KASDI MERBAH UNIVERSITY - OUARGLA

Faculty of New Technologies of Information and Telecommunication

Department of Computer Science and Information Technology

MASTER

Domain : Computer Science

Field : Artificial Intelligence and Data Science

Submitted by : OTHMANE BAZINE and OMAR RAI

Thesis :

A new deep learning-based method for data augmentation: application to Alzheimer dementia diagnosis.

Evaluation Date: June 18, 2023

Before the Jury :

Dr.Hocine Merbati	MCA	President	UKM Ouagrla
Dr.Khadra Bouanane	MCB	Examiner	UKM Ouagrla
Dr.Oussama Aiadi	MCA	Supervisor	UKM Ouagrla
Dr.Rachid Hedjame		Co-supervisor	Sultan Qaboos University Oman

Academic year: 2022/2023

Abstract

Deep learning has demonstrated remarkable capabilities in the field of medical imaging, one of the significant advantages of deep learning in medical imaging is its ability to automate and enhance various processes. For example, deep learning models can automatically detect abnormalities in medical images, and assist in tumor segmentation. However, the availability of medical imaging data can be a significant challenge in developing deep learning models. Collecting large amounts of high-quality annotated medical imaging data can be time-consuming, expensive, and sometimes limited due to privacy concerns and data-sharing regulations.

By utilizing few-shot learning, researchers and medical professionals can work with smaller datasets. The objective of this thesis is to advance the field of few-shot learning by focusing on the data perspective. In order to achieve this goal, we developed a novel architecture based on delta autoencoder, Our proposed architecture focuses on learning the transformation between different classes, enabling the generation of data that exhibits characteristics of multiple classes.

To evaluate the effectiveness of the proposed architecture, the OASIS 1 dataset was utilized as a benchmark. The experiments conducted on this dataset demonstrated significant improvements in few-shot learning performance compared to existing methods. Specifically, the achieved result of 82% accuracy showcases the remarkable capabilities of the developed architecture in enhancing the learning process.

The findings showcase the potential of the proposed method in improving generalization and adaptability when faced with limited labeled data. These results encourage further exploration and development of innovative approaches to enhance machine learning algorithms in few-shot learning scenarios.

Keywords: deep learning, medical imaging,few-shot learning, autoencoder, OASIS1

الخلاصة

أظهر التعلم العميق قدرات رائعة في مجال التصوير الطبي ، ومن المزايا المهمة للتعلم العميق في التصوير الطبي قدرته على أتمتة العمليات المختلفة وتعزيزها. على سبيل المثال ، يمكن لنماذج التعلم العميق اكتشاف التشوهات في الصور الطبية تلقائيًا ، والمساعدة في تجزئة الورم ، ومع ذلك ، يمكن أن يكون توفر بيانات التصوير الطبي تحديًا كبيرًا في تطوير نماذج التعلم العميق. جمع كميات كبيرة من بيانات التصوير الطبي المشروح عالية الجودة يمكن أن تستغرق وقتًا طويلاً ومكلفة ومحدودة في بعض الأحيان بسبب مخاوف الخصوصية ولوائح مشاركة البيانات. من خلال استخدام FEW SHOT LEARNING ، يمكن للباحثين والمهنيين الطبيين العمل مع مجموعات بيانات أصغر ، والهدف من هذه الأطروحة هو تطوير مجال FSL من خلال التركيز على منظور البيانات. من أجل تحقيق هذا الهدف ، قمنا بتطوير بنية جديدة تعتمد على DELTA AUTOENCODER ، تركز بنيتنا المقترحة على تعلم التحول بين الفئات المختلفة ، مما يتيح توليد البيانات التي تعرض خصائص فئات متعددة. لتقييم فعالية البنية المقترحة ، تم استخدام مجموعة البيانات OASIS 1 كمعيار. أظهرت التجارب التي أجريت على مجموعة البيانات هذه تحسينات كبيرة في الأداء مقارنة بالطرق الحالية. على وجه التحديد ، تُظهر النتيجة المحققة بدقة 82% لقدرات الرائعة للبنية في تعزيز عملية التعلم. تظهر النتائج إمكانات الطريقة المقترحة في تحسين التعميم والقدرة على التكيف عندما تواجه مع بيانات محددة . تشجع هذه النتائج على المزيد من الاستكشاف وتطوير الأساليب المبتكرة لتعزيز خوارزميات التعلم الآلي في سيناريوهات قلة البيانات .

االكلمات المفتاحية : التعلم العميق،التصوير الطبي ,few-shot learning, autoencoder, OASIS1

Acknowledgment

We would like to express our heartfelt gratitude to the individuals and institutions that have supported us throughout our journey in completing this master's thesis.

First of all, we want to express our sincere gratitude to Dr. Oussama Aiadi, our supervisor, for his consistent leadership, supportive encouragement, and priceless knowledge. Their perseverance, dedication, and astute criticism were crucial in determining the course and caliber of this research. We sincerely appreciate their guidance and the chances they have given us.

We also owe a debt of gratitude to the professors at Kasdi Merbah University's Department of Computer Science and Information Technology for their commitment to teaching and for providing us with the resources and expertise we needed to complete this thesis. Their passion for their individual professions has served as a constant source of inspiration.

We want to express our appreciation to the study participants who so kindly gave their time, knowledge, and perspectives. This research would not have been possible without their assistance. We appreciate their support and confidence.

We would want to express our gratitude for all of our families' and friends' support and encouragement along this trip. Our pillars of strength have been their faith in us, their comprehension of our situation, and their constant support. Their presence in our life has increased the significance of this academic work.

Last but not least, we would like to express our gratitude to all the researchers, academics, and writers whose work has contributed to the development of this thesis. We owe them for their ground-breaking work in the field and their invaluable contributions to publications and research.

We sincerely thank everyone who contributed to the successful completion of this thesis. Your assistance and contributions have greatly influenced our academic development and success.

Contents

1 General introduction **4**
- 1.1 Introduction . . . 4
- 1.2 Problematic . . . 4
- 1.3 Overview on the related works . . . 5
  - 1.3.1 Methods based on data . . . 5
  - 1.3.2 Methods based on model . . . 5
  - 1.3.3 Methods based on algorithms . . . 6
- 1.4 motivation . . . 6
- 1.5 Overview on proposed method . . . 7
- 1.6 Conclusion . . . 7

2 Work background **8**
- 2.1 Introduction . . . 8
- 2.2 Machine learning . . . 8
  - 2.2.1 Supervised machine learning . . . 9
  - 2.2.2 Unsupervised machine learning . . . 10
  - 2.2.3 limitations of traditional machine learning . . . 10
- 2.3 Deep learning . . . 11
  - 2.3.1 Basic deep learning concepts . . . 11
  - 2.3.2 Types of neural networks . . . 16
  - 2.3.3 The Impact of Hyperparameters on Deep Learning Performance . . . 19
- 2.4 Deep learning and medical images . . . 20
  - 2.4.1 Types of Medical Imaging . . . 20
  - 2.4.2 Alzheimer's dementia (AD) diagnosis . . . 22
- 2.5 Based works . . . 22
  - 2.5.1 Delta Auto Encoder . . . 22
  - 2.5.2 Convolution Neural Networks and Self-Attention Learners . . . 23
- 2.6 Conclusion . . . 23

3 Proposed methods **24**
- 3.1 Inroduction . . . 24
- 3.2 The proposed augmentation method . . . 24
  - 3.2.1 Architecture's Layers . . . 25
- 3.3 The used hyperparameters . . . 26
  - 3.3.1 "ReLU" activation function . . . 26
  - 3.3.2 Loss function . . . 26
  - 3.3.3 ADAM optimizer . . . 27
- 3.4 The proposed approach for the task of dementia identification . . . 27
  - 3.4.1 Augmentation model . . . 27
  - 3.4.2 classification model . . . 30

3.5 The proposed approach for the task of generating images based on artistic transformation . . . 31
3.6 Conclusion . . . 35

4 EXPERIMENTAL RESULTS 36
4.1 The dataset . . . 36
4.1.1 The first datasets (OASIS-1) . . . 36
4.1.2 The second dataset (artistic transformation) . . . 38
4.2 Our Framework . . . 39
4.3 Performance metrics . . . 39
4.3.1 Confusion matrix . . . 39
4.3.2 Receiver operating characteristic curve (ROC) . . . 41
4.4 Experiment 1 . . . 42
4.4.1 Result of our proposed augmentation method . . . 42
4.4.2 Result of our proposed classification method . . . 43
4.5 Result of the proposed approach for the task of generating images based on artistic transformation . . . 47
4.5.1 Transformations result of class A . . . 47
4.5.2 Transformations result of class A . . . 48
4.5.3 Model evaluation . . . 49
4.6 Discussion . . . 49
4.7 General Conclusion . . . 50

List of Figures

2.1 the difference between AI, ML, and DL. . . . 8
2.2 Relu and its derivative Graph. . . . 12
2.3 ResNet architecture. . . . 18
2.4 DenseNet architecture. . . . 18
2.5 U Net architecture. . . . 19
2.6 example of VGG architecture. . . . 19
2.7 example of X-ray. . . . 20
2.8 Example of Computed Tomography. . . . 21
2.9 Details for medical images. . . . 21
2.10 Ultrasound Imaging equipment. . . . 22
2.11 Delta autoencoder. . . . 23

3.1 Training phase. . . . 24
3.2 Our proposed augmentation architecture. . . . 27
3.3 the summary table of our proposed augmentation method. . . . 29
3.4 Our proposed classification architecture. . . . 30
3.5 Our second proposed augmentation method. . . . 32

4.1 An example dataset . . . 37
4.2 An instance of the dataset. . . . 38
4.3 An example of two rows of the dataset. . . . 39
4.4 Results evaluation using The Confusion matrix. . . . 40
4.5 example of the ROC curve . . . 41
4.6 diagram of first proposed augmentation method loss . . . 42
4.7 Some examples of proposed augmentation outputs. . . . 43
4.8 Loss and accuracy updating of classification without augmentation. . . . 44
4.9 Loss and accuracy updating of classification with handcrafted augmentation. . . . 44
4.10 Loss and accuracy updating of classification with first proposed augmentation. 45
4.11 ROC and confusion matrix of classification. . . . 46
4.12 Confusion matrix of classification after changing threshold . . . 47
4.13 Result of an image from class A. . . . 48
4.14 Result of an image from class B. . . . 48
4.15 Training and validation losses. . . . 49

Chapter 1

General introduction

1.1 Introduction

Machine learning (ML) is a subfield of artificial intelligence (AI) that plays a crucial role in the development of algorithms and models capable of learning from data and making predictions or decisions based on that learning. Traditional machine learning approaches involve training algorithms to identify patterns and relationships within the data. These algorithms heavily rely on a process called feature engineering, where domain experts manually select and extract relevant features from the data to be used as inputs for the models.

However, in recent years, deep learning (DL), a subset of machine learning, has gained significant popularity. DL utilizes artificial neural networks with multiple layers to learn and represent patterns and relationships within the data. Unlike traditional machine learning techniques, deep learning models can automatically learn features and representations directly from raw data, eliminating the need for manual feature engineering.

Deep learning has experienced remarkable advancements, driven by the availability of large datasets, powerful hardware, and innovative algorithmic techniques. These advancements have led to the emergence and widespread use of various deep learning architectures. Some of the most widely used deep learning architectures include convolutional neural networks (CNNs)[37] for image and video analysis, recurrent neural networks (RNNs)[46] for natural language processing and time-series analysis, and generative adversarial networks (GANs)[23] for creating synthetic data and generating realistic images.

The success of deep learning techniques has often surpassed human-level performance in many tasks. However, one limitation of deep learning models is their reliance on large amounts of data to learn new categories. In contrast, humans can learn from just a few or even one example. To address this challenge, researchers have developed few-shot learning algorithms that enable models to learn from only a small number of examples. Few-shot learning becomes particularly valuable when obtaining extensive labeled data is difficult or expensive.

In this thesis, we focus our attention on developing a deep learning-based few-shot learning framework. We will also examine the practical applications of few-shot learning (FSL) in various task. Finally, we will discuss the future directions and challenges in the field of few-shot learning and provide recommendations for future research.

1.2 Problematic

Despite the recent advancements in deep learning, the requirement for a large amount of labeled data remains a significant problem for many real-world applications. Few-shot learning is a promising approach to address the limitation of traditional deep learning. However, there are

still several challenges and limitations that need to be addressed. In cases where available data are limited or expensive to obtain, we resort to augmenting the data set.

Traditional data augmentation techniques have been widely used in machine learning to artificially expand training datasets and improve model performance[16]. These techniques involve applying various transformations to the existing data, such as rotations, translations and flips, to create new training examples. While these methods have shown effectiveness in certain scenarios, they also have inherent limitations and can fail to achieve the desired improvements in model generalization.

manually designing traditional augmentation strategies can have adverse effects on model performance. The reason is that manually designing augmentation techniques requires domain expertise and prior knowledge about the dataset and the specific problem being addressed. However, it is challenging to determine the optimal set of transformations and their parameters that will enhance generalization and improve model performance[16].

1.3 Overview on the related works

Few-shot learning is a challenging machine learning problem that involves training a model to recognize new classes using only a few training examples[51]. Traditional machine learning algorithms frequently struggle with this task because learning meaningful representations requires large amounts of data. However, several approaches to few-shot learning have been proposed, which involve leveraging prior knowledge from related tasks to learn generalizable patterns that can be applied to new tasks with limited data, let us provide an overview of these methods :

1.3.1 Methods based on data

These strategies extend the dataset and enhance the number of samples by utilizing prior information. The enriched data may then be used with standard machine-learning models and methods. Manual data augmentation, such as flipping, shearing, scaling, reflection, cropping, and rotation, is commonly employed as pre-processing [41][52], but it cannot entirely solve the few-shot learning (FSL) problem, from these methods augment data from the same data, This method adds to the data collection by changing one sample into many samples with variances for e.g in this paper [35] the model learns a set of geometric transformations from a similar class by iteratively aligning each sample with the other samples, in[40] The learned transformation from different intraclass (x_i, y_i) to form a large data set, which can then be learned by standard machine learning methods and generate new samples by adding the learned variations to x_i to y_i. and there's the method of augmenting data from a Weakly Labeled or Unlabeled data set For example, in photos taken by a surveillance camera, there are people, cars, and roads but none of them are labeled. This method augments the data set by labeling the similar and unlabeled big data set like there are people, cars, and roads but none of them are labeled, and also there is augmented data from similar data. for e.g in [19] generative adversarial network has two generators designed to generate indistinguishable synthetic x˜ aggregated from a data set of many samples. This strategy expands the data set by collecting and modifying input-output pairs from similar and larger data sets[51].

1.3.2 Methods based on model

These strategies improve accuracy by using prior knowledge and Narrowing and constraining the hypotheses space to approximate the ground-truth hypothesis that presents the best model, then minimizing the risk of overfitting is reduced by prior knowledge is used [40][17][19].

Of these strategies, there is multi-task learning, we can assume there are many related tasks $T_1, ..., T_C$, and for each task, there is corresponding data and some of them have very few samples while some have a larger number of samples. Among these tasks, we regard the few-shot tasks as target tasks, and the rest as source tasks, As these tasks are jointly learned, the parameter learned for task Tc is constrained by the other tasks, we can with this strategy solve the few-shot problem with multi-task learning.[25, 33]

1.3.3 Methods based on algorithms

The algorithm is a method for searching in the hypotheses space for the optimal hypothesis' hyper-parameter θ ; the approach attempts to obtain better parameters. When there is a lot of supervised data, there are enough training samples to update θ and determine a suitable step-size using cross-validation. This approach influences how θ is produced by either supplying a well-initialized parameter θ or directly learning an optimizer to output search steps.

For example, for the popular stochastic gradient descent (SGD) that is the optimization function :

$$\theta_t = \theta_{t-1} - \alpha_t * \nabla\theta_{t-1} \tag{1.1}$$

where α_t is the stepsize, with θ initialized at θ_0.

Rich supervised datasets have enough to update θ and cross-validate to determine the suitable stepsize α_t. The offered few-shot supervised dataset is not big enough for few-shot learning, though. The techniques in this section make use of prior knowledge to alter how is obtained, either by giving a good initialization for the parameter θ_0 or by directly training an optimizer to produce search steps.

1.4 motivation

Handly data augmentation, in general, can be particularly advantageous for machine learning models since it increases the diversity and variability of the training data, which can improve the model's accuracy and robustness. However, in other circumstances, data augmentation manually may hurt the performance of the model or you should be an expert to know how to augment the data [16].

Data Augmentation refers to a collection of approaches for increasing the amount and quality of training datasets so that stronger Deep Learning models may be generated using them like the paper [40] that our approach is based on it, which uses data augmentation to address a few-shot problem, and we aim to improve the approach based on this paper despite some of its drawbacks. the images were converted to vectors using vgg16, but since vgg16 was created for classification, the feature vector extracted from it belonging to the same class would be similar. The work was done with a basic autoencoder. Despite the fact that the images are different and there is a problem with reconstructing images that we cannot use in this situation, we are still compelled to work exclusively with feature vectors throughout.

Additionally, since its architecture lacks a skip connection, we are unable to delve further into it without risking a vanishing gradient[27]. This makes it challenging to reconstruct the desired results, particularly when dealing with medical data that demands accuracy. We can then classify dementia using a convolution neural network (CNN) and compare the results to some papers like[14].

