MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE ABBES LAGHROUR DE KHENCHELA
FACULTE DES SCIENCES ET DE LA TECHNOLOGIE

Département Mathématique et Informatique

N° de série : ……

Mémoire de fin d'études

Pour l'obtention du diplôme de Master (L.M.D)

Spécialité : Sécurité et Technologie Web

L'apprentissage profond pour la génération et l'identification automatique de DeepFakes

Présenté et soutenu publiquement par :

DjebabraSeif El islam &AbboudiTakieddine

Le :

Membres de jury :

Président du jury : Abd Elhadi Adel

Examinateur : Fellah Hajer

Encadreur : *Abbas Fayçal.*

Année universitaire : 2020/2021

Remerciements

Lorsque nous réussissons un travail et achevons nos buts c'est en grande partie grâce à toutes les personnes qui nous ont assistés durant notre travail.

C'est pour cela, on remercie Mr.Abbas Fayçal de nous avoir donné la plus grande aide pendant toute la durée de notre travail. De nous avoir procuré plus que des directions mais aussi d'avoir été disponible, clairvoyante et pour avoir été un vrai modèle par ses qualités humaines.

Nous présentons nos vifs remerciements à l'ensemble des membres du jury pour avoir pris le temps de rédiger et juger notre modeste travail et de nous aider à nous améliorer pour de futurs travaux.

Pour avoir été d'une aide précieuse, que ce soit professionnel ou émotionnel, nous réservons un remerciement spécial à nos parents en particulier et nos familles en général pour avoir été nos guides et pour avoir ajouté de la légèreté pendant la durée de notre travail.

Mes remerciements ne seront pas complets sans mentionner mes amis qui me sont chers.

Mes remerciements aussi à toute personne qui m'a aidé de près ou de loin pour réussir ce modeste travail.

Merci de nous avoir inspirés et nous avoir encouragés à toujours faire de notre mieux.

Dédicaces

Avec un cœur plein de sentiments sincères et reconnaissants, je dédie ce modeste travail à :

La femme qui a dédier sa vie à moi en plus de donner naissance à mon existence, elle a tellement souffert pour garantir mon éducation et que je grandis pour être un homme honnête et responsable que dieu te protège de tout malheur ma chère mère (ZINEB).

L'homme qui a sacrifié son temps et sa santé pour mètre du pain sur la table, même si tu ne le montre pas mais on sait comment c'est difficile de se lever chaque matin avec la responsabilité de toute une famille sur tes épaules, papa (KHELLAF) je ne serais jamais comment te repayer pour tous ce que tu as fait pour moi.

Une personne spécial dans ma vie qui ne se lasser pas de m'encourager et de me pousser à faire de mon mieux chaque jour, merci pour ton soutien infini.

Mon cher binôme (TAKI) pour être une personne à qui je peux compter sur et à qui je peux toujours faire confiance.

Mon frère que je ne partage pas de sang avec mais que je partage mon âme avec dédicace à toi (ZAKI) et toute la famille Berdouk.

Mon bras droit (RAMZI) dont je peux toujours compter sur, j'espère vraiment que tu réaliseras ton rêve de décrocher le doctorat.

Merci à tous mes chers amis que je n'ai pu mentionner ici.

Merci tous.

ISLEM

Dédicaces

Je dédie ce travail :

A l'ensemble de ma famille et plus particulièrement à mes parents pour leur amour, leur confiance, leurs conseils ainsi que leur soutien inconditionnel qui m'a permis de réaliser les études pour lesquelles je me destine et par conséquent ce mémoire,

Merci Maman (SAMRA) Merci papa (MOHAMED) pour être des parents modèles.

A mes chers frères (AKRAM) et (CHAKER) merci pour être la lumière de ma vie.

A mon ami et binôme (ISLEM) merci pour ton aide et support dans cette long aventure qui a été une merveilleuse expérience dans ma vie.

Merci à tous mes amis pour leur support et soutien moral.

Merci à tous.

TAKI

Résumé :

Dans ce mémoire notre travail se focalise sur la technique de synthèse de vidéo Deepfake crée à l'aide de Deep Learning, en effet cette dernière ouvre des opportunités dans certains domaines de la vidéographie. Son principe consiste à créer de faux contenus générés par des techniques de l'apprentissage profond. L'utilisation de cette technique aujourd'hui peut susciter des applications malveillantes, en termes de sécurité et d'éthique. Aujourd'hui il est nécessaire de disposer d'un système de détection fiable de Deepfake .

Dans ce mémoire nous proposons deux méthodes : la première pour la création d'une vidéo à faux contenu et la deuxième pour la détecter des vidéos Deepfake. En effet notre solution consiste à utiliser un réseau de neurone antagoniste (Genrative advarsarial network) afin de relever le défi imposé par les médias à faux contenus. Notre réseau de neurones est composé de deux réseaux un encoder et un décodeur, notre solution produit de bons résultats en termes de précision et performance.

Mots Clés : Deepfake, modèle détection, modèle création, un réseau de neurone antagoniste un réseau de neurone antagoniste).

Abstract:

In this thesis we will focus on the technique of video synthesis (Deepfake) created with the help of Deep learning, this method can open a lot of opportunities in some domain of videography. The principle consists in creating fake contents generated by deep learning. The use of this technique nowadays stimulate the creation of malicious applications security and ethics wise. Today it is necessary to have a reliable Deepfake detection system.

In this thesis we are proposing two methods: the first one being the creation of a fake video and the second is the detection of a Deepfake video. Our solution consists of using a generative adversarial network (GAN) in the purpose of winning the challenge against fake media content. Our neural network is composed of two networks an encoder and a decoder, our solution generate good results precision and performance wise.

Keywords: Deepfake, detection model, creation model, generative adversarial network (GAN).

ملخص:

في هاته المقالة عملنا سيرتكز على اسلوب تركيب الفيديوهات (Deepfake) المنتجة بواسطة Deep Learning, هاته الاخيرة تفتح ابواب كثيرة في مجال الVIDEOGRAPHY .مبدأ عملها يكون بانشاء محتوى مغشوش مولد بواسطة تقنيات التعلم العميق. استعمال هاته التقنية في يومنا هذا قد يؤدي الى انتاج تطبيقات ضارة من ناحية الحماية و الجانب الاخلاقي. اليوم من الضروري امتلاك جهاز موثوق به لكشف DeepFake.

نقترح في هاته المقالة طريقتين للعمل : الاولى لانشاء فيديو بمحتوى مغشوش و الثانية لكشفها. حلنا يتمحور حول استعمال شبكة الخلايا العصبية المناهضة (Genrative advarsarial network) من أجل مواجهة التحدي الذي تفرضه الفيديهات ذات المحتوى الكاذب. الشبكة الخاصة بنا مقسمة الى شبكتين ENCODER و DECODER, حلنا يتميز باعطاء نتائج جيدة من ناحية الدقة و الاداء.

الكلمات المفتاحية: Deepfake، نموذج الصناعة، نموذج الكشف، شبكة الخلايا العصبية المناهضة (Genrative advarsarial network).

Table des matières

Liste des Figures

Liste des formules :

Introduction générale

Introduction générale :

Le cerveau humain étant le seul être doté d'intelligence sur terre a été le sujet de nombreuses études en cours des années visant à résoudre son mystère, en effet sa capacité de collecter des données à partir de son environnement grâce à ses sens et à les analysés et à en déduire des théories et des faits a toujours été son ultime atout pour survivre et pour évolué et construire tous les nombreuses civilisations connues et non connues au cours des milléniaux passé.

En 1950 le mathématicien renommé Alan Touring a publié un article qui a néanmoins surpris la majorité des informaticiens à l'époque intitulé « Computing Machinery and Intelligence » insinuant la possibilité que la machine peut adapter le comportement intelligent de l'humain et c'est de là où est née la première idée qui a donnée vie à l'intelligence artificiel (IA).

Au cours des années l'IA a évolué lentement mais surement mais ça n'a pas été le cas dans les dizaines d'année passée, en effet une branche de cette dernière a vu le jour et a ouvert beaucoup de portes à des contributions mondial dans le domaine de l'informatique, on parle bien sûr du machine learning et de la technologie évoluée de celle-ci je nomme le deep learning.

On a parlé au début de cette introduction comment le cerveau humain se sert de ces capacités pour collecter les informations et ensuite agir selon eux, et la question qui se pose Est-ce qu'il y a moyen pour le trempé ? Une méthode qui altère la réalité à un certain niveau ?

Une telle technologie existe dite deepfake, dans notre travail nous proposerons un algorithme basé sur l'utilisation du deep learning, en effet ce dernier est un réseau de neurones convolutif entrainé sur une base de données d'images extraites à partir des vidéos contenant les personnes concernées par ce thème et enfin nous terminerons ce mémoire sur l'exposition de quelques résultats de détections d'une vidéo deepfake en utilisant un réseau de neurones convolutif.

Notre mémoire est organisé comme suite :

1- Dans le premier chapitre, nous introduiront la science de l'intelligence artificielle ainsi que ces différents types et usages, ensuite nous aborderont un des champs d'études de ce domaine, on parle d'apprentissage automatique, cella nous servira après de transition vers le deuxième chapitre.

Le deuxième chapitre parlera de l'apprentissage profond, on détaillera ce sujet on profondeur puis on expliquera la différence entre l'apprentissage automatique et profond et finalement nous discuteront l'architecture général de ce domaine.

2- Dans le troisième chapitre nous aborderons la notion des deepfakes, un petit historique à ce sujet puis une définition détaillée pour mieux comprendre cette notion, après nous parlerons de quelques méthodes utilisées pour la création de ces derniers.
3- Le chapitre final exposera notre approche pour la création et la détection des deepfake, nous présenterons les résultats que nous avons atteint et nous donnerons une comparaison avec les travaux anciens.
4- Finalement nous concluront notre travail et nous exposons quelques pistes futures pour améliorer ce dernier.

Chapitre I :
L’intelligence Artificielle

1. Introduction

Au lieu d'exécuter les ordres d'un programme, la machine peut désormais acquérir par elle-même, par l'expérience, les capacités nécessaires pour accomplir les tâches qui lui sont assignées, y compris celles que l'on croyait réservées à l'humain. Les applications adoptant ce type de raisonnement sont importantes :

Reconnaissance des formes, des voix, des images et des visages, voiture autonome, traduction de centaines de langues, détection des tumeurs dans les images médicales [1].

L'IA est un domaine hybride entre le data science et les mathématiques qui se base sur l'analyse d'une base de données et/ou une base de règles et en déduire de nouveaux selon un comportement donné.

Cette technologie qui ne cesse de grandir est sujet de plusieurs applications dans la vie de tous les jours on nomme le coté médical, l'astronomie, les études économiques…

Dans un autre regard, l'IA est devenue un terme surutilisé pour les applications qui effectuent des tâches complexes qui auparavant était réalisé manuellement par les humains, comme communiquer avec les clients en ligne ou jouer aux échecs, malgré ses contributions substantielles à la recherche scientifique, l'Intelligence Artificielle est concentrée sur les approches mathématiques, visant à résoudre des problèmes formels avec un ensemble d'objectifs bien définis [2].

2. L'Intelligence Artificiel

1.1 Définition

Malheureusement Il n'existe pas de définition universelle pour l'IA chacun a sa propre définition du terme en cite quelques un :

- « La faculté de connaître et comprendre, incluant la perception, l'apprentissage, l'intuition, le jugement et la conception. » (Petit Robert)
- « La faculté de connaître et de raisonner. » (Dictionnaire American Heritage)
- « Application de la connaissance à la résolution de problèmes. » (Newell et Simon).

