algorithms and architectures [47].

Deep Learning, a subset of Artificial Neural Networks (ANNs) (see Figure 2.5), has emerged as a powerful approach in the field of Machine Learning (ML). Inspired by the information processing mechanisms in biological systems, Artificial Neural Networks (ANNs) are flexible structures inspired by biological systems that consist of interconnected processing units called

neurons. ANNs can be modified for various Machine Learning (ML) contexts and are characterized by the transmission of signals between neurons, which are weighted and adjusted during the learning process. Neurons are organized into networks with input, hidden, and output layers, and the connections between them allow for non-linear mapping.

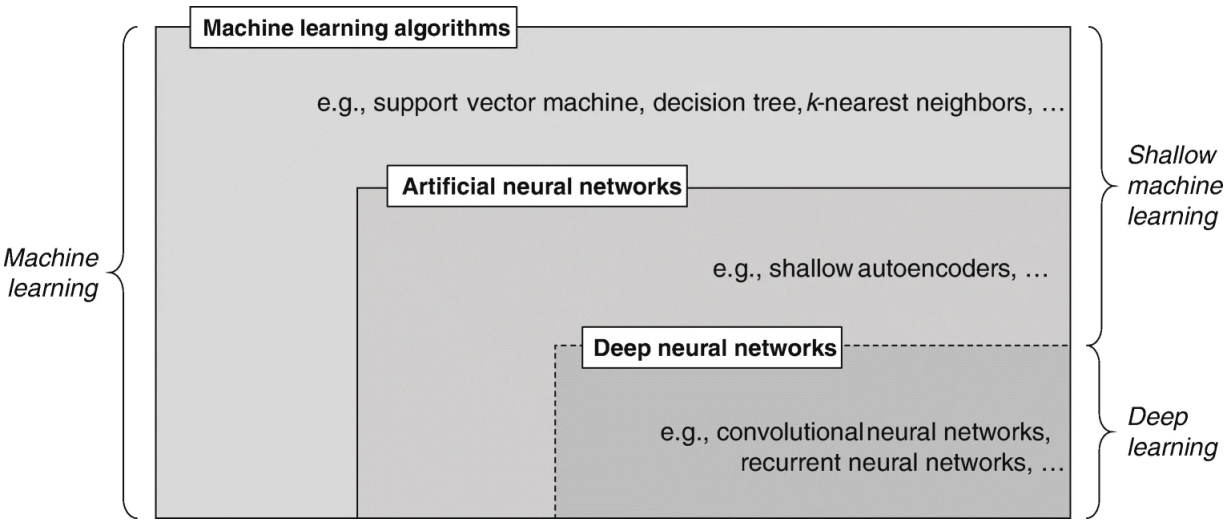


Figure 2.5: Venn diagram of machine Machine Learning algorithms learning concepts and classes (inspired by Goodfellow et al. 2016, p. 9) [48].

Deep Neural Networks, a type of ANN, have multiple hidden layers and advanced neurons that enable them to automatically learn representations from raw input data, known as Deep Learning (see Figure 2.6). Deep Learning excels in processing large and high-dimensional data like text, images, and audio. However, for low-dimensional data and limited training data, shallow ML algorithms can still produce superior and more interpretable results. While Deep Learning can achieve superhuman performance in certain tasks, it falls short in solving problems requiring strong AI capabilities [48].

2.3.1 Deep Learning's Developement

The development of Neural Networks can be categorized into three generations (Figure 2.7) : the perceptron, the Deep Neural Network, and the Liquid State Machine [49].

1. **The Perceptron** : proposed by Frank Rosenblatt in 1957, represented the first generation of Neural Networks. It was a single-layer network

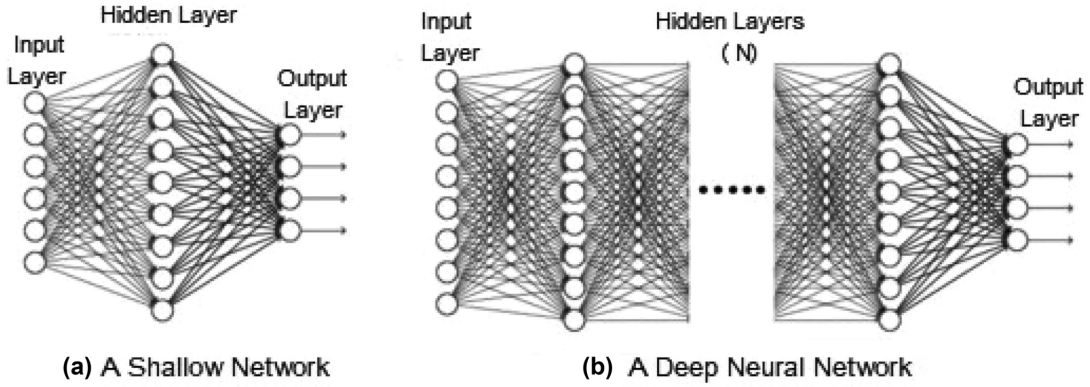


Figure 2.6: A general architecture of a a shallow network with one hidden layer and b a deep Neural Network with multiple hidden layers [30].

capable of binary classification tasks. Rosenblatt's work laid the foundation for the study of Artificial Neural Networks and their learning capabilities. However, perceptrons had limitations in their ability to handle complex patterns and learn non-linear relationships, leading to a decline in interest in Neural Networks during the 1960s and 1970s [50].

2. **The Deep Neural Networks** : emerged in the 1980s and gained significant attention in the early 2000s [47]. Pioneering work by researchers such as Geoffrey Hinton [51], Yann LeCun [52], and Yoshua Bengio [53] contributed to the resurgence of interest in Deep Learning. Hinton's research on backpropagation algorithms and the development of multilayer Neural Networks enabled the training of deep architectures. LeCun's introduction of Convolutional Neural Networks (CNNs) revolutionized image recognition tasks, while Bengio's work on Deep Learning algorithms and architectures established the theoretical foundations of Deep Neural Networks. The success of deep Neural Networks in various domains, including computer vision and natural language processing, propelled the second generation of Neural Networks to the forefront of research and applications.

3. **Liquid State Machines** : emerged in the early 2000s. Proposed by Maass et al. in 2002, Liquid State Machines (see Figure 2.8) aimed to mimic the information processing capabilities of the brain's neural circuits. These networks were designed to process temporal data and demonstrated promising results in tasks such as speech recogni-

tion and prediction. Liquid state machines introduced a new perspective on Neural Network architecture and dynamics, highlighting the importance of recurrent connections and dynamic computations., it's considered as a Spiking Neural Network (SNN) model [54].

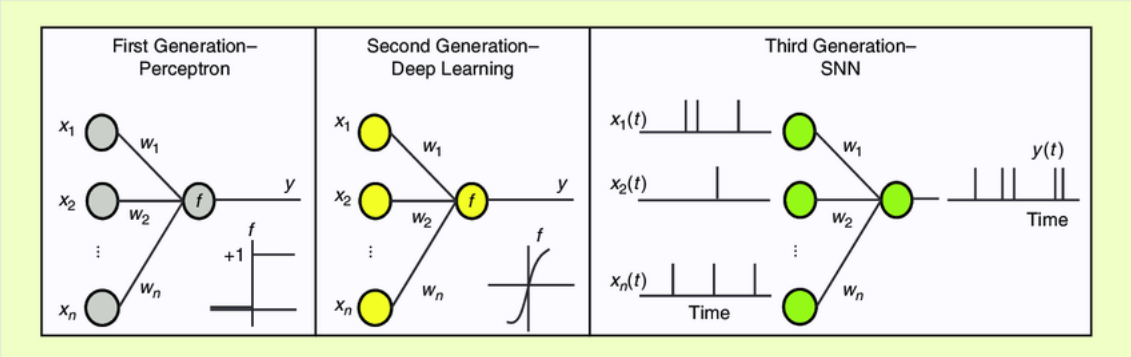


Figure 2.7: Three generations of Neural Network models.[49].

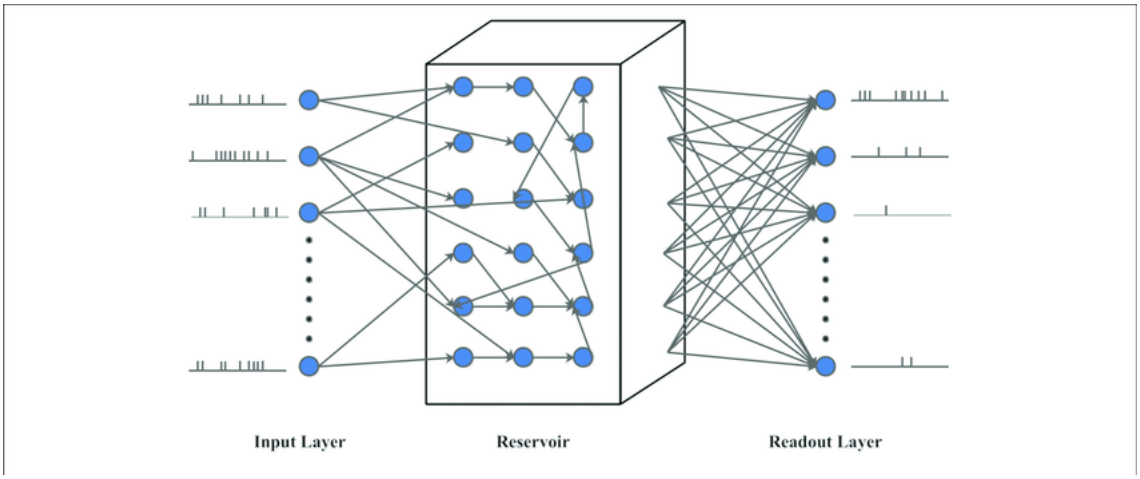


Figure 2.8: The structure of Liquid State Machine LSM[54].

2.3.2 Deep Learning Techniques and Applications

DL models can be categorized into four main types: deep supervised, unsupervised, Reinforcement Learning, and hybrid models. Figure 1 visually represents these categories and provides examples of models within each category. Brief descriptions of each category are provided below. Furthermore, Figure 2.9 presents the commonly used techniques for each category [55]

I Deep Supervised Learning

Deep Supervised Learning models play a crucial role in the field of Deep Learning. They require labeled training data to train effectively. These models employ a loss function to assess their performance and iteratively update the model's weights to minimize the error until it reaches an acceptable level [55].

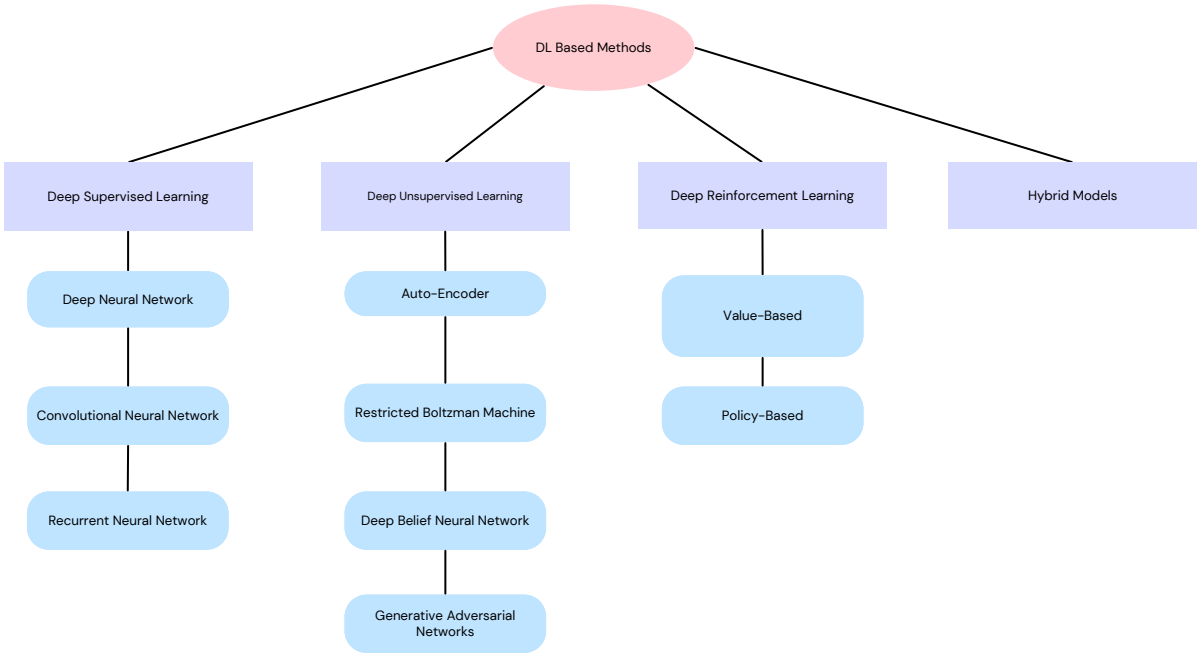


Figure 2.9: DL Based Methods[55].

Among this models ,we have Convolutional Neural Networks (CNN) :

I.1 Convulutional Neural Networks

CNNs (see Figure 2.10) are widely recognized and extensively employed in Deep Learning, offering several advantages over previous algorithms. One notable benefit is their ability to autonomously identify relevant features without human supervision. CNNs have found applications in various domains, including computer vision, speech processing, and face recognition. The structure of CNNs is inspired by the Neural Networks present in human and animal brains, particularly the visual cortex.

[56] states that CNNs leverage shared weights and local connections to exploit the structures of input data, such as image signals. This approach re-

duces the number of parameters, simplifies training, and improves network speed. The concept is analogous to how cells in the visual cortex only sense small regions of a scene rather than the entire scene, extracting local correlations like local filters over the input.

A commonly used CNN architecture, similar to a Multi-Layer Perceptron (MLP), comprises Convolution layers followed by sub-sampling (Pooling) layers, with Fully Connected (FC) layers at the end.

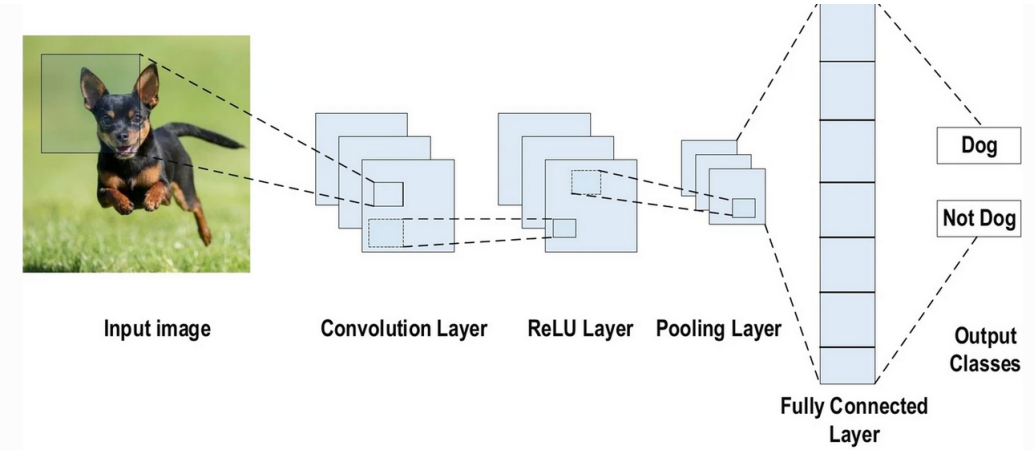


Figure 2.10: CNN model [56].

In a CNN model, the input data is organized in three dimensions: height, width, and depth (channel number) or $n{*}m{*}r$ where $n{=}m$. For example, in an RGB image, the depth r is equal to three. Each Convolutional Layer consists of multiple kernels k (filters) with the same dimensions as the input image where they are of size $n{*}n{*}q$, but with a smaller depth where $n{<}m$ and $q \leq r$. These kernels are responsible for generating feature maps by convolving with the input using shared parameters (bias b^k and weight x^k) (see Figure 2.11).

The Convolution Layer performs a dot product between the input and the weights, followed by the application of a nonlinearity or activation function.

$$h^k = f(w^k \cdot x + b^k) \tag{1}$$

where h^k are the features maps got by for each convulotion operation of size $m - n - 1$.

