الجزائرية الديمقراطية الشعبية
The People's Democratic Republic of Algeria
وزارة التعليم العالي والبحث العلمي
Ministry of Higher Education and Scientific Research
جامعة محمد بوضياف بالمسيلة
University Mohamed Boudiaf of M'sila

قسم الإعلام الآلي
Department of Computer Science

كلية الرياضيات والإعلام الآلي
Faculty of Mathematics and Informatics

Domain: Mathematics and Computer Science

Thesis Presented to Fulfill the Partial Requirement
for Master's Degree in Computer Science

Specialty: Artificial Intelligence

Prepared By: Merzougui Abdellatif

Supervised By: Dr. Rahima Bentrcia

ENTITLED

Face Prediction System for Missing Children

Jury Members

Dr. Nourelhouda Chalabi	President
Dr. Rahima Bentrcia	Supervisor
Dr. Khadidja Derdour	Examiner

Academic Year 2024/2025

Dedication

To my beloved family, I dedicate this work to you as an expression of my thanks and gratitude for everything you have given me. To those who have had the credit after God for everything I have achieved, to my support and support, to those who have provided me with continuous, support and encouragement.

Acknowledgement

First and above all, we thank “ALLAH” for giving me the patience, the health, the courage to have come this far.

I would like to thank Dr. Rahima Bentrcia, my supervisor, for her continuous support, valuable guidance and valuable comments. Her experience and encouragement were crucial to the completion of this work.

Abstract

This study aims to explore and use competitive generative networks (GANs) with the PixtoPix approach to predict images of a child based on a photo taken of him. The generator is trained to create images of the child, while the discriminator is trained to distinguish real images from generated images. The model was applied to a data set contains different images. The results showed that this model can generate predicted images.

ملخص

تهدف هذه الدراسة إلى استكشاف واستخدام الشبكات التوليدية التنافسية للتنبؤ بصور طفل بناءً على صورة تم التقاطها له. يتم تدريب المولد لإنشاء صور للطفل، بينما يتم تدريب المميز للتمييز بين الصور الحقيقية والصور المولدة. تم تطبيق النموذج على مجموعة بيانات. تحتوي على صور مختلفة. أظهرت النتائج أن هذا النموذج يمكنه توليد صور متوقعة

Résumé

Cette étude vise à explorer et à utiliser des réseaux génératifs compétitifs (GAN) avec l'approche PixtoPix pour prédire les images d'un enfant à partir d'une photo prise de lui. Le générateur est formé pour créer des images de l'enfant, tandis que le discriminateur est formé pour distinguer les images réelles des images générées. Le modèle a été appliqué à un ensemble de données. Contient différentes images. Les résultats ont montré que ce modèle peut générer des images prédites

Table of Contents:

List of Figures

List of Tables

General Introduction

The world is witnessing many problems, including what is known as lost children. In light of the development of artificial intelligence technology in general and adversarial neural networks in particular, we proposed a system to predict the face of a lost child based on the captured image of him, and through a research and analytical methodology that links the theoretical background and the application of the proposed system, and as a result we evaluated the performance of this system through practical experiments, which may contribute to returning children to their families.

CHAPTER 1:

Face prediction for missing children

1.1 Introduction:

In this chapter, we review a comprehensive view of the phenomenon of child loss and its impact on society and families. We also briefly discuss the method used and the data, and then we report and discuss the general organization of the thesis.

1.2 Overview:

Throughout the ages, the phenomenon of loss of children has been one of the greatest human tragedies in many countries of the world. This phenomenon has emerged and the suffering of those who suffer it and their families have increased in recent times due to the exacerbation of armed conflicts and the spread of internal violence, as well as other factors that have contributed to exposing many people to material and moral loss, it is known that the issue of loss has been included in many legal and social studies, but the issue of losing children was not of interest in these studies and did not receive sufficient research even though this topic is considered a human tragedy with a serious dimension because children are from the vulnerable groups and are most at risk of missing.

In our system, we relied on the generative adversarial networks (GANs) technology, which generates images using generative adversarial networks. In this context, we exploited this technology to predict the shape of children’s faces based on the available image.

A dataset named FGnet is used in this research, which contains a large collection of images of real faces with age-related metadata. GANs model is trained on this data set to predict and generate facial images of missing children.

Finally, this system contributes to strengthening efforts to find missing children and provide technical support to solve this important humanitarian problem.

1.3 Thesis hypothesis:

The model is provided with two images of the same face across two different age stages, trying to use the facial features of the first image to create the expected image based on the training it took from the other images. The images are usually blurry or low-resolution, but clear shapes can be expected with the Image-to-Image Translation method an effective solution to the challenges of searching for young children through their faces in advance of their previous photos, this can lead to effective results in searches, which unreasonably limits the need for its own hardware analysis.

Deoxyribonucleic acid (DNA) analysis, although considered an expensive and time-consuming process, is widely researched in daily operations. In addition, these analyzes depend on obtaining biological patterns, which may not always be available in a timely manner. Hence, there is an urgent need for alternative technologies that are less expensive and faster in bringing about change.

This project will contribute to improving the accuracy and efficiency of tracing systems, and reduce the costs associated with traditional procedures such as DNA analyses by providing an innovative and effective technical solution, we seek to achieve a tangible positive impact in the lives of missing children and their families and strengthen the community response to this critical humanitarian problem.

1.4 Thesis outline:

This research is organized as follows:

In **chapter 1,** general concepts about the problem and the proposed solution are illustrated.

In **chapter 2,** We talk about the background of this model and this part is divided into artificial intelligence, machine learning and deep learning.

In **chapter 3,** We delve deeper into the structure of GANs and also clarify how the generator and discriminator work.

In **chapter 4,** We document all the methodological stages, from our handling of the data to the production of the expected images.

In **chapter 5,** one of the important stages is the evaluation stage. We used metrics including precision, recall, and F1-score to evaluate the model, and the results are clearly reported and analyzed.

1.5 Conclusion:

After reviewing these concepts, we found that we can coordinate between the method and the current problem to create a system that can predict the face of the missing child, as we will see in the next chapter the general background of the system.

CHAPTER 2:

Background

2.1 Introduction:

This section aims to provide the comprehensive conceptual background that lays the foundation for understanding the research method. All the basic concepts and components related to the generative adversarial network method is addressed.

2.2 Artificial Intelligence:

Artificial intelligence (AI) refers to computer systems capable of performing complex tasks that historically only a human could do, such as recognizing speech, making decisions, or identifying patterns. Today, the term "AI" encompasses a wide variety of technologies, including machine learning, deep learning, and natural language processing (NLP) [1]. These technologies power many of the services and goods we use daily, from recommendation apps to real-time customer support chatbots. While there are philosophical debates about whether these technologies truly constitute "true" artificial intelligence, most people commonly associate AI with machine learning-powered applications like Chat GPT or computer vision, enabling machines to perform tasks previously exclusive to humans.
AI is an umbrella term that encompasses a wide variety of technologies, including machine learning, deep learning, and natural language processing, as shown in Figure 2. 1.

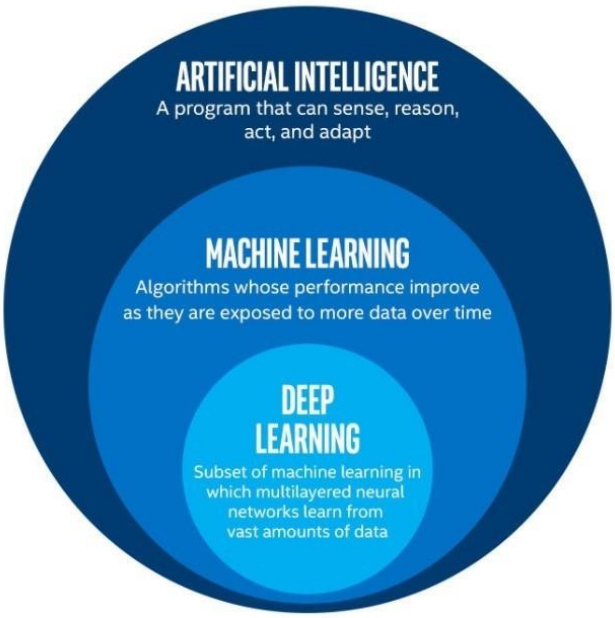

Figure 2.1: Artificial intelligence

2.3 Machine Learning:

Machine learning (ML) is generally considered a subfield of AI, and even a subfield of computer science from some perspectives. In the last few years, especially thanks to the recent advancements in the field of Deep Learning (DL), ML has been drawing a lot of attention.