Une définition plus au moins répondue est que l'IA réunit des sciences, théories et techniques (notamment logique mathématique, statistiques, probabilités, neurobiologie computationnelle et informatique) et dont le but est de parvenir à faire imiter par une machine les capacités cognitives d'un être humain [2].

1.2 Court historique sur l'IA

Le développement des techniques informatiques (augmentation de la puissance de calcul) aboutit à plusieurs avancées :

- Dans les années 1980, l'apprentissage automatique se développe, notamment avec la renaissance du connexionnisme. L'ordinateur commence à déduire des « règles à suivre » en analysant seulement des données.
- Parallèlement, des algorithmes « apprenants » sont créés qui préfigurent les futurs réseaux de neurones (l'apprentissage par renforcement, les machines à vecteurs de support, etc.). Ceci permet par exemple en mai 1997 à l'ordinateur Deep Blue de battre Garry Kasparov au jeu d'échecs16 lors d'un match revanche de six parties ;
- L'intelligence artificielle devient un domaine de recherche international, marquée par une conférence au Dartmouth College à l'été 195617,18 à laquelle assistaient ceux qui vont marquer la discipline ;
- Depuis les années 1960, la recherche se fait principalement aux États-Unis, notamment à l'université Stanford sous l'impulsion de John McCarthy19, au MIT sous celle de Marvin Minsky20, à l'université Carnegie-Mellon sous celle de Allen Newell et Herbert Simon21 et à l'université d'Édimbourg sous celle de Donald Michie22, en Europe et en Chine, ainsi qu'au Japon avec le projet « ordinateurs de cinquième génération (en) » du gouvernement. En France, l'un des pionniers est Jacques Pitrat23 ;
- Dans les années 2000, le Web 2.0, le big data et de nouvelles puissances et infrastructures de calcul permettent à certains ordinateurs d'explorer des masses de données sans précédent ; c'est l'apprentissage profond (« deep learning »), dont l'un des pionniers est le français Yann Le Cun [3].

1.3 Les Types de l'IA

Il y'a plusieurs façons de classer l'IA, on cite 2 méthodes de classement :

- Le premier est que l'intelligence artificielle peut être considérée comme faible ou forte.
  - En effet l'IA faible est un système d'intelligence artificielle conçu pour reproduire une tâche précise à laquelle il est formé, les assistants virtuels comme Siri et Cortana en sont l'exemple.
  - Par contre l'IA fort (IA général) est un système doté de capacités cognitives humaines générales qui, lorsqu'on lui présente une tâche inhabituelle, est assez intelligent pour trouver une solution.

– Le deuxième est proposé par Arend Hintze, professeur en biologie intégrative et ingénierie informatique à la Michigan State University. Il classe l'IA en quatre types allant de celui des systèmes actuels aux systèmes sensibles à venir. Ses catégories sont :

- **Les machines réactives :** Ce sont des systèmes qui font des prédictions et ne s'appuie pas sur des expériences précédentes, ils ont été conçus à des fins précises et ils ne sont pas facilement transposables à une autre situation donc ils ne sont pas vraiment flexibles, on donne l'exemple de Deep Blue, le programme d'IBM qui a battu Garry Kasparov aux échecs dans les années 1990.
- **Machines à mémoire restreinte** : Ces systèmes d'IA s'appuient sur leurs expériences passées pour prendre les décisions présentes, ils sont des systèmes qui apprend tout comme l'humain des expériences passé pour mieux améliorer les décisions futuristique, le deepfake est un exemple commun de ce type d'IA.
- **Théorie de l'esprit :** Il s'agit d'un concept de psychologie qui se rapporte à la compréhension des gens en tant qu'êtres ayant des pensées, désirs et raisons propres qui les poussent à prendre leurs décisions. Ce type d'IA n'existe pas encore.
- **Conscience de soi :** Dans cette catégorie, les systèmes d'IA ont une identité, une conscience. Ces machines douées de conscience connaissent leur état actuel et utilisent ces informations pour inférer ce que les autres ressentent. Ce type d'IA n'existe pas encore [4].

1.4 Les Technologies utilisé dans l'IA

L'IA est un domaine vaste donc il n'est pas surprenant que diverses technologies existent, On nomme les 3 les plus répondue de nos jours :

- **L'apprentissage automatique** : Il met en œuvre les techniques qui permettent à un ordinateur d'agir sans programmation préalable, en effet il extrait les connaissances dont il a besoin à partir des données qu'on lui fournit.
- **Le traitement automatique des langues (TAL/NLP) :** Comme son nom l'indique c'est une technologie réservée à l'aspect linguistique, un des meilleurs et des plus anciens exemples de TAL est la détection des courriers indésirables qui analyse l'objet et le corps d'un e-mail pour le classer ou non en indésirable.
- **La vision artificielle :** Elle permet aux ordinateurs d'analyser, de traiter et de comprendre une ou plusieurs images. Elle sert dans de nombreuses applications, de l'identification des signatures à l'analyse d'imagerie médicale ou à la création de trucage visuel [5].

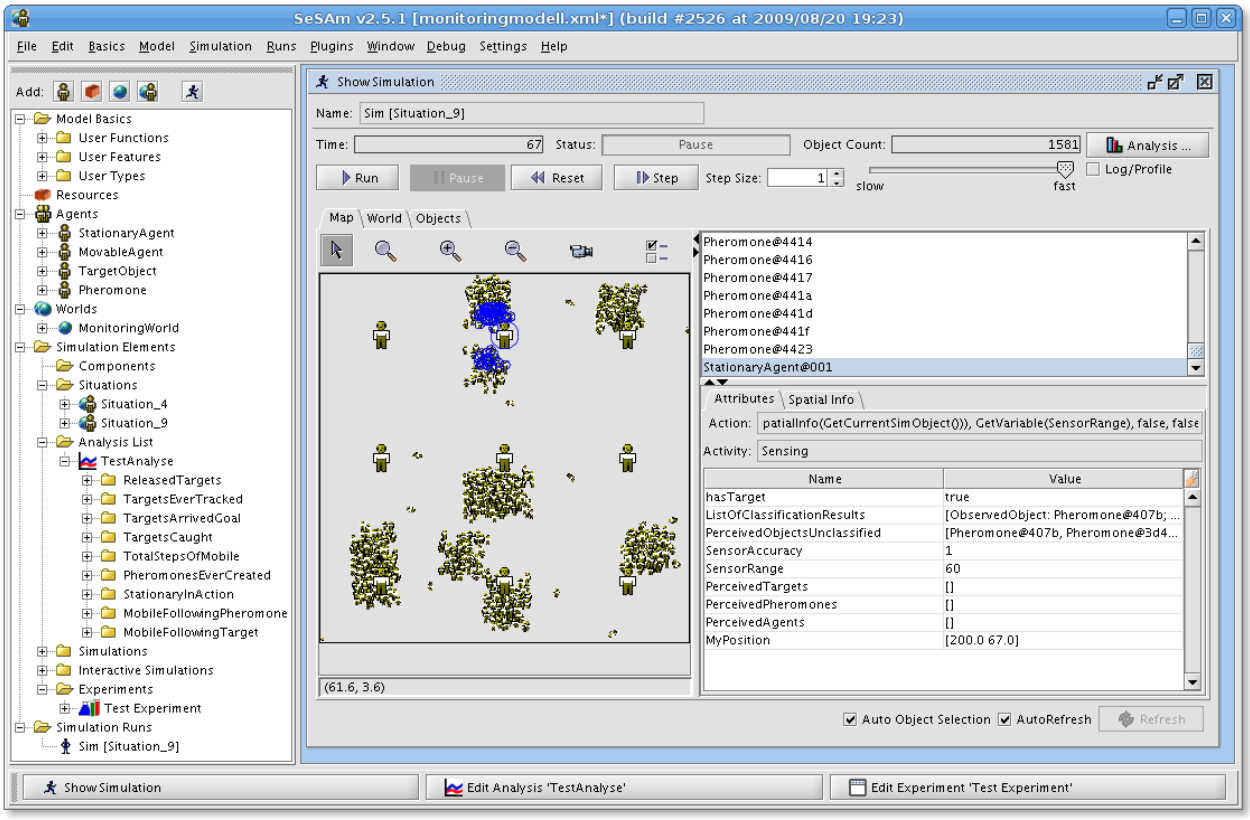

Figure 1.1 *: Un système multi-agent crée avec le programme SeSAm [6]*

1.5 Les techniques utilisées dans l'IA

Plusieurs techniques ont été développées au cours des années, chacune ayant une approche assez spéciale de l'IA, on va se contenter d'expliquer les 3 les plus connue :

1.5.1 Les systèmes Multi-Agent

Un système multi-agents est comme le nom l'indique un système composé de multiples agents/entités (processus, bots, agents), les systèmes multi-agents peuvent résoudre des problèmes difficiles ou impossibles à résoudre pour un agent individuel ou un système monolithique [6].

1.5.2 Les systèmes experts

D'après Feigenbaum c'est un logiciel intelligent qui utilise des connaissances et des inférences logiques pour résoudre des problèmes qui sont suffisamment difficiles qui nécessite une expertise humaine importante pour trouver une solution [7].

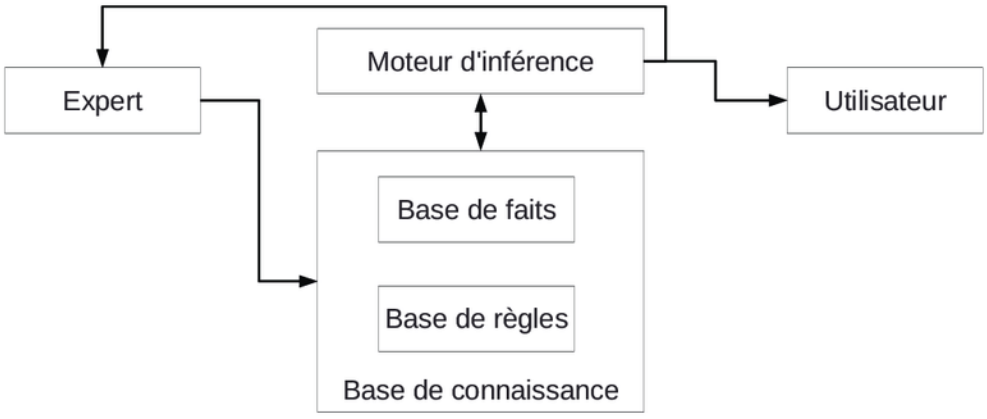

Figure 1.2 : *Architecture d'un système expert [7]*

Les systèmes experts et l'apprentissage automatique ne sont pas à confondre, en effet ils ont quelques similarités, en revanche ils sont deux entités différentes, voici un schéma qui illustre cette différence :

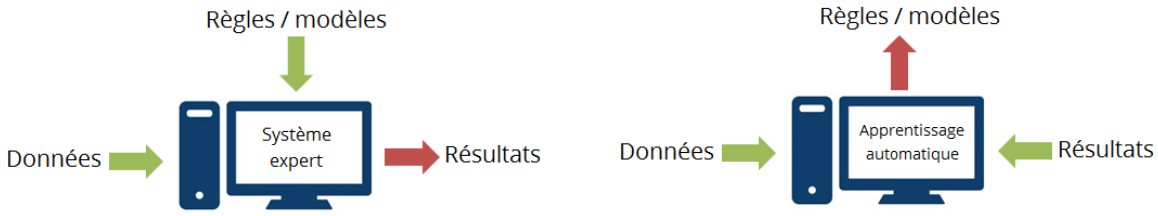

Figure 1.3 : *différence entre S.E et A.A [7]*

1.5.3 Les réseaux de neurones :

Ensemble de neurones (nœud) interconnectés permettant la résolution de problèmes complexes tels que la reconnaissance des formes ou le traitement du langage naturel, grâce à l'ajustement des coefficients de pondération dans une phase d'apprentissage [8].