1.5 Overview on proposed method

From what we said earlier we made some changes:

- dealing with the images themselves instead of the feature vectors, and of course, in this situation, we should replace the basic autoencoder with a convolution autoencoder, because the results in the basic autoencoder are less effective in capturing the visual patterns and dependencies crucial in image analysis, On the other hand, convolution captures the spatial relationships and -patterns in images more effectively, making them the preferred choice for image classification, object detection, and other image-related tasks[27]. Also, his architecture consists of 3 inputs as feature vectors, two inputs belong to the same class, and the last input belongs to another class and enters directly into decoding, in that case since we are working on images directly, we suggested working with 2 encoders and 1 decoder.

- And because medical images need accuracy, it was better to add a skip connection, In particular, skip connections can help with many limitations of sample autoencoder, such as Better feature extraction, especially with complex medical images[28], It also allows us to add layers and create deeper architecture[27], And of course, it has an effect on Improved accuracy, with add skip connection between the corresponding encoder and decoder layers, our architecture changed from convolutional autoencoder to convolution Unet.

- And the architecture was applied to the brain Cross-sectional MRI Data in Young, Middle Aged, Nondemented and Demented older Adults(oasis-1), for detect dementia we augment the dataset and create special classification architecture (CNN) with 2 inputs (images and features vector), where the images are inserted into the convolutions layers and features vectors are concatenated after the convolutions, and the accuracy was increased to 82%, as the most prominent results reach 75.6% in [14].

1.6 Conclusion

Few-shot learning is a promising approach to address labeled data limitations in deep learning, it uses prior knowledge from related tasks to learn generalizable patterns for limited data, using data-based, model-based, and algorithm-based methods. In our work, we focus on data-based. however, challenges remain. Traditional data augmentation techniques, like rotations and translations, can have inherent limitations and may not achieve desired improvements in generalization.

In this thesis we improved an approach based on a few-shot problem using a convolution neural network to classify dementia, we replaced basic autoencoders with convolution autoencoders for image analysis, as we capture spatial relationships more effectively. The architecture consists of 3 inputs, 2 encoders, and 1 decoder. A skip connection was added for better feature extraction and deeper architecture. The architecture was applied to brain MRI data to detect dementia, increasing accuracy to 82%.

Chapter 2

Work background

2.1 Introduction

AI is the replication of human intelligence in machines that are programmed to do activities that normally require human intellect, such as perception, reasoning, learning, decision-making, and natural language processing. AI technologies are intended to analyze large volumes of data, identify patterns, and then make predictions or suggestions based on that analysis.

Natural language processing, picture and audio recognition, robotics, self-driving cars, and virtual assistants are just a few of the many uses of AI. AI has the ability to transform industries such as healthcare, banking, transportation, and manufacturing by increasing efficiency, lowering costs, and offering customers individualized experiences.

AI is classified into numerous categories, including rule-based AI, machine learning, deep learning, and neural networks. Machine learning methods enable systems to learn from data and improve over time, whereas rule-based AI systems use a set of established rules and decision trees. Deep learning simulates the human brain with neural networks and can recognize complicated patterns in data.

Overall, AI is an exciting field that has the potential to transform the way we live and work. As AI technologies continue to evolve, it is critical to understand their capabilities and limitations to harness their full potential while addressing ethical and societal issues.

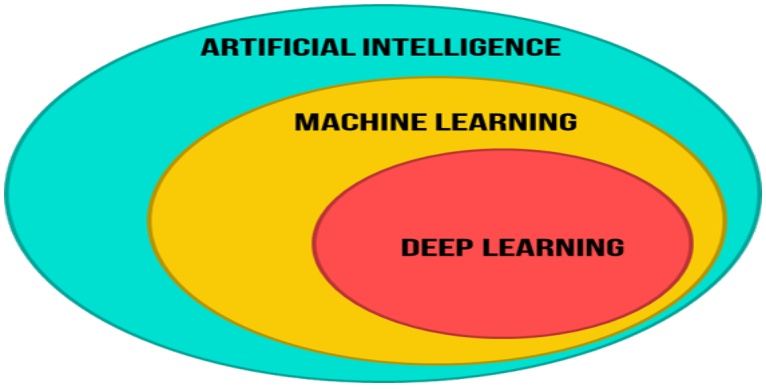


Figure 2.1: the difference between AI, ML, and DL.
[5]

2.2 Machine learning

Machine learning is a rapidly growing field of artificial intelligence that has seen significant advancements in recent years, It involves the development of algorithms and statistical models

that enable computers to analyze and make predictions from data, without being explicitly programmed. There are three main types of machine learning: supervised learning, unsupervised learning, and reinforcement[36].

2.2.1 Supervised machine learning

Supervised learning is a sort of machine learning that involves training an algorithm on a labeled dataset. The dataset is made up of input data and output labels, which are used to train the algorithm to produce predictions or classifications. The goal of supervised learning is to build a mapping function that can predict output labels reliably for new unknown input data. This is accomplished by the algorithm examining the patterns in the input-output pairings and modifying its internal parameters to minimize the discrepancy between predicted and actual outputs. Supervised learning can be further divided into two main types: regression and classification.[10]

2.2.1.1 Regression

The output variable in the regression is continuous, and the algorithm is taught to predict a numerical value. In a housing price prediction task, for example, the input variables could include the size of the house, the number of bedrooms, and the location, and the output variable would be the price. Based on the input features, the algorithm would learn to forecast the price of a new house.[10]

2.2.1.2 Classification

The output variable in classification is categorical, and the algorithm is taught to predict a label or category. In a spam email detection task, for example, the input variables might be the email content, sender information, and subject line, while the output variable might be whether or not the email is spam. Based on the input features, the algorithm would learn to categorize fresh emails as spam or not spam.[10]

2.2.1.3 Algorithms

Here are some commonly used algorithms in supervised machine learning:

1. **k-Nearest Neighbors (k-NN) :** K-NN is a machine learning technique for classification and regression tasks, determining k nearest neighbors based on distance measures, with sensitivity to metric and k value.[10]

2. **Decision Trees:** A decision tree is a classification model using recursive data splitting and output values.[10]

3. **Random Forests:** Random forests improve ensemble learning by combining multiple decision trees, reducing overfitting, and averaging individual tree predictions.[10]

4. **Support vector machines (SVMs):** Superlevel classification algorithm finds super levels for class separation, reduces mean squared error, and uses kernel function for data transformation.[10]

5. **Linear Regression:** Linear regression is a type of supervised learning algorithm that is used for regression tasks. The algorithm finds the line (or hyperplane in higher dimensions) that best fits the training data in the least squares sense. [10]

6. **Logistic Regression:** A form of supervised learning method used for binary classification tasks is logistic regression. The procedure determines the line (or hyperplane in higher dimensions) that best separates the two classes, transforming the output if necessary with a logistic function.[10]

2.2.2 Unsupervised machine learning

Unsupervised learning is a sort of machine learning in which the model is trained on unlabeled data, i.e. data with no target variables or labels. Instead, the idea is to find the underlying structure or patterns in the data without knowing what those structures or patterns are, There are several types of unsupervised learning techniques, including:

2.2.2.1 Clustering

This involves grouping similar data points together into clusters, where the similarity is measured based on some distance metric. Clustering algorithms, such as k-means and hierarchical clustering, can be used to discover natural groupings in the data. [34].

2.2.2.2 Dimensionality Reduction

This entails reducing the data's dimensionality by translating it into a lower-dimensional space while maintaining as much of the original information as possible. This can be used to visualize the data, compress it, or prepare it for further investigation. Dimensionality reduction techniques such as Principal Component Analysis (PCA) and Independent Component Analysis (ICA) are routinely utilized.[44].

2.2.2.3 Algorithms

1. **K-Means :**K-Means is a powerful clustering algorithm for image segmentation, text mining, and bioinformatics, dividing large datasets into K clusters based on similarity..[10]

2. **Gaussian Mixture Model (GMM):** The Gaussian Mixture Model (GMM) is a probabilistic model used in image processing, natural language processing, and speech recognition for clustering and density estimation.[10]

3. **Singular Value Decomposition (SVD):** (SVD) divides a matrix into orthogonal U, and V matrices, reducing dimensionality, compressing data, and approximating singular values. It is widely used in image processing, data analysis, and recommendation systems.[10]

4. **Principal Component Analysis (PCA):** PCA is a dimensionality reduction technique that converts high-dimensional data into lower-dimensional representations while retaining critical information, used in image, pattern, and signal processing .[10]

2.2.3 limitations of traditional machine learning

Despite its potential, traditional machine learning has its limitations. One significant limitation is the need for feature engineering, which involves hand-crafting features to be used in the learning algorithm[12], Feature engineering can be time-consuming, and the quality of the features is critical to the success of the learning algorithm. Additionally, traditional machine learning algorithms may struggle with complex patterns and relationships in the data, requiring more advanced techniques such as deep learning.

2.3 Deep learning

Deep learning is a subset of machine learning that utilizes artificial neural networks with multiple layers to model and solve complex problems. It is important in artificial intelligence and machine learning because it has revolutionized many applications in various fields, such as computer vision, natural language processing, speech recognition, and robotics. It has enabled its ability to automatically extract features from raw data and surpass human-level accuracy in many tasks. Deep learning has also enabled the development of intelligent systems that can learn from their environment and adapt to changing conditions, allowing for more efficient and personalized systems such as recommendation engines, virtual assistants, and self-driving cars. As a result, deep learning has become a crucial component in the Development of Artificial Intelligence and Machine Learning, paving the way for many exciting new applications and innovations.

Deep learning has a long history dating back to the 1940s, but it wasn't until the early 2000s that significant progress was made. Key milestones include the development of deep belief networks in 2006 and the success of deep learning in image recognition competitions in 2012. Breakthroughs in recent years have led to advancements in natural language processing, speech recognition, and robotics. Today, deep learning is a fundamental tool in artificial intelligence and machine learning.

Deep learning is a rapidly growing field in artificial intelligence and machine learning, and there is a high demand for professionals with deep learning skills. Studying deep learning can lead to career opportunities in research, development, and implementation of deep learning systems, as well as the tools and knowledge necessary to tackle complex problems and create innovative solutions. It is an interdisciplinary field that draws on mathematics, statistics, computer science, and neuroscience, and can provide a broad range of knowledge and skills that can be applied to many different fields and industries.[22]

2.3.1 Basic deep learning concepts

Deep learning is a subset of machine learning that uses artificial neural networks to model and solve complex problems. Deep learning has achieved remarkable success in various fields, including computer vision, natural language processing, and robotics. In this report, we will discuss the basic building blocks of deep learning, including artificial neural networks, activation functions, backpropagation, and optimization algorithms.

2.3.1.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) are machine learning algorithms that are designed to mimic the structure and function of the human brain. ANNs are made up of linked neurons that are grouped into layers. The input layer takes input data, while the output layer generates output predictions. The intermediate layers, known as hidden layers, convert the input data into a usable representation for the output layer. Weighted connections connect each neuron to the neurons in the subsequent layers, and the strength of these connections is modified during training to increase the network's accuracy.[22]

The simplest form of a neuron is a perceptron, and for ANN each neuron is considered a perception that takes a vector of input values X and produces an output Y using a weighted sum and an activation function: $Y = f(w \cdot X + b)$.[22]

ANNs have several limitations, such as requiring a large amount of labeled data to learn and generalize effectively, overfitting the training set, and being computationally costly. They can also overfit the training data, memorize training examples, and use regularization techniques such as dropout and weight decay to prevent overfitting. Additionally, they are often referred to

as black box models, making it difficult to detect and resolve problems that may develop during training or testing, and they lack explainability, which can be difficult in certain contexts.[22]

2.3.1.2 Activation functions

Activation functions are mathematical functions that are applied to the output of each neuron in a neural network, The purpose of activation functions is to introduce non-linearity into the network. There are several types of activation functions that are commonly used in deep learning, including :

1. **Sigmoid:** The sigmoid function is a commonly used activation function in artificial neural networks. It is a mathematical function that maps any input value to a value between 0 and 1, The sigmoid function is defined as:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

   However, the sigmoid function also has some limitations. For example, when the input values are too large or too small, the output values become saturated, meaning that they are close to 0 or 1. This can lead to the problem of vanishing gradients [49], which makes it difficult to train deep neural networks.

2. **The tanh function :** The tanh function is similar to the sigmoid function, but outputs values between -1 and 1. The tanh function is defined as:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.2}$$

   The tanh function is commonly used in the hidden layers of neural networks.

3. **ReLU:** The rectified linear unit (ReLU) is one of the most widely used activation functions. the rectified linear unit (ReLU) activation function operates by thresholding values at zero[9]. It can be defined as $f(x) = \max(0, x)$ where x represents the input to the function. In simple terms, ReLU outputs 0 when the input is negative $x < 0$, effectively "turning off" the neuron, and outputs a linear function when the input is non-negative $x \geq 0$. The derivative of ReLU is defined as follows:

$$f'(x) = \begin{cases} 0, & \text{if } x < 0 \\ 1, & \text{if } x \geq 0 \end{cases} \tag{2.3}$$

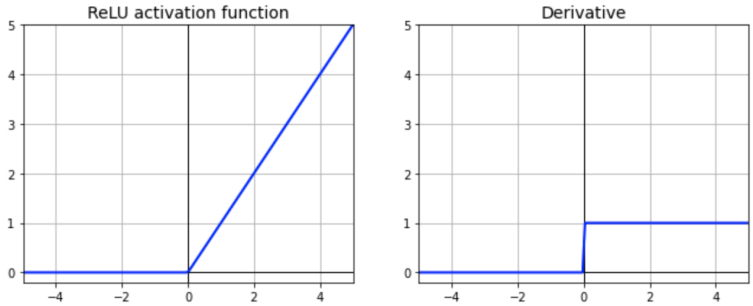


Figure 2.2: Relu and its derivative Graph.
[3]

2.3.1.3 Backpropagation

The fundamental goal of backpropagation is to adjust the weights of the network by propagating the error back through the layers of the network and using this information to update the weights in a way that reduces the error. The basic equation for backpropagation can be written as follows [39]:

1. Compute the output of the neural network denoted as y-out, given the input vector x and the current weights W:
$$y_{out} = f(Wx) \tag{2.4}$$
where f is the activation function.

2. Calculate the error between the predicted output and the target output y:
$$E = \frac{1}{2}\sum_{i=1}^{n}(y_i - y_{out,i})^2 \tag{2.5}$$
where n is the number of output neurons.

3. Compute the gradient of the error with respect to the weights using the chain rule of differentiation:
$$\frac{\partial E}{\partial w_{i,j}} = (y_{out,j} - y_j)f'(h_j)y_i \tag{2.6}$$
where $w_{i,j}$ is the weight between neuron i and neuron j, $y_{out,j}$ is the output of neuron j, y_j is the target output for neuron j, $f'(h_j)$ is the derivative of the activation function f evaluated at the input to neuron j, and y_i is the input to neuron i.

4. Update the weights using an optimization algorithm:
$$w_{i,j} = w_{i,j} - \alpha\frac{\partial E}{\partial w_{i,j}}. \tag{2.7}$$

2.3.1.4 Optimization algorithms

Deep learning relies on optimization algorithms to update the weights of artificial neural networks during training and minimize the loss function. There are various optimization algorithms for training neural networks, each with unique advantages and disadvantages. The choice of optimization algorithm depends on the characteristics of the dataset, the architecture of the neural network, and the available computational resources.[22]

1. **Stochastic Gradient Descent (SGD):** The weights in this optimization approach are adjusted after each individual training example. The SGD update rule has the following formula:
$$W_i = W - \alpha * \Delta L(W_{i-1}) \tag{2.8}$$
where:
   - W_i is the new weight matrix.
   - W_{i-1} is the old weight matrix.
   - α is the learning rate.
   - $L(W)$ is the loss function.

- $\Delta L(W)$ is the gradient of the loss function with respect to the weights.

SGD has several advantages, including the following: it is simple and easy to implement, it requires less memory and processing resources than other optimization techniques, and it works well with small datasets. However, it has some drawbacks: It can be sluggish to converge and get stuck in local minima. It may also require a low learning rate, which might slow down convergence.[22]

2. **Momentum:** Momentum is a modification of SGD that adds a momentum term to the weight update. The momentum term helps the weights to keep moving in the same direction when the gradient changes direction, which can help the algorithm converge faster. The formula for the momentum update rule is :

$$V_i = \beta * V_{i-1} + (1 - \beta) * \Delta L(W_{i-1}) \tag{2.9}$$

$$W_i = W_{i-1} - \alpha * V_i \tag{2.10}$$

where:

- β is the momentum coefficient.
- V is the velocity of the weight update.
- $\Delta L(W)$ is the gradient of the loss function with respect to the weights.

The momentum benefits include accelerating convergence and preventing local minima, dampening oscillations in weight updates, and supporting noisy or sparse gradients. Additionally, if the momentum coefficient is too high, it may overshoot the ideal weights, and if the learning rate is too low, it may converge slowly.[22]

3. **Adagrad :** Adagrad is an adaptive learning rate optimization algorithm that modifies the learning rate according to the historical gradients of each weight. The goal is to enhance the learning rate for slowly changing weights while decreasing the learning rate for quickly changing weights. The Adagrad update rule is calculated as follows:

$$G_i = G_{i-1} + (\Delta L(W_{i-1}))^2 \tag{2.11}$$

$$W_i = W_{i-1} - (\frac{\alpha}{\sqrt{G_i + \epsilon}}) * \Delta L(W_{i-1}) \tag{2.12}$$

where:

- G is the sum of the squared gradients for each weight.
- ϵ is a small value to prevent division by zero.