In ML, observation of data, such as examples, direct experiences, or instructions, is the first step in learning so that patterns in the data can be seen and better decisions can be made based on the examples provided. Generally, the goal is to make computers learn automatically without human intervention and adjust their actions accordingly. The following definition is a better description for machine learning. " A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E " [2].

Many kinds of tasks can be solved with machine learning. Some of the most common machine learning tasks include the following: Classification, Classification with missing inputs, Regression, Transcription, Machine translation, Structured output, Anomaly detection, Synthesis and sampling, Imputation of missing values, and Density estimation [3].

Machine learning algorithms can be broadly categorized as unsupervised or super vised by what kind of experience they are allowed to have during the learning process. A semi-supervised learning

algorithm is a form of learning that falls somewhere between supervised and unsupervised. Supervised Learning, unsupervised learning, and semi-supervised learning are discussed in (Section 2.1.1). Some machine learning algorithms do not just experience a fixed dataset. For example, reinforcement learning algorithms interact with an environment, so there is a feedback loop between the learning system and its experience. Reinforcement learning is a machine learning training method based on rewarding desired behaviors and/or punishing undesirable ones. An agent that uses reinforcement learning is capable of perceiving and interpreting its environment, taking actions, and learning through trial and error [4].

2.3.1 Supervised versus Unsupervised Learning:

Supervised learning algorithms experience a dataset containing features, but each example is also associated with a label or target. Over time, the model can learn from both examples and correct labels in this training dataset. Loss functions are used to measure the accuracy of the algorithm, adjusting until the error has been sufficiently minimized. In general, supervised learning algorithms are applicable to two types of problems; classification problems and regression problems.

Classification refers to a predictive modeling problem where unlabeled test data would be assigned to specific classes by feeding it into a trained model. There are a number of common classification algorithms such as Neural networks, Naive Bayes, Support Vector Machine (SVM), K-nearest neighbor, and Random forest.

Regression is used to understand the relationship between independent and dependent parameters. It is widely used to make projections, such as for a business sales revenue. Popular regression algorithms include linear regression, logistic regression, and polynomial regression.

Unsupervised learning algorithms experience a dataset containing many features, then learn useful properties of the structure of this dataset [3]. Therefore, there is no label or target associated with training data. Using these data, the unsupervised learning algorithms find patterns that aid in clustering or association problems. When subject matter experts are unsure about common properties within a data set, this is especially useful. Hierarchical, k-means, and Gaussian mixture models are all commonly used clustering algorithms [5].

Semi-supervised learning is a type of learning that falls between supervised and unsupervised. It combines labeled and unlabeled examples to expand the available data pool for model training. It uses pseudo-labeling methods for predicting labels for unlabeled examples. Pseudo-labeling is the process

of using the labeled data model to predict labels for unlabeled data. This results in improved model performance and time and costs savings because we don't have to manually label thousands of examples.

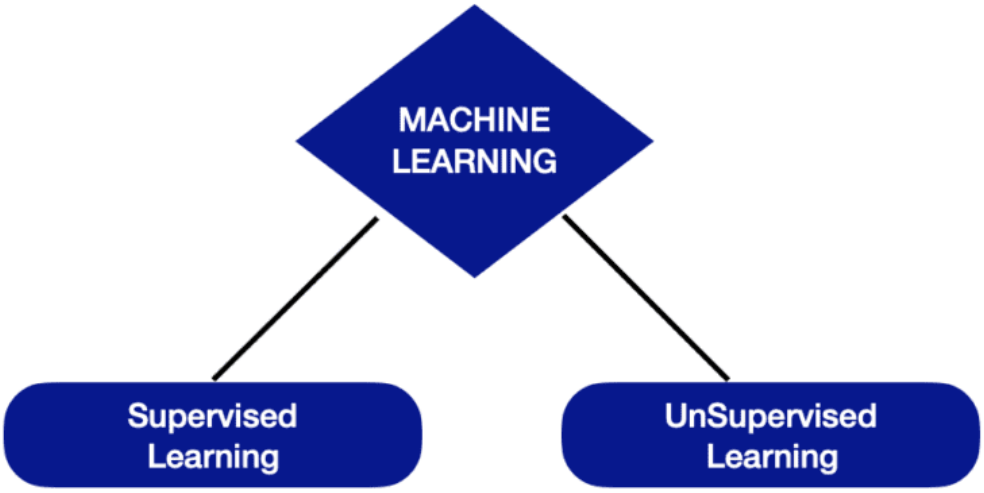

Figure 2.2: Machine learning

2.4 Deep Learning:

Deep learning is an advanced branch of artificial intelligence that relies on building and training deep neural networks, which are mathematical structures inspired by the way the human brain works. These networks are distinguished by their ability to automatically learn from data, as they gradually learn features and patterns from simple to complex layers, which can Performing complex tasks with high accuracy. The practical applications of deep learning are wide and diverse, including machine translation improvement, image classification, speech recognition, and even gaming.

One important innovation in the field of deep learning is generative adversarial networks (GANs). GANs are a special type of deep neural networks that consist of two models competing against each other: a generator and a discriminator. The generator creates new data that mimics the real data, while the discriminator tries to distinguish this fake data from the real data. This competitive interaction drives each model to improve its performance, generating high-quality data that closely resembles real data. Applications of GANs include image and video generation, image quality enhancement, and artistic style transformation, making them a powerful and innovative tool in many fields.

2.4.1 Generative Adversarial Networks:

Generative adversarial networks or GANs [6] are a generative modeling approach based on differentiable generator networks. In generative adversarial networks, the generator network competes

against an adversary based on a game-theoretic model. These networks work with two neural network models simultaneously. The first feed-forward neural network is a generative model, which creates synthetic examples of objects similar to real examples in the dataset. It is also intended to produce examples so realistic that a trained observer cannot distinguish whether a sample belongs to the original data set or it is generated. When we have a repository of car images, a generative network can generate synthetic examples of those images using a generative model. Thus, we now have images of real as well as fake cars. On the other hand, the second feed-forward neural network is a discriminative model that has been trained on a set of data that is tagged with the fact of whether the images are synthetic or real. This model takes inputs of either real examples from the original dataset or synthetic objects generated by the generator network and attempts to distinguish between the real and fake objects [7].

Using this analogy, the generative network could be viewed as a counterfeiter, producing fake notes, while the discriminative network could be viewed as the police, attempting to catch the counterfeiter. Due to this, the two networks are adversaries, and training makes them both better, until they reach an equilibrium [8].

When the discriminative network correctly flags a synthetic object as fake, the generative network modifies its weights to make it more difficult for the discriminative network to recognize samples generated from it, new samples are generated from the generator network after the weights are modified, and the process is repeated. It gets better and better at creating counterfeits over time. Eventually, the discriminator cannot differentiate between real objects and synthetic ones [9]. Formally, it can be shown that the Nash equilibrium of the minimax game is a parameter setting (generator) in which the distribution of points generated by the generator corresponds to that of the data samples .To make this approach work , it is crucial that the discriminator is a high-capacity model, and that it has access to a lot of data [10]. Figure 2.3 illustrates an overview of GAN framework. Due to the fact that both the networks have their own individual objectives, both are trying to optimize themselves to accomplish the same goal, i.e., D wants to maximize the cost function, and g wants to minimize the cost function, as described (1):

$$LGAN = V(D,g) = Ex[\log(D(x))]+Ez[\log(1 - D(g(z))] \quad .....(1)$$

where D(x) is the discriminator's estimate of the probability that real data instance x is real, Ex is the expected value over all real data instances, g(z) is the generator's output when given noise z, D(G(z)) is the discriminator's estimate of the probability that a fake instance is real, and Ez is the expected value

over all random inputs to the generator. From the cross-entropy between the real and generated distributions, the formula is derived.