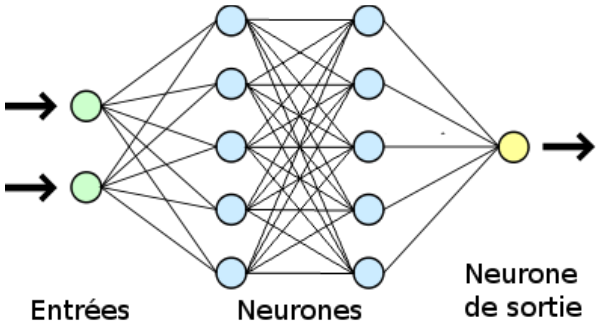

Figure 1.4 : *Structure d'un réseau de neurones [8]*

1.6 Les domaines d'application de l'IA

1.6.1 L'IA dans la santé

Des sociétés utilisent l'apprentissage automatique pour accélérer et affiner les diagnostics.

IBM Watson est l'une des technologies les plus connues dans le domaine de la santé.

Capable de comprendre le langage naturel, le système répond aux questions qu'on lui pose. Il analyse les données des patients, ainsi que d'autres sources de données, pour formuler une hypothèse qu'il présente avec un score de fiabilité [9].

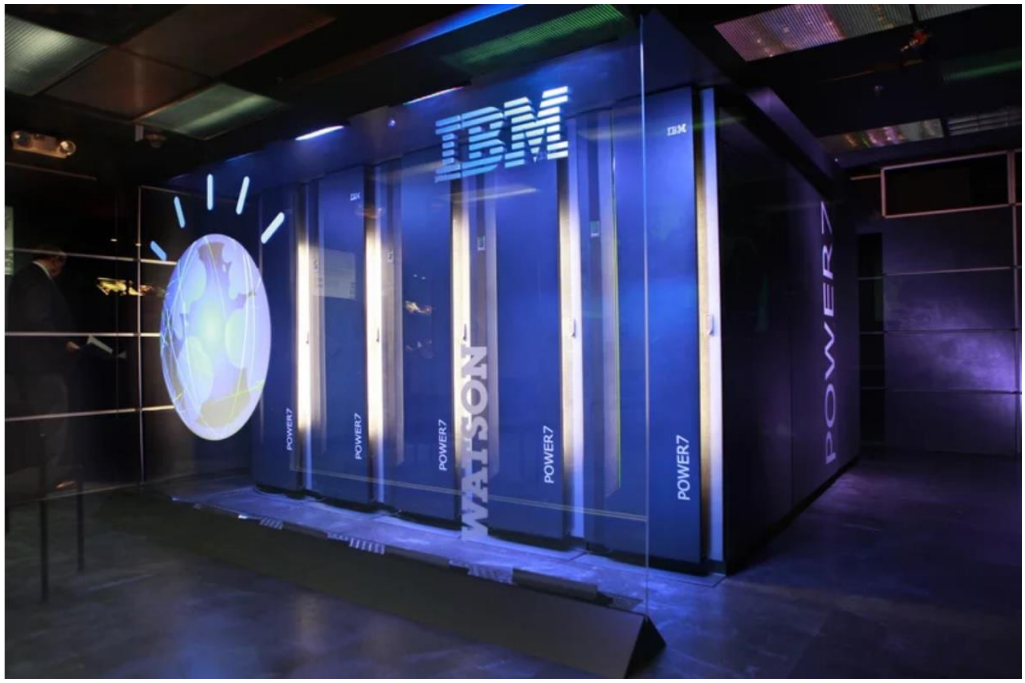

Figure 1.5 : *Le superordinateur IBM Watson [9]*

1.6.2 L'IA dans l'enseignement

L'IA peut automatiser la notation et faire gagner du temps aux enseignants et prédire le potentiel de ces derniers. Elle peut évaluer les élèves et étudiants et s'adapter à leurs besoins pour qu'ils travaillent à leur propre rythme. Des enseigneurs adjoints artificiels peuvent apporter une aide supplémentaire aux étudiants pour qu'ils gardent leur niveau éducationnel.

1.6.3 L'IA dans la sécurité et la surveillance

Les caméras de surveillance dotées d'une intelligence artificielle sont en tête des technologies les plus prisées par le secteur bancaire. Ces solutions vidéo offrent des fonctions de télésurveillance et des fonctionnalités avancées d'intelligence artificielle. Leur système d'analyse envoie des alarmes en fonction de scénarios prédéterminés ou d'images de situations à haut risque, telles que des

criminels identifiés entrant dans le bâtiment ou le sabotage présumé d'un distributeur automatique de billets [10].

Figure 1.6 *: caméra de surveillance utilisant l'IA* [10]

2. L'apprentissage automatique :

Vous vous demandez peut-être pourquoi on reparle de l'apprentissage automatique lorsque on l'a déjà mentionné et bien simplement car c'est une transition vers le deuxième chapitre l'apprentissage profond.

Pour enlever toute ambigüité on définit l'apprentissage profond comme un sous-ensemble de l'apprentissage automatique qui, en termes plus simples, peut s'envisager comme l'automatisation de l'analyse prédictive.

2.1 Définition :

L'apprentissage automatique (machine-learning en anglais) est une discipline scientifique, qui est aussi l'un des champs d'étude de l'intelligence artificielle.

L'apprentissage automatique fait référence au développement, à l'analyse et à l'implémentation de méthodes qui permettent à une machine d'évoluer grâce à un processus d'apprentissage, et ainsi de remplir des tâches qu'il est difficile ou impossible de remplir par des moyens algorithmiques plus classiques [11].

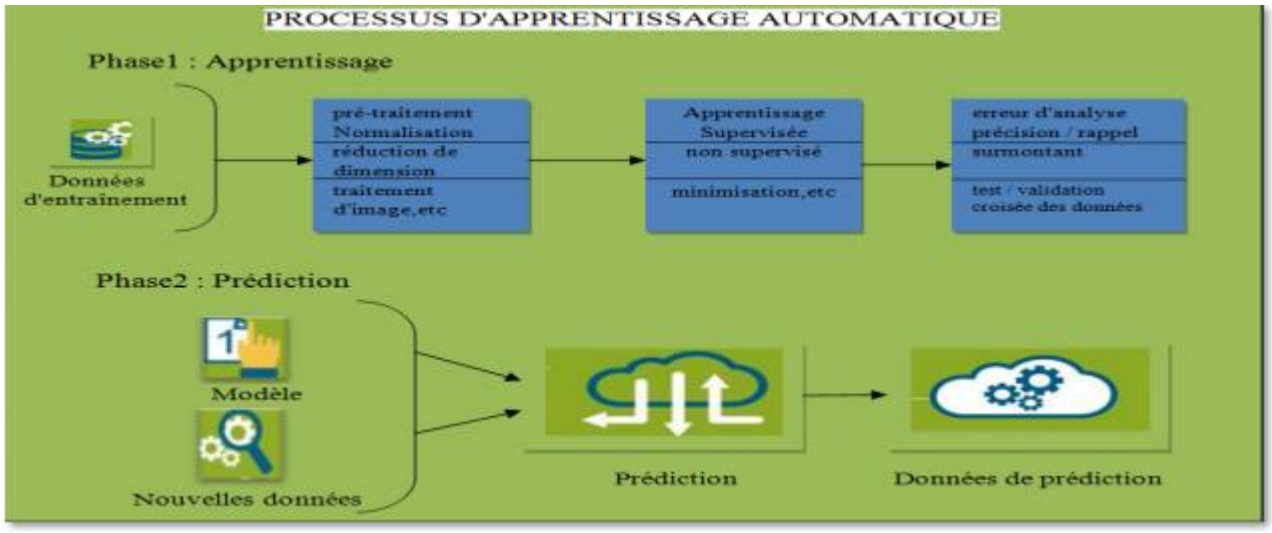

Figure 1.7 : *Processus d'apprentissage automatique [11]*

2.2 Les différents types d'apprentissage automatique

Il existe en tout cinq types d'apprentissage automatique, il se catégorise selon le mode d'apprentissage qu'ils emploient :

2.2.1 L'apprentissage supervisé

L'apprentissage supervisé (supervised learning) s'intéresse aux données étiquetées. L'objectif est de prédire l'étiquette (inconnue) y associée à une nouvelle observation x, à partir de la connaissance fournie par les N observations étiquetées du jeu de données (X_n, Y_n) $1 \leq n \leq N$ [12]. On prend l'exemple suivant pour plus clarifié ce qu'on a dit précédemment

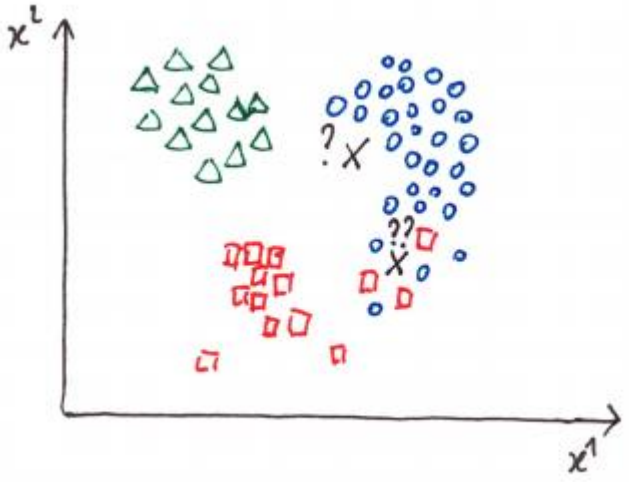

Figure 1.8 : *Classification supervisé [12]*

Ici, l'espace des observations est le plan bidimensionnel. Les observations ont des coordonnées (X1 et X2) et sont étiquetées par trois catégories (triangle, rond, carré). L'objectif est de déterminer quelle catégorie associer à une nouvelle observation non-étiquetée (représentée par une croix), à partir des observations étiquetées formant la base d'apprentissage représentée sur la figure. Si pour l'observation marquée «? » la tâche peut sembler facile (la classe est vraisemblablement « rond »), on voit que la classification de l'observation «??» est plus discutable (« rond » ou « carré » ?) [12]

2.2.2 L'apprentissage non-supervisé

Quand le système ou l'opérateur ne disposent que d'exemples, mais non d'étiquettes, et que le nombre de classe et leur nature n'ont pas été prédéterminés, on parle d'apprentissage non supervisé ou clustering. Aucun expert n'est disponible ni requis. L'algorithme doit découvrir par lui-même la structure plus ou moins *cachée* des données. Le clustering est un algorithme d'apprentissage non supervisé [12].

Le système doit ici -dans l'espace de description (la somme des données) - cibler les données selon leurs attributs disponibles, pour les classer en groupe *homogènes* d'exemples. La similarité est généralement calculée selon la fonction de distance entre paires d'exemples. C'est ensuite à l'opérateur d'associer ou déduire du sens pour chaque groupe et pour les patterns d'apparition des groupes ou groupes de groupes dans leur « espace » [12].