Subsequently, the sub-sampling layers downsample each feature map by applying a pooling function (such as Max or Average Pooling) to adjacent areas of a specific size of length p, that is, of size $p * p$. This reduces the network parameters, speeds up training, and helps address overfitting (see Figure 2.12).

The Fully Connected (FC) (see Figure 2.13) layers receive the extracted mid- and low-level features and generate high-level abstractions, similar to a Conventional Neural Network. The last-stage layers produce classification scores using techniques like softmax. These scores represent the probabilities of different classes for a given input instance, the whole process is shown in figure.

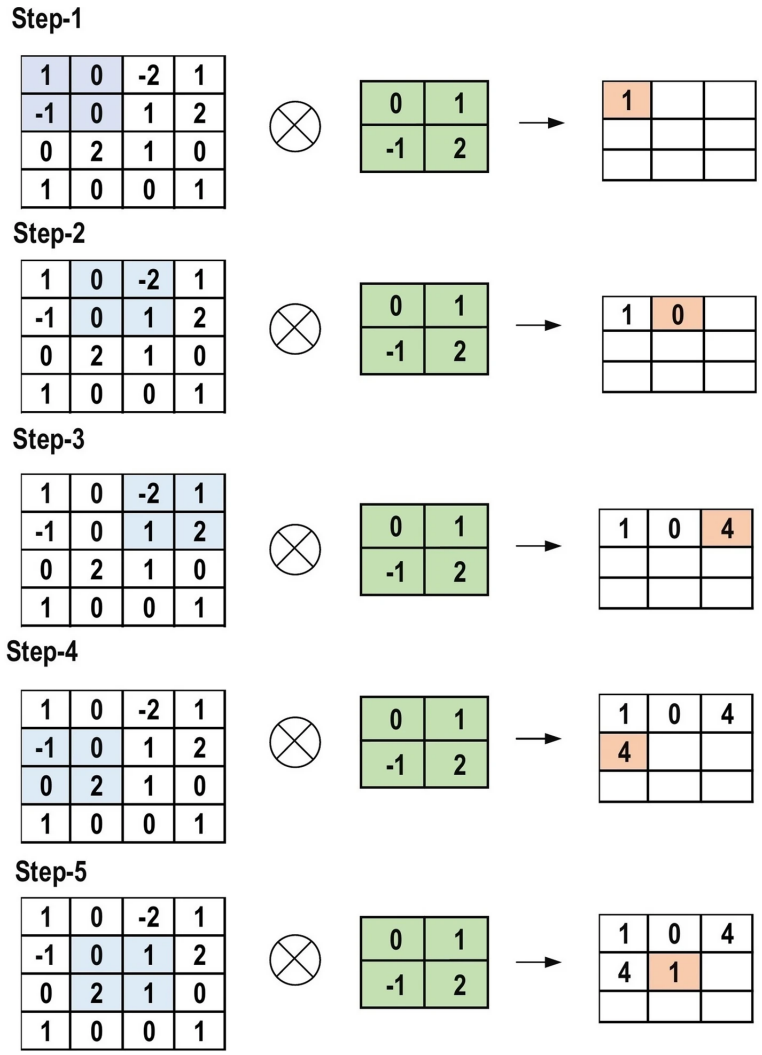


Figure 2.11: The process of convolution [56].

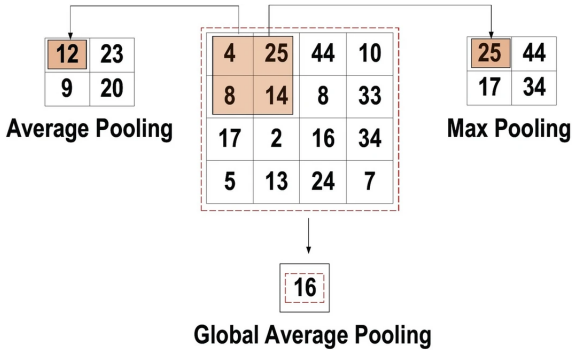


Figure 2.12: The process of pooling [56].

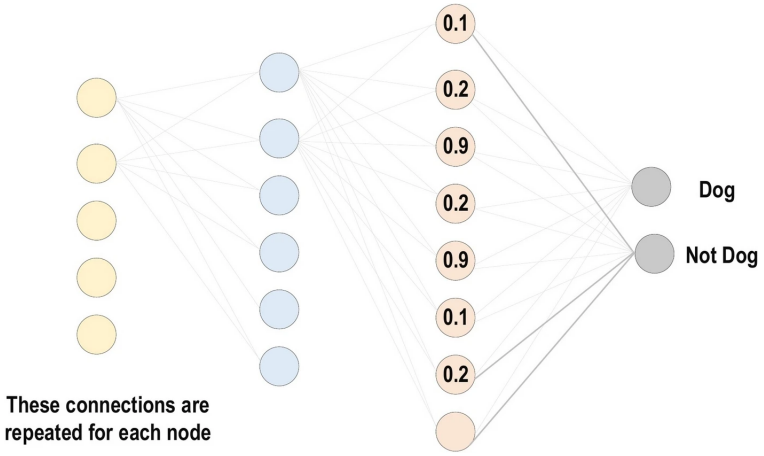


Figure 2.13: Fully connected layer [56].

Common Convolution networks

There are numerous CNN's which differ in their architecture, model architecture is an important key in enhancing the performance of different applications. Various modifications have been achieved in CNN architecture from 1989 until today, so we present below some of the important and popular models:

a. **Visual Geometry Group** : VGG (Visual Geometry Group) (see Figure 2.14) is a convolutional Neural Network (CNN) architecture developed by the University of Oxford. It is known for its deep network structures with multiple stacked Convolutional Layers. VGG models, such as VGG16 and VGG19, have achieved state-of-the-art performance in image classification tasks. The architecture features small 3x3 filters to capture intricate image details. However, VGG's main drawback is its high computational cost and memory requirements.[57]

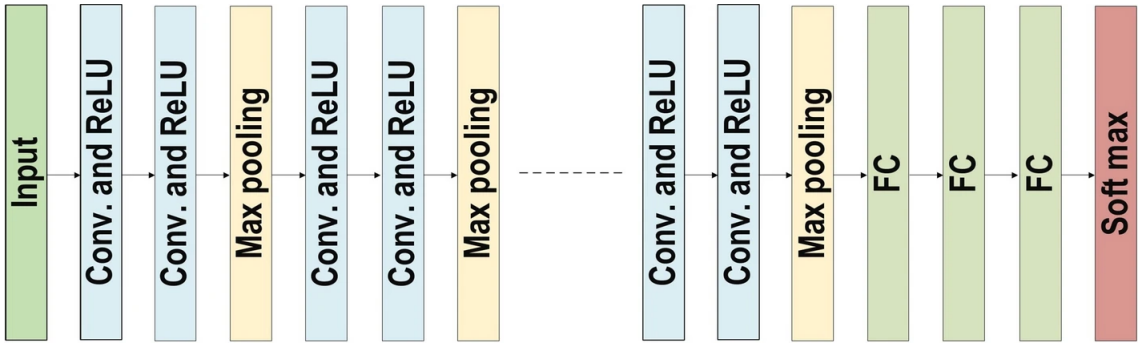


Figure 2.14: VGG model [56].

b. **ResNet** : ResNet (Residual Network) (see Figure 2.15) is a popular Convolutional Neural Network (CNN) architecture that addresses the vanishing gradient problem in deep networks. It introduces the concept of residual connections, which allow the network to skip over layers and directly propagate information from earlier layers to later ones. This enables the training of much deeper networks (e.g., ResNet-50, ResNet-101) [58] while maintaining good accuracy. ResNet architectures have demonstrated outstanding performance in various computer vision tasks, including image classification, object detection, and segmentation[58, 59, 60]. The residual connections help alleviate the degradation problem and enable the successful training of extremely deep Neural Networks with improved accuracy and training efficiency.

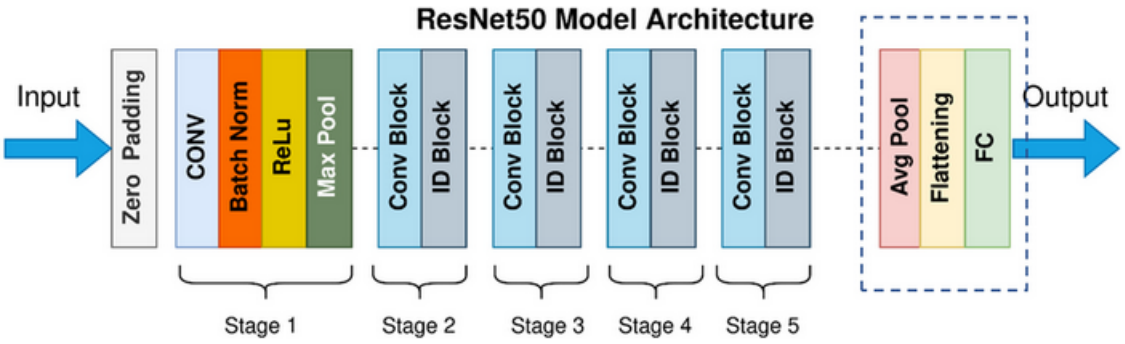


Figure 2.15: Resnet model [56].

c. **Unet** : UNet (see Figure 2.16) is a U-shaped Convolutional Neural Network (CNN) architecture designed for semantic segmentation tasks. It consists of an encoder path and a decoder path to capture and recover spatial information. Skip connections enable the fusion of low-level and high-level features for precise object boundary localization. UNet has gained popularity in medical imaging. Its effectiveness, simplicity, and ability to handle limited training data make it widely adopted. The encoder path extracts hierarchical features through Convolutional and Pooling Layers. The decoder path uses upsampling and concatenation operations to generate pixel-wise segmentation maps. UNet's skip connections facilitate accurate delineation of objects or regions. Overall, UNet has made significant contributions to the field of computer vision, particularly in Semantic Segmentation tasks [1].

I.2 Transformers Neural Networks

Originally designed for machine translation, the Transformer model [61] has found utility in a range of natural language processing tasks (see Figure 2.17). It combines a CNN with attention to address parallelization challenges and enhance computational efficiency. The Transformer follows a sequence-to-sequence architecture, comprising six encoders and a decoder. Its key feature is self-attention, a mechanism that enables the model to focus on different parts of the input sequence during processing. By leveraging self-attention, the Transformer model has demonstrated impressive performance in various NLP applications and recently in image processing with the inception of Vision Transformer (ViT) (see Figure 2.18) Model especially

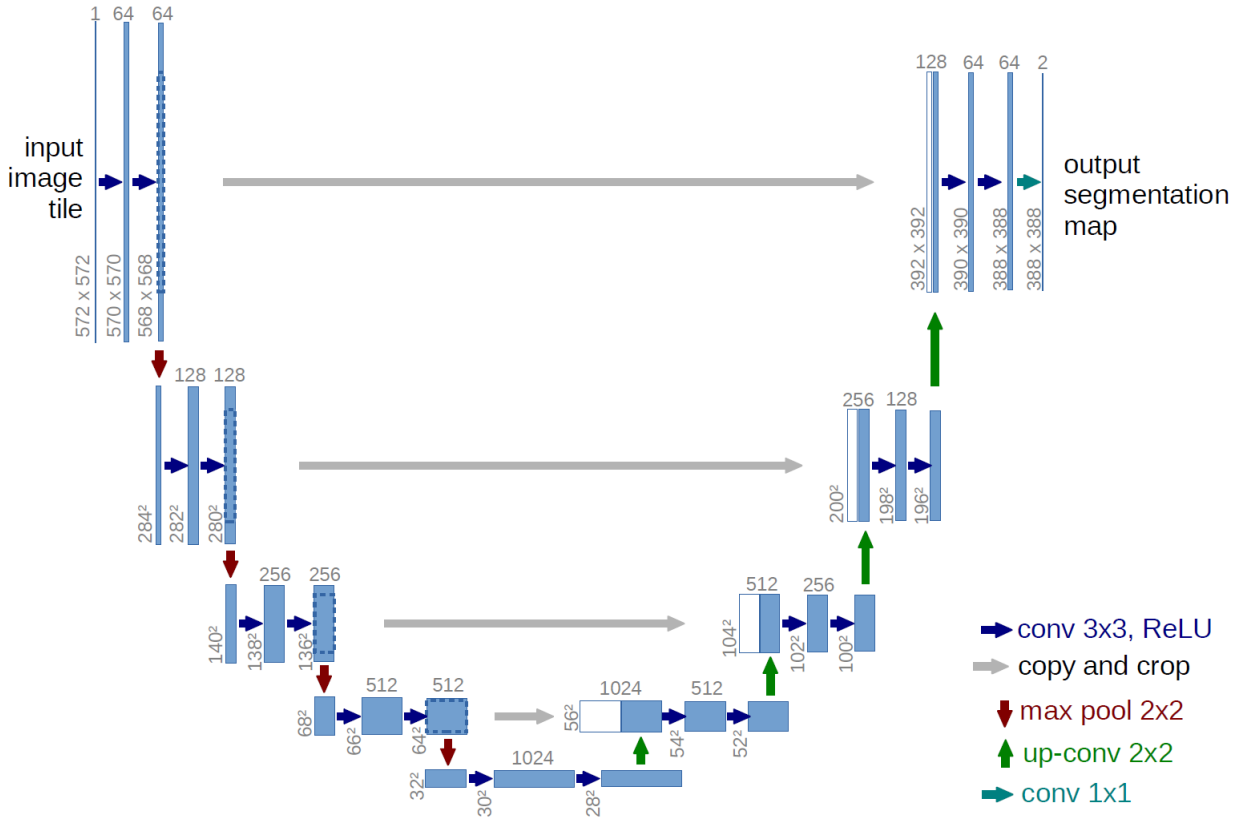


Figure 2.16: 2D Unet model's [1] example with 3 layers. Each one of the blue boxes corresponds to a multichannel feature map. The number of channels is indicated on top of each box. The spatial size is provided at the lower left edge of each box. The white boxes represent copied feature maps. The arrows represent different operations.

[62].

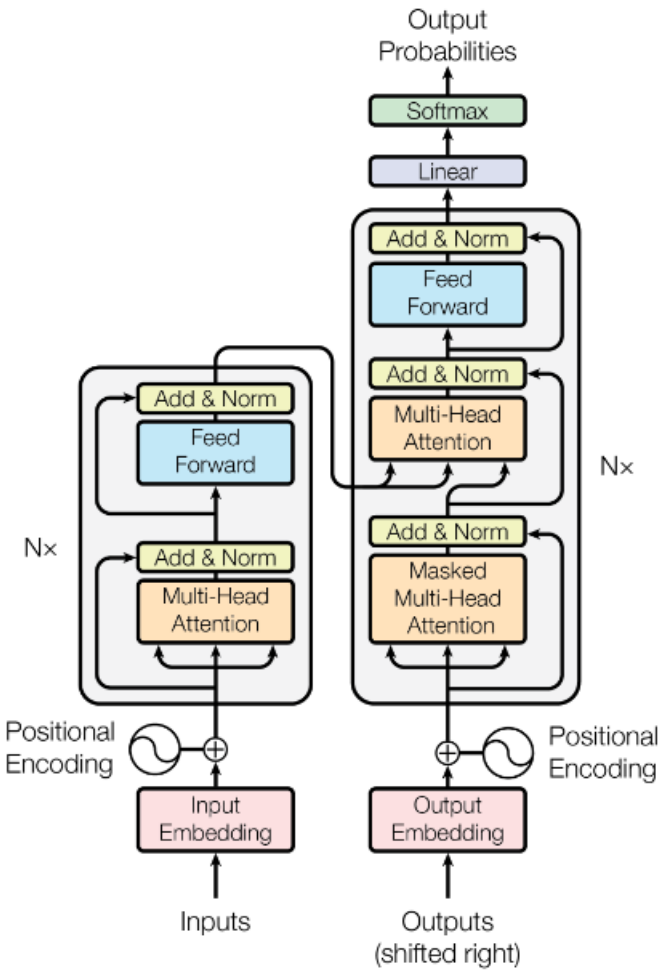


Figure 2.17: Transformer model [61].

The architecture of Transformers is composed of [63]:

Encoder

Within the Transformer architecture, each encoder is composed of two key components: a feed-forward Neural Network and a self-attention layer. The encoder's input undergoes processing through the self-attention layer before further encoding. This self-attention mechanism enables the encoder to take into account surrounding words within the input text, enhancing its understanding and representation of individual words.