The benefits of Adagrad adjust the learning rate for each weight based on its past gradients, which can be effective for problems with many parameters and sparse or noisy gradients. and its Drawbacks Can eventually bring the learning rate to zero and slow convergence by accumulating the squared gradients over time. May not be effective for problems involving non-stationary or non-convex objective functions.[22]

4. **RMSprop:** RMSprop is a modification of Adagrad that addresses the problem of the learning rate decreasing over time. It does this by using a moving average of the squared gradients to adjust the learning rate. The formula for the RMSprop update rule is:

$$G_i = \beta * G_{i-1} + (1 - \beta)(\Delta L(W_{i-1}))^2 \tag{2.13}$$

$$W_i = W_{i-1} - (\frac{\alpha}{\sqrt{G_i + \epsilon}}) * \Delta L(W_{i-1}) \tag{2.14}$$

where:

- G is the moving average of the squared gradients.
- β is the moving average coefficient.

RMSprop has the advantages of being able to adaptively change the learning rate for each weight based on its history gradients and working well for non-convex or non-stationary objective functions. Its drawbacks include the possibility of slow convergence if the learning rate is too low and the potential need for careful adjustment of the moving average coefficient and learning rate[22].

5. **Adam**: Adam is a combination of momentum and RMSprop, which has become one of the most popular optimization algorithms in deep learning. The formula for the Adam update rule is:

$$m_i = \beta_1 * m_{i-1} + (1 - \beta_1)(\Delta L(W_{i-1})) \tag{2.15}$$

$$V_i = \beta_2 * V_{i-1} + (1 - \beta_2)(\Delta L(W_{i-1}))^2 \tag{2.16}$$

$$W_i = W_{i-1} - (\frac{\alpha}{\sqrt{V_i + \epsilon}}) * m_i \tag{2.17}$$

where :

- β_1 and β_2 are the exponential decay rates for the moving averages.
- m is the moving average of the gradient.
- V is the moving average of the squared gradient.

Adam's advantages include combining the advantages of momentum and RMSprop, has the ability to adaptively change the learning rate for each weight based on its previous gradients and momentum, and is effective for a variety of applications and datasets. Additionally, for some problems or datasets, it may be necessary to carefully tune the decay rates and learning rates, and it can converge slowly.[22]

2.3.1.5 Performance enhancers

1. **Dropout:** A portion of the neurons in a layer are randomly removed during training using the common regularization approach known as "dropout." By requiring the network to acquire more robust and generalizable characteristics, this can help prevent the network from overfitting. When many neural networks with various subsets of neurons are trained concurrently and their outputs are averaged at test time, this process is known as dropout and can be compared to ensemble learning. The dropout rate is a hyperparameter that regulates the fraction of neurons that are dropped out, and it is commonly applied after a layer's activation function. The dropout rate is typically set at 0.5, but this can be altered depending on the issue and dataset.[22]

2. **L1/L2 regularization:** A penalty term is added to the loss function using the L1 and L2 regularization techniques, which promotes the network to have smaller weight values. L1 regularization, also referred to as Lasso regularization, increases the loss function's absolute weights total, whereas L2 regularization, also referred to as Ridge regularization, increases the loss function's squared weights total. The trade-off between limiting the training loss and minimizing the magnitude of the weights is controlled by the regularization strength hyperparameter λ. While L2 regularization tends to produce

weight vectors with small non-zero values, L1 regularization tends to produce sparse weight vectors with many weights that are exactly zero.[22]

3. **Batch normalization:** The activations of each layer in the network are normalized using the batch normalization technique. By doing so, the training process can be stabilized and local minima can be avoided by the network. Batch normalization involves taking the mean of the activations in each mini-batch and dividing it by the standard deviation. The network can then learn to scale and shift parameters γ and β for each layer, allowing it to modify the normalized activations as necessary. Typically, batch normalization is used after a layer's linear transformation but before the activation function.[22]

4. **Skip connections :** Skip connections can be thought of as a deep learning performance booster." By using skip connections to link the input of one layer to the output of another, the network is able to learn a residual mapping rather than attempting to directly fit the desired output. Adding skip connections to the network yields many advantages: alleviating the vanishing-gradient problem, strengthening feature propagation, encouraging feature reuse, reducing parameters substantially [26], and allowing for the deeper exploration of the network [24]. So, skip connections can be considered a performance enhancer because they can help to improve the accuracy and efficiency of deep neural. skip connections, also known as residual connections, have shown their powerful technique in deep learning.

2.3.2 Types of neural networks

There are several types of neural networks, each with its own unique architecture and purpose. Some of the most common types of neural networks are:

1. **Convolutional Neural Networks (CNNs):** Convolutional neural networks (CNNs) are a type of deep learning algorithm that has been widely used for image recognition, object detection, and segmentation tasks in computer vision applications[30]. CNNs are inspired by the visual cortex of the human brain, where neurons respond to specific visual stimuli and are arranged in a hierarchical manner. One of the key components of CNNs is the convolutional layer, which applies a set of filters to the input image to extract features such as edges, corners, and textures. The filters are learned during the training process using backpropagation [30], The convolutional layer is followed by a pooling layer, which reduces the dimensionality of the feature maps and increases the robustness of the network to small variations in the input image. In recent years, The versatility and effectiveness of CNNs have made them one of the most popular and powerful tools in deep learning.

2. **The concept of convolution :** In a CNN, the input image is processed by convolutional layers, which apply a set of filters to extract local features. Each filter performs convolution on a small region of the input image, sliding over the entire image and producing a feature map. The filters are learned through backpropagation during training [22].

   Convolutional layers are characterized by three key hyperparameters: the number of filters, the size of the filters, and the stride. The number of filters determines the number of feature maps produced by the layer, while the size of the filters determines the receptive field of the layer (i.e., the size of the region over which each filter operates). The stride determines the distance between adjacent filter applications, allowing for downsampling and controlling the size of the output feature maps [29].

3. **Convolutional Neural Network Architecture:** The architecture of a typical CNN consists of several layers, each of which performs a specific type of computation on the input data [22], The first layer in a CNN is typically a convolutional layer, which applies a set of learnable filters to the input image, producing a set of feature maps that highlight different aspects of the image, After the convolutional layers, there are usually one or more pooling layers, which downsample the feature maps to reduce their size and make the network more efficient.The final layers of a CNN are usually fully connected layers, which take the flattened output of the preceding layers and produce the final classification output [22].

4. **the process of training a convolutional neural network:** Training a CNN involves iteratively adjusting the weights of the network to minimize the difference between the predicted output and the actual output. This is done by optimizing a cost function using an algorithm (such as stochastic gradient descent), The first step in training a CNN is to initialize the weights of the network. This can be done randomly or using a pre-trained network, next, the network is trained on a labeled dataset by feeding the input data through the network and comparing the predicted output to the actual output. This process is repeated for many iterations, with the weights being updated after each iteration to improve the performance of the network [22].

5. **the applications of convolutional neural networks (CNNs):** Convolutional neural networks (CNNs) have found a wide range of applications, particularly in computer vision tasks such as image classification, object detection, and face recognition. They have also been used for medical image analysis, natural language processing, and autonomous driving applications. CNNs have achieved state-of-the-art performance in many of these tasks, and their success can be attributed to their ability to automatically learn features from input data through convolutional layers.[29]

2.3.2.1 Auto encoder

Autoencoders are a type of neural network that has become increasingly popular in recent years for their ability to learn efficient representations of data, Autoencoders can be used for a variety of tasks, including dimensionality reduction, feature extraction, and noise removal. autoencoder consists of two parts: an encoder and a decoder. The encoder takes in the input data and compresses it into a lower-dimensional representation. The decoder then takes this lower-dimensional representation and reconstructs the original input data. The goal of the autoencoder is to learn a representation of the data that is as efficient as possible, while still being able to reconstruct the original input data with high accuracy, Autoencoder can be used in many tasks :

1. **Denoising autoencoders :** This kind of autoencoder is made to learn a clean representation of noisy data, it is trained to predict the original, given a corrupted data point as input [22], and it is useful for removing noise from images and other data.

2. **Convolutional autoencoder** this type is commonly used for image denoising, compression, and generation. Unlike standard autoencoders, CAEs utilize convolutional layers in their encoder and decoder networks, allowing them to better handle spatial information in images, the encoder learns to capture important features of the image by applying convolutional operations, while the decoder reconstructs the image from the encoded representation [21].

2.3.2.2 Popular convolutional neural network

1. **ResNet :** ResNet stands for Residual Network, a deep convolutional neural network for image recognition. ResNet uses skip connections to add the output of a previous layer to the output of a residual block [24].

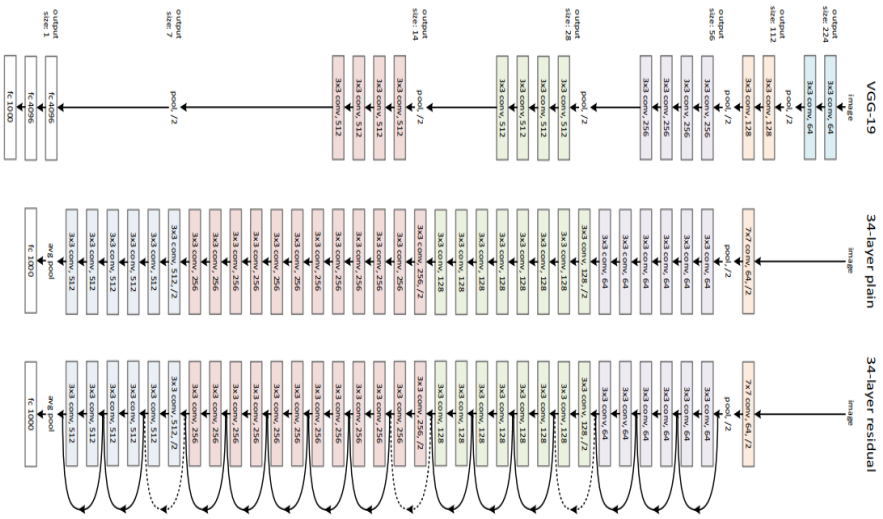


Figure 2.3: ResNet architecture.
[7]

2. **DenseNet :** In DenseNets, each layer is connected to every subsequent layer through concatenation or summation [26].

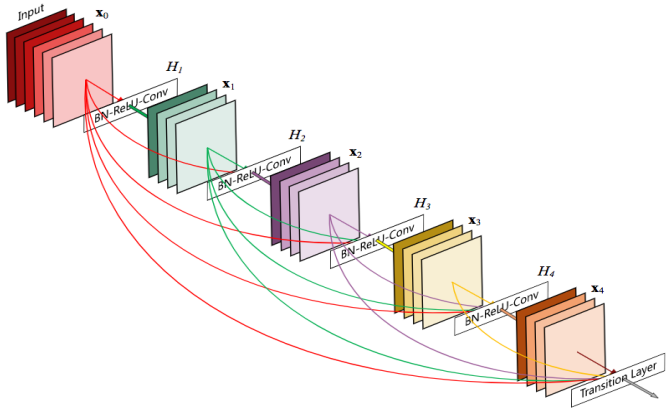


Figure 2.4: DenseNet architecture.
[26]

3. **U-Net :** Convolutional neural network UNet uses skip connections to combine low-level and high-level features from the network's encoder and decoder parts. UNet is used for semantic segmentation. This enhances the segmentation output's localization and context information [38].

   The U-Net architecture has been extensively In several medical image segmentation tasks, including the segmentation of organs, tumors, and cells in multiple imaging modalities. It has also been modified and used in other fields, such as industrial inspection, natural scene segmentation, and satellite photography.

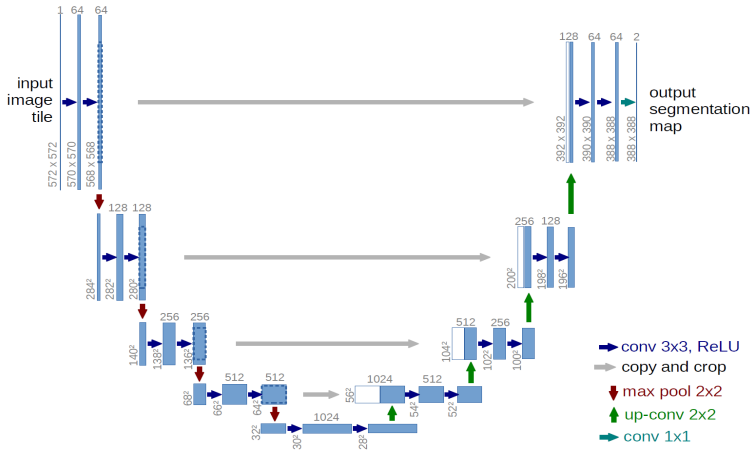


Figure 2.5: U Net architecture.
[38]

4. **Visual Geometry Group (VGG):** Because of the VGG architecture's famed simplicity, it is widely used and simple to comprehend. The number of weight layers was the main factor that the authors looked at when examining how network depth affected CNN performance. They put forth a number of VGG network variations with various depths, ranging from 11 to 19 weight layers. VGG's consistent use of 3x3 convolutional filters throughout the network, which permits the use of multiple stacked layers while keeping the number of parameters manageable, is one distinguishing feature. Max-pooling layers with a 2x2 filter and a stride of 2 are also used in VGG networks. The experiments conducted by the authors demonstrated that performance was significantly enhanced by increasing network depth[42].

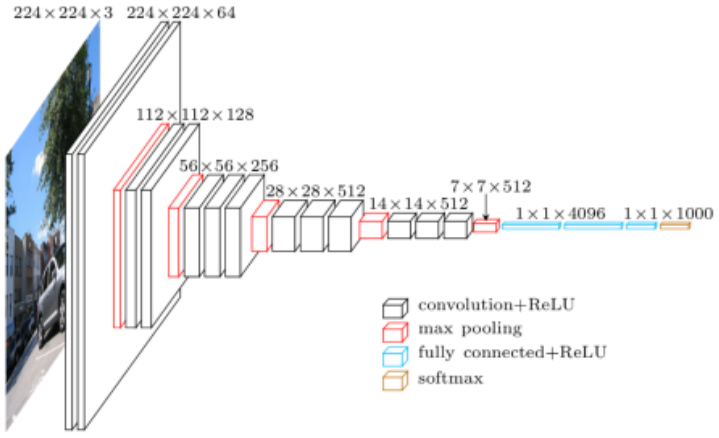


Figure 2.6: example of VGG architecture.
[1]

2.3.3 The Impact of Hyperparameters on Deep Learning Performance

The performance of deep learning models is highly dependent on the choice of hyperparameters, which are parameters that are not learned during training but rather set by the user prior to training, Hyperparameters include the learning rate, batch size, number of layers, number of neurons per layer, activation functions, regularization methods, and many others. A careful

selection of hyperparameters can significantly improve the performance of deep learning models, while a poor choice can lead to poor performance, slow convergence, or even model instability. Various approaches have been suggested for hyperparameter optimization, such as grid search, random search [13], and Bayesian optimization [43].

2.4 Deep learning and medical images

Medical image analysis is essential to modern medicine, helping with disease diagnosis, planning treatments, and keeping track of patient outcomes, the integration of deep learning in medical image analysis has led to breakthroughs in various tasks, including image classification, segmentation, reconstruction, and predictive modeling. Deep learning models, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have demonstrated superior performance in detecting diseases, segmenting anatomical structures, and predicting patient outcomes [31].

2.4.1 Types of Medical Imaging

The term "medical imaging" refers to a number of methods used to produce images of the internal organs and bodily processes for the purpose of clinical diagnosis and treatment. For healthcare professionals to diagnose and treat a variety of medical conditions, medical imaging is a crucial tool. The use of medical imaging has transformed medical diagnosis and treatment.

2.4.1.1 X-ray imaging

The most popular imaging method for diagnosing conditions like bone fractures, dental issues, and conditions of the chest is X-ray imaging. X-ray imaging creates images of the internal body structures by using X-rays, an electromagnetic radiation that can pass through soft tissue. X-ray imaging is an excellent tool for the diagnosis of a variety of medical conditions because it is straightforward, quick, and non-invasive [45].

Figure 2.7: example of X-ray.
[4]

2.4.1.2 Computed Tomography (CT)

CT imaging uses computer technology and X-rays to produce in-depth images of the body's internal structures. For the diagnosis of diseases like cancer, heart disease, and internal bleeding, CT imaging is especially helpful. CT imaging is a great tool for the diagnosis of complex medical conditions because it gives a more accurate and detailed picture of the body's internal structures than X-ray imaging does[45].

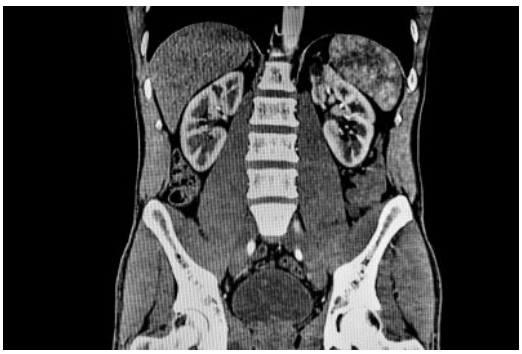
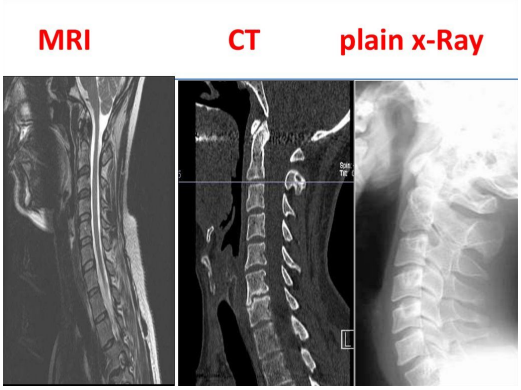


Figure 2.8: Example of Computed Tomography.