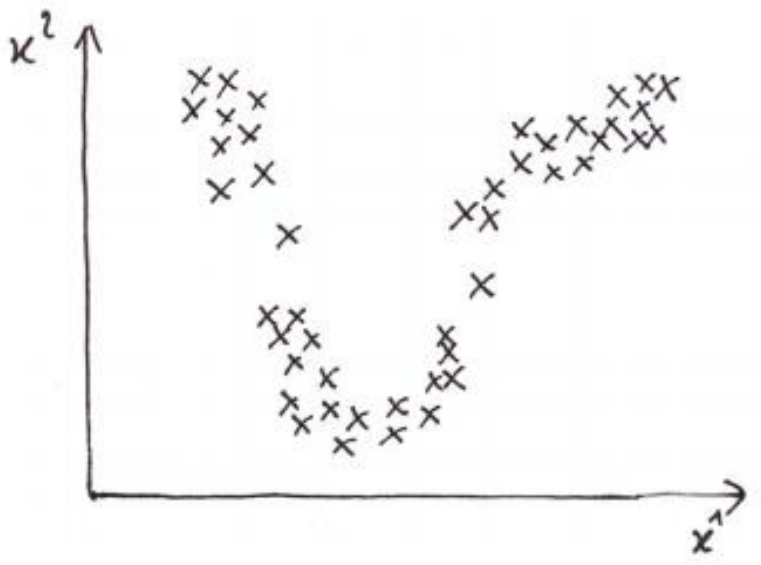

Figure 1.9 : *Classification non-supervisé [12]*

On distingue 3 clusters qui se forme avec 2 X plus au moins au milieu entre 2 clusters.

2.2.3 L'apprentissage semi-supervisé

Il effectué de manière probabiliste ou non, il vise à faire apparaitre la distribution sous-jacente des « exemples » dans leur espace de description. Il est mis en œuvre quand des données (ou « étiquettes ») manquent... Le modèle doit utiliser des exemples non-étiquetés pouvant néanmoins renseigner.
Ex : En médecine, il peut constituer une aide au diagnostic ou au choix des moyens les moins couteux de tests de diagnostics [12].

2.2.4 L'apprentissage partiellement supervisé (probabiliste)

Quand l'étiquetage des données est partiel. C'est le cas quand un modèle énonce qu'une donnée n'appartient pas à une classe A, mais peut-être à une classe B ou C (A, B et C étant 3 maladies par exemple évoquées dans le cadre d'un diagnostic différentiel) [12].

2.2.5 L'apprentissage par renforcement

L'algorithme apprend un comportement étant donné une observation. L'action de l'algorithme sur l'environnement produit une valeur de retour qui guide l'algorithme d'apprentissage [12].

3. Conclusion

L'IA est un domaine encore jeune mais qui reste très vaste, il a été sujet de plusieurs critiques depuis le début de sa création mais a pu quand même se développer au point d'achever des résultats assez surprenants.
L'IA a encore plus d'espace pour ce développé encore plus. En effet Il montre un grand potentiel pour faciliter la vie de tous les jours et même de pouvoir exploré des horizons étant non atteignable auparavant.
Ce chapitre a juste été un bref aperçu sur l'IA et en même temps une introduction vers le deuxième chapitre de notre thème intitulé l'apprentissage profond.

Chapitre II :
L’apprentissage profond

1. Introduction

Fondé dans les années 2010 c'est une technologie évoluée de l'apprentissage automatique, Andrew Yan-Tak Ng le co-fonder et le directeur de l'équipe google brain qui est spécialisé dans le domaine de l'IA a déclaré dans une interview de la magazine WIRED que l'apprentissage profond est un domaine très prometteur et qui ne cessera d'évolué, en effet le nombre de domaine important que ce dernier a touché au cours de cette décennie est juste époustouflant, on nomme quelques-uns : la médecine, les voiture intelligente (self driving cars, la cyber security….) [13].

Un des termes les plus important de l'apprentissage profond proviennent des sciences neurologiques, plus précisément de la notion de réseau de neurones, en effet les neurones dans l'apprentissage profond sont l'unité de base composant le logiciel cérébral, ils sont tout comme leurs homologues responsables des calculs et la liaison des données d'entré et de sortie se fait entre elles par l'intermédiaire d'un réseau complexe (réseau, cerveau) capable de prendre des décisions complexes.

Les couches intermédiaires entre les neurones permettent de traiter des problèmes complexes, sans elles, le système ne résout que des calculs simples. Le nombre de couches est donc un facteur décisif pour la complexité du système, et de l'apprentissage, les données s'associent d'une couche à l'autre, les résultats d'une première couche servant d'entrée à la prochaine, et ainsi de suite. Ce fonctionnement donne toute sa profondeur au réseau et à l'apprentissage. C'est ainsi que les termes apprentissage profond et les réseaux de neurones profonds sont nés.

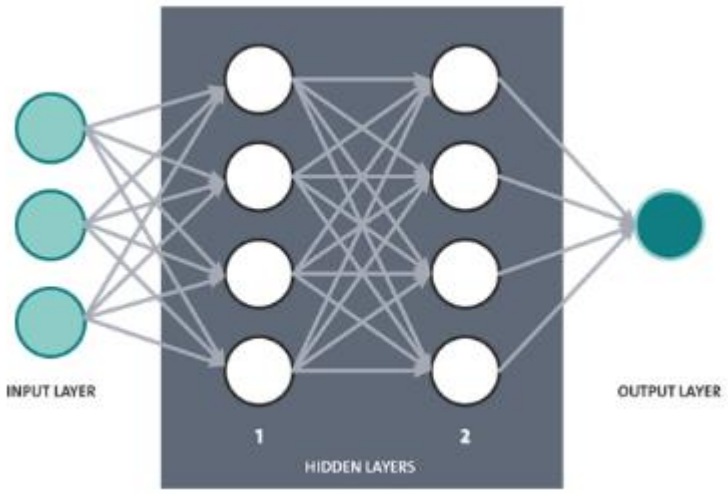

Figure 2.1 : *structure générale d'un réseau de neurones [13].*

2. La Différence entre l'apprentissage automatique et l'apprentissage profond

Le deep learning ou apprentissage profond est un sous-domaine de l'apprentissage automatique qui est apparu bien après la création de celle-ci, on illustre ça dans la figure suivante :

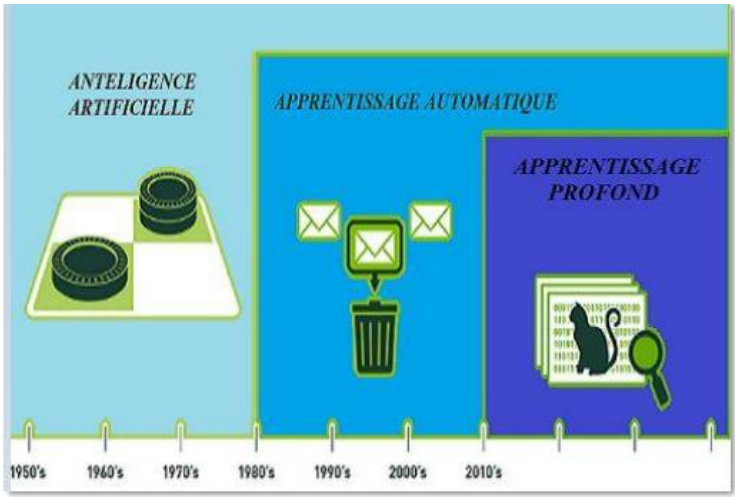

Figure 2.2 : *l'apprentissage profond par rapport à l'automatique et l'IA [16].*

En terme plus "approfondi" on prenant le traitement d'images comme exemple on aperçoit qu'un processus de l'apprentissage automatique commence par l'extraction manuelle de caractéristiques pertinentes à partir d'images. Un modèle qui catégorise les objets de l'image est ensuite créé en se basant sur les caractéristiques extraies. Dans un processus d'apprentissage profond, l'extraction de caractéristiques pertinentes à partir d'images est automatique. En outre, l'apprentissage profond effectue un apprentissage « de bout en bout » : à partir de données brutes, un réseau se voit assigner des tâches à accomplir (une classification, par exemple) et apprend comment les automatiser.

Une autre différence se voit dans la progression des deux apprentissages. L'apprentissage automatique évolue de manière plus au moins exponentielle, en effet le modèle s'améliore aux files de l'entrainement en ajoutant de plus en plus de données jusqu'à s'arrêté après un certain niveau, tandis que l'apprentissage profond a un graphe d'évolution linéaire, en d'autres termes il ne cesse d'évoluer tant qu'on lui fait nourrir plus de données [16].

3. Apprentissage profond

3.1 Définition

La notion d'apprentissage profond est tout d'abord une traduction directe du terme anglais « deep learning », que certain préfère traduire par la notion d'apprentissage statistique. De même que sa traduction, sa définition varie également, mais principalement au niveau des détails [14].

Pour définir cette notion dans les grandes lignes, on pourrait dire que :

L'apprentissage profond est un algorithme d'abstraction de haut niveau qui permet de modéliser les données à partir de grands ensembles de données apprises.

Précisons quelques termes :

L'abstraction suppose que les données initiales diffèrent largement des donnés de sorties, avec pour résultat possible la classification d'images, la prédiction d'un comportement ou une traduction. L'abstraction signifie qu'il n'y a pas de relation simple entre l'entrée et la sortie. Dans notre cas, il s'agit selon toute vraisemblance d'une relation inconnue, une sorte de « boîte noire ».

La modélisation signifie que nous tentons de créer un certain scenario réaliste de sorte qu'une classification ou un résultat réaliste en découle.

La notion relative aux grands ensembles de données apprises que les données d'entrée sont extrêmement diverses. L'apprentissage profond ou l'apprentissage automatique implique généralement que les propriétés importantes de ces données sont détectées lors du processus d'apprentissage [15].

3.2 Les architectures de l'apprentissage profond

Différents algorithmes ont été développés au fur et à mesure depuis la création de la notion du deep learning, on va aborder juste ceux qu'on va utiliser dans ce thème d'étude, on commence par :

3.2.1 Les réseaux de neurones convolutionnels (CNN)

Historiquement parlant, les premiers CNN ayant eu du succès ont été inventés en 1989 par LeCun [17] et appliqués à la reconnaissance d'écriture manuscrite. Ce système a été perfectionné par son auteur en 1998 [18].

Comme le nom l'indique c'est un type de réseau de neurones utilisant l'apprentissage supervisé et un certain type de neurones appelé convolutionnels.

Ils sont utilisés généralement dans la reconnaissance et le traitement d'images et spécifiquement conçu pour traiter les données de pixels. Ces derniers sont de puissants systèmes de traitement de données qui effectue des tâches à la fois génératives et descriptives, souvent à l'aide de Machine

Vision (ex caméra) qui inclut la reconnaissance d'images et de vidéos, ainsi que des systèmes de recommandation et le traitement du langage naturel (NPL) [19].

Les couches des réseaux de neurones convolutionnels sont liée par une convolution de 2 dimensions et non par des opérations matricielles et les entrées et sorties sont des tensors de dimension 3, dont 2 sont utilisé pour représenter les valeurs des pixels de l'image et la 3eme étant réservé pour le nombre de canaux, on appelle ça la feature map [19].

– Une image noire et blanc de dimension L*H : dimension 2 (L*H*1) car un seul canal.
– Une image en couleur de dimension L*H : dimension 3 (L*H*3) car plusieurs canaux d'entrées (Rouge, Vert et Bleu).