Decoder

Similar to the encoder, the decoder in the Transformer architecture consists of the feed-forward Neural Network and self-attention layers. However, the decoder also incorporates an additional attention layer. This attention mechanism empowers the decoder to selectively emphasize important elements within the input text, enabling it to generate more accurate and contextually relevant outputs. By integrating the attention layer, the decoder enhances its ability to effectively process and interpret the information provided by the encoder.

Self-Attention

In the Transformer's self-attention mechanism, each word within the encoder independently follows its designated path. However, in the self-attention layer, these paths become interdependent, enabling the model to capture the relationships among different words. Unlike the feed-forward layer, which lacks these interdependencies, the self-attention layer allows for parallel execution of multiple paths as the data flows through. This self-attention mechanism plays a crucial role in capturing the intricate relationships between regions and words, facilitating the model's understanding of contextual dependencies within the input.

Multi-Head Attention

The multi-head attention technique in the Transformer design employs h different learned projections to linearly project queries, keys, and values. These h projections are processed in parallel through the attention mechanism, resulting in h outputs. These outputs are then concatenated and projected once more to yield the final result. The purpose of multi-head attention is to enable the attention function to extract information from multiple representation subspaces, which is not achievable with a single attention head. This approach enhances the model's ability to capture diverse and nuanced relationships within the input data, contributing to its overall effectiveness and performance.

Positional Encoding

In the Transformer's encoding process, positional encoding plays a vital role in representing the position of each word. Encoding the position of each word is crucial as it directly impacts its translation. By incorporating positional encoding, the model can differentiate between words based on their position in the input sequence, allowing for accurate and contextually relevant translations. This mechanism enables the Transformer to effectively capture the positional information and leverage it during the encoding process, contributing to the overall quality of the translation .

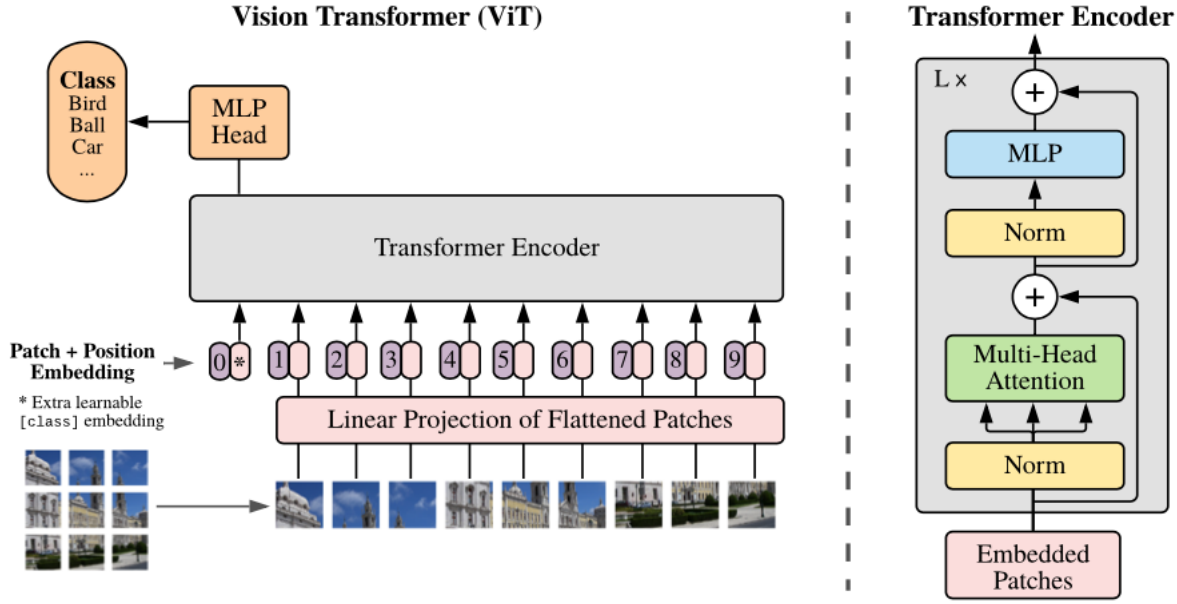


Figure 2.18: ViT model [62].The input image is first split in fixed-size patches, which are linearly embedded, then position embeddings are added and fed to a Transformer encoder. In order to perform classification, an extra learnable "classification token" (also known as [CLS] or [CLASS]) is added to the sequence.

II Deep Unsupervised Learning

Deep unsupervised learning has emerged as a prominent branch of Deep Learning models, garnering considerable attention. These models have proven to be highly effective in scenarios where only a limited number of unlabeled samples are available for training [55]. There are several types of Deep Unsupervised models, including Autoencoders, restricted Boltzmann Machines, and Generative Adversarial Networks. Notably, the Deep Belief

Network encompasses the Variational Autoencoder (VAE) (see Figure 2.19), which is the focus of our further explanation. VAEs are a crucial component of the deep belief network and have been extensively utilized for tasks involving unsupervised learning and generative modeling.

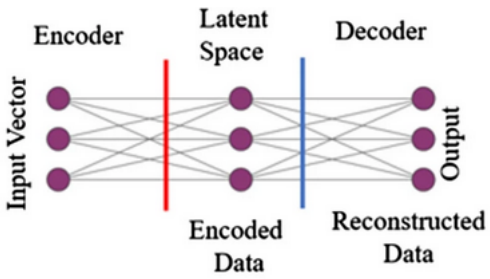


Figure 2.19: AutoEncoder model [55].

II.1 Variational Autoencoder

A variational autoencoder (VAE) is a Neural Network designed to learn a condensed representation of input data by encoding it into a lower-dimensional latent space and subsequently reconstructing the original data from this compressed representation. Unlike traditional autoencoders, VAEs employ a probabilistic approach during the encoding and decoding process. This probabilistic framework allows VAEs to capture the inherent structure and variability in the data, enabling them to generate new data samples from the learned latent space. VAEs find utility in diverse tasks, including anomaly detection, data compression, as well as image and text generation. The work by Kingma and Welling in 2013 introduced the concept of VAEs and their probabilistic nature [64].

VaE Architecture

Autoencoder Neural Networks consist of an encoder and a decoder. The encoder learns to map the input sequence to a latent space, while the decoder reconstructs the input sequence. Vanilla autoencoders lack regularity in the latent space, making it hard to interpolate for missing data points.

Variational autoencoders (VAEs) (see Figure 2.20) address this by introducing KL divergence regularization. The VAE encoder (E) maps data to vectors

representing the mean and standard deviation of a Gaussian distribution. Minimizing the Lprior loss encourages the encoder to compress the input into a Gaussian distribution. This regularization improves reconstruction in the decoder, which samples from a continuous distribution.

The decoder loss is computed using the distance between the reconstructed sequence $(\hat{x})$ and the original input (x). Both Lprior and Lreconstruction losses are backpropagated to train VAE parameters. This regularization and reconstruction loss allow VAEs to learn a structured latent space, facilitating interpolation and generating new data samples [65].

$$L_{\text{prior}} = \text{DKL}\left(E(x)||N(0,1)\right) \tag{2}$$

$$L_{\text{reconstruction}} = L_{\text{prior}} + |\hat{x} - x|_2 \tag{3}$$

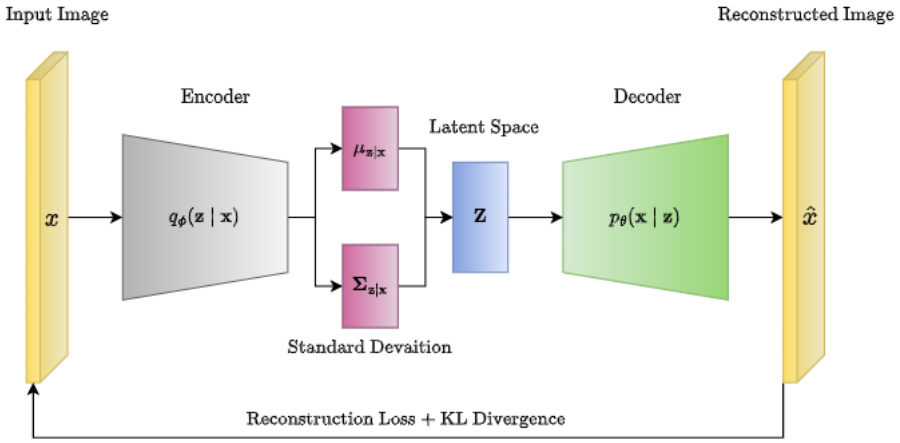


Figure 2.20: Vae model [65].

III Deep Reinforcement Learning

Deep Reinforcement Learning combines Deep Learning techniques with Reinforcement Learning algorithms to enable agents to learn and make decisions in complex environments. It leverages deep Neural Networks to approximate the value or policy functions, allowing agents to handle high-dimensional input spaces. By using deep Neural Networks as function approximators, Deep Reinforcement Learning can learn directly from raw sen-

sory inputs, such as images or text. This approach has achieved significant success in various domains, including robotics, game playing, and autonomous driving. However, training Deep Reinforcement Learning models can be challenging due to issues such as sample efficiency and instability[55].

IV Hybrid Deep Learning

Deep Learning models have strengths and weaknesses in terms of hyperparameter tuning and data exploration. This weakness can limit their effectiveness in different applications. To address these limitations, hybrid Deep Learning models have been proposed, combining different models to overcome specific shortcomings. Convolutional Neural Networks (CNNs) and VaEs are popular and highly applicable in various studies, demonstrating significant potential compared to other Deep Learning models[55].

2.4 Conclusion

In summary, this chapter provided a comprehensive overview of Machine Learning, Deep Learning, and Artificial Intelligence. We explored different algorithms and their applications across various domains. Machine Learning algorithms, including Supervised, Unsupervised, and Reinforcement Learning, were discussed in terms of their strengths and limitations.

Deep Learning emerged as a powerful subset of Machine Learning, capable of learning intricate patterns from large datasets. We highlighted its applications in computer vision, natural language processing, and robotics.

In the next chapter, we will build upon the theoretical concepts covered in this chapter to introduce our methodology and work. By leveraging the knowledge gained, we will outline our approach, applying Machine Learning and Deep Learning techniques to address a specific problem or task. Our aim is to contribute to the advancement of the field and provide innovative solutions by incorporating the foundational understanding established in this chapter.

Chapter 3

Proposed Method

3.1 Introduction

In this chapter, we present our novel method for Semantic Segmentation in seismic data images, building upon the state-of-the-art Deep Learning techniques explored in the previous chapter. Focusing on the specific domain of Seismic Survey and Salt Deposits, our proposed method addresses the limitations of existing approaches. By leveraging the power of Deep Learning architectures, such as Convolutional Neural Networks (CNNs) with encoder-decoder structures and attention mechanisms, we aim to achieve highly accurate and efficient segmentation results. Throughout this chapter, we provide an explanation of our methodology, including the architecture, training process, and optimization strategies employed. By introducing this novel methodology, we aspire to contribute to the field of geological interpretation, enabling more accurate reservoir characterization and exploration in the oil and gas industry.

3.2 Semantic Segmentation

Semantic Segmentation is a computer vision technique that labels each pixel in an image with its corresponding category, such as 'car,' 'tree,' or 'road.' Unlike object detection, which identifies and locates objects within an image, Semantic Segmentation provides a detailed map by classifying every part of the image. This detailed labeling is essential for tasks requiring a comprehensive understanding of the scene.

One of the significant challenges in Semantic Segmentation is dealing with the complexity and variability of real-world images. Objects can overlap, come in different shapes and sizes, and be seen from various angles, making accurate segmentation a difficult task. Factors such as lighting, occlusions, and changing scales further complicate the process.

To tackle these challenges, modern approaches use Deep Learning, particularly Convolutional Neural Networks. CNNs excel in learning and recognizing patterns from raw image data. They can identify both fine details and broader features necessary for accurately labeling each pixel. Semantic Segmentation is widely used across different fields. In medical imaging, it helps in detecting tumors and segmenting organs. In autonomous driving, it enables the understanding of road scenes and the identification of objects like pedestrians and vehicles. In remote sensing, it assists in land cover classification and urban planning. By providing precise pixel-level classification, Semantic Segmentation plays a crucial role in these and many other applications [66].

3.3 Methodology

In this section, we detail the methodology employed in our study, Guided by principles of the data science methodology (except for Data Deployment), encompassing our dataset, data processing, model development, and evaluation (see Figure 3.1).

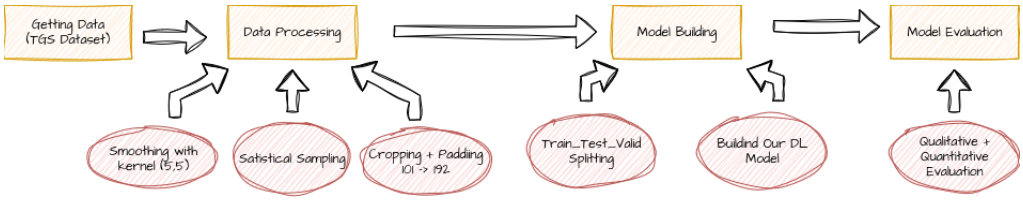


Figure 3.1: Our Methodology

3.3.1 The Dataset

There are three open datasets existed on the web:

- **Netherlands Offshore F3 Block Dataset:** This dataset consists of seismic data collected offshore in the Netherlands, specifically in the F3 block area. It is commonly used in geophysics and seismic imaging

research for testing and benchmarking various algorithms and techniques related to seismic interpretation, imaging, and inversion. The dataset typically includes 2D or 3D seismic reflection data, well logs, and geological interpretations.

- **LANDMASS Dataset:** The LANDMASS dataset is a collection of seismic reflection data acquired onshore, primarily used for research and development in seismic imaging and interpretation. It may cover various geological regions and terrains, providing seismic data from different environments such as basins, mountains, or plains. Researchers utilize the LANDMASS dataset for studying geological structures, imaging subsurface features, and testing seismic processing algorithms.

- **TGS Dataset:** The TGS dataset, often referred to as the TGS Salt Identification Challenge dataset, is a publicly available dataset provided by the geoscience company TGS. It contains seismic reflection data with a focus on salt dome identification within the subsurface. Salt domes are geological structures formed by the movement of salt deposits beneath the Earth's surface and are of significant interest in hydrocarbon exploration. The TGS dataset is commonly used for developing and evaluating algorithms for automatic salt dome detection and seismic interpretation tasks.

We used the data of the TGS Salt Identification Challenge competition

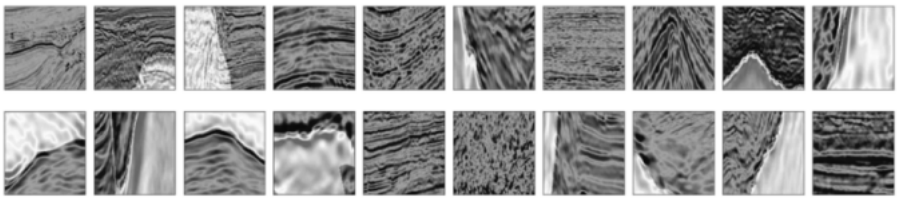


Figure 3.2: Overview of original data

3.3.2 Data Processing

Data processing plays a pivotal role in the context of deep learning, serving as the cornerstone for model training, validation, and inference. In the realm of deep learning, where algorithms learn from vast amounts of data to make decisions or predictions, the quality and efficiency of data processing directly impact the performance and reliability of the models. Effec-

tive data preprocessing techniques, such as normalization, augmentation, and feature extraction, ensure that the input data is appropriately formatted and optimized for consumption by deep learning algorithms. Moreover, data processing facilitates tasks such as data cleaning, outlier detection, and handling imbalanced datasets, which are critical for improving model robustness and generalization capabilities. By streamlining data processing workflows and harnessing the power of advanced techniques, deep learning practitioners can unlock the full potential of their models, driving advancements in various domains, in our case, Semantic Segmentation.

Prior to model development, there are multiple ways to alter images and preprocessing techniques (For instance data sampling), but this project will fixed to the following after multiple experiments:

1. **Blurring**: To enhance convergence and mitigate noise, we applied a 5x5 kernel blur to the images.