2.4.1.3 Magnetic Resonance Imaging (MRI)

MRI imaging uses a magnetic field and radio waves to create detailed images of the body's internal structures. For the diagnosis of diseases like brain tumors, spinal cord injuries, and multiple sclerosis, MRI imaging is particularly helpful. MRI imaging is an excellent tool for the diagnosis of complex medical conditions because it produces an extremely accurate and detailed image of the body's internal structures [45].

(a) MRI scan[2].



(b) the difference between MRI,CT,and x-ray [8].

Figure 2.9: Details for medical images.

2.4.1.4 Ultrasound Imaging

High-frequency sound waves are used in ultrasound imaging to produce images of the body's internal organs, Ultrasound imaging is commonly used to diagnose conditions such as pregnancy, liver disease, and heart disease[45].

The main equipment of ultrasound imaging includes a handheld device called a transducer, a pulser and receiver, and a display system.

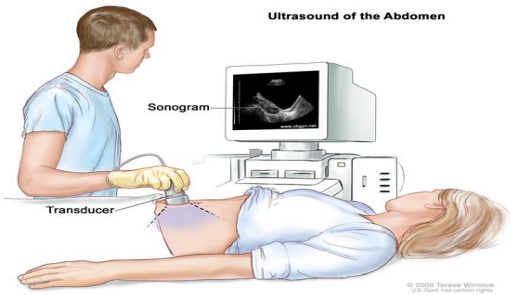


Figure 2.10: Ultrasound Imaging equipment.
[6]

2.4.2 Alzheimer's dementia (AD) diagnosis

The term "dementia" is used in general to refer to a premature decline in cognitive function that goes beyond biological aging. The most prevalent type of dementia, accounting for 70% of cases, is Alzheimer's disease (AD). It gradually interferes with daily tasks and functions by altering memory, thinking, and behavior. Early identification of any accompanying cognitive impairment enables the administration of preventive drugs to slow the development of the illness [48]. Observing people with mild cognitive impairment (MCI) and evaluating cognitive changes over time are the traditional methods for diagnosing AD.

However, AD is caused by the gradual loss (degeneration) of brain cells, which can be seen in brain scans even before or with very mild symptoms[47]. The structures of the brain can be seen using structural imaging techniques like magnetic resonance imaging (MRI), which can also show damage to particular areas of the brain and the atrophy of neurons and their connections. For the diagnosis and assessment of dementia, MRI-based diagnostics have become a crucial component of clinical practice[18, 32].

2.5 Based works

2.5.1 Delta Auto Encoder

This approach focuses on teaching a neural network to generate samples from distributions of new visual categories, using only a limited number of observed examples. it aims to learn and leverage transferable intra-class deformations, referred to as "deltas," between pairs of training examples belonging to the same class. By capturing these deltas, our model can effectively apply them to a limited set of examples from a novel class, unseen during training, to efficiently synthesize additional samples[40].

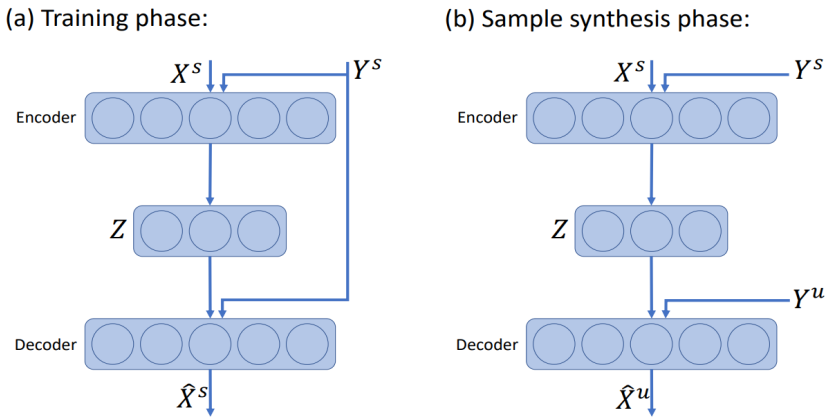


Figure 2.11: Delta autoencoder.
[40]

2.5.2 Convolution Neural Networks and Self-Attention Learners

This work aimed to enhance the automatic detection of dementia in MRI brain data by employing computer-aided diagnosis (CAD). The study explored three deep convolutional models (ResNet, DenseNet, and EfficientNet) and two transformer-based architectures (MAE and DeiT) for mapping input images to clinical diagnosis. Two publicly available datasets (ADNI and OASIS) were used for fair comparison through various benchmarks. Results indicated that the deeper ResNet and DenseNet models outperformed the shallower versions, and transformer architectures, particularly DeiT, demonstrated superior classification accuracy and robustness to added noise. Compared to existing methods, this approach achieved a significant improvement in accuracy (up to 7%), opening up possibilities for CAD implementation in real-world applications[14]. They trained the models using different numbers of slices per subject, specifically 4, 8, and 16 slices. By training the models on varying slice configurations, the researchers aimed to assess the models' performance under different levels of input data granularity.

2.6 Conclusion

AI replicates human intelligence in machines, enabling tasks like perception, reasoning, learning, and decision-making. It can transform industries like healthcare, banking, transportation, and manufacturing by increasing efficiency, lowering costs, and offering personalized experiences.

Medical image analysis is crucial for disease diagnosis, treatment planning, and patient outcomes tracking. Deep learning models, like CNNs and RNNs, excel in image classification, segmentation, reconstruction, and predictive modeling.

Alzheimer's disease is prevalent dementia causing cognitive decline beyond biological aging. Early identification and early detection enable preventive drugs, in our work we attempted to automate the diagnosis of AD, and we constructed two new architectures, the first one is for data augmentation which is inspired by Deltaencoder[40], and the second one for classification which is based on CNNs and Self-Attention Learners[14].

Chapter 3

Proposed methods

3.1 Inroduction

Few-shot learning (FSL) is an important area of research in deep learning. however, the ability to learn new concepts with limited data is a key challenge in these fields, and several approaches have been proposed to address this problem.

One of the key approaches to FSL performance is through the data perspective, which uses prior knowledge to augment the supervised experience. This perspective involves generating additional training data and applying this augment method to improve the dementia classification task.

In our work, we focus on the data perspective of FSL and propose a new approach to improve few-shot learning by generating new data with an enhanced autoencoder-based architecture. We apply the proposed approach to the task of dementia identification and ordinary task.

3.2 The proposed augmentation method

The proposed solution involves training a network consisting of two encoders and a decoder. The first encoder learns transferable deformations between pairs of examples of the same class (A), while the second encoder extracts features from the class (B). Our suggested method uses a combination of U-net, CAE, and ResNet, where the decoder applies the transferable deformations to the example from the second encoder to generate a transferred sample of class (B). 3.1 present the general flowchart of the proposed method :

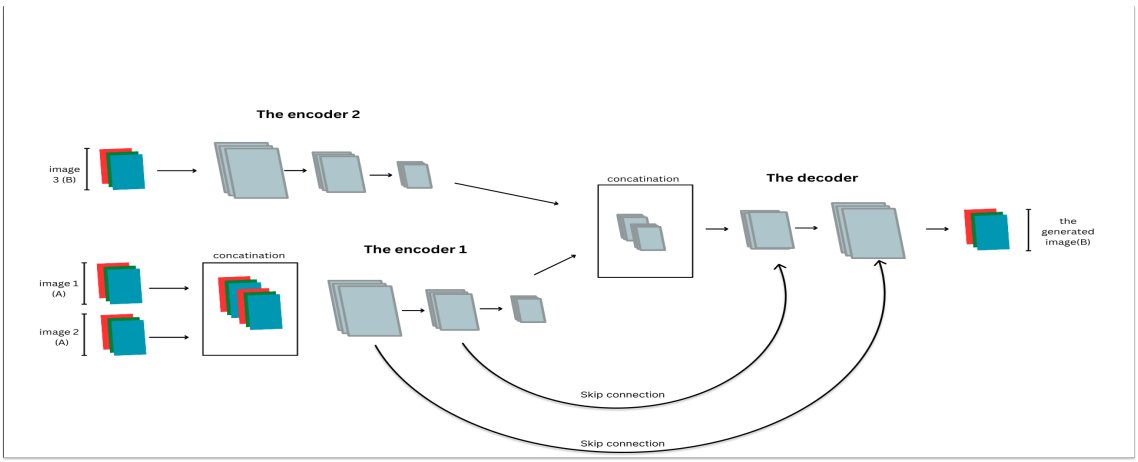


Figure 3.1: Training phase.

The network parameters are updated accordingly to minimize the difference between the generated image and the original pair image of class (B). Hereafter, we provide more details on the components of the proposed approaches.

3.2.1 Architecture's Layers

3.2.1.1 Concatenation layers

In our proposed method, we utilized a concatenation layer as a key component in the architecture. we employed concatenation layers in three specific cases to enhance the information flow and capture richer features :

1. **3.2.1.2 Concatenate input images:** We utilized a concatenation layer to combine two input images. By concatenating the pixel values along a specific dimension, we effectively merged the information from both images into a single input tensor. This allowed the subsequent layers to leverage the combined information from multiple sources, enabling the model to extract more comprehensive features.

2. **Concatenate output of encoders:** We utilized concatenation layers to merge the outputs from two separate encoders. After passing the inputs through two distinct encoder pathways, we concatenated their respective output feature maps. This merging operation facilitated the fusion of features learned at different levels of abstraction.

3. **Concatenation layers in the skip connections:** Skip connections are commonly used to improve gradient flow and facilitate the propagation of information across different layers. In our architecture, we incorporated concatenation layers in the skip connections, allowing the direct concatenation of feature maps from encoder layers with decoder layers.

Convolution layers

In our proposed method, we incorporated convolutional layers with a 3x3 filter size and "same" padding. The convolutional operation is a fundamental building block in convolutional neural networks (CNNs) and plays a crucial role in feature extraction.

By using a 3x3 filter size, we aimed to capture local patterns and spatial dependencies within the input data. This filter size is commonly chosen due to its effectiveness in detecting meaningful features.

we employed "same" padding in the convolutional layers. The same padding ensures that the output feature maps have the same spatial dimensions as the input. It achieves this by padding the input with zeros before performing the convolution operation.

3.2.1.3 Pooling layers

In our proposed method, we incorporated max pooling layers after every 2 or 3 convolutional layers. Max pooling is a common technique used in convolutional neural networks (CNNs) for downsampling feature maps and capturing the most salient features.

We used a pooling size of 2 by 2, which means that within each pooling region, the maximum value was selected. This downsampling operation reduced the spatial dimensions of the feature maps while retaining the most significant information. By downsampling, the model becomes more computationally efficient and gains a degree of transnational invariance.

Furthermore, we applied "same" padding in the max pooling layers. The same padding helps to compensate for any overlaps that occur when the input size and kernel size do not perfectly align.

3.2.1.4 Dense layers

We design two architectures using our suggested techniques, one of them contains the dense layers; the output of the first and second encoders are flattened, concatenated, and connected by a dense layer with 2000neurons, this is done after the convolution and max pooling layers; finally, another dense layer is added. Its dimensions match those of one of the outputs from the encoder after it has been flattened, this is done so that it can be reshaped, fed into the decoder, and then used to reconstruct a new image.

3.3 The used hyperparameters

3.3.1 "ReLU" activation function

Artificial neural networks need an activation function because it introduces non-linearity to the network's output, Choosing the right activation function is crucial as it directly impacts the network's ability to model complex relationships and effectively learn from data.in our architecture, we used the Relu activation function in whole networks. ReLU, or Rectified Linear Unit, is a popular activation function used in artificial neural networks The main reason that we use ReLU is its Computational Simplicity. it uses simple thresholding operations, so it is computationally efficient. When compared to activation functions that require more complicated computations, this efficiency enables quicker training and inference, Moreover, it does not cause the vanishing gradient problem when increasing layers like sigmoid or tanh[20].

3.3.2 Loss function

The loss function is a key element in many deep learning algorithms and is important for both training and optimizing models.in our method, we used (MAE).

3.3.2.1 Mean Absolute Error (MAE)

From a mathematical perspective, MAE is calculated by summing the absolute differences between predicted values (ŷ) and actual values (y), and then dividing by the total number of samples (n):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.1}$$

3.3.2.2 Mean Square Error (MSE)

Mean Square Error (MSE) is a widely used statistical metric to evaluate the quality of predictions or estimations by measuring the average squared difference between predicted values and actual values[15].

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.2}$$

We used this activation function in our classification model.

3.3.3 ADAM optimizer

In our architecture, we used ADAM as an optimization algorithm.ADAM is a widely used optimization algorithm that combines the advantages of both adaptive learning rate methods and momentum-based methods.

3.4 The proposed approach for the task of dementia identification

In this part, we will show the ability of our method in the context of generating new data and learning the transformations. The purpose of the first experience is to learn the transformation from a longitudinal section to a cross-section of MRI images, which contains we traditional skip connection from the encoder to the decoder as a traditional procedure for increasing the performance of generating approach, which in turn helps us in the classification of dementia.

3.4.1 Augmentation model

To solve the few data problem in cross-section MRI images we proposed this architecture that allows us to augment our dataset by longitudinal section as prior knowledge. The model consists of two encoders and a decoder, and the two images from subject 'A' are placed into one of the encoders, one of these images is a longitudinal section MRI and the second is a cross-section MRI. The first encoder has five blocks in total. Four of the blocks have three convolutions and a max-pool, while the fifth block only has two convolutions. The second encoder, which has 4 blocks and 3 convolutions, and 1 max-pool in each, includes the picture of the longitudinal images that belong to subject B. After that, the output from them is reshaped and inserted after being flattened, concatenated, and connected successively to 2 Dense layers. A new cross-section image with five blocks, four of which include three convolutions and summation, and one block with only two convolutions is created with the decoder. It is noteworthy that the initial encoder and the decoder, such as UNET, are connected via a skip connection, we can see more details and understand the architecture from figure 3.2:

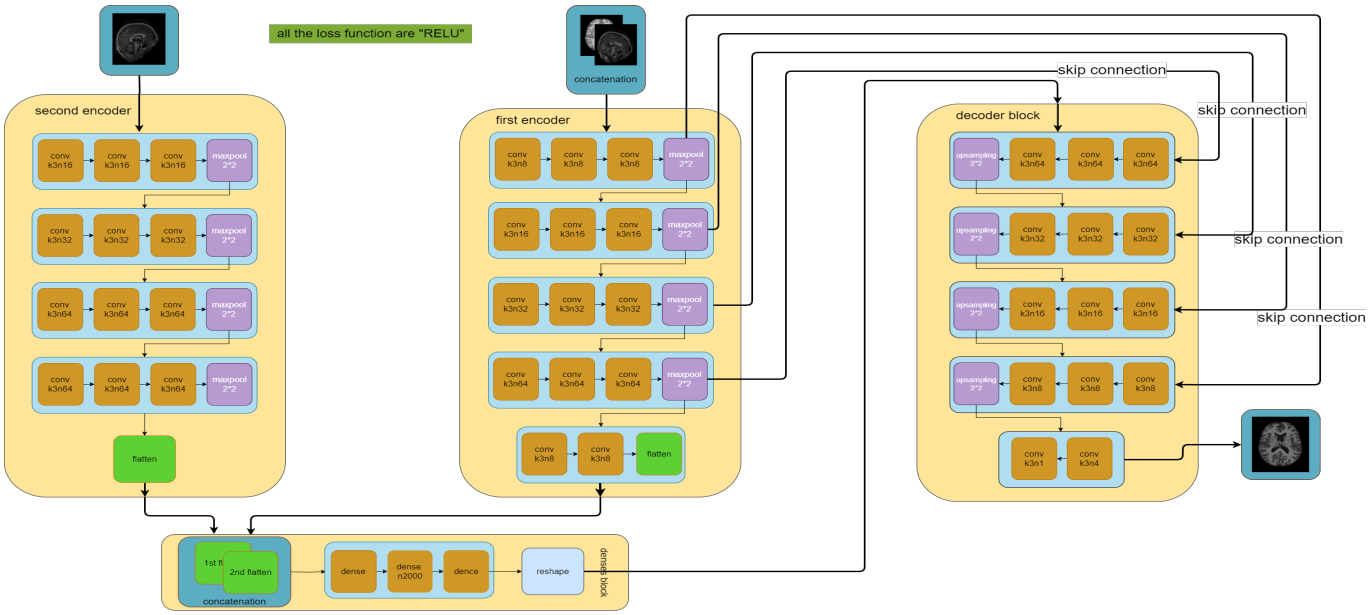


Figure 3.2: Our proposed augmentation architecture.