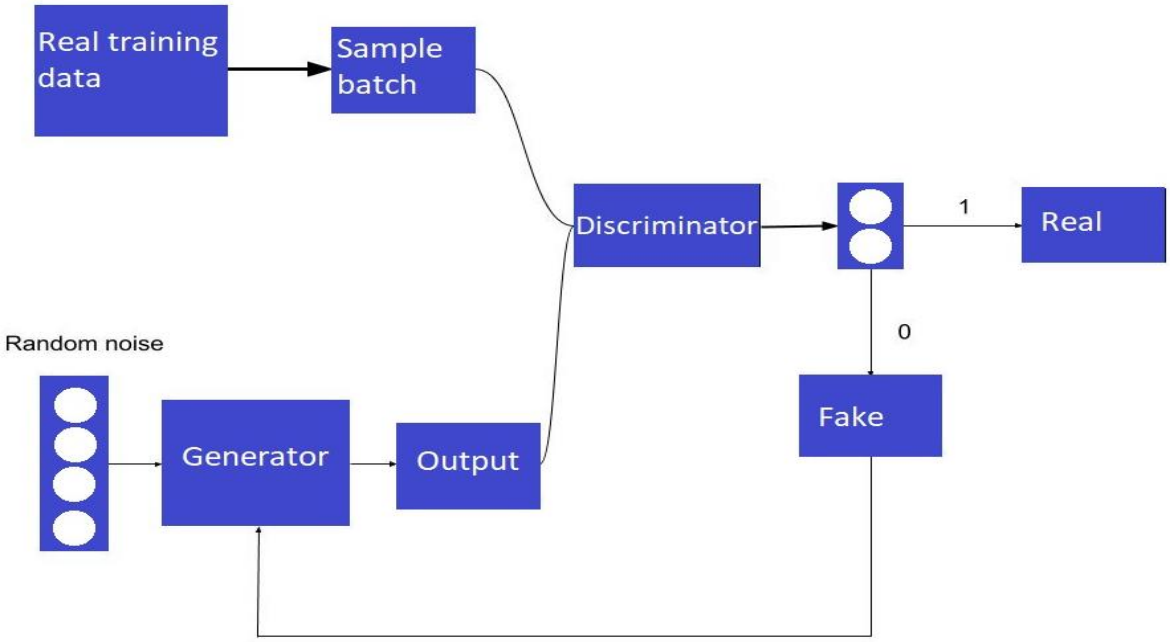

Figure 2.3: Standard GAN architectures

2.4.2 Conditional Adversarial Networks:

Generative adversarial Networks can be extended to a conditional model if both the generator and discriminator are conditioned on some extra information y. y could be any kind of auxiliary information, such as class labels or data from other modalities. We can perform the conditioning by feeding y into the both the discriminator and generator as additional input layer. In the generator the prior input noise pz(z), and y are combined in joint hidden representation, and the adversarial training framework allows for considerable flexibility in how this hidden representation is composed. 1 In the discriminator x and y are presented as inputs and to a discriminative function (embodied again by a MLP in this case) [11], The objective function of a two-player minimax would be as Eq 2 .

min G max D V(DG)=Ex pdata(x)[logD(xy)]+Ez pz(z)[log(1 D(G(zy)))].....(2)

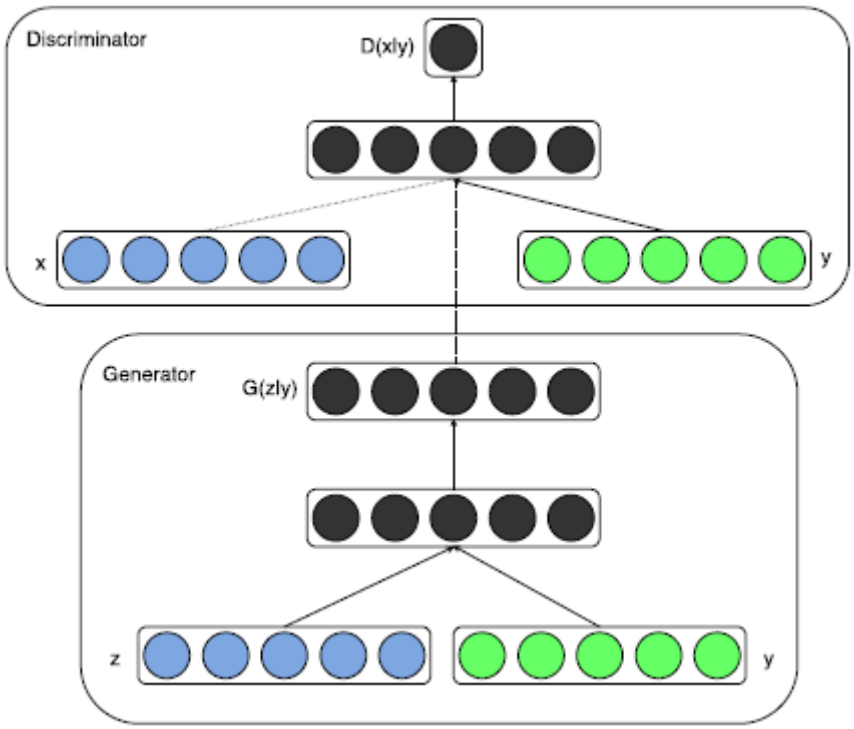

Figure 2.4: Architecture of a C-GANs

2.5 Conclusion:

By gradually going through all these concepts, we can now delve into the study of adversarial generative networks, in a systematic and organized manner.

CHAPTER 3:

Generator Adversarial Networks

3.1 Introduction:

This part is the focus of the research because we will see the details of GANs in the Conditional Generative Adversarial Network CGAN method or what is called PixtoPix and explain the details of the discriminator and generator as well as the loss functions for both of them.

3.2 Overview:

The core methodology of PixtoPix is grounded in the GAN framework but is specifically tailored to handle paired image data, in a GAN, two neural networks the Generator and the Discriminator compete against each other, the Generator creates images attempting to be as realistic as possible, while the Discriminator evaluates whether these images are real or generated. In Pix2Pix, this process is conditional, meaning that both the Generator and the Discriminator are provided with additional information in the form of the input image.

3.3 Generator:

The generator model takes an image as input, and unlike the traditional GAN model, it does not take a point from the latent space as input. Instead, the source of randomness comes from the use of dropout layers that are used during training and when making predictions.

The U-Net model is used for the generator architecture instead of the common encoder-decoder model, the encoder-decoder architecture involves taking an image as input, downgrading its resolution over several layers until it reaches the bottleneck layer [12], and then upgrading the resolution again over several layers until the final image of the desired size is produced.

This model is very similar to encoder-decoder model in that it involves lowering the resolution to the bottleneck and then raising it again, but links or transient connections are made between layers of the

same size in the encoder and decoder, allowing the bottleneck to be bypassed, for example, the first layer of encoding is combined with the last layer of decoding of the same size, and this is repeated with each layer in the encoder and its corresponding layer in the decoder, forming a U-shaped pattern.

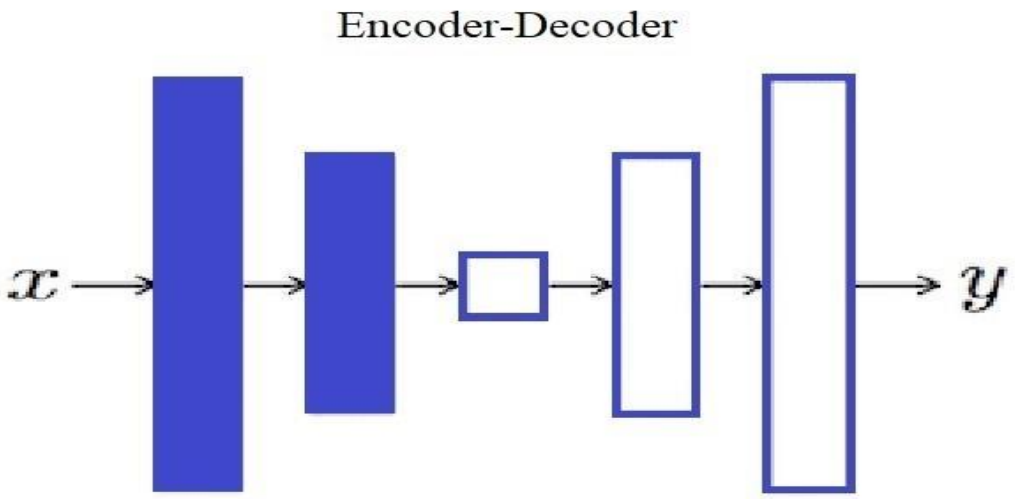

Figure 3.1: Simple encoder-decoder

Skip connection was introduced in Residual Network (ResNet) and showed prediction improvements as well as smoother learning gradients [13]. Inspired by this mechanism, we have to add skip connections to U-Net such that every decoder incorporates the feature map from its corresponding encoder. This is a defining feature of U-Net [14]:

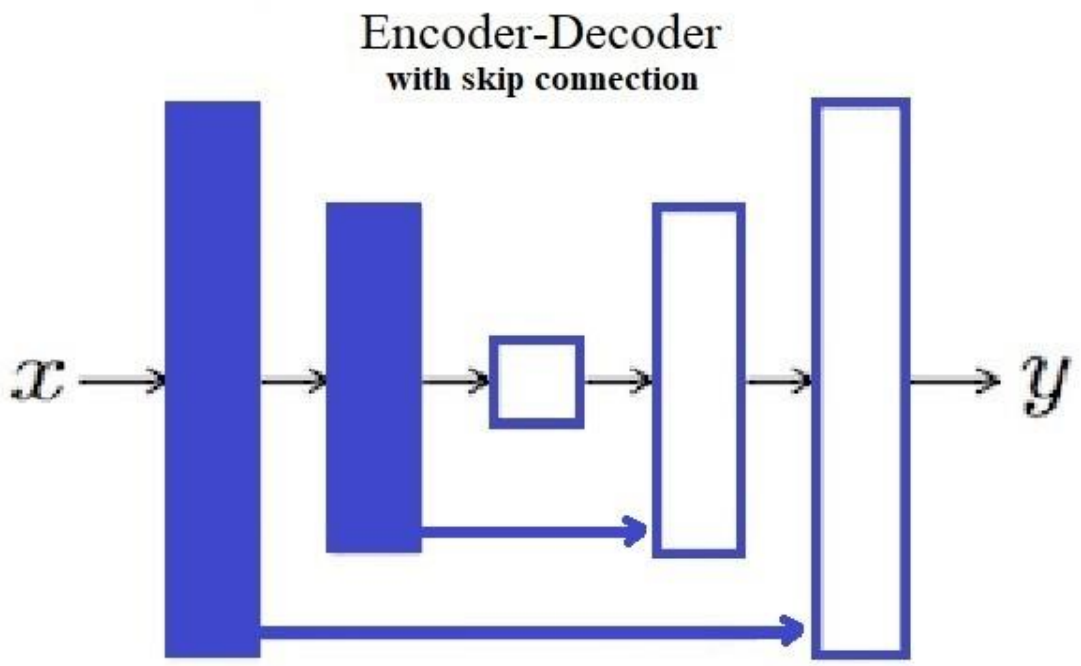

Figure 3.2: With skip connection

The U-Net model was developed for a semantic segmentation task. when images are fed into the neural network, we can choose to classify objects overall or by individual instances, we can predict changes in the outcome of an object in an image (image prediction), determine where all objects are (localization/semantic segmentation) or determine where individual objects are (object detection/instance segmentation). Figure 3. 3 clearly shows how it works.

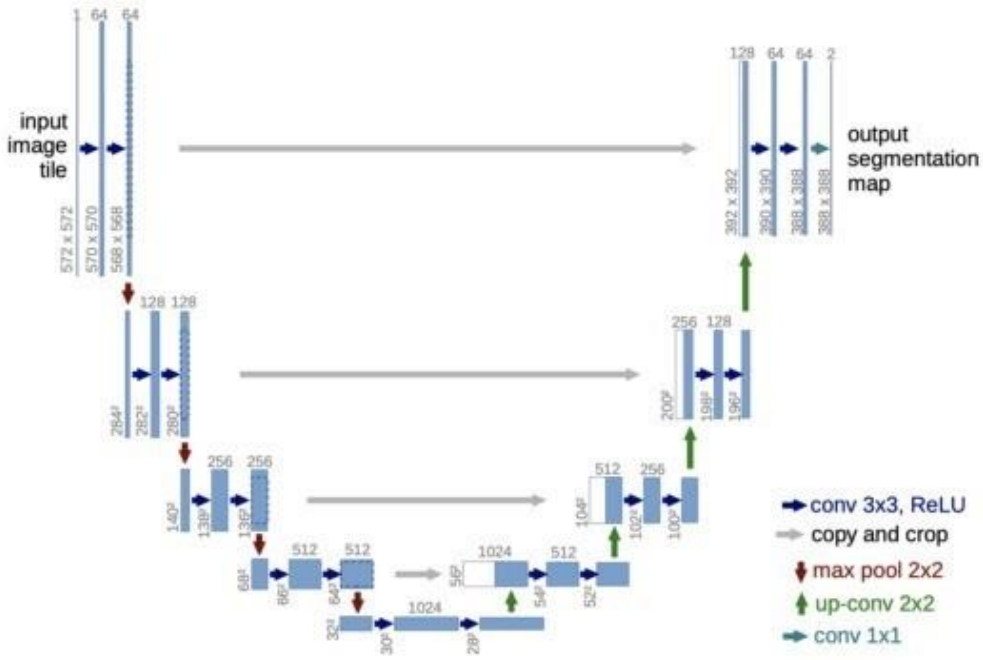

Figure 3.3: The U-Net architecture

3.4 Discriminator:

The discriminator model takes an image from the source domain and an image from the target domain and predicts the likelihood of whether the image from the target domain a real or generated version of the source image, each block of the discriminator contains a convolution layer, batch norm layer, and LeakyReLU. This discriminator receives two inputs:

- The input image and Target Image (which discriminator should classify as real)
- The input image and Generated Image (which they should classify as fake or not).

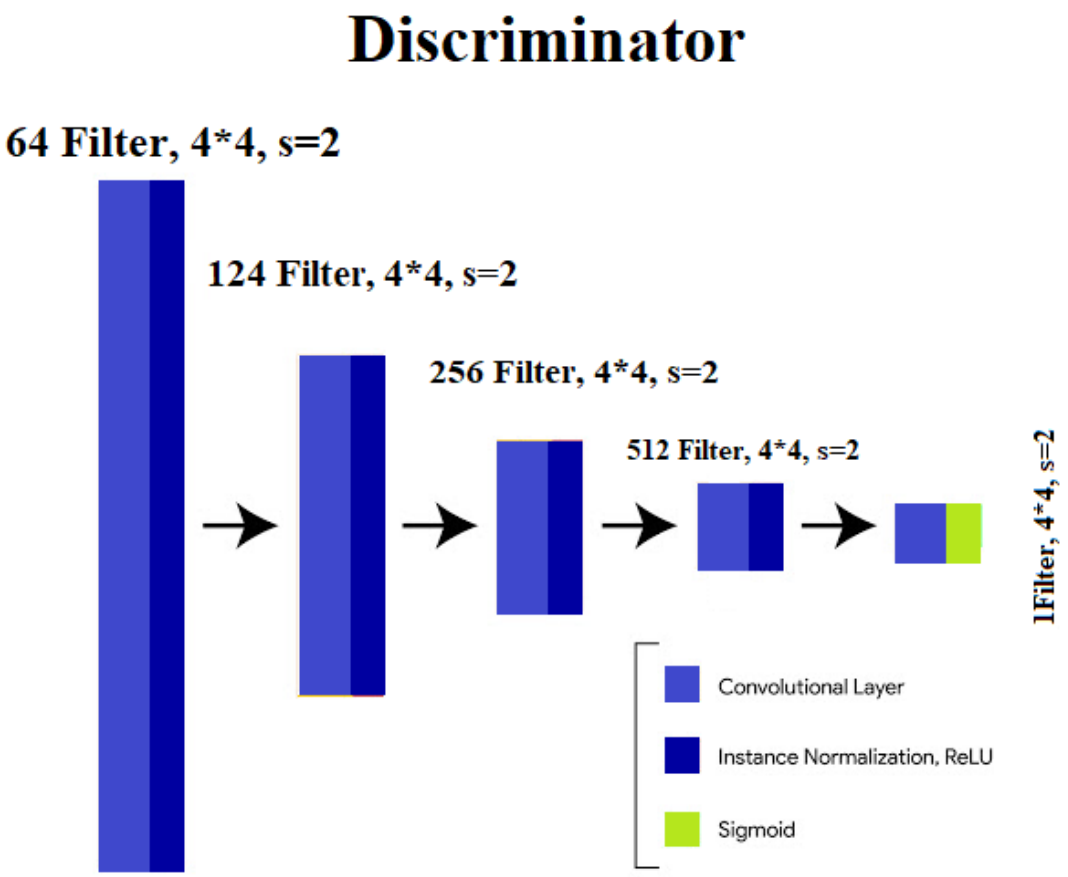

Figure 3.4: The patch GAN architecture

The discriminator of PixtoPix GAN uses Patch GAN architecture [15], which also uses Style GAN architecture. This Patch GAN architecture contains a number of Transpose convolutional blocks. This Patch GAN architecture takes an NxN part of the image and tries to find whether it is real and fake. This discriminator is applied convolutionally across the whole image, averaging it to generate the result of the discriminator D.

The PatchGAN is used because will be able to preserve high-frequency details in the image [15], with low-frequency details that can be focused by loss function of discriminator(we'll see clearly loss function in 3.4).

3.4.1 Benefits of PatchGAN:

The use of Patch GAN has several benefits in image-to-image translation tasks, where the goal is to transform an input image into a corresponding output image. Some of the benefits include:

- Improved Quality: By focusing on local patches rather than the overall structure, PatchGAN can produce images that have more detailed textures and styles.
- Efficient Computation: By assuming independence between pixels separated by more than a patch diameter, PatchGAN reduces the number of computations required. This makes it more efficient to use on large datasets.
- Robustness: PatchGAN can be more robust to image distortions and inconsistencies since it focuses on small, localized areas rather than the entire image.
- Flexibility: PatchGAN can be adapted for a variety of tasks, including object detection, image segmentation, and style transfer.