2. **Statistical Sampling**: Given the class imbalance, where 54% of images lacked salt masks, we undertook statistical sampling. Specifically, we randomly selected 500 non-salt images for empirical reasons, balancing the dataset.

3. **Mask Processing**: Utilizing the `cropNonEmptyMaskIfExists` function, we cropped images based on non-empty masks, focusing on pertinent regions. Subsequently, padding was applied to standardize image dimensions to (192, 192) for empirical reasons.

4. **Train-Test Split**: Following preprocessing, we performed a stratified train-test split, allocating 80% of the dataset for training and reserving 10% each for validation and testing, where the classes are equally distributed for each group.

3.3.3 Model Development

For model development, we designed a Deep Learning architecture tailored to the task of salt identification. The architecture proposed (see Figure 3.2) in our study is seen as a combination of the MT-UNet architecture with the VAE (Variational Autoencoder) model and the custom layer "LiquidLayer". It's a complex structure that integrates different components in an inter-

connected manner for image processing. First, the integration of a VAE encoder enriches the latent representation of image features, favoring more structured representations and the generation of diverse image. This VAE includes strategically stacked convolution layers, such as two layers of 3x3 convolution followed by 2x2 max-pooling layers (see the figure), gradually reducing the dimensionality of the input image. These encoder layers generate the latent space, where each point represents a unique feature of the image. By combining this generation capability with the segmentation power of the U-Net structure with a pre-trained backbone such as ResNet18 (Wang et al., 2015), the model is able to provide accurate results for image segmentation. Adding transformer blocks to the U-Net model output strengthens its ability to capture complex relationships in the data, thereby improving class separability. Drive stability is promoted by the use of VAE loss, providing natural regularization. Furthermore, the flexibility of the model is underlined by the possibility of specifying the number of transformation blocks, allowing adaptation to the complexity of the data. These transformation blocks dynamically adapt the features extracted by the backbone to capture task-specific information. For instance, they may focus on detecting contours, textures, or other important details in the image. Ultimately, the intermediate outputs of the backbone and the transformed intermediate outputs are concatenated to form a single final output. This fusion of information from the backbone and transformed information allows the MT-UNet model to combine high-level features learned by the backbone, potentially comprising multiple deep convolution layers, with task-adapted features from the transformer blocks, consisting of additional convolution layers. Finally, the custom layer "LiquidLayer" introduces an interpretable dimension by adjusting a parameter, thus providing insights into its impact on the model's predictions. The final model, thus constructed using convolution layers for the encoder and backbone, is ready for training and evaluation in specific image segmentation tasks, where it excels in capturing significant features and adapting its representations. This interconnected and adaptable architectural approach is particularly useful for computer vision problems where a rich representation of images is essential. The algorithm is shown below (Algorithm 1) :

Algorithm 1 MT-UNet-VAE-LSM

Require: Input image of size (192, 192, 3), specified backbone for the Unet model, number of transformer blocks

Ensure: Trained MT-UNet-VAE model

procedure MT-UNet-VAE-LSM
    Begin:
    Initialize the VAE encoder:
        • Create a Conv2D model with a LeakyReLU (alpha=0.2) activation for each layer
        • Flatten the encoder output and feed it into two distinct dense layers to obtain the distribution parameters (z_mean and z_log_var) of the latent distribution
        • Use a sampling layer to sample latent samples (z) based on the distribution parameters
    Initialize the VAE decoder:
        • Create a dense and transposed Conv2D model to reconstruct the image
    Initialize the MT-UNet model:
        • Create a Unet model with the specified backbone
        • Add the specified number of mixed transformer blocks to the MT-UNet model
        • Apply the LiquidLayer to the output of the MT-UNet model
    Compile the model:
        • Compile the model with a Dice loss for segmentation and a VAE loss for learning the latent distribution
    Train the model:
        • Train the model with the SGD optimizer, Dice loss, and metrics
    End.
end procedure

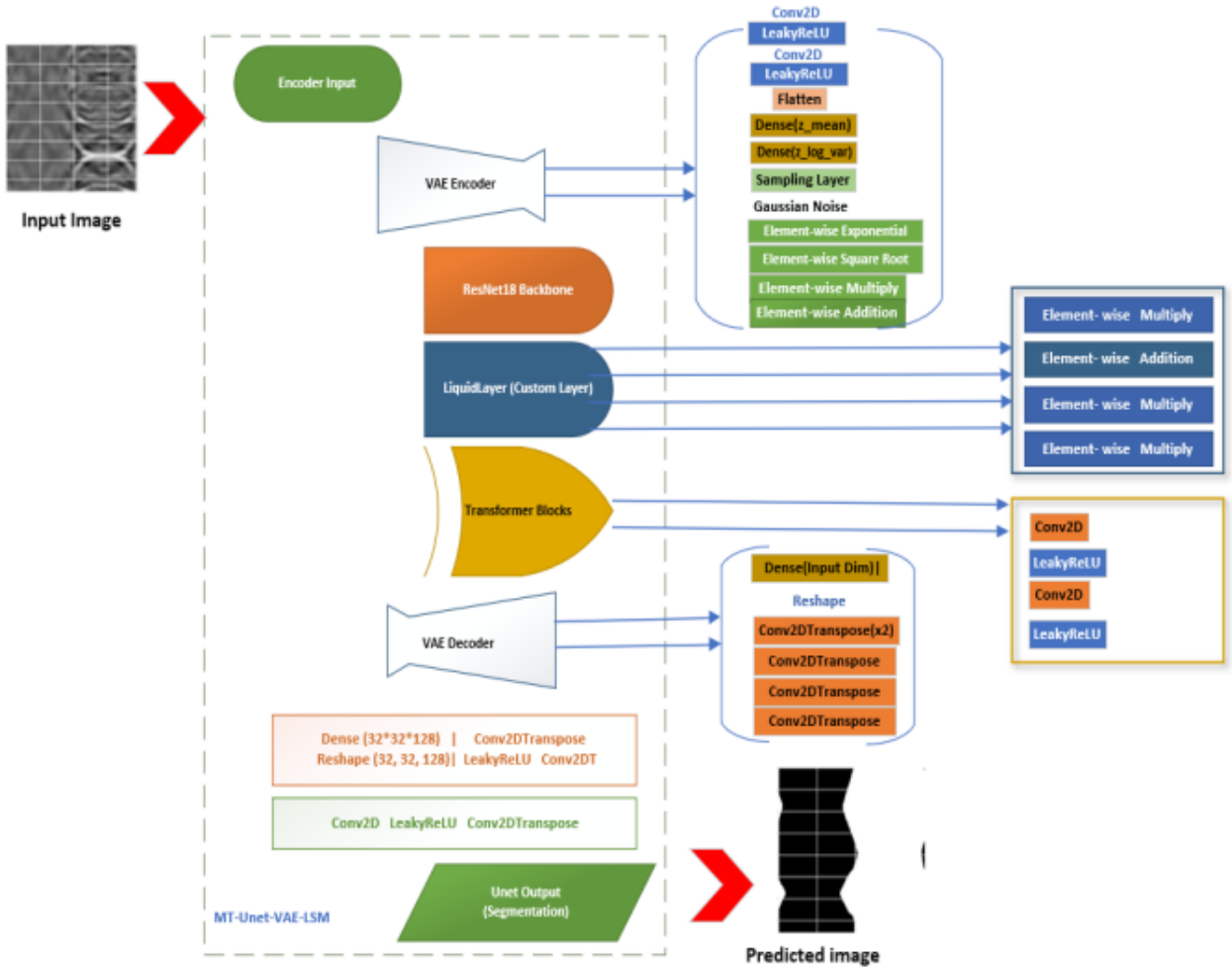


Figure 3.3: Proposed structure of MT-U-net-VAE-LSM model for salt detection.
This model consists of four main components: (1)- A VAE encoder/decoder, which takes the input image and produces a latent representation. (2)- A U-Net model, which takes the latent representation and produces a segmentation mask. (3)- A Liquid Layer (a custom Liquid State Machines layer), which is a custom layer that applies a trainable weight to the output of the U-Net model. (4)- Transformers are used in the MT-Unet model to improve the accuracy of the mask segmentation produced by the U-Net model

3.3.4 Evaluation

We conducted both qualitative and quantitative evaluations to assess the efficacy of our model:

1. **Qualitative Evaluation**: We visually inspected model predictions against ground truth labels to gauge segmentation accuracy and identify any discrepancies or artifacts.

2. **Quantitative Evaluation**: Quantitative metrics, including Intersection over Union (IoU), Dice Similarity Coefficient (DSC), and Hausdorff Distance (HD), were computed to quantify model performance objectively. Furthermore, we compared our model's performance against baseline methods and state-of-the-art approaches to ascertain its effectiveness.

3.4 Conclusion

This chapter has provided an exposition of our methodology for detecting salt domes using Semantic Segmentation in seismic images. By meticulously outlining the steps involved in data preprocessing, model development, and evaluation. By establishing a robust framework for automating this crucial geological task. Our methodology offers a promising avenue for accelerating the identification of subsurface structures. The next chapter provide a detail explanation for each step following by empirical results.

Chapter 4

Implementation and Results

4.1 Introduction

In this chapter, we will delve into the application of Deep Learning for detecting salt domes in seismic images. We will discuss the setup of our development environment, techniques for preprocessing data, the experiments and the implementation of our model, metrics used for evaluation, and the outcomes of our study.

4.2 Data Processing in Details

We created a mosaic using patches from the dataset to visualize the data. The competition's objective is to segment areas containing salt, with a unique consideration: if a 101x101 image encompasses all salt pixels, it's treated as an empty mask, reflecting the emphasis on delineating salt deposit boundaries rather than the entire salt body. The dataset consists of 4000 training images and 18000 test images, divided into two sets: "train" and "depths." The "train" set has three columns: image ID, the depth at which the image was captured, and the tokenized mask. It comprises 4000 rows, each representing specific training image information. As for the "depths" set, it has two columns: image ID and depth, covering a total of 22000 images, including both training and test images. It's worth noting that some test images are intentionally labeled. Out of the 14000 test images, 7499 have empty masks (no salt rocks or the entire image is a salt dome). Figure 4.1 is visual includes 20 random images extracted from the dataset, providing an overview of the

visual representation of information. The brightest white areas correspond to masks identified as salt bodies. Although the images appear sharp, some blurring has been applied with a mask of size (5, 5).

To promote efficient convergence of the machine learning model, normalization and scaling of the data was performed, bringing the pixel values to a common range of 0 to 1 by dividing by 255.0. The images, as well as the masks, have original dimensions of (101, 101), but have been subjected to padding up to (198, 198) for empirical reasons. This decision proved beneficial, improving the performance of the neural network compared to later changing the dimensions after padding. The precise selection of 198 dimensions aligns with deep learning architectures that require images to have dimensions divisible by 48 and 32, while taking into account compute capacity and memory constraints.

To eliminate the imbalance between positive and negative examples in the dataset, imbalance handling techniques, such as oversampling or undersampling, were deployed to balance the classes. Notably, 54% of the images in the train dataset do not have a mask. Therefore, these images were eliminated, and 500 no salt images were randomly selected. Our training dataset consists of 5% non-salt images and 95% salted images. In the training set, 80% of the images are divided into 5% non-salt and 95% salted. Similarly, both the validation and test sets contain 10% of the images each, with 5% being non-salt and 95% salted. This distribution remains consistent across all datasets, resulting in a balanced representation of non-salt and salted images (Figure 4.1).

4.2.1 Dataset Description

We provide a comprehensive description of the dataset utilized in our study. The dataset consists of multiple attributes (Figure 4.2), each playing a crucial role in our analysis and model development.

1. **Image ID (*id*)** : This column contains unique identifiers for each image in the dataset. These identifiers are essential for indexing and referencing specific images during data processing and analysis.

2. **Depth (*z*)** : The depth attribute represents the depth at which the im-

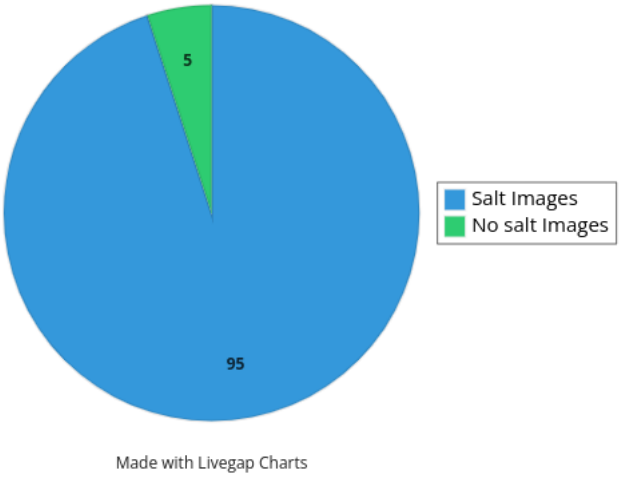


Figure 4.1: Salt/No Salt images Repartition

age was captured. This information is valuable in understanding the geological context of the images and their spatial distribution within the Earth's subsurface, from our analysis, we found that fortunately its distribution follows normal distribution (Figure 4.3), so we didn't have to make any further selection based on this column .

3. **Images** : The dataset includes grayscale images depicting 2D slices representing a 3D perspective of the Earth's interior, obtained through reflection seismology. These images capture the interfaces between different rock types in random subsurface locations, the images are of size 101*101 corresponding to the height and weight respictively.

4. **Masks** : Corresponding to each image, the dataset contains binary masks that classify each pixel as either salt or sediment. These masks are crucial for the segmentation task, delineating regions containing salt deposits from the surrounding sedimentary layers.

5. **Salt Amount** : This attribute denotes the quantity of salt present in each image, ranging from 0 to 1. It provides quantitative information about the salt content within the subsurface formations represented by the images.

6. **Salt Coverage (1-10)** : The salt coverage attribute provides a qualitative assessment of the extent to which salt deposits are present in each

image. It is rated on a scale from 1 to 10, with higher values indicating greater coverage of salt within the image.

7. **RLE Mask (Run-Length Encoded Mask)** :This column contains the run-length encoded representation of the binary masks corresponding to each image. Run-length encoding is a compression technique used to represent consecutive sequences of identical pixels efficiently.

	id	z	images	masks	salt_amount	coverage_class	rle_mask
0	a266a2a9df	794	[[[0.3411764705882353, 0.3411764705882353, 0.3...	[[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,...	0.504950	6	5051 5151
1	75efad62c1	468	[[[0.5686274509803921, 0.5686274509803921, 0.5...	[[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,...	0.993334	10	9 93 109 94 210 94 310 95 411 95 511 96 612 96...
2	34e51dba6a	727	[[[0.5411764705882353, 0.5411764705882353, 0.5...	[[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,...	0.149201	2	48 54 149 54 251 53 353 52 455 51 557 50 659 4...
3	4875705fb0	797	[[[0.06666666666666667, 0.06666666666666667, 0...	[[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,...	0.042839	1	1111 1 1212 1 1313 1 1414 1 1514 2 1615 2 1716...
4	782ae9b7e7	677	[[[0.6078431372549019, 0.6078431372549019, 0.6...	[[1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,...	0.225370	3	1 1815 1819 90 1920 81 2021 73 2122 64 2223 55...