By using the skip connection the network is able to learn some information that helps in the correct construction of the output image rather than attempting to directly fit the desired output, encouraging feature reuse, and also allowing for a deeper exploration of the network.

and the table from figure 3.3 shows the summary of the augmentation model:

Layer (name)	Layer (type)	Output Shape	Param #
input_1	InputLayer	[(None, 224, 192, 3)]	0
tf.__operators__.getitem	SlicingOpLambda	(None, 224, 192, 2)	0
reshape	Reshape	(None, 224, 192, 2)	0
conv2d	Conv2D	(None, 224, 192, 8)	152
conv2d_1	Conv2D	(None, 224, 192, 8)	584
conv2d_2	Conv2D	(None, 224, 192, 8)	584
max_pooling2d	MaxPooling2D	(None, 112, 96, 8)	0
conv2d_3	Conv2D	(None, 112, 96, 16)	1168
tf.__operators__.getitem_1	SlicingOpLambda	(None, 224, 192)	0
conv2d_4	Conv2D	(None, 112, 96, 16)	2320
reshape_1	Reshape	(None, 224, 192, 1)	0
conv2d_5	Conv2D	(None, 112, 96, 16)	2320
conv2d_14	Conv2D	(None, 224, 192, 16)	160
max_pooling2d_1	MaxPooling2D	(None, 56, 48, 16)	0
conv2d_15	Conv2D	(None, 224, 192, 16)	2320
conv2d_6	Conv2D	(None, 56, 48, 32)	4640
max_pooling2d_4	MaxPooling2D	(None, 112, 96, 16)	0
conv2d_7	Conv2D	(None, 56, 48, 32)	9248
conv2d_16	Conv2D	(None, 112, 96, 32)	4640
conv2d_8	Conv2D	(None, 56, 48, 32)	9248
conv2d_17	Conv2D	(None, 112, 96, 32)	9248
max_pooling2d_2	MaxPooling2D	(None, 28, 24, 32)	0
max_pooling2d_5	MaxPooling2D	(None, 56, 48, 32)	0
conv2d_9	Conv2D	(None, 28, 24, 64)	18496
conv2d_18	Conv2D	(None, 56, 48, 64)	18496
conv2d_10	Conv2D	(None, 28, 24, 64)	36928
conv2d_19	Conv2D	(None, 56, 48, 64)	36928
conv2d_11	Conv2D	(None, 28, 24, 64)	36928
max_pooling2d_6	MaxPooling2D	(None, 28, 24, 64)	0
max_pooling2d_3	MaxPooling2D	(None, 14, 12, 64)	0
conv2d_20	Conv2D	(None, 28, 24, 64)	36928

Layer (name)	Layer (type)	Output Shape	Param #
conv2d_12	Conv2D	(None, 14, 12, 64)	36928
conv2d_21	Conv2D	(None, 28, 24, 64)	36928
conv2d_13	Conv2D	(None, 14, 12, 64)	36928
max_pooling2d_7	MaxPooling2D	(None, 14, 12, 64)	0
flatten	Flatten	(None, 10752)	0
flatten_1	Flatten	(None, 10752)	0
concatenate	Concatenate	(None, 21504)	0
dense	Dense	(None, 2000)	43010000
dense_1	Dense	(None, 10752)	21514752
reshape_2	Reshape	(None, 14, 12, 64)	0
conv2d_22	Conv2D	(None, 14, 12, 64)	36928
conv2d_23	Conv2D	(None, 14, 12, 64)	36928
conv2d_24	Conv2D	(None, 14, 12, 64)	36928
concatenate_1	Concatenate	(None, 14, 12, 128)	0
up_sampling2d	UpSampling2D	(None, 28, 24, 128)	0
conv2d_25	Conv2D	(None, 28, 24, 32)	36896
conv2d_26	Conv2D	(None, 28, 24, 32)	9248
conv2d_27	Conv2D	(None, 28, 24, 32)	9248
concatenate_2	Concatenate	(None, 28, 24, 64)	0
up_sampling2d_1	UpSampling2D	(None, 56, 48, 64)	0
conv2d_28	Conv2D	(None, 56, 48, 16)	9232
conv2d_29	Conv2D	(None, 56, 48, 32)	4640
conv2d_30	Conv2D	(None, 56, 48, 32)	9248
concatenate_3	Concatenate	(None, 56, 48, 48)	0
up_sampling2d_2	UpSampling2D	(None, 112, 96, 48)	0
conv2d_31	Conv2D	(None, 112, 96, 8)	3464
conv2d_32	Conv2D	(None, 112, 96, 8)	584
conv2d_33	Conv2D	(None, 112, 96, 8)	584
concatenate_4	Concatenate	(None, 112, 96, 16)	0
up_sampling2d_3	UpSampling2D	(None, 224, 192, 16)	0
conv2d_35	Conv2D	(None, 224, 192, 1)	145
Total			65060945

Figure 3.3: the summary table of our proposed augmentation method.

3.4.2 classification model

Our classification model has two inputs: a cross-section MRI of the brain and a corresponding feature vector. The image is traversed through a total of five blocks. the first four of the blocks have three convolutions and a max-pool, while the fifth block only has two convolutions, and all of its convolutions contain the "relu" activation function. The output from the previous layer is then flattened, reformed, and connected consecutively to two dense layers, each of which contains 1000 neurons . then connected it to the two dense layers with dropout regularization are then linked. the first one, which had 500 neurons concatenated with the second input (a feature vector), and the second one had 100 neurons, to create the output, which had just one neuron. With the "sigmoid" activation function, all of that, From here, we can see more specifics and grasp the architecture from the figure 3.4:

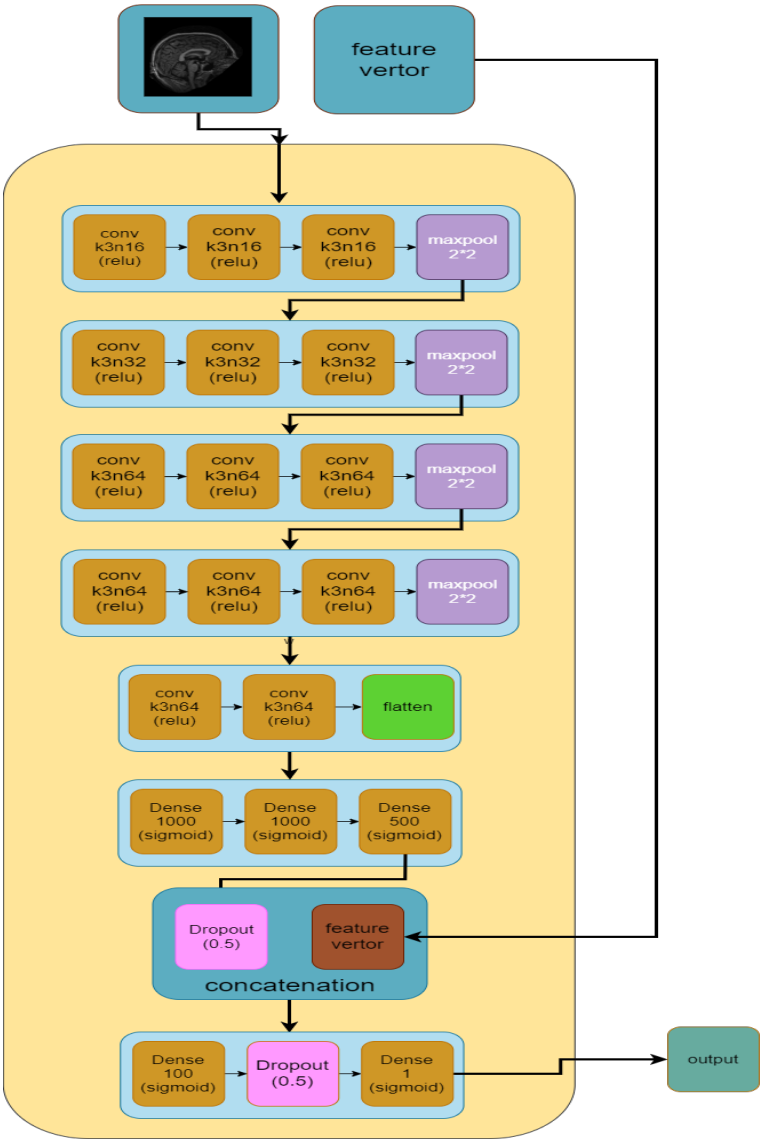


Figure 3.4: Our proposed classification architecture.

the table 3.1 represent the summary of the augmentation model::

Layer (name)	Layer (type)	Output Shape	Param #
input_1	InputLayer	[(None, 224, 192, 1)]	0
conv2d	Conv2D	(None, 224, 192, 8)	80
conv2d_1	Conv2D	(None, 224, 192, 8)	584
conv2d_2	Conv2D	(None, 222, 190, 8)	584
max_pooling2d	MaxPooling2D	(None, 111, 95, 8)	0
conv2d_3	Conv2D	(None, 111, 95, 16)	1168
conv2d_4	Conv2D	(None, 111, 95, 16)	2320
conv2d_5	Conv2D	(None, 109, 93, 16)	2320
max_pooling2d_1	MaxPooling2D	(None, 55, 47, 16)	0
conv2d_6	Conv2D	(None, 55, 47, 32)	4640
conv2d_7	Conv2D	(None, 55, 47, 32)	9248
conv2d_8	Conv2D	(None, 53, 45, 32)	9248
max_pooling2d_2	MaxPooling2D	(None, 27, 23, 32)	0
conv2d_9	Conv2D	(None, 27, 23, 64)	18496
conv2d_10	Conv2D	(None, 27, 23, 64)	36928
conv2d_11	Conv2D	(None, 25, 21, 64)	36928
max_pooling2d_3	MaxPooling2D	(None, 13, 11, 64)	0
conv2d_12	Conv2D	(None, 13, 11, 128)	73856
conv2d_13	Conv2D	(None, 13, 11, 128)	147584
flatten	Flatten	(None, 18304)	0
dense	Dense	(None, 1000)	18305000
dense_1	Dense	(None, 1000)	1001000
dense_2	Dense	(None, 500)	500500
dropout	Dropout	(None, 500)	0
input_2	InputLayer	[(None, 8)]	0
concatenate	Concatenate	(None, 508)	0
dense_3	Dense	(None, 100)	50900
dropout_1	Dropout	(None, 100)	0
dense_4	Dense	(None, 1)	101
Tottal			20201485

Table 3.1: summary table of our proposed classification method

3.5 The proposed approach for the task of generating images based on artistic transformation

In the second experiment, we aim to learn two transformations , the transformation from face to cartoon face , and the transformation from anime to sketch art. Our aim was to address the generation performance of the models, and we explored adjustments to the skip connections within the architecture. We added skip connections from the input to each final block of the model. These skip connections allowed the model to directly access the original image information at multiple scales throughout the network. By concatenating the feature maps of the final blocks with the original images and applying max pooling, we aimed to preserve the spatial information while incorporating learned features. As a result of these modifications, we observed significant improvements in generating performance.

As previously mentioned, a key modification we made to the skip connections involved passing the original images through both the encoder and decoder pathways of the model. Before concatenation with the end of each block, we applied max pooling to the original images to ensure they had the same size as the target layers. Additionally, we reduced the number of convolution blocks in our modified architecture, resulting in a reduction in the total number of parameters as shown in 3.2 3.3 tables. The figure below show the architecture of our model 3.5

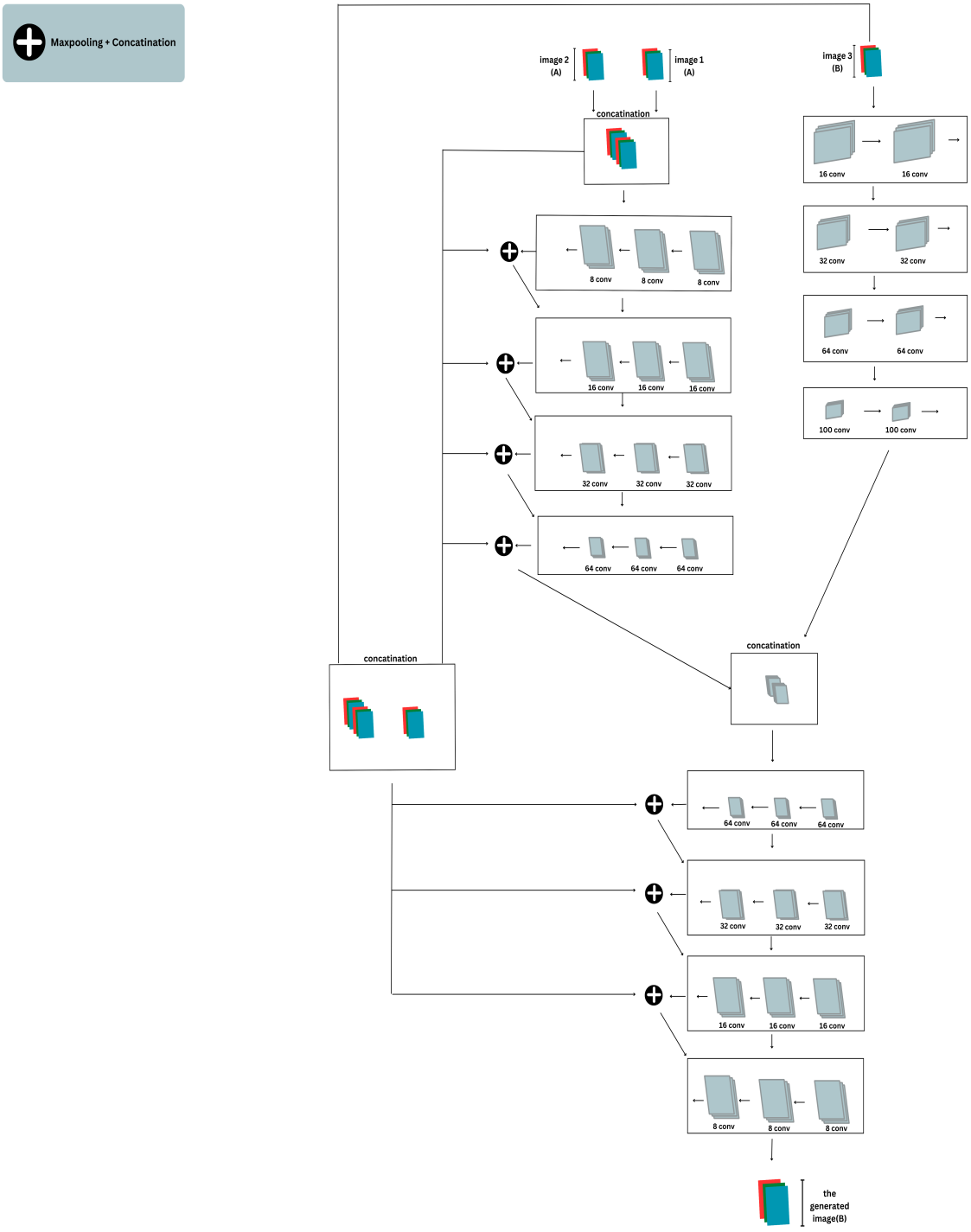


Figure 3.5: Our second proposed augmentation method.