Briefly, the PatchGAN discriminator helps improve the quality of generated images by considering local patches rather than the entire image. It encourages the generator to produce realistic details at a fine scale

3.5 Loss functions:

The loss function plays a crucial role in training the two adversarial models the Generator and the Discriminator. Each of these models has a different loss function aimed at achieving opposing objectives. Let's discuss the loss function for each model in detail.

3.5.1 Generator loss (G_loss):

The loss function of the generator encourages to improve the quality of the generated images such that the discriminator is unable to differentiate them from the real images, it consists of two main parts:

3.4.1.1 Adversarial Loss:

Adversarial Loss function encourages the generator to produce images that appear realistic and match the target images [16],and the mathematical expression for Adversarial Loss [17] as follows :

$$L_{GAN}(G,D) = E_{x,y}[\log D\,(x,y)] + E_{x,z}\left[\log\left(1 - D\big(x, G(x,z)\big)\right)\right] \ldots\ldots (3)$$

where:

- G is the generator.
- D is the discriminant.
- x is the input image.

- y is the real target image.
- z is random noise

The goal of Adversarial Loss with this loss is to have the generator generate images realistic enough to fool the discriminator, while the discriminator seeks to differentiate between real and fake images.

3.4.1.2 Reconstruction Loss:

Reconstruction Loss (L1_Loss) Function is used to minimize the error which is the sum of the all the absolute differences between the true value and the predicted value [16], this function to reduce the error between true and predicted values in predictive models. and works by calculating the sum of the absolute differences between each true value and the value predicted by the model.

The formula for the L1 loss function [17] is as follows:

$$\sum_{i=1}^{n} | yi - y^{i} | = L_{l1} \ldots\ldots (3)$$

Where:

- y_i It is the real value of the point
- y^i It is the value predicted by the model for the point
- n is the number of points in the data set.

Briefly, the L1 loss function is used to ensure that the image generated by the model is like the original image in terms of content. The L1 loss function is calculated between the real image and the generated image to reduce the absolute differences between them, which contributes to improving the quality of the generated images.

3.5.2 Discriminator loss (D_loss):

While the discriminator is trained, it classifies both the real data and the fake data from the generator. It penalizes itself for misclassifying a real instance as fake [16], or a fake instance (created by the generator) as real, by maximizing the below function.

The discriminator loss function measures the discriminator's ability to differentiate between real images and generated images. The goal is to improve the performance of the discriminator so that it can distinguish real images from images generated by the generator, this loss has two main components:

- Real Image Loss [17]:

$$L_{real} = -E_{x,y}[\log D\,(x, y)] \ldots\ldots (5)$$

Here, *x* represents the input image and *y* is the corresponding real image. D(x, y) is the probability that the discriminator assigns to the pair being real. The goal is to maximize this probability, so the loss is minimized when D(x, y) is close to 1 for real images.

- Generated Image Loss:

$$L_{Discrimitor} = -\left(E_{x,y}[\log D\,(x, y)] + E_x\left[\log\left(1 - D\big(x, G(x)\big)\right)\right]\right) \ldots\ldots (6)$$

In this case, G(x) is the image generated by the generator from input *x*. D(x, G(x)) is the probability that the discriminator assigns to the pair being real. The goal here is to minimize this probability, so the loss is minimized when D(x, G(x)) is close to 0 for generated images.

- Combined Discriminator Loss [17]:

The overall discriminator loss combines these two components [18]:

$$L_{Discrimiator} = -\left(E_{x,y}[\log D\,(x, y)] + E_x\left[\log\left(1 - D\big(x, G(x)\big)\right)\right]\right) \ldots\ldots (7)$$

Where:

- $E_{x,y}$ [logD(x, y)] It expresses the discriminant loss when dealing with real images.
- E_x [log(1−D(x, G(x)))] It expresses the discriminator loss when dealing with generated images.

This combined loss function is aimed at continuously improving the discriminator's performance in distinguishing real images from generated ones, by optimizing this loss, the discriminator becomes more accurate in identifying real images, which in turn compels the generator to produce higher quality images to deceive the discriminator.

3.6 Mechanism of PixtoPix GAN:

PixtoPix leverages the conditional GAN framework to perform image transformation tasks very efficiently, by adapting the generator and discriminator to the input image, high-quality transformations that closely correlate with the features of the original image are achieved. Using U-Net architecture in the generator ensures that fine details are preserved, while using PatchGAN in the discriminator provides high accuracy in detail evaluation, this makes PixtoPix a powerful and efficient tool for a wide range of applications in the field of computer vision.

3.7 Conclusion:

After delving into the details of Pixtopix and understanding all its aspects and how it works, we can now access the practical application of this system in the next chapter.

CHAPTER 4:

Proposed System

4.1 Introduction:

The methodology section of a research project or study is an essential part that explains the processes and techniques used to collect, analyze, interpret, and train data.

4.2 Work environment:

4.2.1 Hardware environment:

We mainly train deep neural networks and work with large datasets of images in this project. Both tasks require a considerable amount of computing power, so maximizing the efficiency of computing resources is a top priority. To accomplish this, our computations are typically performed on GPUs (Graphics Processing Units), and we take advantage of hyper-threading on our CPUs to pre-process datasets and render them efficiently.

Since this training requires large and complex calculations, and because the process is adversarial and competitive between two neural networks (generator and discriminator), the training process requires special and specific equipment, here:

- **Graphics Processing Unit (GPU):** NVIDIA provides devices such as the Tesla T4, V100, A100, or the RTX 30xx and 40xx series, these devices high performance and speed in deep computing, is due to CUDA technology.
- **Central Processing Unit (CPU):** The Intel(R) Xeon(R) CPU @ 2.20GHz played a major role in accelerating data preloads and completing non-GPU operations.

- **Random Access Memory (RAM):** During training, we need a large amount of random memory (32GB or more), especially when dealing with huge data, and we have 54GB.
- **Storage:** It is necessary to use an SSD to speed up access to data, and we have more than 100GB of space.

We found all this hardware, specifications, and features in one place, which is Google Colab Notebook.

4.2.2 Software environment:

This training was implemented using Python programming language, at variance many other programming languages, Python is easy to read and less complex, a wide range of open source packages is available in Python, allowing it to be used for many different tasks, including mathematical applications, image processing, computer vision, and machine learning.

We relied on some open source offices that provide features and facilities for building and training models efficiently:

- **TensorFlow:** an open-source library for machine and deep learning too, maintained and developed by google brain, used for building and training machine and deep learning model because it's flexible and comprehensive ecosystem.
- **Matplotlib:** an open-source library in python the most popular and easy-to-use. It is widely used to create 2D graphs and charts. Here are some of its key features: High Flexibility, Good Integration, Customizability, Interactive Plots.
- **IPython:** (Interactive Python) is an enhanced interactive interpreter for Python, providing a rich interactive computing experience. It offers features like interactive shells, rich media integration, support for parallel computing, In IPython, you can too display various types of content, including images, videos, HTML, LaTeX, and more, using display.
- **OS:** a module in Python provides a way of using operating system-dependent functionality, allowing you to interact with the operating system in a platform-independent manner.
- And more library for facilitate operations like: Pathlib, Time, Datetime.

4.2.3 Source Code:

You can access the Python implementations of our models, evaluation metrics, and pretrained models at the following link.
”
https://drive.google.com/drive/folders/1lXTqcriN1mKpUCMFoxn7wu29sgFxHoCb?usp=sharing “

4.3 Data:

4.3.1 Data Source:

The FGNet dataset is a valuable resource in computer vision, which contains 1,002 images of 82 people in different age stages, ranging from 1 to 70 years old. We chose these for the diversity of facial expressions, lighting, coloring, and backgrounds, which makes training easier for us.