Figure 4.2: Dataset Description

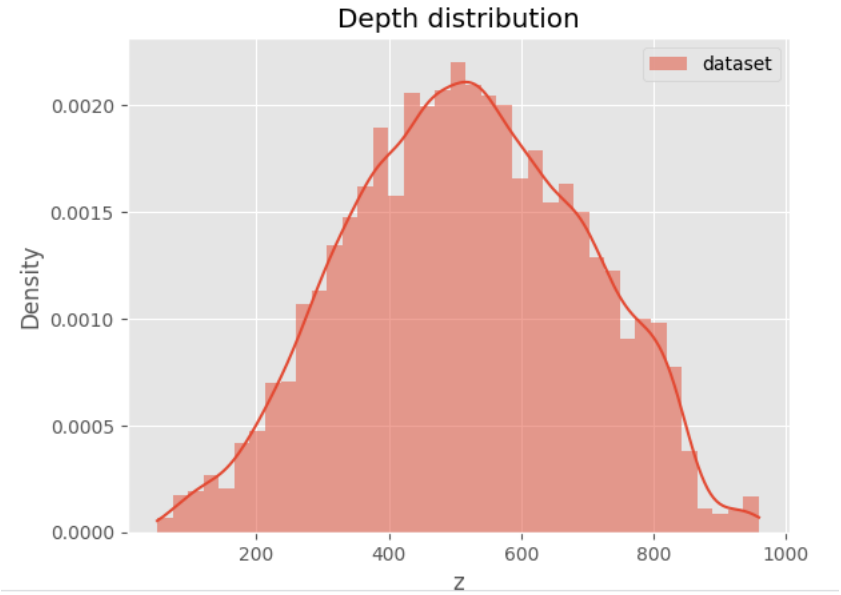


Figure 4.3: Depth Description

4.3 The Development Environement

A programming environment refers to the set of tools, libraries, and resources that developers use to write, test, and debug software applications. It

provides an integrated platform where programmers can create, modify, and execute their code efficiently. A well-designed programming environment streamlines the development process, enhances productivity, and facilitates collaboration among developers. For this project we have used :

4.3.1 Kaggle

Kaggle (its website picture given in Figure 4.4) is an online platform known for hosting data science and machine learning competitions. It serves as a community hub for data scientists, machine learning practitioners, and researchers to collaborate, learn, and showcase their skills. Kaggle offers a wide range of datasets, competitions, tutorials, and forums, making it a valuable resource for data-driven projects. Users can access diverse datasets across various domains and participate in competitions to solve real-world problems. Kaggle also facilitates collaboration through code sharing and discussion forums. The platform provides opportunities for data scientists to showcase their expertise, gain recognition, and connect with potential employers or collaborators. In summary, Kaggle is a vibrant platform that fosters learning, collaboration, and problem-solving in the field of data science [67].

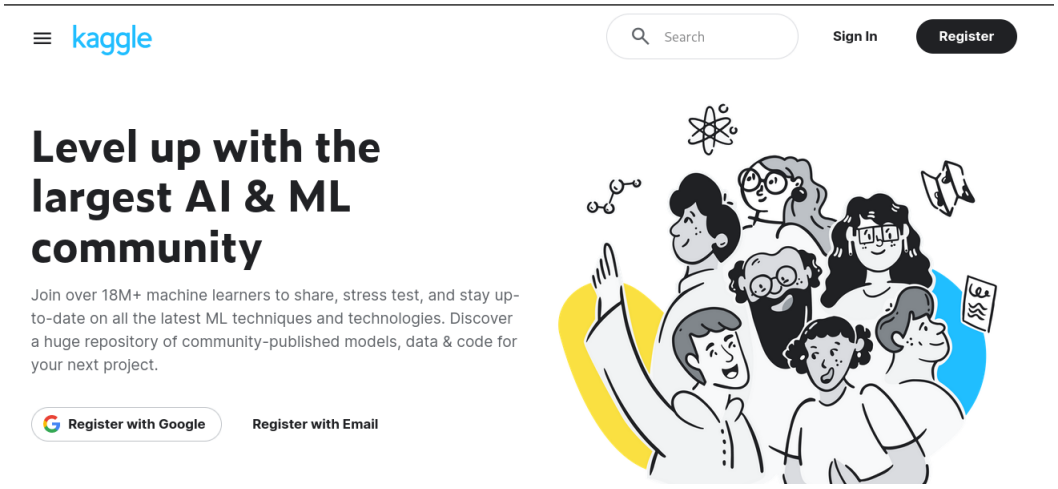


Figure 4.4: Kaggle's website [68]

4.3.2 Python

Python [69] is a high-level, interpreted programming language that is renowned for its simplicity and readability. It features a clean and intuitive syntax, utilizing indentation and minimalistic punctuation, making it easy to read

and understand. Python has a vast ecosystem of libraries and frameworks, such as NumPy, Pandas, and TensorFlow, which provide powerful tools for scientific computing, data analysis, and machine learning. Its versatility extends to web development, automation, scripting, and system administration. Python's strong community support fosters collaboration, with developers actively contributing to open-source projects and sharing knowledge. It is cross-platform compatible and runs on various operating systems, allowing for seamless deployment. Python's popularity and ease of use have led to its adoption in diverse domains, making it a preferred choice for both beginners and experienced programmers (see Figure 4.5).

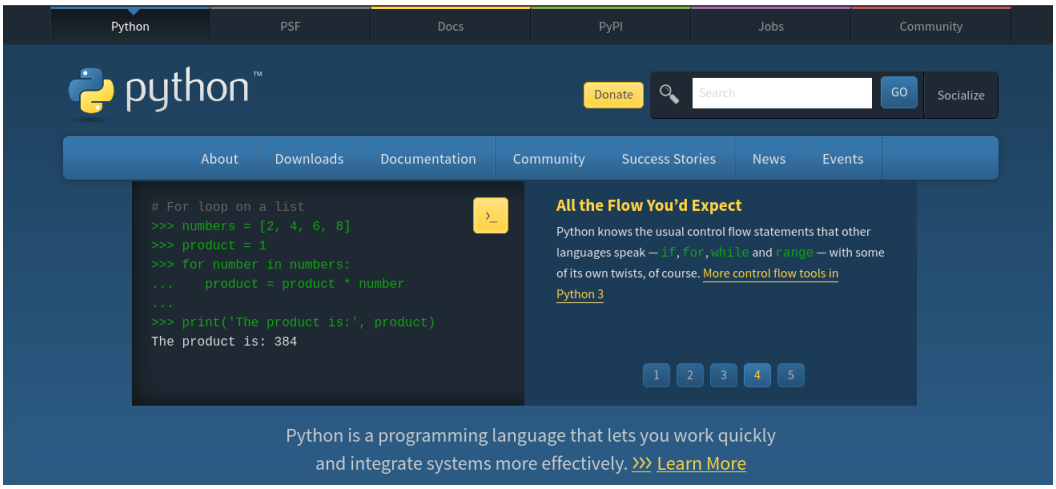


Figure 4.5: Python's website

4.3.3 Tensorflow and Keras

TensorFlow [70] was developed by the Google Brain team, led by Jeff Dean and Rajat Monga. It was released in 2015 and has become one of the most widely used frameworks for machine learning and AI. TensorFlow has applications in computer vision, NLP, speech recognition, and reinforcement learning, enabling tasks like image classification, object detection, language translation, and more.

Keras, created by François Chollet, is a high-level API running on top of TensorFlow. It was released in 2015 and gained popularity for its simplicity. Keras is widely used for building and training neural networks, particularly for tasks like image recognition, sentiment analysis, and sequence generation.

Both TensorFlow and Keras have active communities, providing extensive documentation, tutorials, and resources. TensorFlow's flexibility and scalability make it suitable for research and production. Keras, with its user-friendly interface, allows for quick prototyping and experimentation.

TensorFlow 2.0 onwards includes Keras as its official high-level API, solidifying their integration. This merger enhances the capabilities and usability of both libraries, combining TensorFlow's power with Keras' simplicity.

The integration of TensorFlow and Keras has made them leading libraries in deep learning. They offer a wide range of applications and are accessible to users of different skill levels, contributing to their widespread adoption in academia and industry.

Figure 4.6: Keras & Tensorflow

4.4 Evaluation metrics

To evaluate the performance of our segmentation model, we opted to use metrics commonly used in semantic segmentation problems. These metrics include Intersection over Union (IoU), Dice Similarity Coefficient (DSC), Hausdorff Distance (HD), and Mean Absolute Distance (MAD). Our aim is to provide an accurate and realistic evaluation of our model. Thus, to gauge accuracy and robustness, we prioritized the use of IoU as the primary evaluation criterion.

Concerning the model's sensitivity and tolerance, we measured the Dice Coefficient. MAD and HD metrics were also employed to assess the similarity between segmentation results and ground truth.

4.4.1 Intersection over union (IoU)

The IoU between a predicted set of salt pixels and a set of true salt pixels is determined by the formula (4.1).

$$\text{IoU} = \frac{Intersection(Pred, GT)}{Union(Pred, GT)} = \frac{Pred \cap GT}{Pred \cup GT} \tag{4.1}$$

For binary classification, it is written as shwon in equation (4.2).

$$\text{IoU} = \frac{TP}{TP + FP + FN} \tag{4.2}$$

In the context of binary classification, we define:

$$\text{TP} = \text{True Positive}$$
$$\text{FN} = \text{False Negative}$$
$$\text{FP} = \text{False Positive}$$

4.4.2 The Dice Similarity Coefficient (DSC)

The dice Coefficient (DSC), one of the most common methods for evaluating segmentation results, indicates a level of similarity between the reference (manual segmentation) and segmented result (automatic segmentation) (Raina et al., 2023). The formulation of DSC is given by the equation (4.3).

$$\text{DSC} = \frac{2 \times (Pred \cap GT)}{|Pred| + |GT|} \tag{4.3}$$

where **DSC** $\in [0, 1]$

4.4.3 The Hausdorff distance (HD)

HD is metric represents the spatial distance between two point sets. The formulation of HD is given by the equation (4.4).

$$HD(A, B) = \max \left\{ \sup_{a \in A} \inf_{b \in B} d(a, b), \sup_{b \in B} \inf_{a \in A} d(b, a) \right\} \tag{4.4}$$

where $d(a, b)$ is the distance metric between points a and b, and A and B are sets of points.

4.4.4 The Mean Absolute Distance (MAD)

MAD is a measure of the average distance between each data point and the mean of the data set (Formula (4.5)). It is a robust measure of variability, meaning that it is not as sensitive to outliers as some other measures, such as the standard deviation.

$$\mathrm{MAD}(A, B) = \frac{1}{|A|} \sum_{a \in A} \min_{b \in B} d(a, b) \tag{4.5}$$

where A is the predicted set and B is the Ground Truth set.

4.5 Implementation Details

All experiments were conducted using the computational power of a 15 GB GPU provided by Kaggle (see Figure 4.7 and Figure 4.8). Our dataset comprised 22,000 images divided into two main groups: 4,000 compressed images for training and 18,000 compressed images originally, fortunately more than 14,000 images for testing were later labeled where more than 50% of them have empty masks, so at the end there are 22,000 images, but we worked only on the labeled data. Each group was accompanied by a pandas DataFrame containing the image ID and the corresponding mask in run-length encoding (RLE), additionally depth column for each image given, each image can be either salt images or just sediment (which means they have simply empty masks - simply just empty background images).

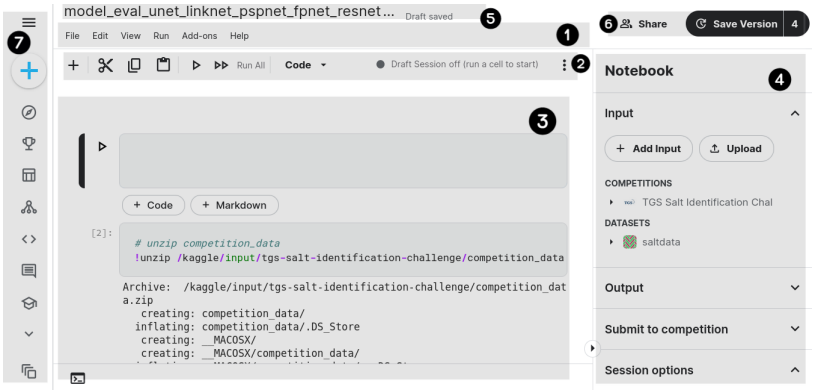


Figure 4.7: Kaggle's environment : (1) Menu bar for editing the notebook (2) Execute menu bar (3) Notebook and command line interface at the bottom (4) Metadata about the environment including input files directory and output directory (5) Notebook file name (6) Share button and Save Version button (7) Kaggle competitions panel.

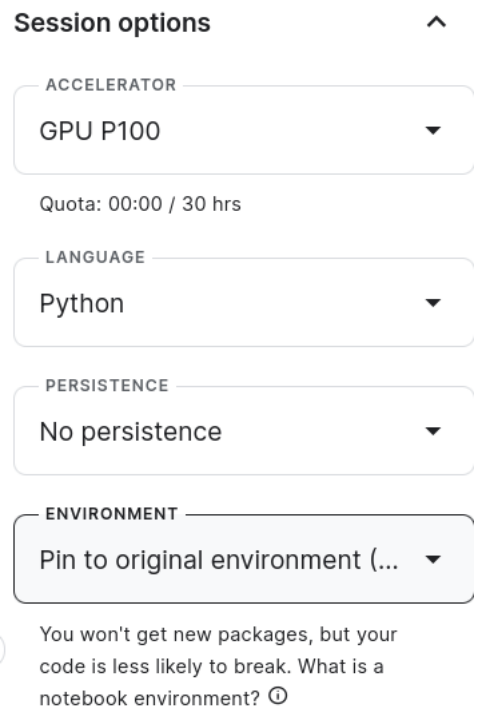


Figure 4.8: Kaggle's environnement of execution

We preprocessed these images by converting them into matrices of size $101 \times 101 \times 3$ using OpenCV. This conversion applied to both sets of images, which were then combined into a single dataset. We ensured there were no duplicate images by cross-referencing IDs. The RLE masks were decoded into 101×101 matrices in grayscale level (matrices of 0s and 1s) and added to a new column named 'masks' as shwon below in the first code snippet.

```
# LOAD IMAGES AND MASKS + NORMALISATION

data_df["images"] = [cv2.imread("{}/{}.png".format(image_path,idx),cv2.
    IMREAD_COLOR)/255.0  for idx in tqdm_notebook(data_df.id)]
data_df["masks"] = [cv2.imread("{}/{}.png".format(mask_path,idx),cv2.
    IMREAD_UNCHANGED)/65535.0 for idx in tqdm_notebook(data_df.id)]
```

```
def rle2mask(mask_rle, shape=(101,101)):
    '''
    mask_rle: run-length as string formated (start length)
    shape: (width,height) of array to return
    Returns numpy array, 1 - mask, 0 - background

    '''
    s = mask_rle.split()
    starts, lengths = [np.asarray(x, dtype=int) for x in (s[0:][::2], s
    [1:][::2])]
    starts -= 1
```

```
    ends = starts + lengths
    img = np.zeros(shape[0]*shape[1], dtype=np.uint8)
    for lo, hi in zip(starts, ends):
        img[lo:hi] = 1
    return img.reshape(shape).T
```

To analyze the data, we examined the distribution of depths, which followed a normal distribution (Figure 4.3). Additionally, we calculated the "salt coverage" for each mask, representing the percentage of non-empty pixels (the equation is given below (4.6)), and scaled this value to a range of 0 to 10. This was added as a new column to the dataset. A bar plot revealed that 54% of the masks had zero salt coverage. To address this, we downsampled the dataset to retain only 500 images with zero coverage, ensuring a balanced representation as shown on Figure 4.9, so . Than we moved to make blurring to all the images presented in the dataset:

```
# DENOISING
training['images'] = training['images'].apply(lambda x :cv2.blur(x,(5,5))
    )
```

$$\text{Coverage} = \frac{\sum_{i \in \text{mask}} \text{pixel}_i}{\text{Image Size}^2} \tag{4.6}$$

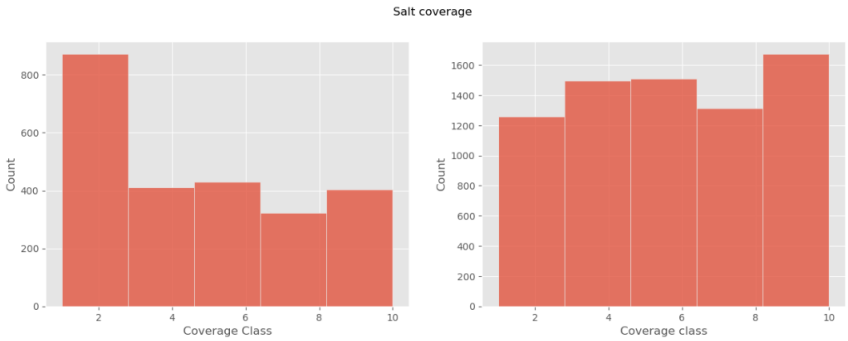


Figure 4.9: Distribution of salt coverage in the dataset (Left) before sampling (Right) after sampling.