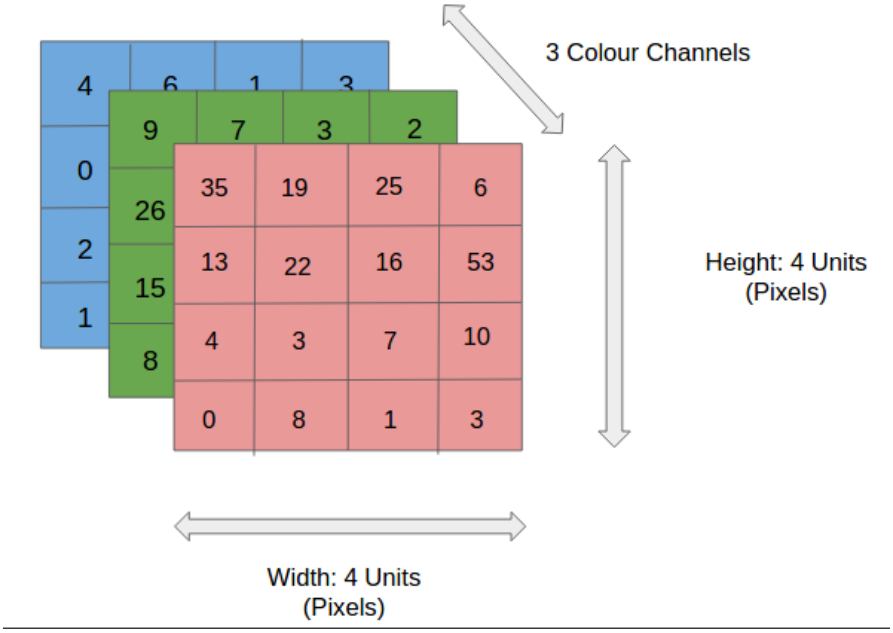

***Figure 2.3** : une image colorée transformé en un tensor de 3 dimensions (feature map)[20]*

- **Comment ça marche ?**

Les réseaux de neurones non convolutionnels apprend les paternes des pixels présente dans les photos et peuvent la détecter une autre fois mais les coordonnées ne doivent pas changer. Cependant le CNN apporte une approche dynamique à la détection parfaite, et ce en obligeant le réseau à réapprendre les paternes dans de nouveaux emplacements jusqu'à qu'il figure ce qu'il est censé de faire [20].

Dans l'exemple ci dessue d'un réseau non convolutionnels il suffisait juste de flipper l'image pour que la détection soit fausse.

Figure 2.4 : *detection des yeux d'un chien[20].*

Une autre méthode est d'appliquer un filtre qui nous aidera à identifier une paterne spécifique, on donne l'exemple suivant :

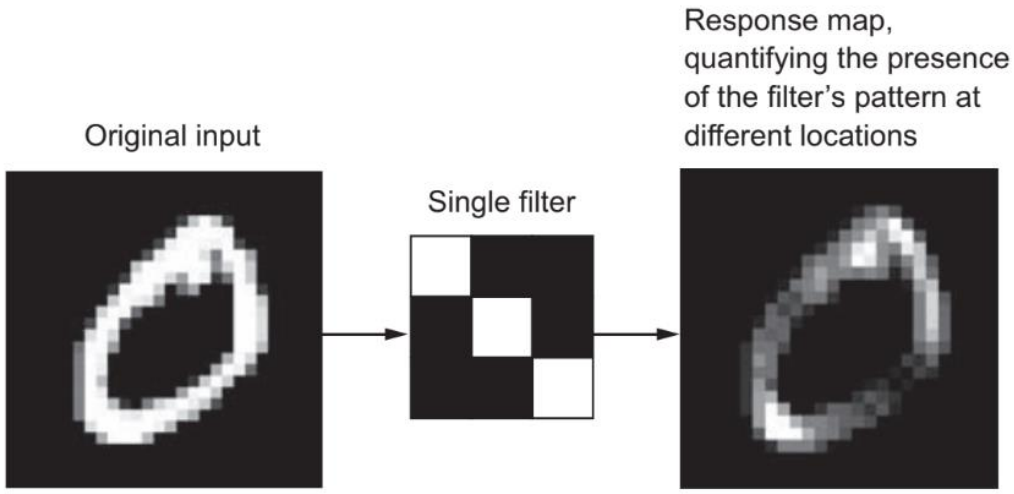

Figure 2.5 : *exemple d'un filtre appliqué sur une image [21]*

Sur notre gauche est l'original photo, au milieu est le filtre appliquer et la dernière photo est le résultat qui nous montre le taux de présence de la paterne qu'on va appeler response map (en général plusieurs response map sont générés car plusieurs filtres sont appliqués) [21].

- **Les couches fréquemment utilisé avec les couches convolutionnels**

Plusieurs de disaines de couches existes donc on va juste mentionner ceux qu'on a utilisé dans notre projet.

- Pooling :

Le principe du pooling est de sélectionner un nombre spécifique d'une sous matrice de la response map puis décaler d'un certain nombre de pixel (le décalage s'appelle stride), le type du pooling (min pooling, max pooling, average pooling) varie avec le nombre choisit (min, max, moyenne) [21].

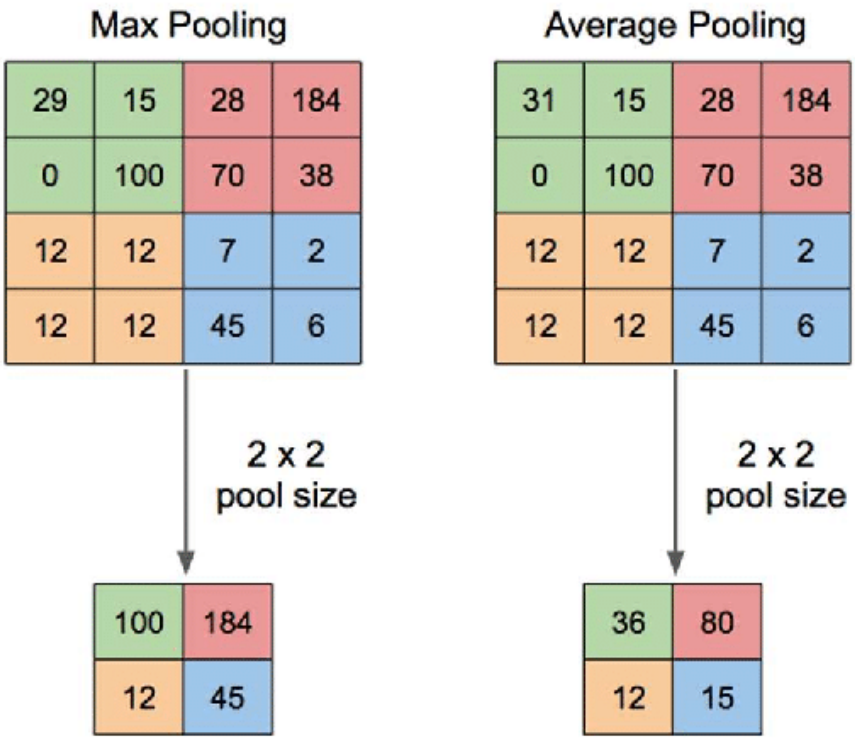

Figure 2.6 : *exemple sur un max pooling et un min pooling avec un stride de 2 pixels [21]*

- Padding :

Comme vous pouvez le constater déjà les cases au bordes des matrices ne peuvent jamais être au centre des calculs du pooling ce qui n'est pas vraiment grave mais qui comme même aide un peu pour avoir des résultats plus au moins meilleurs, pour y remédier on ajoute des pixels noirs autour de l'image [21].

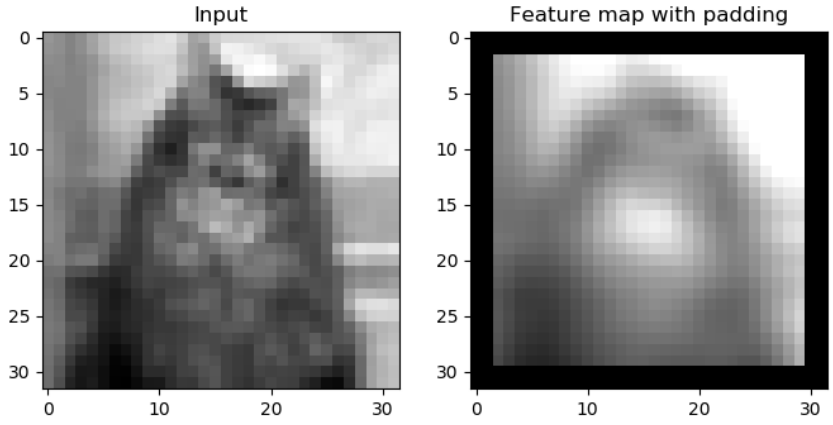

Figure 2.7 : *exemple d'un padding d'image [21]*

3.2.2 Les autos encodeurs (autoencoder)

Un auto-encodeur, ou auto-associateur est un réseau de neurones artificiels utilisé pour l'apprentissage non supervisé, il se compose toujours de deux parties, l'encodeur et le décodeur [22].

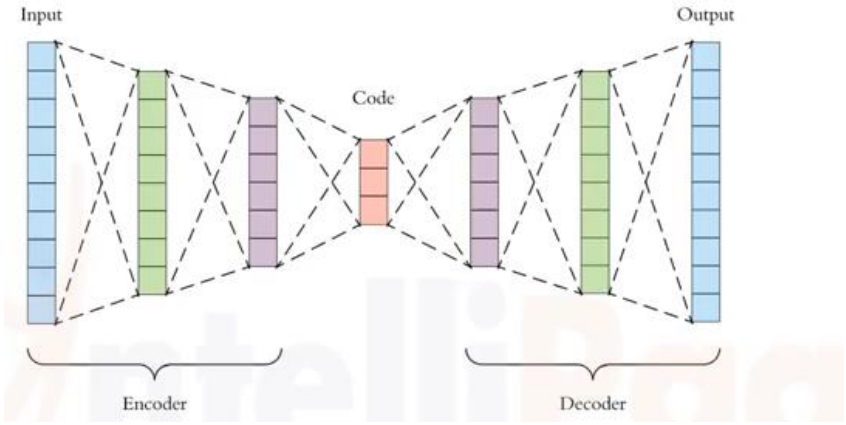

Figure 2.8 : *schéma général d'un auto-encodeur [15]*

Le principe de programmation d'un tel réseau en général se fait en imposant ce qu'on appelle un engorgement (bottleneck) au cours du l'apprentissage, en terme plus simple on limite la machine exprès lorsqu'elle s'entraine.

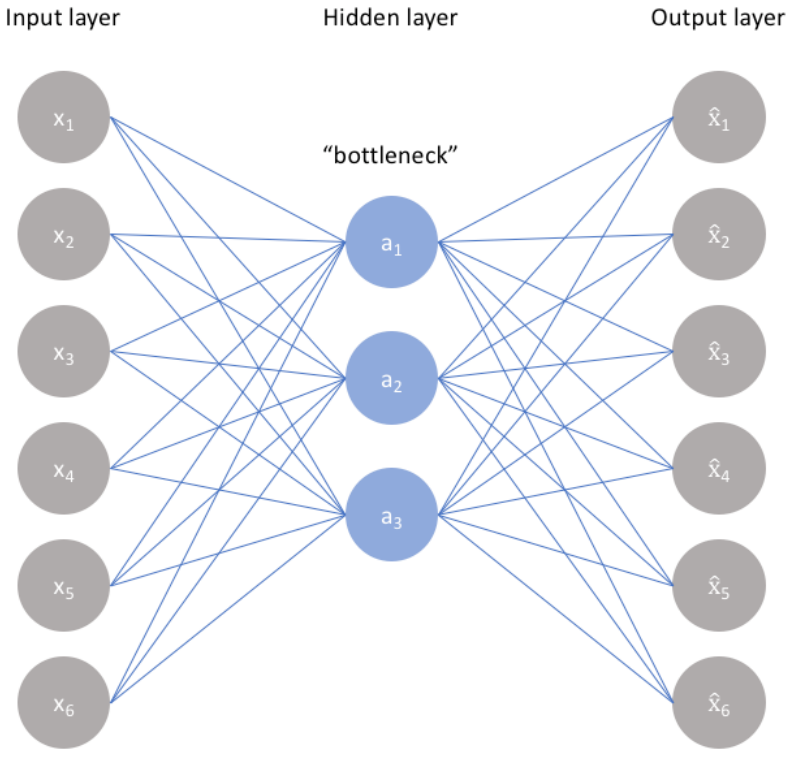

Figure 2.9 : *Principe de travail d'un auto-encodeur [22].*

Maintenant qu'on connait le principe de marche des auto-encodeurs, on va parler des deux types dérivés de cette technologie :

- **Les sparse auto-encodeurs:**

Ce sont des auto-encodeurs spécialisé dont la compression et la décompression des fichiers, la compression se fait vers des formats connus comme le rar et le zip, un exemple sur un auto-encodeur est le fameux winrar [22].