Layer (name)	Layer (type)	Output Shape	Param #
input_2	InputLayer	[(None, 128, 128, 9)]	0
tf.__operators__.getitem_2	SlicingOpLambda	(None, 128, 128, 6)	0
reshape_2	Reshape	(None, 128, 128, 6)	0
conv2d_34	Conv2D	(None, 128, 128, 8)	440
conv2d_35	Conv2D	(None, 128, 128, 8)	584
conv2d_36	Conv2D	(None, 128, 128, 8)	584
concatenate_13	Concatenate	(None, 128, 128, 14)	0
max_pooling2d_20	MaxPooling2D	(None, 64, 64, 14)	0
conv2d_37	Conv2D	(None, 64, 64, 16)	2032
conv2d_38	Conv2D	(None, 64, 64, 16)	2320
conv2d_39	Conv2D	(None, 64, 64, 16)	2320
max_pooling2d_21	MaxPooling2D	(None, 64, 64, 6)	0
tf.__operators__.getitem_3	SlicingOpLambda	(None, 128, 128, 3)	0
concatenate_14	Concatenate	(None, 64, 64, 22)	0
reshape_3	Reshape	(None, 128, 128, 3)	0
max_pooling2d_22	MaxPooling2D	(None, 32, 32, 22)	0
conv2d_47	Conv2D	(None, 128, 128, 16)	448
conv2d_40	Conv2D	(None, 32, 32, 32)	6368
conv2d_48	Conv2D	(None, 128, 128, 16)	2320
conv2d_41	Conv2D	(None, 32, 32, 32)	9248
max_pooling2d_23	MaxPooling2D	(None, 64, 64, 6)	0
max_pooling2d_30	MaxPooling2D	(None, 64, 64, 16)	0
conv2d_42	Conv2D	(None, 32, 32, 32)	9248
max_pooling2d_24	MaxPooling2D	(None, 32, 32, 6)	0
conv2d_49	Conv2D	(None, 64, 64, 32)	4640
concatenate_15	Concatenate	(None, 32, 32, 38)	0
conv2d_50	Conv2D	(None, 64, 64, 32)	9248
max_pooling2d_25	MaxPooling2D	(None, 16, 16, 38)	0
max_pooling2d_31	MaxPooling2D	(None, 32, 32, 32)	0
conv2d_43	Conv2D	(None, 16, 16, 64)	21952
max_pooling2d_26	MaxPooling2D	(None, 64, 64, 6)	0
conv2d_51	Conv2D	(None, 32, 32, 64)	18496
conv2d_44	Conv2D	(None, 16, 16, 64)	36928
max_pooling2d_27	MaxPooling2D	(None, 32, 32, 6)	0
conv2d_52	Conv2D	(None, 32, 32, 64)	36928
conv2d_45	Conv2D	(None, 16, 16, 64)	36928
max_pooling2d_28	MaxPooling2D	(None, 16, 16, 6)	0
max_pooling2d_32	MaxPooling2D	(None, 16, 16, 64)	0
concatenate_16	Concatenate	(None, 16, 16, 70)	0
conv2d_53	Conv2D	(None, 16, 16, 100)	57700

Table 3.2: First part of our second proposed augmentation method's summary.

conv2d_53	Conv2D	(None, 16, 16, 100)	57700
max_pooling2d_29	MaxPooling2D	(None, 8, 8, 70)	0
conv2d_54	Conv2D	(None, 16, 16, 100)	90100
conv2d_46	Conv2D	(None, 8, 8, 100)	63100
max_pooling2d_33	MaxPooling2D	(None, 8, 8, 100)	0
concatenate_17	Concatenate	(None, 8, 8, 200)	0
conv2d_55	Conv2D	(None, 8, 8, 64)	115264
conv2d_56	Conv2D	(None, 8, 8, 64)	36928
max_pooling2d_34	MaxPooling2D	(None, 64, 64, 9)	0
up_sampling2d_4	UpSampling2D	(None, 16, 16, 64)	0
max_pooling2d_35	MaxPooling2D	(None, 32, 32, 9)	0
conv2d_57	Conv2D	(None, 16, 16, 64)	36928
max_pooling2d_36	MaxPooling2D	(None, 16, 16, 9)	0
concatenate_19	Concatenate	(None, 16, 16, 73)	0
conv2d_58	Conv2D	(None, 16, 16, 32)	21056
conv2d_59	Conv2D	(None, 16, 16, 32)	9248
up_sampling2d_5	UpSampling2D	(None, 32, 32, 32)	0
max_pooling2d_37	MaxPooling2D	(None, 64, 64, 9)	0
conv2d_60	Conv2D	(None, 32, 32, 32)	9248
max_pooling2d_38	MaxPooling2D	(None, 32, 32, 9)	0
concatenate_21	Concatenate	(None, 32, 32, 41)	0
conv2d_61	Conv2D	(None, 32, 32, 16)	5920
conv2d_62	Conv2D	(None, 32, 32, 16)	2320
up_sampling2d_6	UpSampling2D	(None, 64, 64, 16)	0
conv2d_63	Conv2D	(None, 64, 64, 16)	2320
max_pooling2d_39	MaxPooling2D	(None, 64, 64, 9)	0
concatenate_23	Concatenate	(None, 64, 64, 25)	0
conv2d_64	Conv2D	(None, 64, 64, 8)	1808
conv2d_65	Conv2D	(None, 64, 64, 8)	584
up_sampling2d_7	UpSampling2D	(None, 128, 128, 8)	0
conv2d_66	Conv2D	(None, 128, 128, 8)	584
concatenate_25	Concatenate	(None, 128, 128, 17)	0
conv2d_67	Conv2D	(None, 128, 128, 3)	462
Tottal			654602

Table 3.3: Second part of our second proposed augmentation method's summary.

3.6 Conclusion

Our experiment focused on developing an architecture for dementia identification using data augmentation and classification techniques. We constructed two architectures: one for data augmentation and the other for classification. The data augmentation architecture employed two encoders and decoders to learn transferable deformations and generate augmented samples. This approach aimed to enhance the dataset for dementia identification. For classification, we utilized a convolutional neural network (CNN). The CNN model was designed to classify images into dementia and non-dementia categories.

To evaluate the effectiveness of our data augmentation architecture in different tasks, we applied it to artistic transformation as an additional experiment, we introduced skip connections from the original images into the network to preserve the artistic qualities.

Chapter 4

EXPERIMENTAL RESULTS

4.1 The dataset

.

4.1.1 The first datasets (OASIS-1)

Cross-sectional MRI Data in Young, Middle-Aged, Nondemented, and Demented Older Adults dataset (Oasis-1) consists of 416 people ranging in age from 18 to 96, and it comprises a collection of brain MRI scans with multiple sectionals. However, for our study, we only use the longitudinal section and Cross-section. Three or four distinct T1-weighted MRI scans performed in a single scan session are presented for each subject. Men and women, both right-handed, are represented among the subjects. Over the age of 60, 100 of the patients who were involved in the study received a clinical diagnosis of very mild to moderate Alzheimer's disease (AD). A trustworthy data set with 20 non-demented participants that were photographed 90 days after their initial session is also included in the dataset link https://www.oasis-brains.org/

But when checking the CSV file accompanying the dataset, we noticed that 220 subjects are unlabeled and there is a lot of missing information, and the rest 216 subjects are complete. Therefore, we decided to use the missing subjects to augmentation approach and consider it as prior knowledge. As for dementia classification, we only worked with 216 completed subjects.

4.1.1.1 Prepossessing of the dataset for augmentation task

Since we said previously that almost half of the dataset is unlabeled, we had to work on only 216 subjects and divided it into 75%: 25%, which means about 55 subjects for testing, of which 23 subjects had dementia and 32 didn't have, and 161 for training in the classification task, of which 60 subjects had dementia and 101 didn't have, where we did the cross-validation technique in training the classification approach with k-fold for 75% of the training dataset. As for the task of augmentation of the dataset, we add 220 to the training data. subjects that we said earlier that it does not contain sufficient information because we don't need the information of this part in the augmentation approach.

In the step of augmentation of the dataset, we had to form triplets of images as inputs and one image as output, where the triplets are formed from two images of subject A, one image of a longitudinal MRI section, and one image of an MRI cross-section, while the third image belongs to another random subject B, but from a longitudinal MRI section, while the MRI cross-section of element B is considered As outputs, we can thus form many triads to train our model, given that each element contains two cross-section images and four or five longitudinal images, it is worth mentioning that we did not normalize the images between 0 and 1, but left

them between 0 and 255 because the images have many details and it is difficult to reconstruct images with many details in a semi-accurate manner and their range is between 0 and 1, and also because the images between 0 and 255 take At least half of the space in RAM for images between 0 and 1. Where the dataset is as follows shown in figure 4.1:

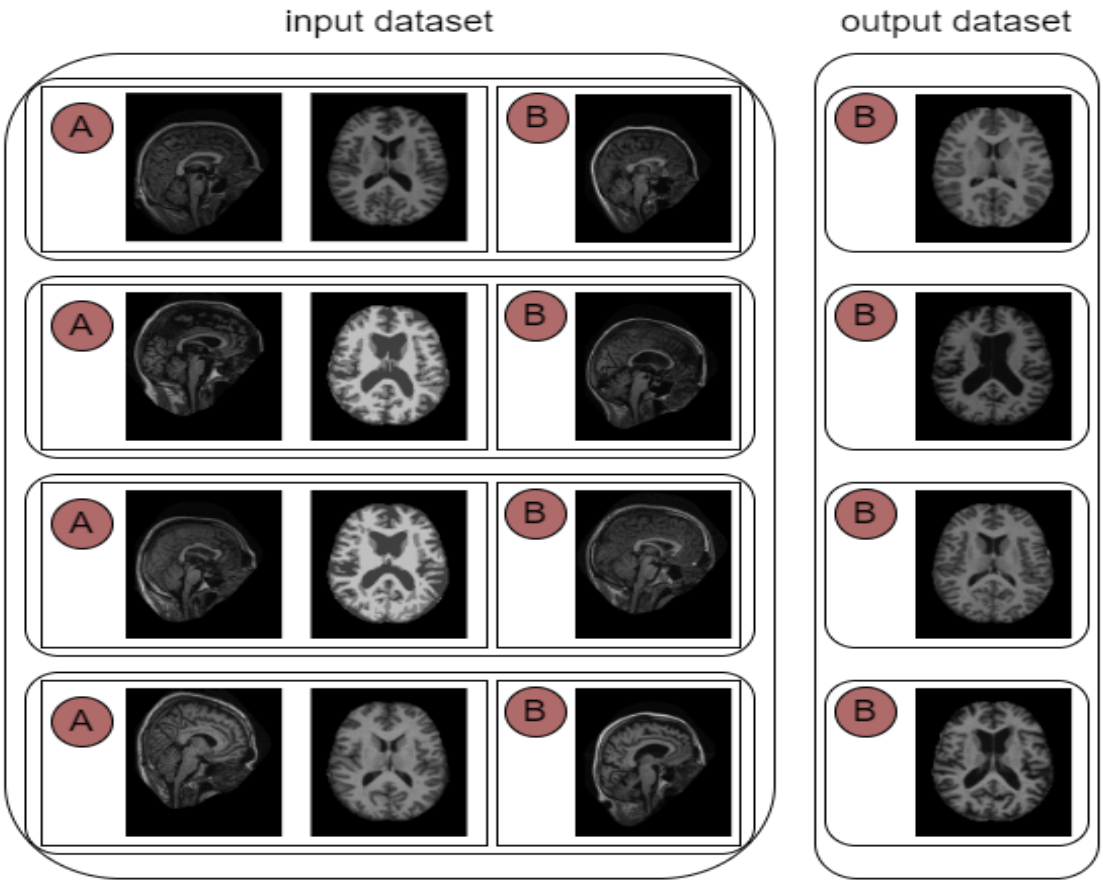


Figure 4.1: An example dataset

4.1.1.2 Preprocessing of the dataset for classification task

In the training of augmentation task, we randomly form the data from the subjects, and only make sure of the conditions that we said previously, but to implement the classification task, we make sure that subject "B", whose data we are increasing, belongs to 161 who have complete information.

In the classification task, we also rely on the information in the CSV file, as it contains a set of features, which are: Gender, Dominant Hand, Age, Education Level(Educ), Socioeconomic Status(ses), Mini-Mental State Examination(MMSE), Estimated Total Intracranial Volume(eTIV), Normalize Whole Brain Volume(nWBV), Atlas Scaling Factor(ASF), and Clinical Dementia Rating(CDR) is the output. we delete Dominant Hand features because there is one value and normalize the features between 0 and 1, Thus we are able to enter it as one of the inputs in the classification task. The outputs contain four values (0, 0.5, 1, 2) that represent the degree of dementia, but due to the lack of a dataset, we decided to do a binary classification, and we converted the values 0.5, 2 to 1, which means that the subject is sick, while the value 0 means that The subject is not sick, then in the classification task, we enter the images with the feature vector corresponding.

4.1.1.3 Preprocessing the dataset with Hand-Crafted augmentation for classification task

Hand-crafted augmentations were employed to facilitate a comprehensive comparison with our model. These augmentations included flipping and rotation operations at angles of 20 and -20 degrees, as well as shifting the images horizontally and vertically. By applying these techniques, we aimed to enhance the diversity and variability of our dataset, enabling a more thorough evaluation of our model's performance.

4.1.2 The second dataset (artistic transformation)

We have two different datasets, the dataset of face2cartoon (A) and anime2scetch (B), both are paired datasets, which means each image has its transformation, the purpose is to learn the transformation features and generate new images based on this transformation, we collected 1000 images from the dataset (A) and 975 from (B), then we allocated 10 images from each class as test datasets. The image below shows an instance of the dataset 4.2.

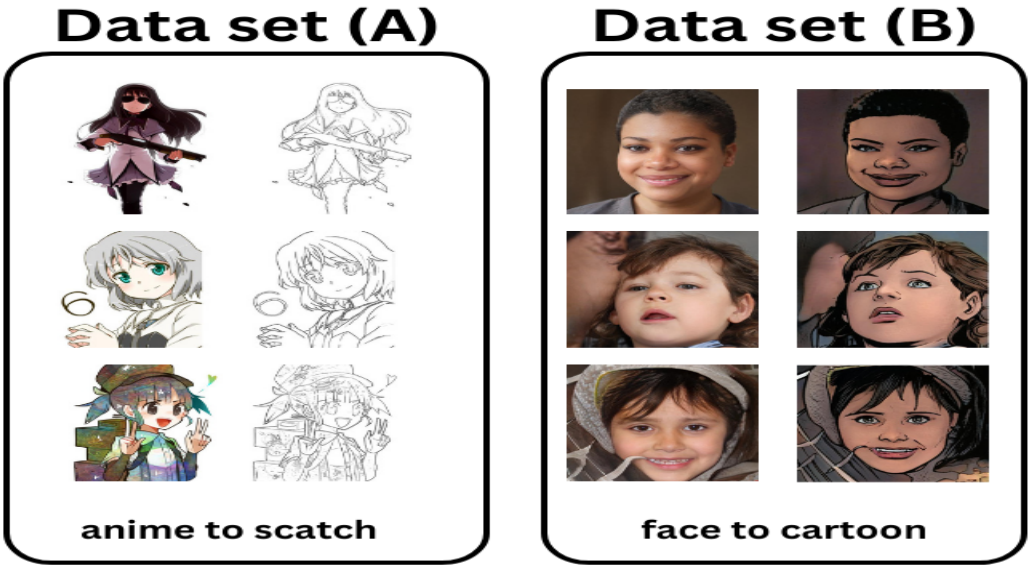


Figure 4.2: An instance of the dataset.

4.1.2.1 Prepossessing of data

In our prepossessing pipeline, we handled a dataset composed of two distinct datasets, denoted as A and B. Each dataset contained paired images, represented as A = a1i, a2i and B = b1i, b2i. To ensure a fair evaluation of our model, we allocated 5 images from each class for testing, leaving us with 995 images from class A and 970 images from class B for training.

To create our training dataset, we followed the following steps:

1. We sampled paired images from class A, resulting in (a11, a21). We then randomly selected another sample from class A, denoted as a12, and concatenated it with the (a1i, a2i), while the second image,a22, was set as the Y train label.

2. Similarly, we sampled paired images from class B.

3. We repeated steps 1 and 2 to further expand our training dataset, ensuring a balanced representation of both classes.

As a result, our X train dataset consisted of threes a11, a21, a12 and b11, b21, b12 , while the Y train dataset contained the corresponding labels, either a22 or b22. The figure below show an example of the training dataset 4.3.

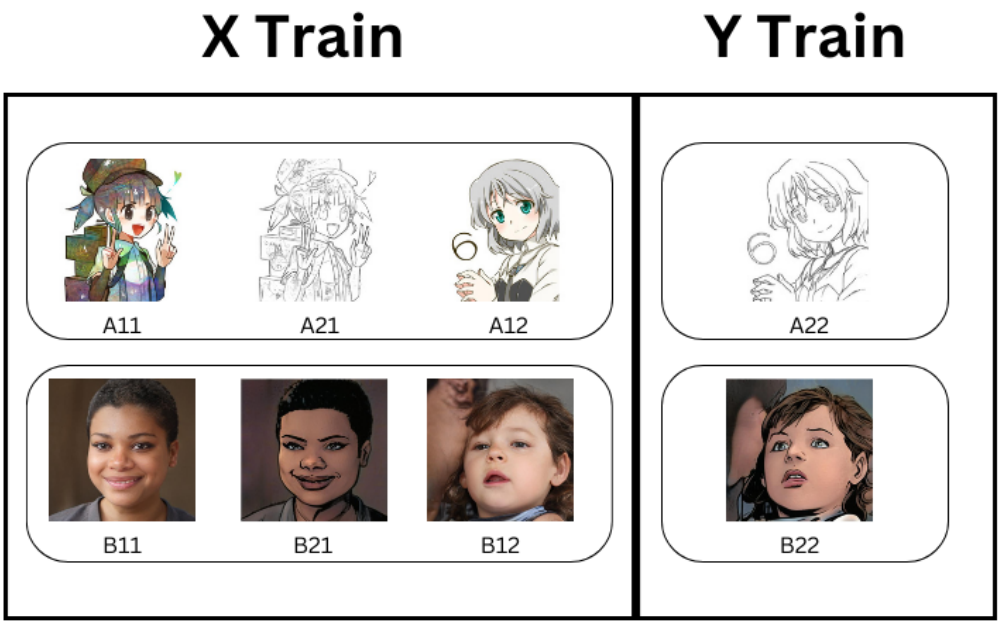


Figure 4.3: An example of two rows of the dataset.

Due to resource constraints, including limited GPU and RAM availability, we resized the images to a resolution of 125 by 125 pixels. This reduction in resolution allowed us to manage the computational requirements while still preserving essential visual features for the task at hand.

By following this prepossessing approach, we ensured a balanced training dataset, paired images from both classes, and appropriately formatted input and output data for our model. The downsampling of the images to 125 by 125 pixels enabled efficient processing.

4.2 Our Framework

We developed our model using Python and Keras. it was trained using the GPU T4 and 12.7 GB of RAM provided by Colab, enabling efficient training.

4.3 Performance metrics

Performance metrics are numerical measurements that are used to assess the efficacy and standard of machine learning models and algorithms. These metrics give information on a model's performance, the precision of its predictions, and whether it achieves the goals of a particular task, like :

4.3.1 Confusion matrix

A performance evaluation tool for classification and machine learning applications is the confusion matrix. By summarizing the outcomes of the predictions made on a set of test data, it aids

in our understanding of the performance of a classification model. The matrix, which shows the actual and anticipated classes or labels, is commonly a square matrix. figure 4.4 represents the confusion matrix for binary classification.