4.3.2 Data processing:

In data processing, it is necessary to process the data in a proper manner and prepare it well. All images go through four stages, which we mention:

- **Data cleaning:** We worked to filter out low-quality images, such as white images, black images, and images that we do not need.
- **Detect & crop faces:** After ensuring the image is not empty, we move on to the stage of face detection, determining its dimensions, and saving them. Then, we use these dimensions to crop the face from the image as illustrated in Figure 4.1:

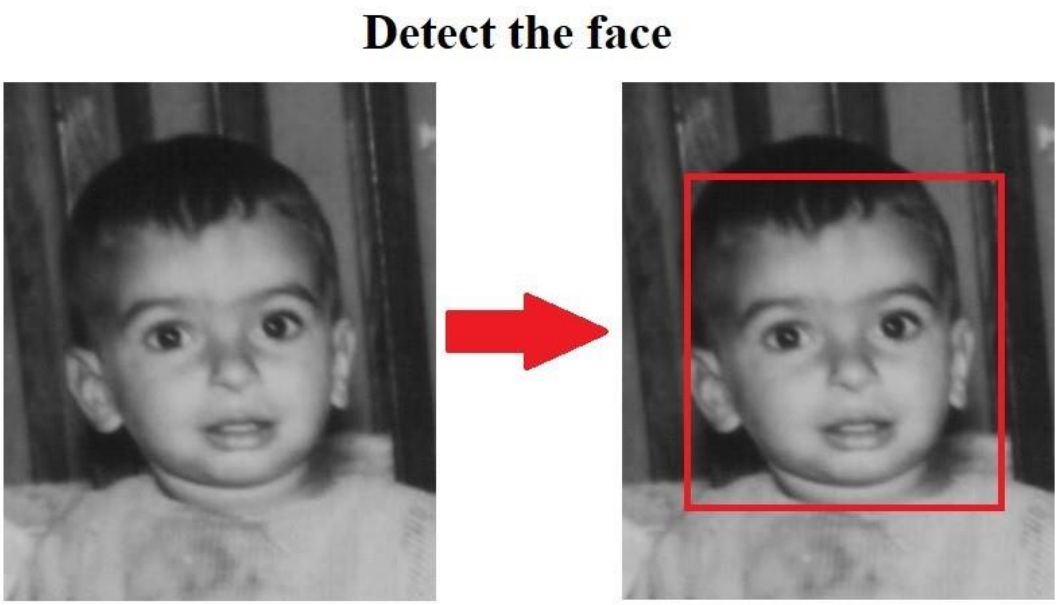

Figure 4.1: Detect the face

this is how we detect, let see the image after the crop

Figure 4.2: The image after the crop

- **Data sizing:** After determining the appropriate size, we re-size the image, to train our model and produce appropriate results. We will continue with the previous example:

Figure 4.3: Re-size image

- **Collect images:** In order to combine two pictures of one person of different ages. We must talk about naming files. The name contains two parts, the first is the person's number and the second is his age, such as 001A02. To combine them, we take the first image of each number and combine it with every other image that contains the same person's number, as shown in the Figure 4.4:

Figure 4.4: Collect image

4.4 GANs mechanism:

4.4.1 Generator:

All GANs models share the same mechanism in the generator, but in our model there is a special case called U-net. In addition to the encoder-decoder, it uses SkipConnect to form the stages of these steps in the shape of a U. We divide these stages and steps as follows:

Firstly, image in-put: We pass the original image into the generator, which is converted into digital pixel matrices using the mathplot library.

Secondly, Encoder or Down-sampling: This part is very important in the network and includes several stages of image processing. These stages consist of: Convolutional Layers, Batch Normalization, Max Pooling

- Convolutional Layers: These layers perform a mathematical operation called convolution, which involves sliding a filter or kernel over the input image to produce a feature map [18].
- Batch Normalization: Batch Normalization is used after the convolutional layers at each level of the path. It helps reduce the internal variance of the data as it passes through the layers, facilitating the learning process. It speeds up the training process by allowing higher learning rates and reducing the need for precise weight initialization.
- Max Pooling: In Encoder, we used the Maxpooling process to reduce the dimensions of images and preserve important features in the images by choosing the maximum value in each part to which the process is applied, as shown in the Figure 4.5.

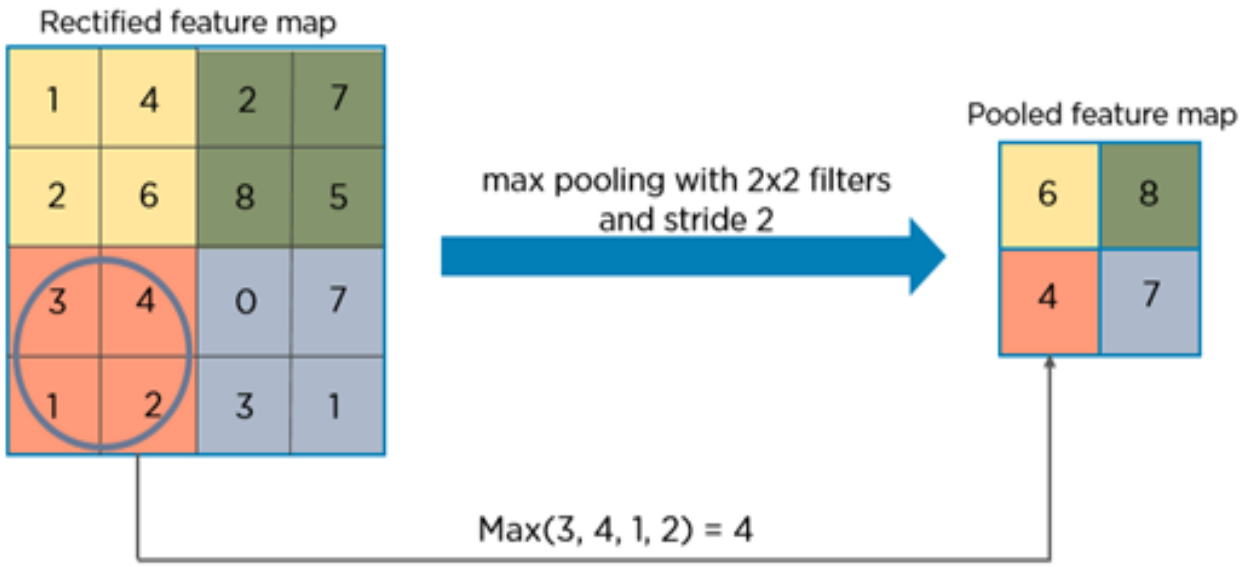

Figure 4.5: Max Pooling

Thirdly, the Bottleneck is the part that connects two networks, Encoder-Decoder, and works to integrate the features extracted from the previous stage, Encoder, and transfer them to the next stage, Decoder.

Fourthly. after Encoder and Bottleneck, by processing the images and extracting features from them, we pass to a stage Decoder in which this part works to increase the dimensions of the images and combine the features using skip connection methods, each Decoder level consists of three stages:

- Up-sampling: U-net uses the Up-sampling process, which is a process of increasing resolution and dimensions. It works to increase the apparent size of the image by means of new data between the existing data. This technique relies on mathematical calculations to estimate the values of new pixels based on the pixels surrounding them.
- skip-connection: After each stage of Up-sampling, the skip-connection extracted from the En-coder side layers is merged with the corresponding level layers on the De-coder side, which were enlarged using Up-conv from the lower level.
- Convolutional Layers of Decoder: Convolutional Layers of Decoder works to reconstruct the details that were lost during the up-sampling process using the Relu function, which is a common and effective activation function. This function keeps the inputs as outputs if they are positive and if the inputs are negative then the result is 0, as shown in Figure 4.6.

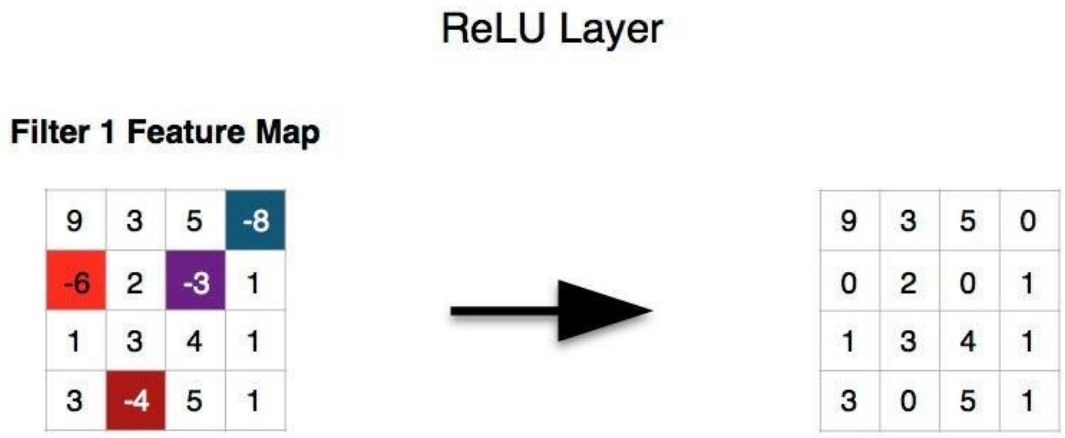

Figure 4.6: Convolutional Layers

In brief, images are generated from inputs through an integrated process that includes a downstream (Encoder) to extract important features and gradually reduce dimensions, a Bottleneck layer to combine and compress information, and an upstream (Decoder) to gradually increase the dimensions of features and reconstruct the image using up-sampling and merging of features from the downstream process. Through Skip Connections, which enables the model to produce expected images. The following figure 4.7 shows everything we worked on in the generator.