- **Les denoisers :**

Les denoisers sont un type d'auto-encodeurs dont le but est de restaurer à un certain degré une image très pixélisé et flou, le général fonctionnement de ce type est qu'il s'entraine à décompresser l'image flou et essaie de restaurer l'original, un exemple d'un denoiser connue et l'application remini qui restaure les visages des gens en une qualité full HD, c'est ce type d'auto-encodeur qui nous intéresse dans ce thème [22].

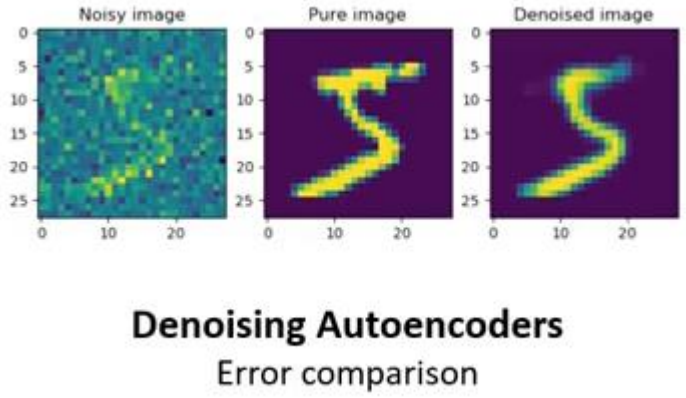

***Figure 2.10** : Exemple d'un denoiser d'image [22]*

3.2.3 Réseaux antagonistes génératifs (GAN)

Un GAN ou Generative Adversarial Network (réseau antagoniste génératif en français) est une technique de machine Learning non-supervisé qui servent à créer des imitations parfaite de données. On s'en sert dans beaucoup de domaine (traitement d'images, de texte, de sons …). Elle repose sur le fait que deux réseaux de neurones artificiels compètent dans un scénario de jeu à somme nulle. Les deux réseaux composant ce système sont :

- **Le générateur :**

C'est un réseau de neurones convolutif dont le rôle est de créer de nouvelles instances d'un objet.

- **Le discriminateur :**

C'est un type de réseau neuronal déconvolutif (CNN inversé) qui détermine l'authenticité de cet objet ou son appartenance à un jeu de données.

Ces deux entités sont en compétition pendant la phase d'apprentissage où les pertes se confrontent les unes aux autres afin d'améliorer les comportements, ce mécanisme étant appelé rétropropagation [22].

Les GAN sont de plus en plus connus comme une forme évoluée d'apprentissage automatique. Des chercheurs et des développeurs ont expérimenté l'utilisation de GAN pour produire des copies, même imparfaites, d'œuvres célèbres telles que la Joconde et des portraits de personnes qui n'existent pas (PersonNotExists) [23].

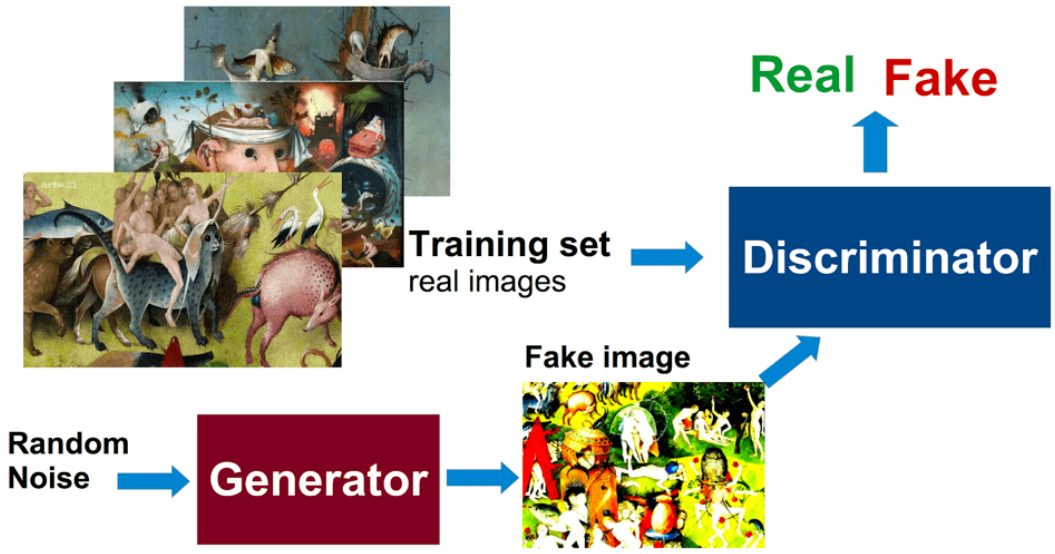

Figure 2.11 : *Exemple sur un système de GAN [23]*

4. Conclusion

L'apprentissage profond est sans doute le domaine le plus intéressant et le plus efficace au niveau des résultats. Il ouvre une multitude de portes qu'auparavant était juste un rêve lointain.

Dans ce chapitre on a abordé les points les plus cruciaux à notre travail que cette technologie nous a offerts dans nos mains car comme vous le doutez il reste un nombre considérable de points qu'on a laissé intentionnellement pour but d'être efficace tout comme cette branche de l'IA.

Le chapitre suivant parlera du deepfake la méthode utilisée et comment peut-on détecter un deepfake.

Chapitre III :
Les méthodes de Deepfake

1. Introduction

Le deep learning (Hyper Trucage) étant un des champs de l'IA les plus prometteurs ces années passées ne cesse de donner naissance à plusieurs technologies surprenantes qui nous ouvre des possibilités qu'auparavant été juste un rêve lointain. Une de ces technologies est celle qu'on a abordée dans ce mémoire : le deepfake.

La première apparition d'une deepfake était en 2016 avec la présentation du premier programme audio deepfake, Adobe Voco.

Tandis qu'en juillet 2017, la BBC diffuse un discours prononcé par une intelligence artificielle reproduisant Obama, discours essentiellement indiscernable de la réalité et ça été la première apparition d'une vidéo deepfake.

Vous vous demandez peut-être c'est quoi un deepfake ?

Un Deepfake est une vidéo ou un enregistrement audio produit ou altéré grâce à l'intelligence artificielle. Le terme désigne non seulement le contenu ainsi généré, mais aussi les technologies utilisées à cet effet.

Le mot Deepfake est une contraction entre " Deep Learning " et " Fake " que l'on pourrait traduire par " faux profond ". En effet, il s'agit de contenus fallacieux, rendus profondément crédibles grâce à l'intelligence artificielle. Plus précisément au Deep Learning ou apprentissage approfondi [24].

Figure 3.1 : *Exemple d'un deepfake faceswap [24]*

Il existe deux types de deepfake, les faceswap (echange de visage) dont le résultat est l'échange des traits d'un visage d'une personne avec une autre, le deuxième est le « facial reenactment » (reconstitution faciale) qui quant à lui imite les mouvements facials d'une personne et crée un réplica avec un visage d'une autre.

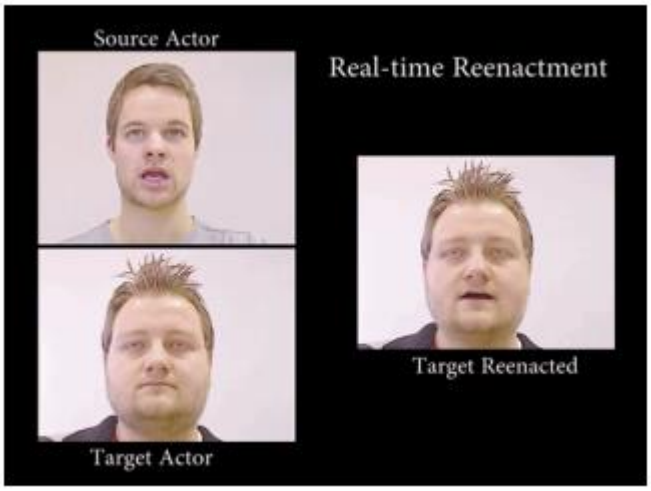

Figure 3.2 : *exemple d'un deepfake facial reenactment [24]*

2. Principes de marche d'un deepfake

La structure générale de tous les deepfakes est la même, en effet on a besoin de 2 fichiers multimédia source d'un système GAN et d'un auto-encodeur (denoiser).

Figure 3.3 : *Structure général d'un Deepfake [24]*

Les auto-encodeurs de type denoiser se charge de compresser la résolution des images extraites de la vidéo et de faire en sorte d'éliminer la partie pixélisée flou, ensuite la partie génératrice du système GAN commence par dessiner le visage à l'aide de ces mêmes auto-encodeurs source dans le visage du destinataire, ensuite la partie discriminatrice juge si la photo semble correcte ou pas pour dire au générateur que le travail est à refaire ou non. Quand on obtient un résultat satisfaisant ou quand le GAN atteint le choc point c'est alors là qu'on s'arrête

3. Quelques méthodes de création du deepfake (faceswap)

3.1 Avec un GAN basé sur le CNN

C'est l'une des premières méthodes utilisées dans la création des deepfake, elle utilise un GAN dont le générateur est un système traditionnel d'encoder-decoder et un réseau CNN comme discriminateur, l'avantage de ces réseaux est qu'ils sont efficaces et donnent des résultats excellents. Par contre ils sont connus pour avoir une structure rigide et non flexible, en effet dans notre cas les photos doivent être d'une même dimension sinon le discriminateur échoue et ne traite pas les images, plusieurs applications utilisent ce modèle on nomme la fameuse application android faceswap.

3.2 Méthodes avec un GAN basé sur LSTM :

LSMT est l'abréviation de long-short term memory ce type de GAN a comme les GAN traditionnels un générateur et un discriminateur en plus d'un réseau LSTM reliant les deux. LSMT est un type de réseaux de neurones RNN améliorés qui est utilisé généralement pour la prédiction des séquences de données, il dispose d'un mécanisme d'oubli (forget gate) qu'il lui permet d'éviter les problèmes de perte de données causées par le rétrécissement des gradients de multiplication des matrices utilisées dans les couches de ce dernier et aussi de la perte de données causées par la croissance exponentielle de ces mêmes gradients.

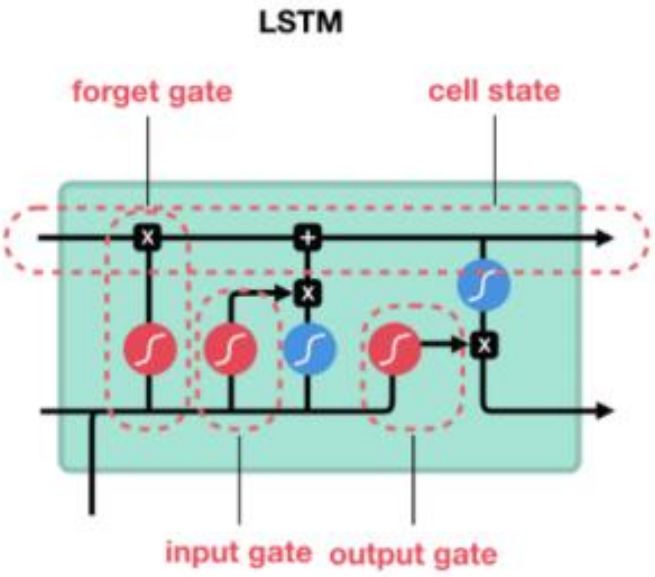

Figure 3.4 : *Structure d'un réseau LSTM [26]*

L'avantage de ces réseaux est qu'ils sont plus flexibles et acceptent la différence de dimensions entre les échantillons mais les résultats dans le traitement d'images restent plus au moins "acceptables ». Un des travaux du deepfake avec cette technologie est celui de Akhil Santha de l'université de Rochester [26].