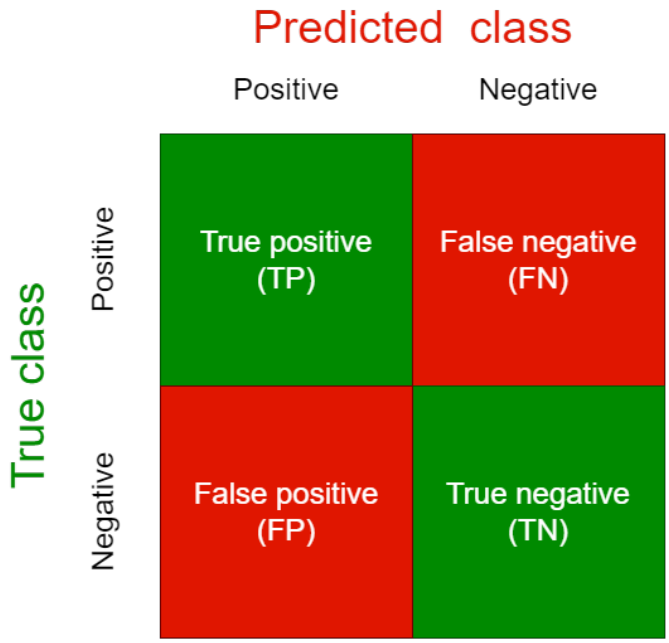


Figure 4.4: Results evaluation using The Confusion matrix.

The four cells of a confusion matrix indicate various combinations of real and anticipated labels:

- True Positive (TP): The model correctly predicted the positive class.
- True Negative (TN): The model correctly predicted the negative class.
- False Positive (FP): The model incorrectly predicted the positive class when the actual class was negative.
- False Negative (FN): The model incorrectly predicted the negative class when the actual class was positive.

We can compute a number of performance metrics, including accuracy, precision, recall (sensitivity), specificity, and F1 score, using the confusion matrix:

1. **Accuracy :** A classification model's accuracy is determined by dividing the number of examples that were properly classified by the total number of instances in the dataset. Accuracy offers a broad evaluation of the model's performance, but when the dataset is unbalanced, it might be deceptive, it is given by the formula:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

2. **Precision :** Precision measures a classification model's capacity to accurately identify positive occurrences. It focuses on reducing false positives when the cost of false positives is high, such as in spam detection or medical diagnosis, it is calculated as:

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

3. **Recall (Sensitivity):**Recall measures the ability of a classification model to correctly identify positive instances out of all actual positive instances. It is important in situations where identifying all positive instances is crucial, such as disease detection, where missing a positive case can have severe consequences, it is computed using the formula:

$$Recall = \frac{TP}{TP + FN} \tag{4.3}$$

4. **Specificity :**Specificity measures the ability of a classification model to correctly identify negative instances, especially in fraud detection and quality control, where false positives need to be minimized, It is given by:

$$Recall = \frac{TN}{TN + FP} \tag{4.4}$$

5. **F1-Score :** The F1 score, which provides a balanced assessment of a model's performance, is a harmonic mean of precision and recall. Both false positives and false negatives are taken into account. The F1 score has a range of 0 to 1, with 1 denoting flawless recall and precision and 0 denoting the worst performance, It is calculated using the formula:

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{4.5}$$

4.3.2 Receiver operating characteristic curve (ROC)

The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various classification thresholds. The thresholds can be adjusted to control the balance between true positives and false positives. By plotting these values on a graph, we can visualize the model's performance across different thresholds. Figure 4.5 gives an example for roc curve.

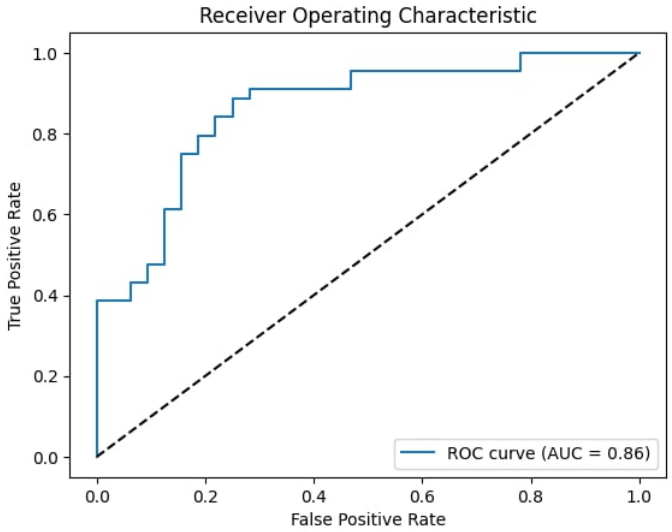


Figure 4.5: example of the ROC curve .

1. **True Positive Rate (TPR) :** The True Positive Rate (TPR), also known as sensitivity or recall, we have mentioned before.

2. **False Positive Rate (FPR):** It represents the proportion of actual negative cases incorrectly classified as positive by the model. It is calculated as the number of false positives divided by the sum of false positives and true negatives:

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{4.6}$$

4.4 Experiment 1

4.4.1 Result of our proposed augmentation method

The cross-section and longitudinal MRI images are present in 1125 original images. To train our model, we generated 19k different triads at random. A total of 65M parameters made up our model. 100 epochs of training were performed on the model with a 128-batch size. To assess the model's performance, we set aside 15% of the data for validation purposes. and the loss function used is "mean absolute error", and we note that the images are between 0 and 255. The difference between training and validation, as well as the very small loss, allowed the model to produce excellent results, as we observe in this figure 4.6 :

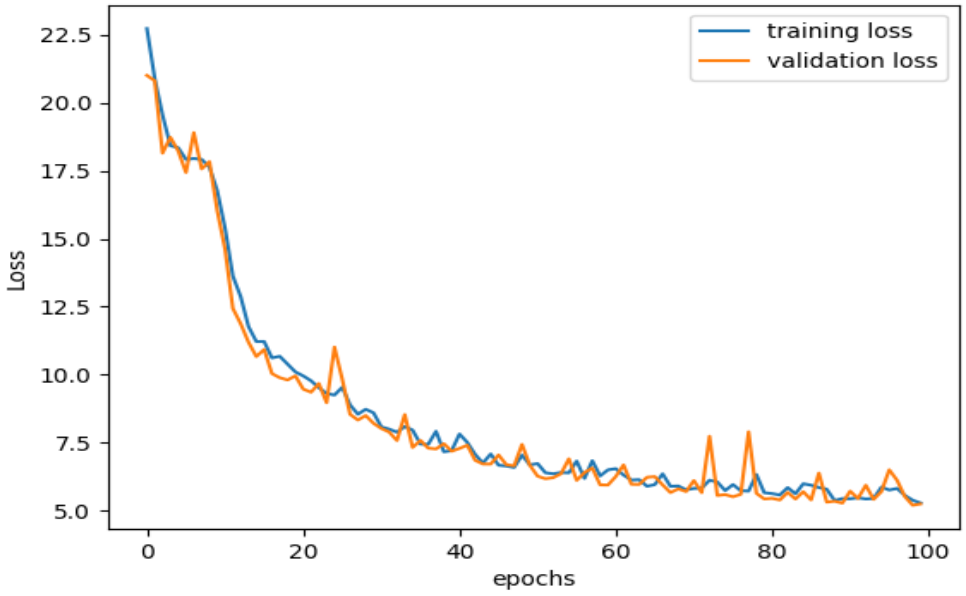


Figure 4.6: diagram of first proposed augmentation method loss

We can note the quality of the images when compared to the original images. To illustrate this, see the figure 4.7 :

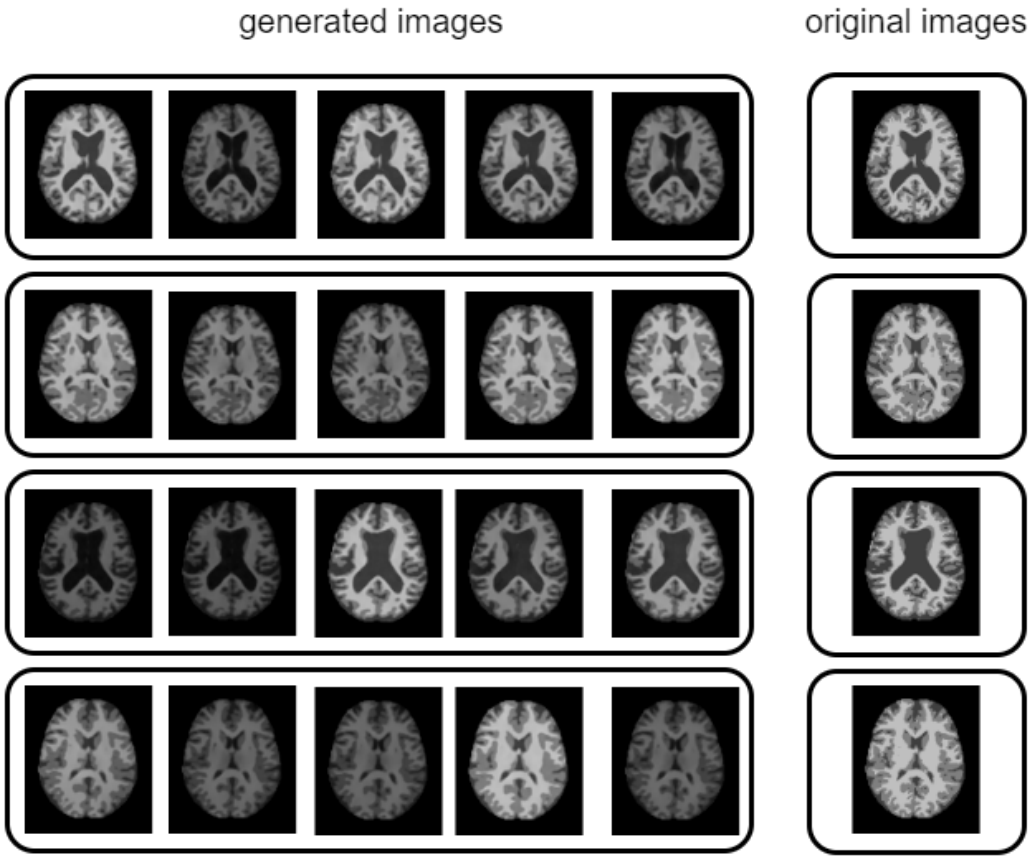


Figure 4.7: Some examples of proposed augmentation outputs.

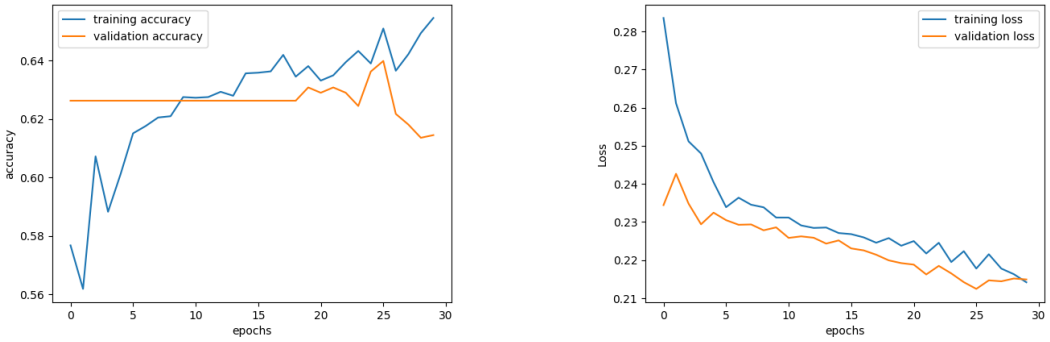
As we said earlier that to carry out the classification task, we make sure that subject B, whose data we are increasing, belongs to the 161 subjects who have complete information. So we tested with these 161 subjects and the loss was by the amount of 5.2737

4.4.2 Result of our proposed classification method

Our model had 20.2M parameters altogether. We use K-fold cross-validation with k=5 and 25 epochs for each k with a 128-batch size for training and validation data, where we calculated the mean of k-fold results in accuracy, loss, and training evolution. We trained 3 models: one of them without augmentation and the second with handcrafted augmentation and the last one with the proposed augmentation method From here, we can see more specifics of loss and accuracy:

4.4.2.1 classification without augmentation

In order to train the classification model with 320 original images of cross-section MRI, we can see the evolution of the results of loss and accuracy in figure 4.8 :



(a) Diagram of accuracy.

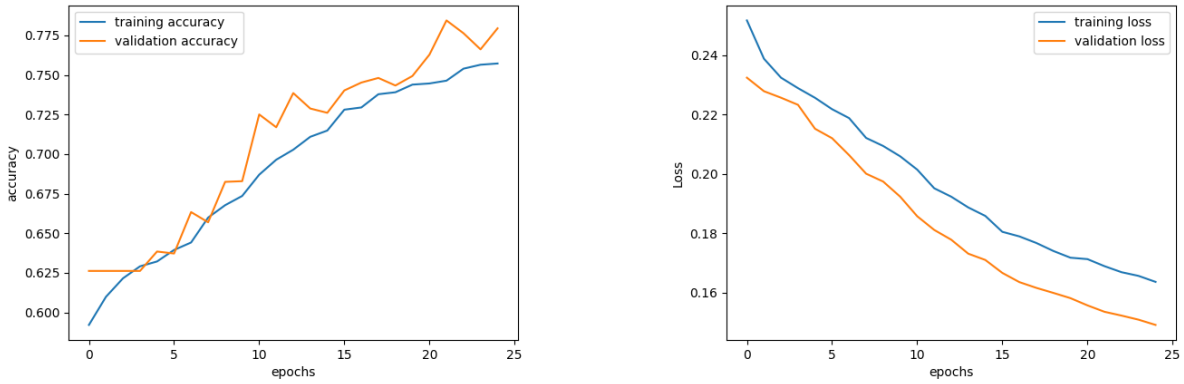
(b) Diagram of loss.

Figure 4.8: Loss and accuracy updating of classification without augmentation.

We can observe training accuracy improvement with each passing epoch. However, validation is wobbly and unstable, although the loss of training and validation is steadily decreasing, that is not enough.

4.4.2.1 Classification with handcrafted augmentation

In order to train the second classification model, we augmented our data from 320 images to 9630 images with flipping, rotation, and shifting. Figure **??** shows how the outcomes of loss and accuracy have changed over time.



(a) Diagram of accuracy.

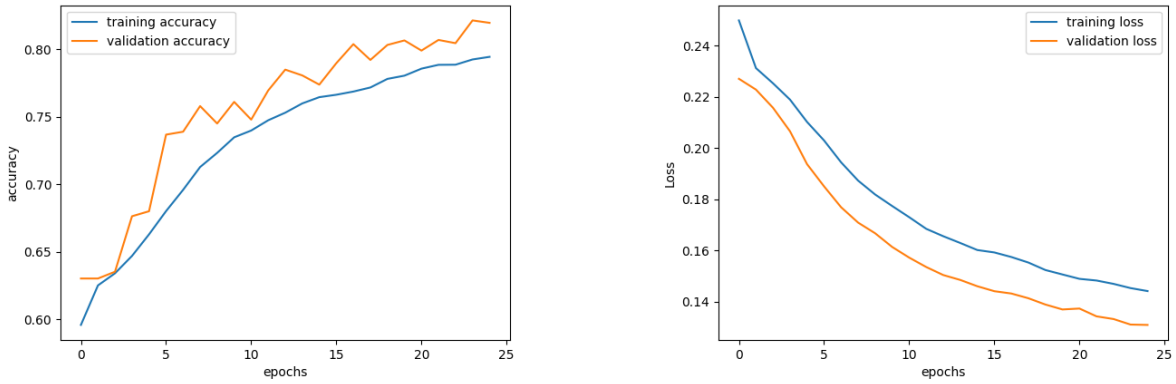
(b) Diagram of loss.

Figure 4.9: Loss and accuracy updating of classification with handcrafted augmentation.

With each succeeding epoch, the training and validation accuracy grew consistent and improved, and the validation accuracy eventually reached 78%. This is an example of the impact of the augmentation strategy.

4.4.2.3 Classification with our proposed augmentation method

We augmented the dataset we used for the augmentation challenge from 320 to 8370 photos in order to train the classification model.Figure 4.10 shows the results for loss and accuracy



(a) Diagram of accuracy.

(b) Diagram of loss.

Figure 4.10: Loss and accuracy updating of classification with first proposed augmentation.