After adjusting the dataset, we redrew the depth distribution, which remained normally distributed (Figure 4.10). Consequently, 95% of the dataset comprised images with non-empty (salty) masks, while 5% had empty masks.

For model training, we split the data which contains more then 12,000 images into training (80%), validation (10%), and test (10%) sets, ensuring stratification based on salt coverage to maintain the 5% no salt and 95% salt ratio

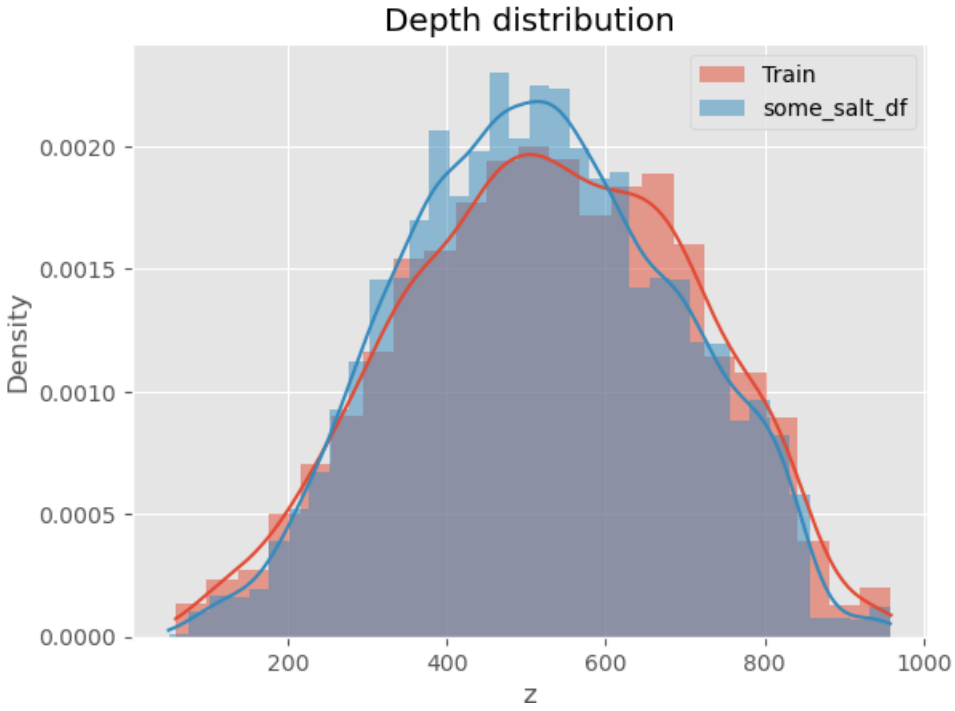


Figure 4.10: Distribution of salt coverage in the dataset where the red distribution before sampling and the blue one after sampling.

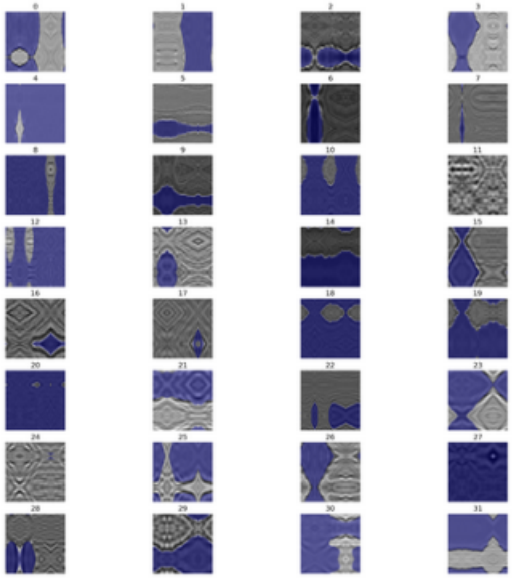


Figure 4.11: Ensemble of test images alongside their masks.

in each group. Figure 4.11 shows a set of images alongside their masks taken randomly from the test set.

```
# SPLIT DATA TO TRAIN, VAL,TEST : 80% , 10%, 10%

train_idx ,val_idx = sklearn.model_selection.train_test_split(training,
    test_size = 0.2, stratify = training.coverage_class, random_state =
    42)

valid_idx,test_idx = sklearn.model_selection.train_test_split(val_idx,
    test_size=0.5,stratify=val_idx.coverage_class,random_state=42)
```

Following the train-test split, we applied further modifications to the data using the Albumentations library to enhance the robustness of our model. Specifically, we employed the following augmentation pipeline:

```
## IMAGE TRANSFORMATIONS LIKE PADDING
import albumentations as A
def aug(image_size = 192, crop_prob = 1, train = True):
    if train:
        return A.Compose([
            A.PadIfNeeded(min_height=image_size, min_width=image_size, p
    =1),
            A.CropNonEmptyMaskIfExists (width = image_size, height =
    image_size, p=crop_prob),
            #A.ColorJitter(p=0.5),
        ], p = 1)
    else:
        return A.Compose([
            A.PadIfNeeded(min_height=image_size, min_width=image_size, p
    =1),
            A.CropNonEmptyMaskIfExists(width = image_size, height =
    image_size, p=crop_prob),
        ], p = 1)
```

- **Padding and Cropping**: Ensured that images were padded to the desired size and cropped non-empty regions.
- **Optional Color Jittering**: Although commented out, this could be used to introduce random variations in brightness, contrast, and saturation.

Than we proceed to train the model over 200 epochs. Our model featured a combination of MT-UNet and VAE architectures, enhanced with a custom

"LiquidLayer." The images were processed in batches of 32 with dimensions of 192×192 pixels. Key parameters included a latent space dimension of 64 and 15 transformer blocks. We used the SGD optimizer with a learning rate of 0.001 and a momentum of 0.9. The segmentation utilized the U-Net model with a ResNet18 backbone, initialized with 'imagenet' weights, and a sigmoid activation function for binary classification (Salt presented by pixels of 1 and the background by 0 pixels). The dice loss function was applied for segmentation, while the VAE loss combined reconstruction loss with Kullback-Leibler divergence.

```
initial_learning_rate = 1e-4
optimizer = SGD(learning_rate=0.001, momentum=0.9)


callbacks = [
    ModelCheckpoint("./keras.model", save_best_only=True, verbose=1),
    ReduceLROnPlateau(factor=0.5, patience=5, min_lr=0.01, verbose=1)


]
model.compile(optimizer=optimizer, loss=[dice_loss, vae_loss], metrics=['
    iou_score', 'accuracy'])
```

The LiquidLayer, with its trainable weight initialized uniformly, provided an interpretable component by adjusting its parameter and analyzing its impact on predictions.

4.6 Results

4.6.1 Impact of each module on the overall model

Examining the impact of individual modules on the performance of our model, our ablation studies reveal instructive subtleties. Starting with the U-net model without transformer layers, VAE, or LSM, we established a state-of-the-art IoU of 0.8964. Integrating transformers alongside the Unet's ResNet 18 backbone showed a significant performance increase, raising the IoU to 0.9515. The subsequent addition of VAE further improved accuracy to 0.9601, with LSM layers reaching a peak IoU of 0.9612. These incremental improvements highlight the synergistic effect of module integration. Beyond technical sophistication, our results underscore a nuanced understanding of individual contributions,advocating for a composite framework

comprising MT-U-net, VAE, and LSM layers to maximize detection accuracy.

4.6.2 Quantitative Evaluations

In the field of advanced image analysis, our MT-UNet-VAE-LSM model surpasses state-of-the-art techniques described in references (Milosavljević, 2020; Karchevskiy et al., 2018; Li K, 2023). The study by (Milosavljević, 2020) provides a detailed comparison of segmentation model performances such as U-net, FPN Lin2016FPN, LinkNet [71], and PSPnet [72], alongside there approach. Our method distinctly stands out for its level of excellence, as evidenced by remarkable results in terms of precision and robustness compared to these studies. Following a meticulous evaluation of 1,000 images, our model achieved an average Intersection over Union (IoU) score of 0.96260, showcasing its undeniable competence.

The comparative analysis, detailed in Table 4.1, demonstrates a significant 6% improvement in the overall average IoU compared to the results reported in (Milosavljević, 2020; Karchevskiy et al., 2018; Li K, 2023). These data compellingly demonstrate the superiority of our model in the field of advanced image analysis for identifying salt domes. Moreover, the model's exceptional performance is further validated by the impressive Dice Coefficient Similarity (DCS) of 0.9417. This metric, approaching unity, elucidates the model's ability to closely align with ground truth in diverse scenarios, attesting to its remarkable versatility. Delving into the minutiae, Table 4.2 showcases minimal averages for Dice Coefficient Similarity (DCS), Hausdorff Distance (HD) and Mean Absolute Deviation (MAD) Metrics, at 1.0781 and 0.01574 respectively. These results underscore the model's precision in capturing nuanced details while maintaining consistency. In the grand tapestry of image analysis tasks, our approach stands as a robust and effective solution, poised to redefine standards in the field. These compelling findings solidify its position as a formidable contender in the ever evolving landscape of advanced image processing and analysis.

To illustrate the training processes figure 4.12 presents two graphs detailing IoU and accuracy measurements on both training and validation sets through- out a training cycle. It's noteworthy that our model demonstrates remarkable stability at every stage of training and validation, enhancing its reliability and effectiveness across various operational scenarios. This sta-

bility is evidenced by comparing the training graphs of our model, as depicted in figure 4.12, with those of corresponding precision and IoU metrics presented in the study (Milosavljević, 2020) and represented in figures 4.13, 4.17, 4.18, 4.19, 4.20. Given the use of the same training framework, all these graphs exhibit similar behavior. Initially, the oscillations of the validation metrics are more pronounced, but they gradually stabilize as the learning rate decreases. A comparison of the IoU metric for the training set reveals a significant difference: after the initial training phase, the models presented in the study achieve around 50% to 60%, while our approach reaches 80%. This behavior can be explained by our utilization of U-net with pre-trained ResNet-18 instead of ResNet-34 alone as indicated in the study, providing us with an initial advantage.

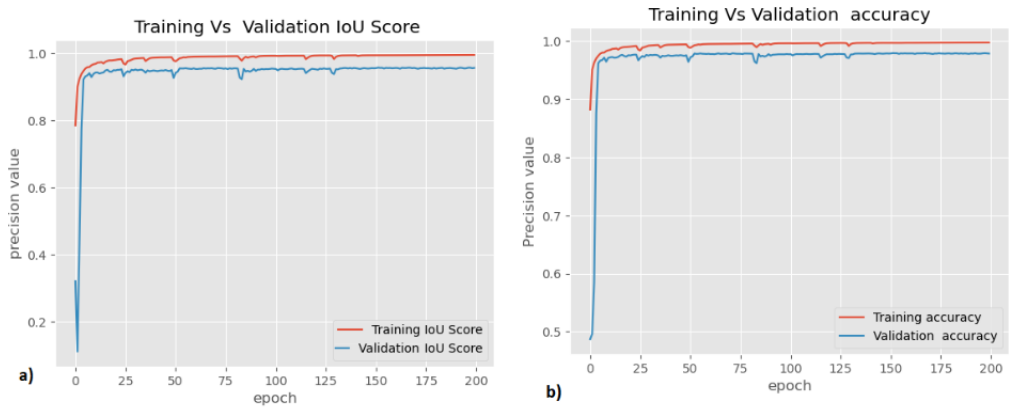


Figure 4.12: Sample accuracy (a) and intersection over union (IoU) (b) plots. Graphs (a) and (b) show the intersection-over-union (IoU) and sample accuracy metrics, respectively, with values plotted for the training and validation sets when evaluating models across its training phase

4.6.3 Qualitative Results

The qualitative results of our segmentation are depicted in Figure 4.21. This visual representation clearly demonstrates that our segmentation (depicted in brown) closely aligns with ground truth (in green), underscoring the precision of our approach. Upon closer examination, our model exhibits a remarkable ability to capture the subtle nuances and details of the seismic images, evident in the precise segmentation regions observed across various contexts. Furthermore, the consistency and robustness of our method are highlighted through Figure 4.22. Part A of this figure displays the seismic images used in our analysis, providing essential visual context for the sub-

sequent evaluations. Part B showcases manual segmentations represented in green, serving as the ground truth for our model's predictions. This comparison illustrates the effectiveness of our approach by visually demonstrating the alignment between our model's predictions (part C) and the ground truth segmentations (part B), thus affirming the accuracy and reliability of our results.

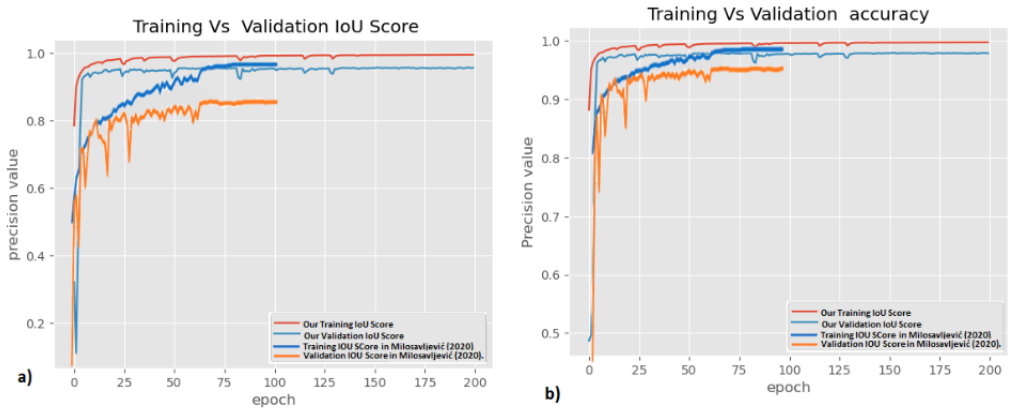


Figure 4.13: Comparison of Intersection over Union (IoU) (a) and accuracy (b) metrics between the model proposed by (Milosavljević, 2020) and our own model.

Table 4.1: Segmentation Model IoU scores

Model	**Description**	**IoU Score**
Model proposed in (Karchevskiy et al., 2018)	Automatic salt deposits segmentation using CNNs	0.8880
Proposed and tested models in (Milosavljević, 2020)	U-Net	0.76996
	FPN	0.85219
	LinkNet	0.84565
	PSPNet	0.83137
Model proposed in (Milosavljević, 2020)	-	0.89646
Model proposed in (Li K, 2023)	Models performance without inversion on TGS dataset	0.90
Our method MT-UNet-VAE-LSM	-	0.96260

Table 4.2: DSC, HD, and MAD results of our model

Measures	DSC	HD	MAD
Our model (Mean)	0.90707	2.31664	0.01753
Our model (Median)	0.98957	2.06614	0.00621
Our model (SD)	0.23555	1.79514	0.04236

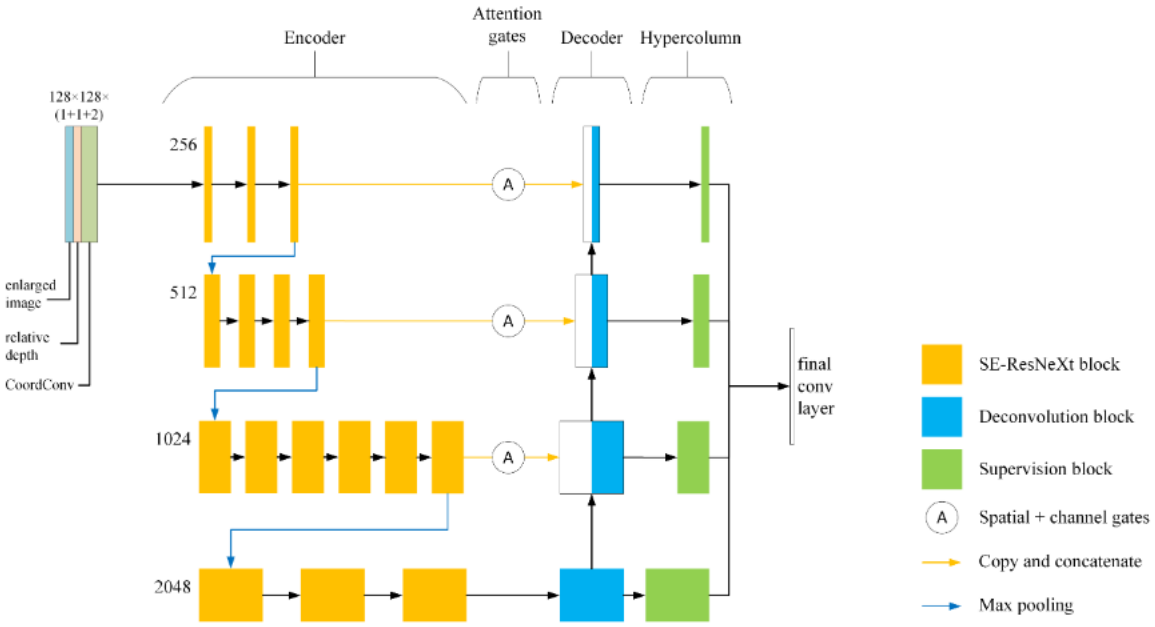
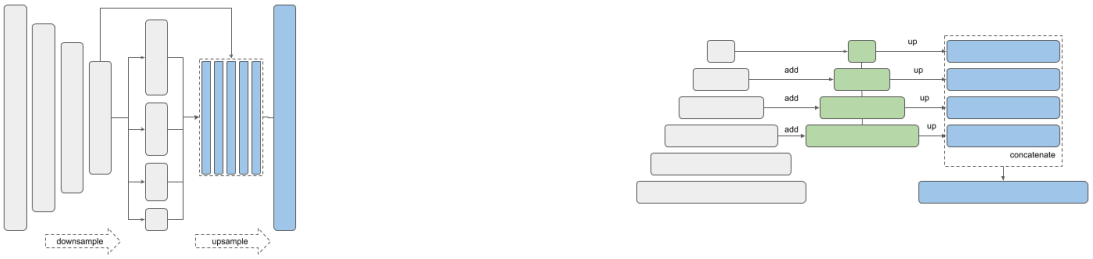


Figure 4.14: Model proposed in (Karchevskiy et al., 2018) : Input images are resized from 101x101 to 128x128 pixels with increased channels, processed through a pre-trained Resnet encoder with Squeeze-and-Excitation blocks, and decoded using deconvolution blocks with attention gates and deep supervision,implementing the Hypercolumn technique finally, all of the DSV outputs are concatenated to the final convolution layer [73].