4. Conclusion

Dans ce chapitre on a abordé la partie création de ce thème d'un point de vue objectif, on a donné le fonctionnement et l'architecture générale d'un deepfake et quelques exemples sur les différents types de modèles.

Les modèles cités ci-dessus étant de très bonne implémentation de l'idée du deepfake, en revanche ces méthodes manquent d'efficacité au niveau de l'entrainement du modèle (il prend beaucoup de temps plusieurs dizaines de jours voir des mois) en plus de donner des résultats plus au moins acceptables.

Dans le chapitre suivant nous proposerons une solution plus efficace basée sur le projet deepfacelab tout en suggérant un modèle basé sur l'architecture CNN qui résoudra le dilemme de la détection.

Chapitre IV :
Implémentation et Résultats

1. Introduction

Toute la partie théorique étant déjà couverte dans les chapitres précédents, ce chapitre abordera le travail conçu pour répondre aux insuffisances et limites des modèles précédents, tout en proposant une solution efficace à la détection des deepfakes.

2. La Configuration du Matériel Utilisé

L'implémentation a été réalisé sur un pc de bureau avec un processeur Intel Core i5-7400 3GHz, 8 Go de RAM et un GPU NVIDIA RTX 2060. Le système d'exploitation était Linux version Ubuntu 20.04.2.0 LTS (Focal Fossa).

3. Logiciel et bibliothèque utilisé dans l'implémentation

a. Python

Python est un langage de programmation puissant de haut niveau, à la fois facile à apprendre, il prend en charge plusieurs modèles de programmation (procédural, fonctionnel et orienté objet). Les bibliothèques (packages) de python encouragent la modularité et la réutilisabilité des codes existants. Python et ses bibliothèques sont disponibles sans difficulté pour la majorité des plateformes et il peut être redistribué gratuitement. On estime que c'est l'un des langages de programmation les plus utilisés au monde [27].

b. Keras

Keras est une API (interface de programmation applicative) de réseaux neuronaux de haut niveau, écrite en Python et capable de fonctionner sur TensorFlow ou Theano. Il a été développé et maintenu par François Chollet pour mettre en place des modèles d'apprentissage en profondeur aussi rapides et faciles que possible pour la recherche et le développement. Keras fonctionne sur Python 2.7 ou 3.5 et peut parfaitement s'exécuter sur les processeurs graphiques GPU et les processeurs (unité centrale de traitement CPU) [28].

c. Tensorflow

TensorFlow est une bibliothèque open-source qui existe dans de nombreux langages de programmation tels que Python, Javascript et C++, il a été initialement lancé par l'équipe de recherche de Google artificielle intelligence pour mener des recherches sur l'apprentissage automatique et les réseaux neuronaux profonds. TF peut également être utilisé dans une grande variété d'autres domaines. [29]

d. Jupyter Notebook

Jupyter notebook est un IDE sous forme d'une application web conçu d'abord pour les langages Julia, R et python mais a évolué à la suite pour supporter plusieurs autres langages.

Basé sur l'architecture client-server il permet de compiler le code programmé sur le navigateur web par défaut offline en créant un server local sur la machine mais il peut aussi être utilisé par distance via internet sur un server.

L'avantage de cet IDE est qu'il facile à utiliser et permit de compiler par block de code sans pour autant compiler tout le programme une seule fois [30].

e. DeepFaceLab

C'est un système d'outils deepfake open-source crée par iperov pour le faceswap, il est a plus de 3,000 fork et 13,000 star dans Github. Il est un des meilleurs discriminateurs du domaine, entrainé avec plusieurs milliers de photos et on va s'en servir pour l'entrainement de notre modèle.

4. Base de données

On a utilisé deux différentes bases de données pour l'entrainement des deux modèles qu'on a conçu, une pour la création des deepfakes et l'autre pour leur détection. La résolution peut varier selon les vidéos utilisés car on va redimensionner toutes les photos vers la résolution 512*512 en couleur RGB format Jpg, quant à la taille elle varie selon la longueur des vidéos.

La base de données réservée à la création contient deux fichiers nommés :

- **data_src :** Dans ce fichier on trouve l'ensemble de photos (2697 photos) extraites de la vidéo source, c'est d'ici qu'on extrait les trait facial qu'on veut copier.

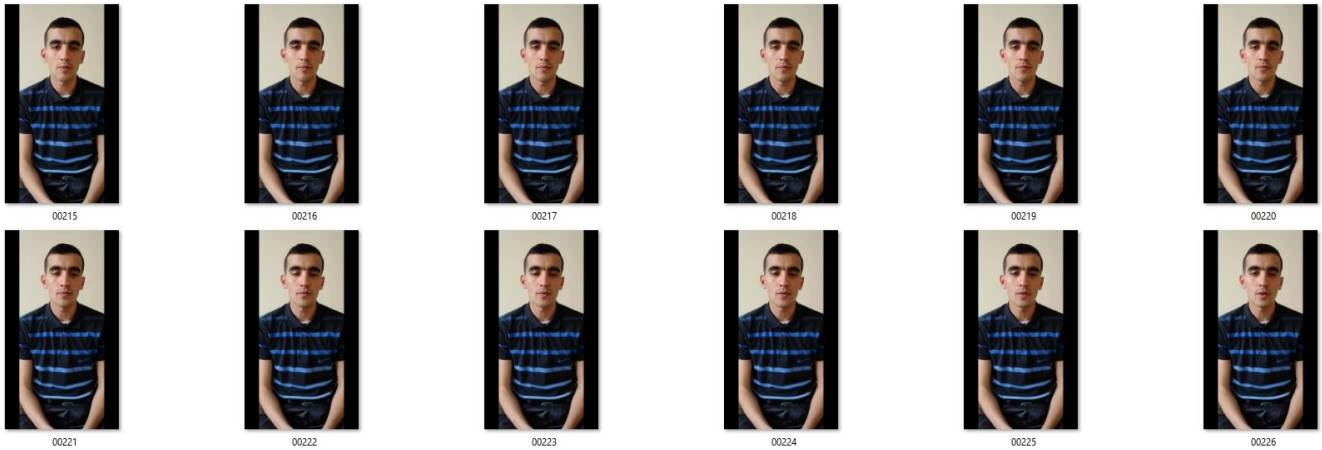

Figure 4.1 : *Quelques images contenue dans le fichier data_src.*

- **data_dst :** Par contre dans celui-ci on trouve l'ensemble de photos (1775 photos) extraites de la vidéo destinataire, c'est dans cette vidéo qu'on va coller les trait facials copier de la vidéo source.

Figure 4.2 : *Quelques images contenue dans le fichier data_dst*

La base de données réservée à la détection contient aussi deux fichiers nommés :

- **fake :** Ce dossier contient les photos extraites de la vidéo qu'on veut tester, la résolution ici va être redimensionnée vers 224*224 en couleur RGB et le nombre de photo dépend de la longueur de la vidéo.

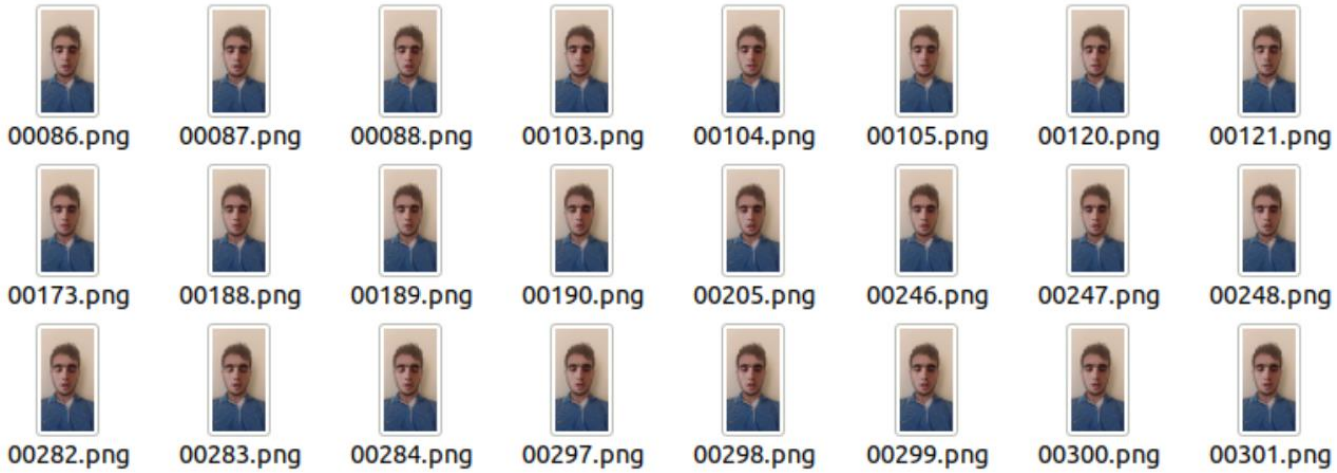

Figure 4.3 : *Quelque image contenue dans le fichier fake*

- **Non fake :** Ce dossier a été créé juste pour éviter l'effondrement de l'entrainement vu qu'on a utilisé un modèle CNN Supervisé, Il contient qu'une image d'une chaise.

5. L'architecture des deux modèles utilisés

5.1 Le modèle de la création des deepfakes

Le modèle de création des deepfake qu'on a conçu repose sur un GAN. Ce dernier est basé sur l'architecture CNN vu l'efficacité de l'entrainement au niveau du résultat et au niveau du temps consumé.

Le problème avec l'architecture CNN comme on a déjà mentionné est le changement de résolution, donc on a utilisé quelques denoisers (proposé par le projet DeepFaceLab) pour le redimensionnement des échantillons vers une résolution standard (512*512) puis on isole les visages et écrase les images précédentes.

***Figure 4.4** : Les images des visages du dossier data_dst isolées*

Après que les échantillons soient près, on les encode avec l'encoder Lavf 58.29.100 ensuite on les décode avec le decodeur Lavc 58.59.100 qui sont un type de denoisers avec l'ajout de quelque couches convolutionelles pour ne pas perdre les paternes du trait du visage.