And we evaluated the three models with the same data and compared them, and the results were summarized in this table with the best results from the paper [14], paper [11], and the paper [50], as the best results based on transformer architectures, its name Data-efficient image Transformers (DeiT) used many slides from MRI images. The table represents these results :

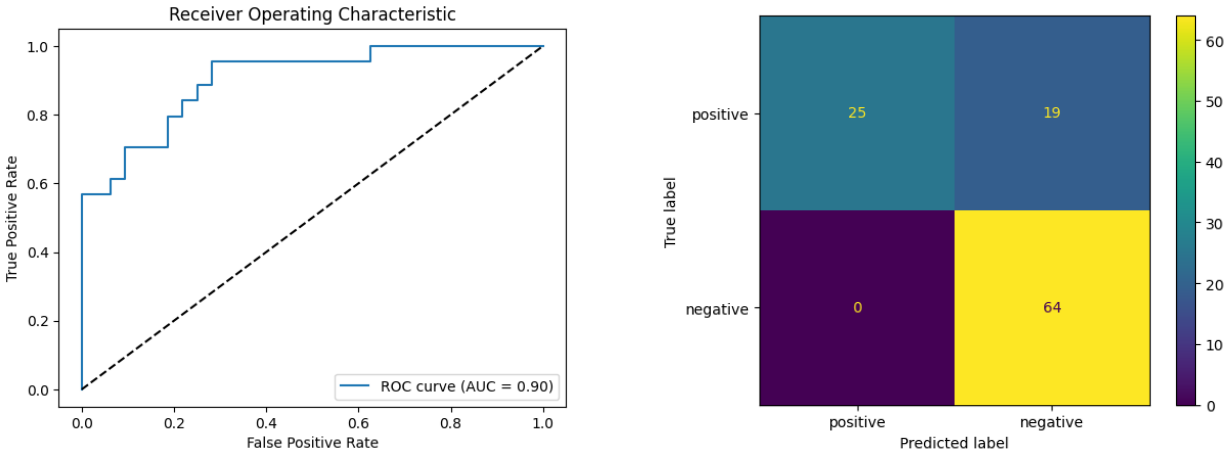
Methods	Accuracy
without augmentation	60.19%
with handcrafted augmentation	76.85%
with proposed augmentation	**82.41%**
DeiT (8 slides) [14]	75.6%
VGG16-v1 (8 slices) [50]	66.0%
VGG16-v2 (8 slices) [50]	66.1%
ResNet-18 (8 slices) [50]	68.8%
VGG16 (10 slices) [11]	71.6%

Table 4.1: Table of comparison between many methods.

We can analyze the results of our proposed method further by seeing the confusion matrix (figure 4.11(b)), receiver operating characteristic (ROC) curve (figure 4.11(a)), and the classification report (table 4.2) below:

	Precision	Recall	F1-score	Support
Class 0 (negative)	0.77	1.00	0.87	64
Class 1 (positive)	1.00	0.57	0.72	44
Accuracy		0.82		108

Table 4.2: Table of classification report of proposed classification method.



(a) Model evaluation with ROC.

(b) Model evaluation with confusion matrix .

Figure 4.11: ROC and confusion matrix of classification.

Although the obtained results seem to be very promising, it must be taken into account that we are working on a medical task, where we pay attention in this task that the recovery value of category 1 or positive category is relatively high because it represents the percentage of accuracy in detecting disease cases, as it is dangerous to make a mistake specifically in this category.

In These cases can sacrifice a little accuracy in exchange for an increase in recall accuracy, by changing the default threshold in the prediction and classification process, where we changed the threshold from 0.5 to 0.34, and the results can be considered better despite the decrease in accuracy to 81%, The results can be clearly seen in the classification report table 4.3 and confusion matrix in figure 4.12 after the changing:

	Precision	Recall	F1-score	Support
Class 0 (negative)	0.88	0.78	0.83	64
Class 1 (positive)	0.73	0.84	0.78	44
Accuracy		0.81		108

Table 4.3: Table of classification report of proposed classification method.

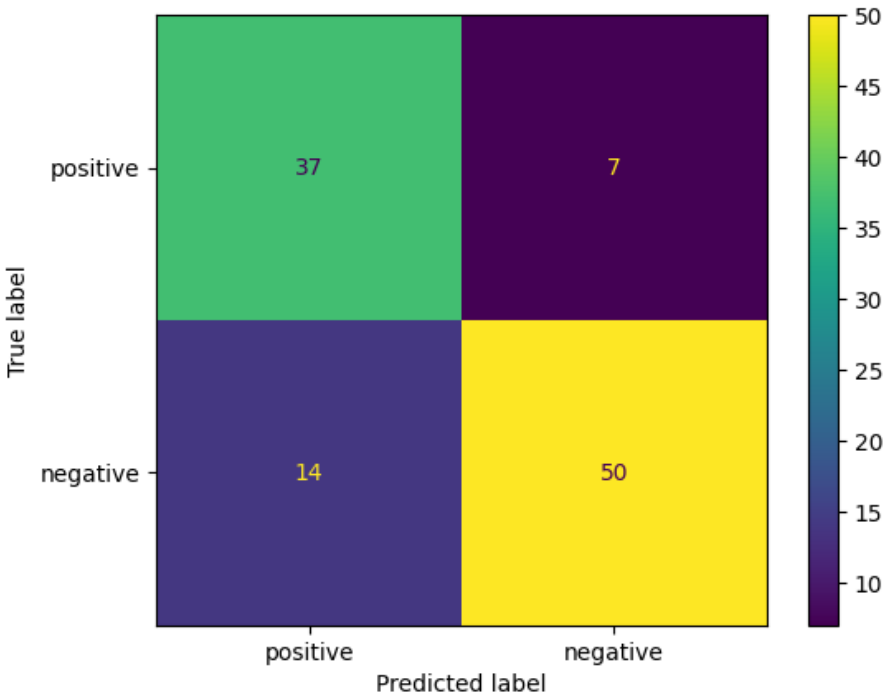


Figure 4.12: Confusion matrix of classification after changing threshold .

4.5 Result of the proposed approach for the task of generating images based on artistic transformation

In this section, we present the results generating performance of our model, which focused on training an architecture capable of generating images based on learned transformations. The model was trained for 25 epochs using a dataset of 5,000 threes images. Our findings indicate that the model successfully learned transformations despite the relatively small amount of data and a limited number of parameters, as we mentioned before, with just approximately 600,000 parameters.

4.5.1 Transformations result of class A

in this example 4.13, we tested the model with a cartoon image (class A), we selected random pairs of images from class A to evaluate the first transformation and pairs of images from class B to evaluate the second transformation, the figure below illustrates the results of applying transformation 1 (corresponding to the pairs from class A) and transformation 2 (corresponding to the pairs from class B).

Figure 4.13: Result of an image from class A.

4.5.2 Transformations result of class A

Additionally, we performed a similar evaluation for class B (real faces) 4.14.

Figure 4.14: Result of an image from class B.

To assess the training progress and the performance of our modified model, we monitored the training and validation losses throughout the training process. It's worth noting that the images range between 0 and 255.

4.5.3 Model evaluation

The loss and validation loss curves of our model 4.15 exhibit an interesting pattern during training. The training loss consistently decreases over the epochs, indicating that the model is learning from the paired images and improving its performance. However, the validation loss shows a more fluctuating behavior. Initially, it decreases slightly, but then it experiences temporary spikes followed by subsequent drops and increases. Despite these fluctuations, the validation loss eventually stabilizes and continues to decrease consistently after epoch 20.

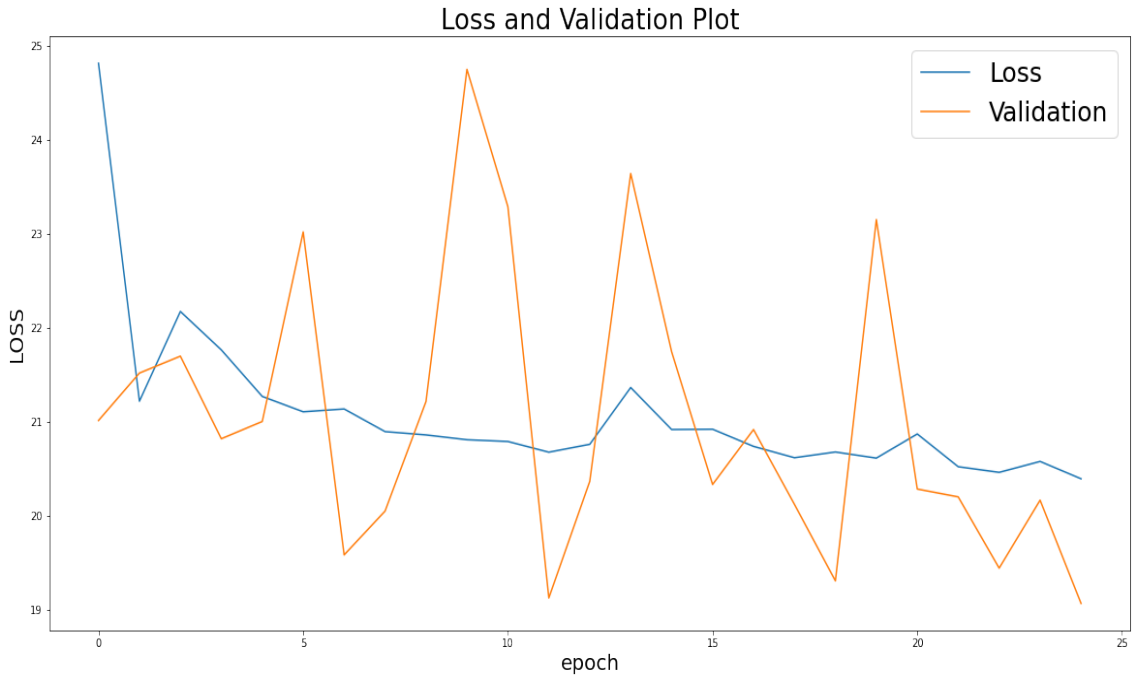


Figure 4.15: Training and validation losses.

4.6 Discussion

In our first experiment, we observed a noticeable enhancement in the results after applying augmentation using our method. This outcome suggests that our model exhibits strong generating capabilities, effectively learning the transformations between different aspects of the images.

While our approach yielded promising results, it is important to acknowledge the potential advantages of hand augmentation in certain cases. Hand augmentation, when performed by individuals with prior knowledge and expertise in the dataset, may yield even better outcomes. The human intervention allows for a nuanced understanding of the dataset characteristics and the ability to tailor the augmentation process accordingly.

However, one of the notable advantages of our method is its ability to extract transformations without requiring human intervention. This is particularly valuable in scenarios where extensive domain knowledge or expertise might not be readily available. By automating the transformation extraction process, our method provides a practical and efficient solution for generating augmented images.

In the second experiment, the effectiveness of skip connections in improving the generation performance was evident in our experiments. Despite the relatively few training steps, which took approximately 30 minutes, and the small number of parameters in the model, it was able to generate acceptable images, indicating successful learning of the transformation. It is worth noting that the datasets used for training and testing, Dataset 1 and Dataset 2,

respectively, were significantly different. Dataset 1 consisted of PNG images with a white background, while Dataset 2 was composed of JPG images. This discrepancy in image format and background color had an impact on the performance of the model. Additionally, due to material limitations, we had to decrease the resolution of the images, which may have affected the overall performance. One significant drawback of this modified method is the substantial consumption of RAM during the training and inference stages. This is primarily attributed to the concatenation of the original images with other layers throughout the architecture. As the image resolution increases, the RAM requirements also escalate, leading to potential memory constraints, especially when dealing with high-resolution images. The need to store and process the concatenated feature maps, along with the original images, imposes significant demands on the available system resources. This high RAM consumption can pose challenges, particularly when working with limited computational resources or large-scale datasets. Mitigating this issue may require careful memory management techniques, such as reducing batch sizes, optimizing data loading strategies, or considering alternative architectures that can handle larger images more efficiently.

Nevertheless, the model demonstrated its ability to adapt and generate reasonable results within these constraints.

4.7 General Conclusion

The ability to improve generalization is of paramount importance, as it allows deep learning models to perform well on unseen data, ensuring robust performance in real-world settings. By addressing the challenge of limited data, our proposed method offers a practical solution for scenarios where obtaining large labeled datasets is challenging or costly.

Furthermore, the method's adaptability is a key advantage, enabling its utilization in a range of applications beyond the medical domain. Through strategic architectural adaptations, the method can be tailored to different tasks, accommodating the specific requirements and characteristics of various problem domains. This flexibility enhances the method's potential for broader impact and opens up possibilities for innovative applications across various industries.

Bibliography

[1] https://paperswithcode.com/method/vgg.

[2] https://premierimaging.org/how-to-prepare-for-an-mri/.

[3] https://towardsdatascience.com/why-rectified-linear-unit-relu-in-deep-learning-and-the-best-practice-to-use-it-with-tensorflow-e9880933b7ef.

[4] https://www.123rf.com/photo$_2$5665107$_v$$iew-of-a-child-xray-film-taken-to-examine-the-lungs.html$.

[5] https://www.advancinganalytics.co.uk/blog/2021/12/15/understanding-the-difference-between-ai-ml-and-dl-using-an-incredibly-simple-example.

[6] https://www.cancer.gov/publications/dictionaries/cancer-terms/def/abdominal-ultrasound.

[7] https://www.geeksforgeeks.org/residual-networks-resnet-deep-learning.

[8] https://www.omegapds.com/difference-between-an-mri-ct-and-x-ray-scan/.

[9] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[10] Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2020.

[11] Selamawet Workalemahu Atnafu and Stefano Diciotti. Development of an interpretable deep learning system for the identification of patients with alzheimer's disease. 2022.

[12] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[13] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.

[14] Pierluigi Carcagnì, Marco Leo, Marco Del Coco, Cosimo Distante, and Andrea De Salve. Convolution neural networks and self-attention learners for alzheimer dementia diagnosis from brain mri. *Sensors*, 23(3):1694, 2023.

[15] Tianfeng Chai and Roland R Draxler. Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. *Geoscientific model development*, 7(3):1247–1250, 2014.

[16] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 113–123, 2019.

[17] Matthijs Douze, Arthur Szlam, Bharath Hariharan, and Hervé Jégou. Low-shot learning with large-scale diffusion. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3349–3358, 2018.

[18] Giovanni B Frisoni, Nick C Fox, Clifford R Jack Jr, Philip Scheltens, and Paul M Thompson. The clinical use of structural mri in alzheimer disease. *Nature Reviews Neurology*, 6(2):67–77, 2010.

[19] Hang Gao, Zheng Shou, Alireza Zareian, Hanwang Zhang, and Shih-Fu Chang. Low-shot learning via covariance-preserving adversarial augmentation networks. *Advances in Neural Information Processing Systems*, 31, 2018.

[20] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, 2011.

[21] Lovedeep Gondara. Medical image denoising using convolutional denoising autoencoders. pages 241–246, 2016.

[22] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[23] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[25] Zikun Hu, Xiang Li, Cunchao Tu, Zhiyuan Liu, and Maosong Sun. Few-shot charge prediction with discriminative legal attributes. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 487–498, 2018.

[26] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[27] Shaoqing Ren Jian Sun Kaiming He, Xiangyu Zhang. Deep residual learning for image recognition. *arXiv:1512.03385*, 2015.

[28] Andrew Zisserman Karen Simonyan. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2015.

[29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[30] Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–44, 05 2015.

[31] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

[32] Xiaojing Long, Lifang Chen, Chunxiang Jiang, Lijuan Zhang, and Alzheimer's Disease Neuroimaging Initiative. Prediction and classification of alzheimer disease based on quantification of mri deformation. *PloS one*, 12(3):e0173372, 2017.

[33] Zelun Luo, Yuliang Zou, Judy Hoffman, and Li F Fei-Fei. Label efficient learning of transferable representations acrosss domains and tasks. *Advances in neural information processing systems*, 30, 2017.

[34] T. Soni Madhulatha. An overview on clustering methods. 2012.

[35] Erik G Miller, Nicholas E Matsakis, and Paul A Viola. Learning from one example through shared densities on transforms. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 1, pages 464–471. IEEE, 2000.

[36] Andreas C Müller and Sarah Guido. *Introduction to machine learning with Python: a guide for data scientists*. " O'Reilly Media, Inc.", 2016.

[37] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks, 2015.

[38] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[39] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[40] Eli Schwartz, Leonid Karlinsky, Joseph Shtok, Sivan Harary, Mattias Marder, Abhishek Kumar, Rogerio Feris, Raja Giryes, and Alex Bronstein. -encoder: an effective sample synthesis method for few-shot object recognition. In *Annual Conference on Neural Information Processing Systems*. Neural information processing systems foundation, 2018.

[41] Pranav Shyam, Shubham Gupta, and Ambedkar Dukkipati. Attentive recurrent comparators. In *International conference on machine learning*, pages 3173–3181. PMLR, 2017.

[42] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[43] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

[44] Carlos Oscar Sánchez Sorzano, Javier Vargas, and A Pascual Montano. A survey of dimensionality reduction techniques. *arXiv preprint arXiv:1403.2877*, 2014.

[45] Paul Suetens. *Fundamentals of medical imaging*. Cambridge university press, 2017.

[46] Kanchan M Tarwani and Swathi Edem. Survey on recurrent neural network in natural language processing. *Int. J. Eng. Trends Technol*, 48(6):301–304, 2017.

[47] US National Institute on Aging. What happens to the brain in alzheimer's disease, Unknown. Accessed: September 27, 2022.

[48] World Health Organization. Global action plan on the public health response to dementia 2017–2025, 2017. Accessed: June 10, 2023.

[49] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[50] Ekin Yagis, Selamawet Workalemahu Atnafu, Alba García Seco de Herrera, Chiara Marzi, Riccardo Scheda, Marco Giannelli, Carlo Tessa, Luca Citi, and Stefano Diciotti. Effect of data leakage in brain mri classification using 2d convolutional neural networks. *Scientific reports*, 11(1):22544, 2021.

[51] James Kwok Lionel M. Ni Yaqing Wang, Quanming Yao. Generalizing from a few examples: A survey on few-shot learning. *arXiv:1904.05046*, 2020.

[52] Yabin Zhang, Hui Tang, and Kui Jia. Fine-grained visual categorization using meta-learning optimization with sample selection of auxiliary data. In *Proceedings of the european conference on computer vision (ECCV)*, pages 233–248, 2018.