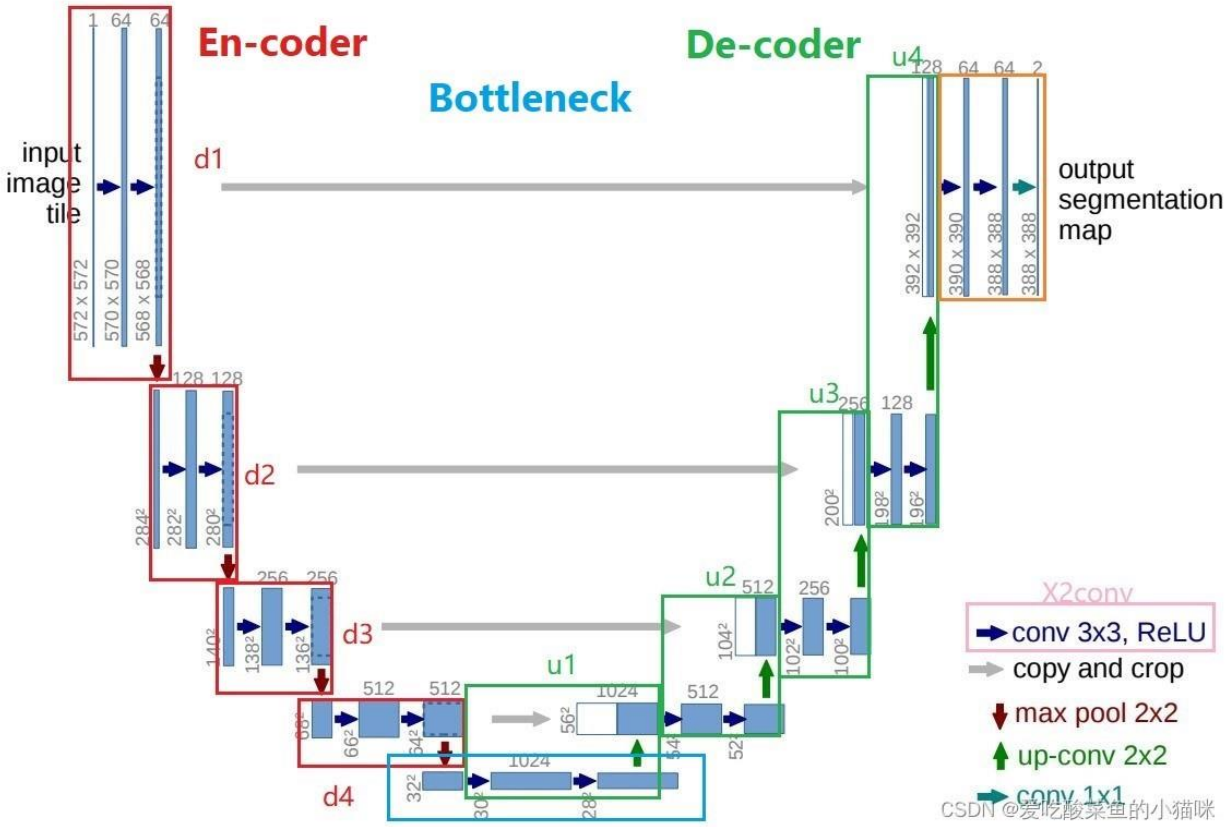

Figure 4.7: The U-Net architecture

4.4.2 Discriminator:

In Discriminator, we use PatchGAN, a model that works to distinguish small parts of images independently. This technology is not like the rest, which discriminates entirely on the image. PatchGAN consists of:

- Convolutional Layers of the discriminator: These layers in the discriminator distinguish between real and generated images, and analyze the input images to extract the important and distinct parts in them, using the Max-pooling layer.
- Batch Normalization: PatchGAN uses Batch Normalization to distribute features learned from convolutional layers where
- Activation Layers: Activation Layers are run in PatchGAN with leaky-Relu. Which helps the Discriminator focus on small and precise details, instead of looking at the images completely.

PatchGAN divides the image into small parts and evaluates each part individually using a convolutional network, which enhances the focus on fine details in the generated images and contributes to improving the overall quality of these images. Figure 3.4 shows how it works.

4.5 Conclusion:

We have gone through all the technical and practical stages that enable us to study the results obtained from the training.

CHAPTER 5:

Experimental Results

5.1 Introduction:

This section presents concrete evidence that answers the research questions, and explains the relationship between findings and research objectives.

5.2 Evaluation Metrics:

There are many evaluation metrics that aim to convert the model's performance into numerical values and ratios, to evaluate it and discuss its results, and in this formulation, we used metrics include precision, recall, and F1-score.

Before starting with these measures, we must define these values and what they represent. In our binary classification results (where there are only two categories: positive and negative), the confusion matrix is a 2x2 table with the following structure:

Table 5.1 Confusion matrix

Actual \ Predicted	**Predicted Positive**	**Predicted Negative**
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

Where:

- True Positive (TP): are measures the extent to which the model correctly predicts the positive class.
- False Positive (FP): False positives occur when the model predicts that an instance belongs to the positive class that it does not.

- False Negative (FN): False negatives can often become more serious than false positives, and so it is important to take them into account when evaluating the performance of a classification model.
- True Negative (TN): occurs when the model predicts that an instance belongs to the positive class that it does not.

After defining the confusion matrix and specifying the values, it will make it easier for us to understand these Precision, Recall, and F1-score, Accuracy metrics:

Precision is a metric that measures how often a model correctly predicts the positive class. You can calculate precision by dividing the number of correct positive predictions (true positives) by the total number of instances the model predicted as positive (both true and false positives) [19], which became the formula Precision:

$$PRECISION = \frac{number\ of\ true\ positives\ generated\ image}{number\ of\ (true\ and\ false)\ positives\ generated\ image}$$

Recall is a metric that measures how often a model correctly identifies positive instances (true positives) from all the actual positive samples in the dataset. You can calculate recall by dividing the number of true positives by the number of positive instances. The latter includes true positives (successfully identified cases) and false negative results (missed cases) [19], The difference between Precision and Recall is that the former is divided by the number of generated images and the latter is divided by all images. the formula:

$$RECALL = \frac{number\ of\ true\ positives\ generated\ images}{number\ of (true\ positives\ and\ false\ negative)\ generated\ image} \quad \ldots.(8)$$

F1-Score is combines both precision and recall and symmetrically represents them via a harmonic mean, F1 can range from 0 to 1, with 1 representing a model that perfectly classifies each observation into the correct class and 0 representing a model that is unable to classify any observation into the correct class.

$$F1 = 2 \cdot \frac{PRECISION \cdot RECALL}{PRECISION + RECALL} \quad \ldots.(9)$$

Accuracy is a metric that measures how often a model correctly predicts the outcome. You can calculate accuracy by dividing the number of correct predictions by the total number of predictions.

$$ACCURACY = \frac{number\ of\ true\ positive\ and\ true\ negative\ generated\ images}{number\ of\ all\ generated\ images} \ldots (10)$$

These metrics help in understanding different types of model performance. Choosing the most appropriate measure depends on the context. By using these measures together, a comprehensive evaluation of the model can be achieved, which helps us understand it and discuss its results.

5.3 Results and Analysis:

In this part, we see a sample of the images predicted by the model that went through many stages in training, and we will also calculate using precision, recall, and F1-score, Accuracy, the metrics for the results obtained, from which we build our analyzes.

To calculate the evaluation metrics, we must define the confusion matrix:

Table 5.2 Defined the confusion matrix

Actual \ Predicted	**Predicted Positive**	**Predicted Negative**
Actual Positive	**451** (TP)	**0** (FN)
Actual Negative	**77** (FP)	**72** (TN)

Where:

- True Positive (TP): The model generates a face and it looks like the first one.
- False Positive (FP): The model generates a face and does not resemble the first one.
- False Negative (FN): The model did not generate any face.
- True Negative (TN): The model generates a scattered face.