(a) (b)

Figure 4.15: (a) Unet and (b) Linknet models. The image is taken from segmentation models repository on GitHub.



(a) (b)

Figure 4.16: (a) PSPnet and (b) FPNet models. The image is taken from segmentation models repository on GitHub.

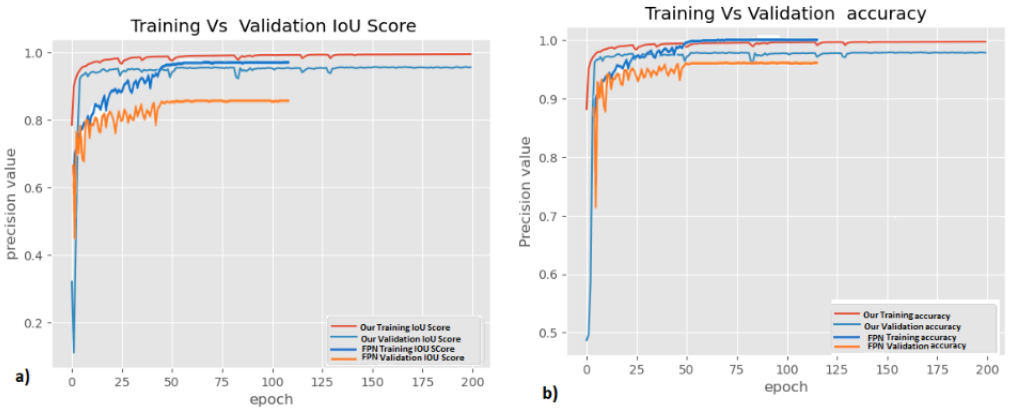


Figure 4.17: Comparison of Intersection over Union (IoU) (a) and accuracy (b) metrics between the FPNet model in (Milosavljević, 2020) and our own model.

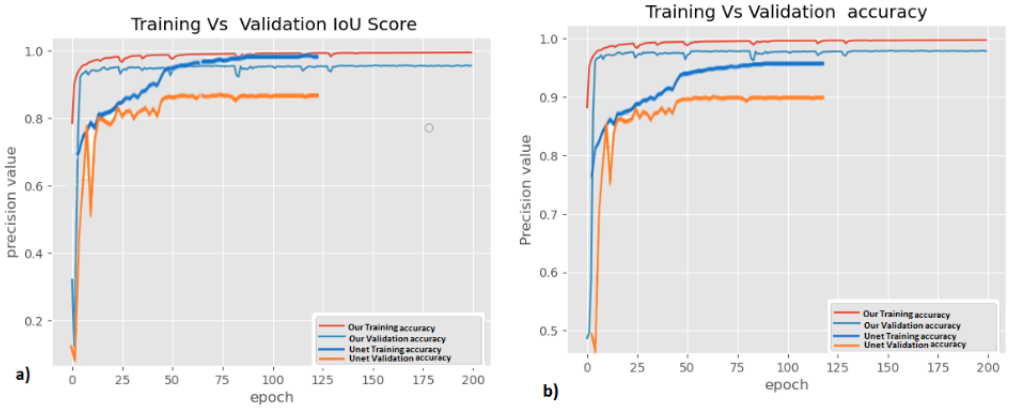


Figure 4.18: Comparison of Intersection over Union (IoU) (a) and accuracy (b) metrics between the Unet model in (Milosavljević, 2020) and our own model.

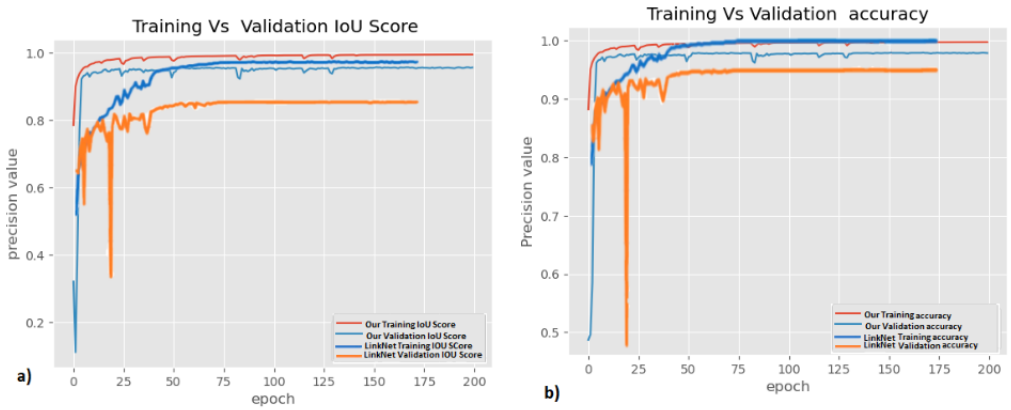


Figure 4.19: Comparison of Intersection over Union (IoU) (a) and accuracy (b) metrics between the LinkNet model proposed by (Milosavljević, 2020) and our own model.

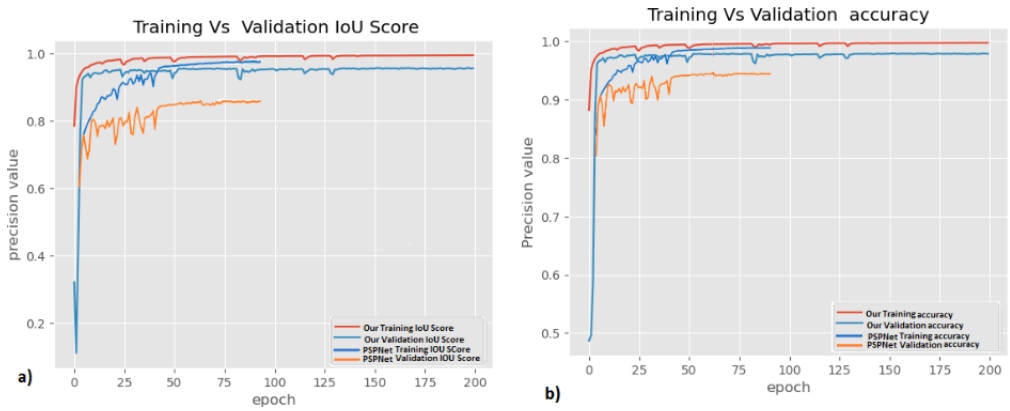


Figure 4.20: Comparison of Intersection over Union (IoU) (a) and accuracy (b) metrics between the PSPNet model proposed by (Milosavljević, 2020) and our own model.

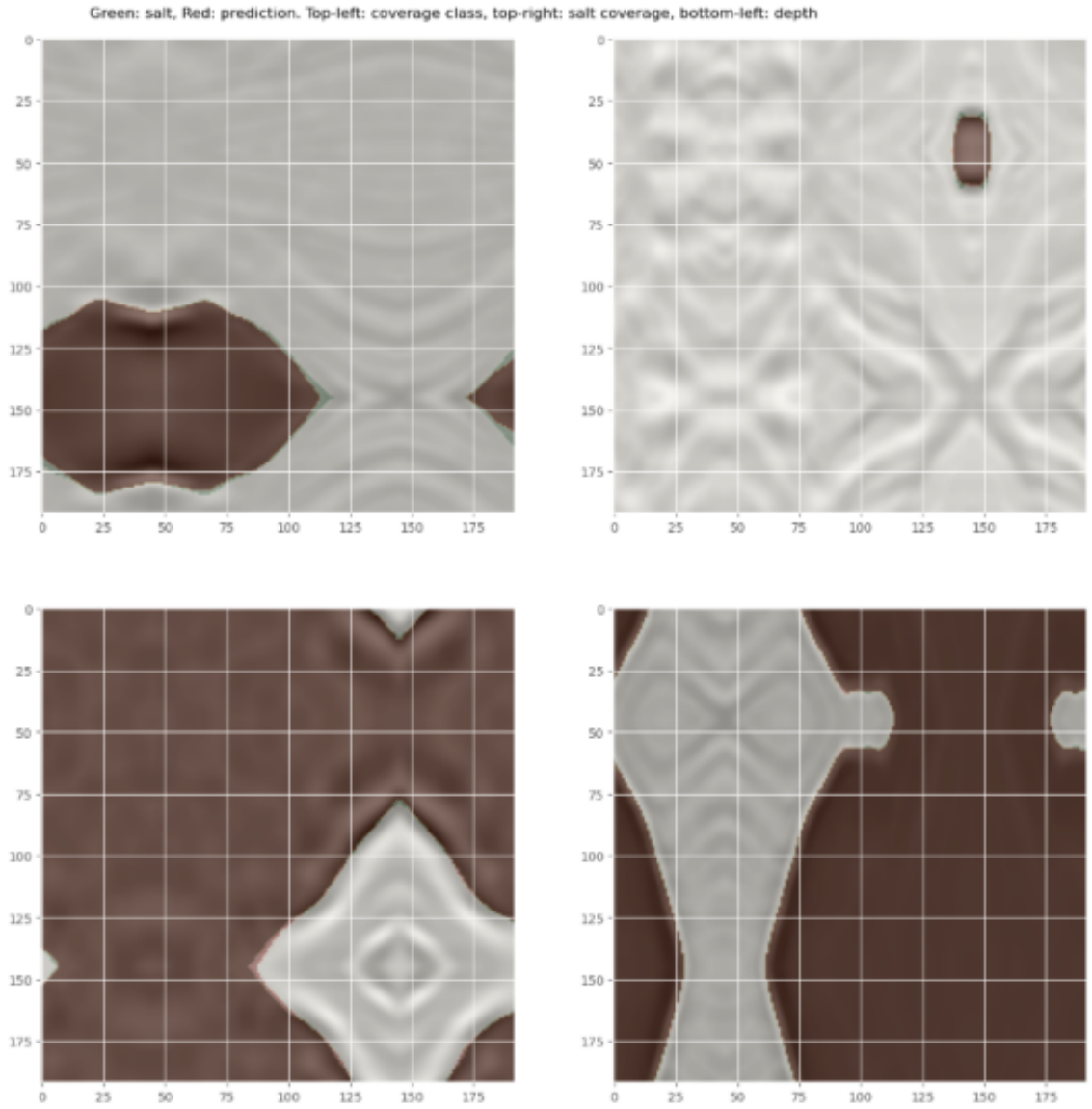


Figure 4.21: Seismic images, real filters, manual segmentations in green and predictions from our model in brown.

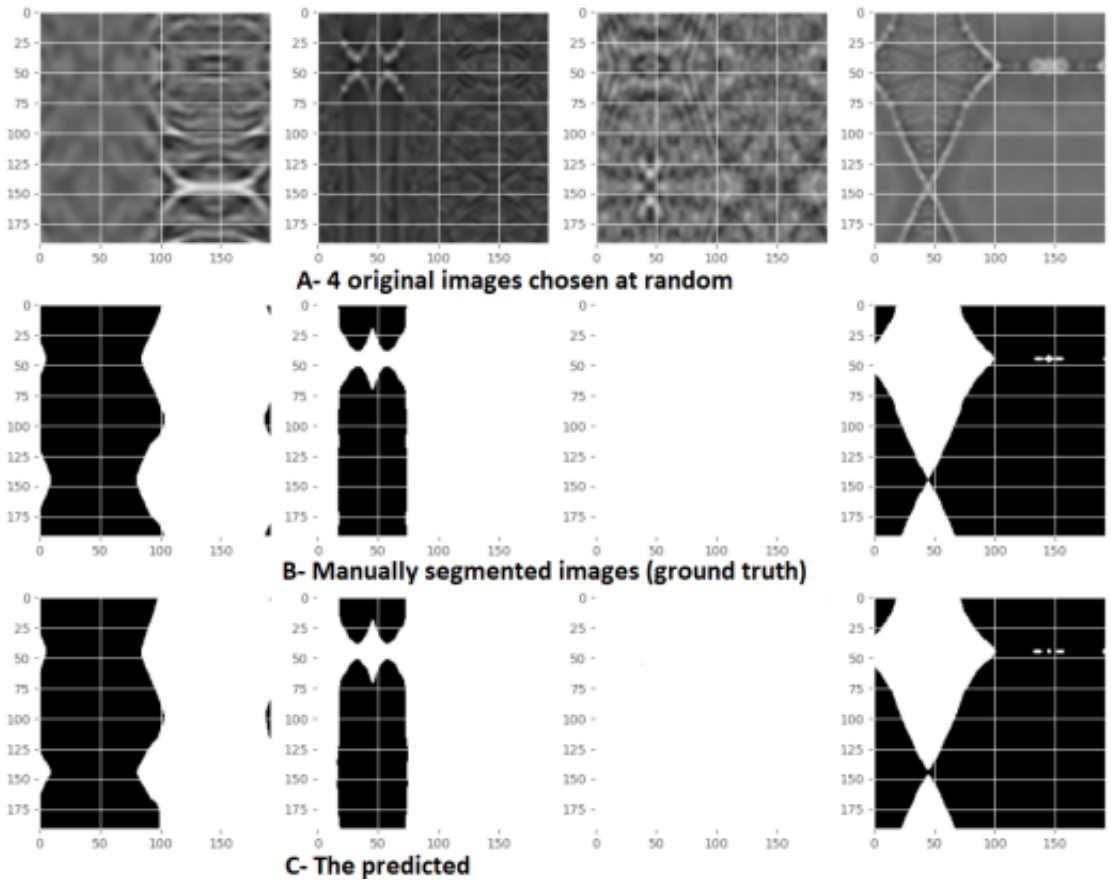


Figure 4.22: Seismic images, real filters, predicted filters.

4.7 Conclusion

In this section, we presented a novel method for segmenting 2D seismic data with the objective of identifying salt deposits, a critical task in subsurface imaging. Our proposed method leverages deep learning techniques, extensive preprocessing, and comprehensive evaluation to achieve state-of-the-art performance.

Through TGS Salt Identification Challenge dataset on Kaggle, we collected a diverse set of images capturing subsurface structures and corresponding binary masks delineating salt and sediment regions. We meticulously processed the data, including blurring to reduce noise, statistical sampling to address class imbalance, and mask processing to focus on pertinent regions.