→ Precision:

$$PRECISION = \frac{451}{451 + 77} = \frac{451}{528}$$

$$PRECISION = 0.85$$

→ Recall:

$$Recall = \frac{451}{451 + 0}$$

$$Recall = 1$$

→ F1-score:

$$F1 = 2 \cdot \frac{1 \cdot 0.85}{1 + 0.85}$$

$$F1 = 0.92$$

→ Accuracy:

$$ACCURACY = \frac{451 + 72}{451 + 72 + 77 + 0} = \frac{523}{600}$$

$$ACCURACY = 0.87$$

The below table shows the scores for all the metrics:

Table 5.3 Table for all the metrics

Metric	Score
Precision	0.85
Recall	1
F1 Score	0.92
Accuracy	0.87

These metrics provide a comprehensive assessment of the expected images on which the model was trained, and here are some random visual examples of pictures of children whose pictures are predicted by this system.

The first example:

Figure 5. 1: The first example

The second example:

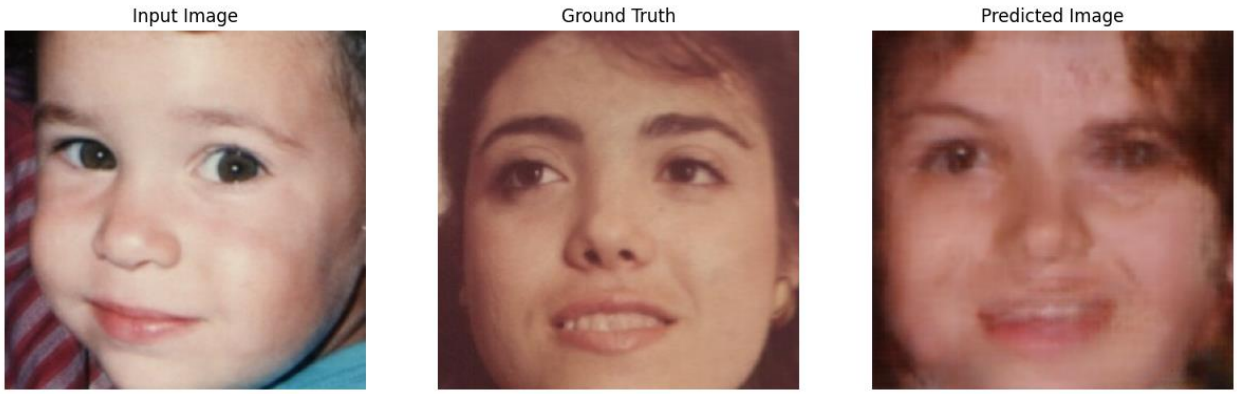

Figure 5.2: The second example

The third example:

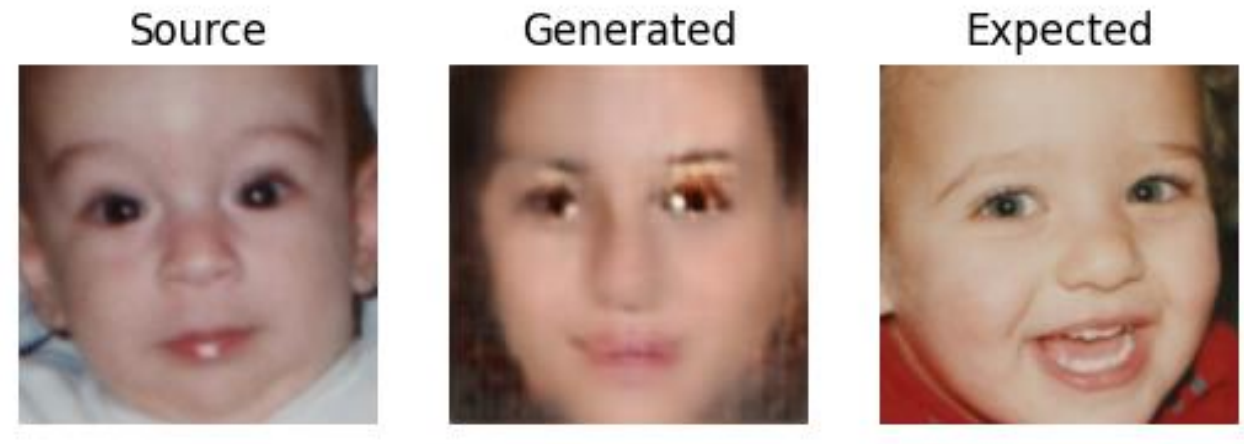

Figure 5.3: The third example

The Fourth example:

Figure 5.4: The Fourth example

Analysis of Results:

These criteria helped us in our evaluation of the system, as they showed us the ratio of Precision to the positive cases that the system predicted were correct, which indicates the effectiveness of the system in creating faces and predicting them correctly. The Recall ratio also proved that the model tried in all cases to create and predict faces. F1-score combines the Precision and Recall scales in one statement, as the model creates faces well while predicting the correct ones at the same time.

The results for the Accuracy of this system showed high efficiency in producing images and creating faces on them.

Overall, after all these results, this model still does not produce clear images, as well as a reduction in the number of face images that were created that the system did not anticipate well, Further refinement is required to enhance the model's performance.

5.4 Conclusion

The results showed their positive aspects in the visual aspect and in the percentage of matching faces, which summarizes the success of the system in prediction.

General Conclusion

Considering the rapid development of GANs, which have proven successful in all the fields they have been applied to, including medical, scientific, artistic, advertising, and video games, we have attempted in our research to keep pace with this development and make improvements to PixtoPix GANs models to suit face prediction for missing children system.

Our model worked to predict the faces of children based on the pictures taken of them. The model was proven according to the measurement standards and by using face recognition to measure the difference between the two faces and the expected face of the captured image. It worked very well.

Despite the good results, the system did not reach the required level. It is necessary to put more effort into research to improve it and use more advanced equipment.

Moreover, we believe that our model can produce expected images better than these, by focusing on improving training techniques and expanding the dataset, that is, using a larger and more diverse dataset, to improve the model's ability to deal with multiple and different types of image data.

Finally, this model showed its effectiveness in prediction, and the results of the measurements proved this. Likewise, we cannot believe that its expected results are inevitable, because children's faces are subject to many factors of change, including genetic, climatic, and many others. However, in general, the model showed respectable results.

Bibliography

[1] Hanson, Emmanuel D., The roles of artificial intelligence in library automation, Sheffield: University of Sheffield, April 2024.

[2] Ian Goodfellow, Yoshua Bengio, Aaron Courville, Deep learning, MIT press, 2016.

[3] Mitchell, Tom, Machine Learning, New York: McGraw-Hill, 1997.

[4] Boris Belousov, Hany Abdulsamad, Pascal Klink, Simone Parisi, Jan Peters, Reinforcement Learning Algorithms: Analysis and Applications., Springer, 2021.

[5] Manoj Kumar Gupta, Pravin Chandra, A comprehensive survey of data mining, International Journal of Information Technology, 2020.

[6] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, Generative Adversarial Networks, Cornel University, 2014.

[7] Charu, C Aggarwal, Neural Networks and Deep Learning, Springer, 2018.

[8] Tianfeng Chai, Roland R Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature," 30 june 2014.

[9] Yashar Deldjoo, Tommaso Di Noia, Felice Antonio Merra, "Adversarial Machine Learning in Recommender Systems: State of the art and Challenges," 20 May 2020.

[10] Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, Jieping Ye, "A Review on Generative Adversarial Networks: Algorithms, Theory, and Applications," 20 JAN 2020.

[11] Mehdi Mirza, Simon Osindero, "Conditional Generative Adversarial Nets," 06 Nov 2014.

[12] Jason, Brownlee, "A Gentle Introduction to Pix2Pix Generative Adversarial Network," *machine learning Mastry*, 2019.

[13] Tran, Minh, "Understanding U-Net," *Towards Data Science*, 2022.

[14] laf Ronneberger, Philipp Fischer , Thomas Brox, U-Net: Convolutional Networks for Biomedical Image Segmentation, CoRR, 2015.

[15] Phillip Isola, Jun-Yan Zhu , Tinghui Zhou , Alexei A. Efros, Image-to-Image Translation with Conditional Adversarial Networks, CoRR, 2018.

[16] Dwivedi, Harshit, "neptune," *Understanding GAN Loss Functions*, 2023.

[17] Lorenzo Ciampiconi, Adam Elwood , Marco Leonardi , Ashraf Mohamed , Alessandro Rozza, A survey and taxonomy of loss functions in machine learning, Switzerland: lastminute.com group, 2023.

[18] Keiron O'Shea, Ryan Nash, "An Introduction to Convolutional Neural Networks," *CoRR*, vol. abs/1511.08458, 2015.

[19] Brendan Juba, Hai S. Le, Precision-Recall versus Accuracy and the Role of Large Data Sets, Washington: Washington University in St. Louis, 2019.