Our model development involved designing a deep learning architecture tailored specifically for salt identification, integrating convolutional neural network layers with advanced techniques such as dropout and batch normalization. Through extensive experimentation and evaluation, we demonstrated the effectiveness of our approach in accurately segmenting salt deposits.

Qualitative evaluation revealed the robustness of our model in capturing intricate salt deposit boundaries, while quantitative evaluation using metrics such as Intersection over Union (IoU) and Dice Similarity Coefficient (DSC) showcased its superior performance compared to baseline methods and existing state-of-the-art approaches.

In conclusion, our proposed method represents a significant advancement in the field of subsurface imaging and salt identification. By achieving state-of-the-art performance, we contribute valuable insights and tools to aid geoscientists and researchers in accurately interpreting seismic data, ultimately facilitating better understanding and characterization of subsurface structures.

Conclusion and Perspectives

In this work, we delve into the realm of seismic image segmentation, focusing on the identification of salt deposits within subsurface structures. Seismic surveys are crucial in geophysical exploration, providing insights into the Earth's subsurface. Salt domes, due to their complex nature, pose significant challenges for accurate delineation in seismic data. We leverage Deep Learning approaches to develop a robust methodology for automatic segmentation of salt deposits.

We began with a literature review on Deep Learning techniques for semantic segmentation, distilling best practices in data processing and model architecture. Through rigorous experimentation, supported by platforms like Kaggle, we achieved a significant improvement in Intersection over Union (IOU) scores, reaching 96%. This was accomplished through strategic data preprocessing and advanced feature extraction.

We also utilized transformer models for capturing long-range dependencies, along with generative models and Liquid State Machines. This integration of diverse techniques led to a unified model architecture.

Future directions include refining model architectures, exploring advanced data augmentation, and combining different neural network paradigms. We plan to deploy our model as a web service, enhancing accessibility and scalability, thus contributing to the broader field of artificial intelligence.

Bibliography

[1] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2015, pp. 234–241.

[4] P. Gillhaus A. Horvarth. "Compilation of geological and geotechnical data of worldwide domal salt deposits and domal salt cavern fields". en. In: (2008).

[5] Martin P. A. Jackson and Michael R. Hudec. "Seismic Interpretation of Salt Structures". In: *Practical Applications of Salt Tectonics*. Cambridge University Press, Jan. 2017. DOI: 10.1017/CBO9781139923777.013.

[6] W.M. Telford, L.P. Geldart, and R.E. Sheriff. "Applied Geophysics". en. In: (1990).

[7] Richard D. Miller et al. "Seismic Investigation of a Surface Collapse Feature at Weeks Island Salt Dome, Louisiana". In: *AAPG Division of Environmental Geosciences Journal* 2.2 (1995), pp. 104–112.

[8] Mustafa Alfarhan, Mohamed Deriche, and Ahmed Maalej. "Robust Concurrent Detection of Salt Domes and Faults in Seismic Surveys Using an Improved UNet Architecture". In: *IEEE Access* (Dec. 2020), pp. 1–1. DOI: 10.1109/ACCESS.2020.3043973.

[9] N. Mondol and K. Bjørlykke. "Seismic Exploration". en. In: (Sept. 2010), pp. 375–402.

[11] Mario Bacic et al. "The Usefulness of Seismic Surveys for Geotechnical Engineering in Karst: Some Practical Examples". In: *Geosciences* 10.10 (Nov. 2020), p. 406. DOI: 10.3390/geosciences10100406.

[12] M. Andrei and Ionelia Panea. "The use of shallow seismic reflection surveys in environmental studies". In: *2nd International Geoscientist Student Conference*. University of Bucharest. Krakow, July 2011.

[13] Chris Gaffney, John Gater, and Susan Ovenden. *The use of Geophysical Techniques in Archaeological Evaluations*. Ed. by Alison Taylor. University of Reading, 2 Earley Gate, PO Box 239, Reading RG6 6AU: Institute of Field Archaeologists, 2002. ISBN: 0 948393 16 3.

[14] Javier Abreu-Torres. “Salt Tectonic Imaging at Crustal and Experimental Scales by Seismic Migration and Adjoint Method: Offshore Application Context”. English. ffNNT: 2022TOU30130ff, fftel-03813706ff. PhD thesis. Université Paul Sabatier - Toulouse III, 2022.

[15] Jingjing Zong et al. “Salt Densities and Velocities with Application to Gulf of Mexico Salt Domes”. In: SEG Annual Meeting Post-convention Workshop, 2015. New Orleans, LA, USA, Oct. 2015.

[17] Naamen Keskes, Abderrahim Boulanouar, and Olivier D. Faugeras. “Application of image analysis techniques to seismic data”. In: *IEEE International Conference on Acoustics, Speech, and Signal Processing*. 1982. URL: https://api.semanticscholar.org/CorpusID:206726992.

[18] Paolo Salvaneschi et al. “Towards a Knowledge-Based System for Seismic Assessment of Buildings”. In: *Computer-Aided Civil and Infrastructure Engineering* 5 (Nov. 2008), pp. 29–41. DOI: 10.1111/j.1467-8667.1990.tb00039.x.

[19] Adam Halpert, Robert Clapp, and Biondo Biondi. “Salt delineation via interpreter-guided 3D seismic image segmentation”. In: *Interpretation* 2 (Mar. 2014), T79–T88. DOI: 10.1190/INT-2013-0159.1.

[20] Asjad Amin et al. “A Novel Approach for Salt Dome Detection using A Dictionary-based Classifier”. In: Aug. 2015, pp. 1816–1820. DOI: 10.1190/segam2015-5925748.1.

[21] Amin Asjad and Mohamed Deriche. “A new approach for salt dome detection using a 3D multidirectional edge detector”. In: *Applied Geophysics* 12 (Sept. 2015), pp. 334–342. DOI: 10.1007/s11770-015-0512-2.

[22] Haibin Di and Ghassan Alregib. “Seismic Multi-attribute Classification for Salt Boundary Detection - A Comparison”. In: June 2017. DOI: 10.3997/2214-4609.201700919.

[23] Mikhail Karchevskiy, Insaf Ashrapov, and Leonid Kozinkin. “Automatic salt deposits segmentation: A deep learning approach”. In: *CoRR* abs/1812.01429 (2018). arXiv: 1812.01429. URL: http://arxiv.org/abs/1812.01429.

[24] Yauhen Babakhin, Artsiom Sanakoyeu, and Hirotoshi Kitamura. “Semi-Supervised Segmentation of Salt Bodies in Seismic Images using an Ensemble of Convolutional Neural Networks”. In: *CoRR* abs/1904.04445 (2019). arXiv: 1904.04445. URL: http://arxiv.org/abs/1904.04445.

[25] Aleksandar Milosavljević. “Identification of Salt Deposits on Seismic Images Using Deep Learning Method for Semantic Segmentation”. In: *ISPRS International Journal of Geo-Information* 9.1 (2020). ISSN: 2220-9964. DOI: 10.3390/ijgi9010024. URL: https://www.mdpi.com/2220-9964/9/1/24.

[26] Mustafa Alfarhan, Mohamed Deriche, and Ahmed Maalej. “Robust Concurrent Detection of Salt Domes and Faults in Seismic Surveys Using an Improved UNet Architecture”. In: *IEEE Access* PP (Dec. 2020), pp. 1–1. DOI: 10.1109/ACCESS.2020.3043973.

[28] N. Kühl et al. “Artificial intelligence and machine learning”. In: *Electronic Markets* 32 (2022), pp. 2235–2244. DOI: 10.1007/s12525-022-00598-0.

[29] Corien Prins, Haroon Sheikh, and Erik Schrijvers. *Mission AI: The new system technology*. English. Research for Policy. Germany: Springer, Jan. 2023. ISBN: 978-3-031-21447-9. DOI: 10.1007/978-3-031-21448-6.

[30] Iqbal H. Sarker. “Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions”. In: *SN Computer Science* 2.420 (2021). DOI: 10.1007/s42979-021-00815-1. URL: https://doi.org/10.1007/s42979-021-00815-1.

[31] Mohammed Mohammed, Muhammad Badruddin Khan, and Bader El-Den Bashier Mohammed. *Machine Learning: Algorithms and Applications*. CRC Press, 2016.

[32] Iqbal H. Sarker. “Machine Learning: Algorithms, Real-World Applications and Research Directions”. In: *SN Computer Science* 2.3 (Mar. 2021). DOI: 10.1007/s42979-021-00592-x.

[33] Jiawei Han, Jian Pei, and Micheline Kamber. *Data Mining: Concepts and Techniques*. Amsterdam: Elsevier, 2011.

[34] Iqbal H. Sarker et al. “Cybersecurity Data Science: An Overview from Machine Learning Perspective”. In: *Journal of Big Data* 7.1 (2020), pp. 1–29. DOI: 10.1186/s40537-020-00342-3.

[35] Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. “Reinforcement Learning: A Survey”. In: *Journal of Artificial Intelligence Research* 4 (1996), pp. 237–285.

[36] George H. John and Pat Langley. “Estimating Continuous Distributions in Bayesian Classifiers”. In: *Proceedings of the Eleventh conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., 1995, pp. 338–345.

[37] Saskia Le Cessie and Jan C. Van Houwelingen. “Ridge Estimators in Logistic Regression”. In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 41.1 (1992), pp. 191–201.

[38] Fabian Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[39] David W. Aha, Dennis Kibler, and Marc Albert. “Instance-based Learning Algorithms”. In: *Machine Learning* 6.1 (1991), pp. 37–66.

[40] S.S. Keerthi et al. “Improvements to Platt’s SMO Algorithm for SVM Classifier Design”. In: *Neural Computation* 13.3 (2001), pp. 637–649.

[41] Leo Breiman. “Random Forests”. In: *Machine Learning* 45.1 (2001), pp. 5–32.

[42] Iqbal H. Sarker, Paul Watters, and A. S. M. Kayes. “Effectiveness analysis of machine learning classification models for predicting personalized context-aware smartphone usage”. In: *Journal of Big Data* 6.1 (2019), pp. 1–28.

[43] Leo Breiman. “Bagging Predictors”. In: *Machine Learning* 24.2 (1996), pp. 123–140.

[44] Yali Amit and Donald Geman. “Shape Quantization and Recognition with Randomized Trees”. In: *Neural Computation* 9.7 (1997), pp. 1545–1588.

[45] James MacQueen et al. “Some Methods for Classification and Analysis of Multivariate Observations”. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*. Vol. 1. Oakland, CA, USA, 1967, pp. 281–297.

[46] Yi Xin et al. “Machine learning and deep learning methods for cybersecurity”. In: *IEEE Access* 6 (2018), pp. 35365–35381.

[47] Ilya Aizenberg, Natalia N. Aizenberg, and Joos P. Vandewalle. *Multi-Valued and Universal Binary Neurons: Theory, Learning and Applications*. Springer Science & Business Media, 2013.

[48] Christian Janiesch, Patrick Zschech, and Kai Heinrich. “Machine learning and deep learning”. In: *Electronic Markets* 31.3 (Sept. 2021), pp. 685–695.

[49] Nandakumar S.R. et al. “Building Brain-Inspired Computing Systems: Examining the Role of Nanoscale Devices”. In: *IEEE Nanotechnology Magazine* 12 (Sept. 2018), pp. 19–35. DOI: 10.1109/MNANO.2018.2845078.

[50] Frank Rosenblatt. “The perceptron: A probabilistic model for information storage and organization in the brain”. In: *Psychological Review* 65.6 (1958), pp. 386–408. DOI: 10.1037/h0042519.

[51] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. “Learning representations by back-propagating errors”. In: *Nature* 323 (1986), pp. 533–536. URL: https://api.semanticscholar.org/CorpusID:205001834.

[52] Y. Bengio and Yann Lecun. “Convolutional Networks for Images, Speech, and Time-Series”. In: (Nov. 1997).

[53] Y. Bengio, P. Simard, and P. Frasconi. “Learning long-term dependencies with gradient descent is difficult”. In: *IEEE Transactions on Neural Networks* 5.2 (1994), pp. 157–166. DOI: 10.1109/72.279181.

[54] Gideon Gbenga Oladipupo. *Research on the Concept of Liquid State Machine*. 2019. arXiv: 1910.03354 [cs.NE].

[55] Tala Talaei Khoei, Hadjar Ould Slimane, and Naima Kaabouch. “Deep learning: systematic review, models, challenges, and research directions”. In: *Neural Computing and Applications* 35 (Sept. 2023). DOI: 10.1007/s00521-023-08957-4.

[56] Laith Alzubaidi et al. “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions”. In: *Journal of Big Data* 8.1 (2021). Epub 2021 Mar 31, p. 53. DOI: 10.1186/s40537-021-00444-8.

[57] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[58] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[59] Jiazhi Liang. “Image classification based on RESNET”. In: *Journal of Physics: Conference Series* 1634 (Sept. 2020), p. 012110. DOI: 10.1088/1742-6596/1634/1/012110.

[60] Tobias Pohlen et al. *Full-Resolution Residual Networks for Semantic Segmentation in Street Scenes*. 2016. arXiv: 1611.08323 [cs.CV].

[61] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in Neural Information Processing Systems*. Vol. 30. 2017.

[62] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV].

[63] Ashwaq Alsayed et al. “A Systematic Literature Review on Using the Encoder-Decoder Models for Image Captioning in English and Arabic Languages”. In: *Applied Sciences* 13.19 (2023). ISSN: 2076-3417. DOI: 10.3390/app131910894. URL: https://www.mdpi.com/2076-3417/13/19/10894.

[64] Stefan Feuerriegel et al. “Generative AI”. In: (May 2023).

[65] Mina Razghandi et al. “Variational Autoencoder Generative Adversarial Network for Synthetic Data Generation in Smart Home”. In: *ICC 2022 - IEEE International Conference on Communications*. 2022, pp. 4781–4786. DOI: 10.1109/ICC45855.2022.9839249.

[66] Jeremy Jordan. *An overview of semantic image segmentation*. https://www.jeremyjordan.me/semantic-segmentation/. Accessed on February 28, 2024. 2018.

[71] Abhishek Chaurasia and Eugenio Culurciello. “LinkNet: Exploiting Encoder Representations for Efficient Semantic Segmentation”. In: *CoRR* abs/1707.03718 (2017). arXiv: 1707.03718 [cs.CV]. URL: http://arxiv.org/abs/1707.03718.

[72] Hengshuang Zhao et al. “Pyramid Scene Parsing Network”. In: *CoRR* abs/1612.01105 (2016). arXiv: 1612.01105 [cs.CV]. URL: http://arxiv.org/abs/1612.01105.

[73] Mikhail Karchevskiy, Insaf Ashrapov, and Leonid Kozinkin. *Automatic salt deposits segmentation: A deep learning approach*. 2018. arXiv: 1812.01429 [cs.LG].

Webography

[2] *Salt dome*. Accessed: 2023-11-02. URL: https://www.britannica.com/science/salt-dome.

[3] *What is a Salt Dome? Columns of salt that intrude through overlying sediment units.* Accessed: 2023-11-02. URL: https://shorturl.at/uvX78.

[10] Encyclopaedia Britannica. *Seismic Survey*. Year. URL: https://www.britannica.com/science/seismic-survey.

[16] Lameez Omarjee. *Seismic Surveys: Scientists Call for Tighter Laws, Greater Oversight*. Jan. 2022. URL: https://www.news24.com/fin24/economy/seismic-surveys-scientists-call-for-tighter-laws-greater-oversight-20220114.

[27] Halliburton. *DecisionSpace 365 Seismic Processing Suite*. Accessed: 2024-06-28. 2024. URL: https://www.halliburton.com/en/software/decisionspace-365-enterprise/decisionspace-365-subsurface/seismic-processing-suite.

[67] Coursera Staff. *What Is Kaggle and What Is It Used For?* Accessed on February 28, 2024. 2023. URL: https://www.coursera.org/articles/kaggle.

[68] Kaggle. *Kaggle: Your Home for Data Science*. Accessed: 2024-06-28. 2024. URL: https://www.kaggle.com/.

[69] Python Software Foundation. *Python Programming Language*. Accessed: 2024-06-28. 2024. URL: https://www.python.org/.

[70] TensorFlow. *TensorFlow: Open Source Machine Learning Platform*. Accessed: 2024-06-28. 2024. URL: https://www.tensorflow.org/.