In [ ]:
```
class Encoder(nn.ModelBase):
    def on_build(self, in_ch, e_ch, ae_ch):
        self.down1 = Downscale(in_ch, e_ch, kernel_size=5)
        self.res1 = ResidualBlock(e_ch)
        self.down2 = Downscale(e_ch, e_ch*2, kernel_size=5)
        self.down3 = Downscale(e_ch*2, e_ch*4, kernel_size=5)
        self.down4 = Downscale(e_ch*4, e_ch*8, kernel_size=5)
        self.down5 = Downscale(e_ch*8, e_ch*8, kernel_size=5)
        self.res5 = ResidualBlock(e_ch*8)
        self.dense1 = nn.Dense( lowest_dense_res*lowest_dense_res*e_ch*8, ae_ch )

    def forward(self, inp):
        x = inp
        x = self.down1(x)
        x = self.res1(x)
        x = self.down2(x)
        x = self.down3(x)
        x = self.down4(x)
        x = self.down5(x)
        x = self.res5(x)
        x = nn.flatten(x)
        x = nn.pixel_norm(x, axes=-1)
        x = self.dense1(x)
        return x
```

In [ ]:
```
class Decoder(nn.ModelBase):
    def on_build(self, in_ch, d_ch, d_mask_ch ):
        self.upscale0 = Upscale(in_ch, d_ch*8, kernel_size=3)
        self.upscale1 = Upscale(d_ch*8, d_ch*4, kernel_size=3)
        self.upscale2 = Upscale(d_ch*4, d_ch*2, kernel_size=3)

        self.res0 = ResidualBlock(d_ch*8, kernel_size=3)
        self.res1 = ResidualBlock(d_ch*4, kernel_size=3)
        self.res2 = ResidualBlock(d_ch*2, kernel_size=3)

        self.out_conv  = nn.Conv2D( d_ch*2, 3, kernel_size=1, padding='SAME')

        self.upscalem0 = Upscale(in_ch, d_mask_ch*8, kernel_size=3)
        self.upscalem1 = Upscale(d_mask_ch*8, d_mask_ch*4, kernel_size=3)
        self.upscalem2 = Upscale(d_mask_ch*4, d_mask_ch*2, kernel_size=3)
        self.out_convm = nn.Conv2D( d_mask_ch*2, 1, kernel_size=1, padding='SAME')

        if 'd' in opts:
            self.out_conv1  = nn.Conv2D( d_ch*2, 3, kernel_size=3, padding='SAME')
            self.out_conv2  = nn.Conv2D( d_ch*2, 3, kernel_size=3, padding='SAME')
            self.out_conv3  = nn.Conv2D( d_ch*2, 3, kernel_size=3, padding='SAME')
            self.upscalem3 = Upscale(d_mask_ch*2, d_mask_ch*1, kernel_size=3)
            self.out_convm = nn.Conv2D( d_mask_ch*1, 1, kernel_size=1, padding='SAME')
        else:
            self.out_convm = nn.Conv2D( d_mask_ch*2, 1, kernel_size=1, padding='SAME')

    def forward(self, inp):
        z = inp

        x = self.upscale0(z)
```

Figure 4.5 : *Code source de la partie encoder-decoder de l'entrainement*

Notre système GAN se compose éventuellement d'une partie génératrice et une partie discriminatrice. L'idée derrière la partie génératrice est de créer des masques du visage source et destinataire à l'aide de Landmarks (coordinations de point de détection des traits du visage dans notre cas).

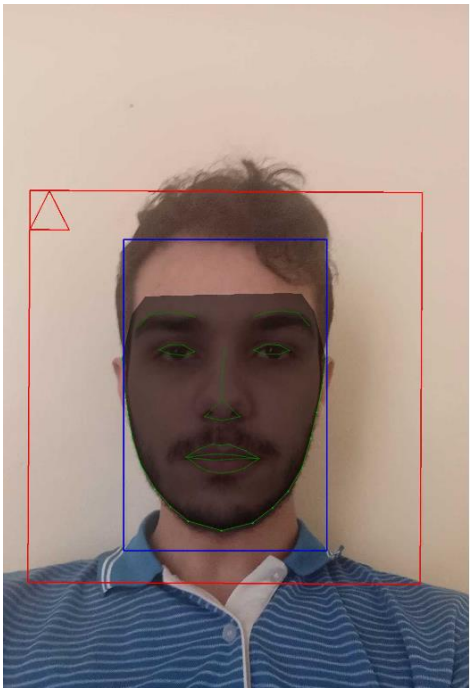

Figure 4.6 : *Mask crée avec les Landmark*

```
from core import imagelib
from core import mathlib
from facelib import FaceType
from core.mathlib.umeyama import umeyama
```

```
In [ ]: landmarks_2D = np.array([
[ 0.000213256,  0.106454  ], #17
[ 0.0752622,    0.038915  ], #18
[ 0.18113,      0.0187482 ], #19
[ 0.29077,      0.0344891 ], #20
[ 0.393397,     0.0773906 ], #21
[ 0.586856,     0.0773906 ], #22
[ 0.689483,     0.0344891 ], #23
[ 0.799124,     0.0187482 ], #24
[ 0.904991,     0.038915  ], #25
[ 0.98004,      0.106454  ], #26
[ 0.490127,     0.203352  ], #27
[ 0.490127,     0.307009  ], #28
[ 0.490127,     0.409805  ], #29
[ 0.490127,     0.515625  ], #30
[ 0.36688,      0.587326  ], #31
[ 0.426036,     0.609345  ], #32
[ 0.490127,     0.628106  ], #33
[ 0.554217,     0.609345  ], #34
[ 0.613373,     0.587326  ], #35
[ 0.121737,     0.216423  ], #36
[ 0.187122,     0.178758  ], #37
[ 0.265825,     0.179852  ], #38
[ 0.334606,     0.231733  ], #39
[ 0.260918,     0.245099  ], #40
[ 0.182743,     0.244077  ], #41
[ 0.645647,     0.231733  ], #42
[ 0.714428,     0.179852  ], #43
[ 0.793132,     0.178758  ], #44
[ 0.858516,     0.216423  ], #45
[ 0.79751,      0.244077  ], #46
[ 0.719335,     0.245099  ], #47
[ 0.254149,     0.780233  ], #48
[ 0.340985,     0.745405  ], #49
```

Figure 4.7 : *Les coordonées Landmark du visage du destinataire*

Après que le traitement des créations des masques de chaque personne est terminé les denoisers commence à apprendre à dessiner chaque visage séparément ensuite ils essaient de dessiner le visage source dans la tête du visage destinataire en encodant ce dernier avec les valeurs trouvées anciennement dans l'entrainement ensuite ils les décodent avec les valeurs du décoder du visage source trouvées dans l'entrainement. La figure ci-dessus explique le fonctionnement général du système :

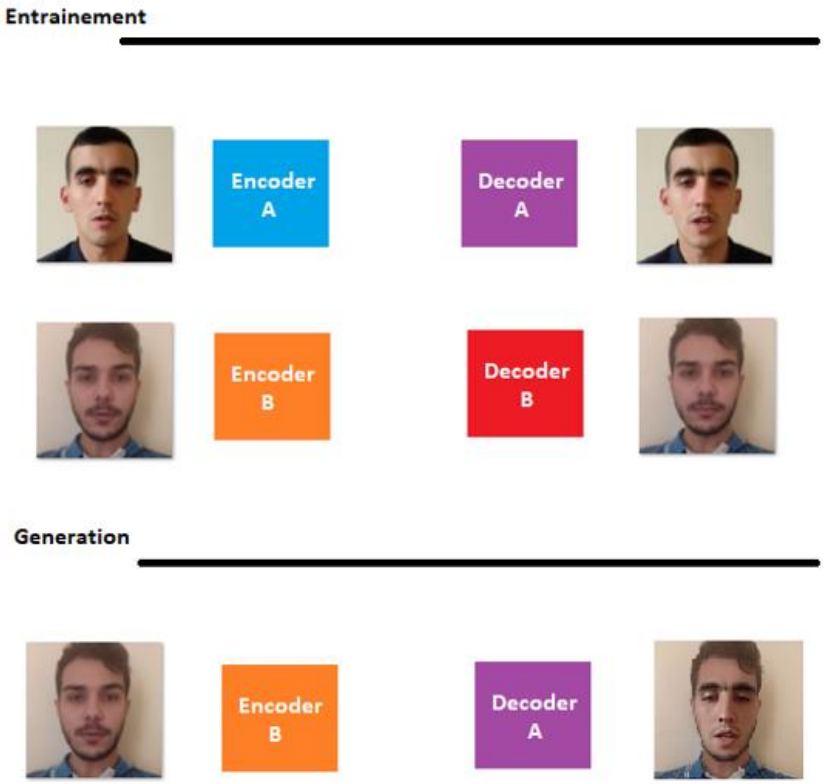

Figure 4.8 : *Fonctionnement général de la partie génératrice du GAN*

La partie discriminatrice du modèle est celle offerte par le projet open source DeepFaceLab, c'est un modèle CNN entrainé par des milliers de visages humains authentiques et non authentiques, voici un schéma que nos proposant qui résumera toutes les idées expliquées dans cette partie :

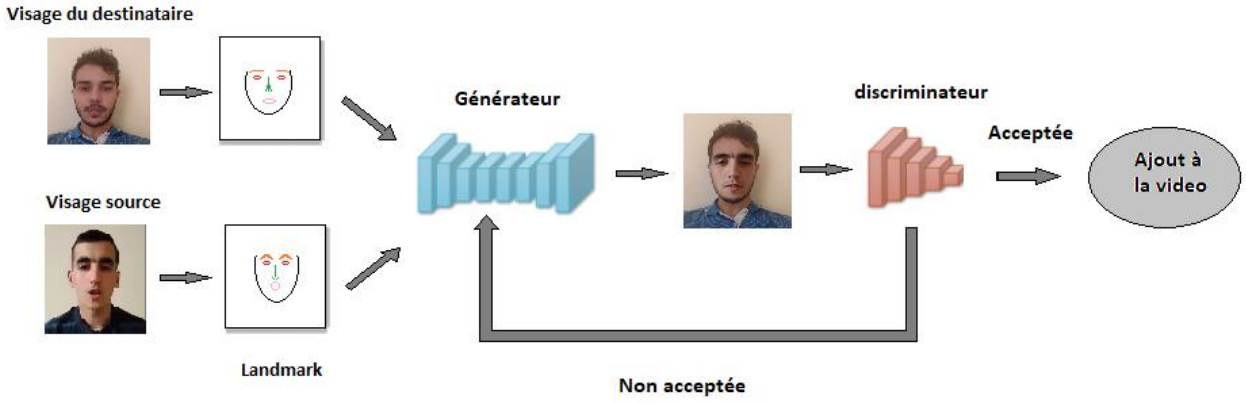

Figure 4.9 : *Le système GAN de la partie création du thème*

5.2 Le modèle de la détection des deepfakes

Les modèles de détection des deepfake suivent en général une approche basée sur les réseaux LSTM (RNN) vu la nature des données séquentielles, par contre notre méthode est semblable à celle de la création, en effet on va utiliser une architecture CNN pour crée notre modèle vu l'efficacité de cette dernière avec le traitement d'image, le modèle exploite une petite faille au niveau du changement des proportions du visage que les deepfake simple en souffre.

5.2.1 Pré-traitement

La première chose qu'on a fait après le découpage de la vidéo à détecter en images est de créer 2 dossiers, un pour l'entrainement contenant environ 50% des images (detection_dataset) et un pour le test d'authenticité contenant le reste (detection_test).

La première partie du code contiendra 2 fonctions génératrices car en effet le fichier detection_dataset après être redimensionner vers une résolution de 244*244 RGB va être classé en 2, une partie pour l'entrainement (80%) et une partie pour la validation (20%) par les fonctions train_generator et val_generator successivement.

```
In [6]: train_generator = data_generator.flow_from_directory(
            base_dir,
            target_size=(IMAGE_SIZE, IMAGE_SIZE),
            batch_size=BATCH_SIZE,
            subset='training')

        Found 709 images belonging to 2 classes.

In [7]: val_generator = data_generator.flow_from_directory(
            base_dir,
            target_size=(IMAGE_SIZE, IMAGE_SIZE),
            batch_size=BATCH_SIZE,
            subset='validation')

        Found 177 images belonging to 2 classes.
```

Figure 4.10 : *code source des fonctions génératrices*

5.2.2 Création du modèle

Maintenant il est temps de crée notre modèle, l'architecture standard et logique est d'utilisé l'algorithme de masquage pour la détection des traits du visage, cependant cette approche étant efficace côté résultat reste un processus très long à entrainer, après quelques études des différentes solutions possible à la détection on a opté pour l'utilisation d'un modèle de détection d'objet qui va nous servir de modèle de base.

```
In [12]: base_model = tf.keras.applications.MobileNetV2(input_shape=IMG_SHAPE,
                                                         include_top=False,
                                                         weights='imagenet')

In [13]: base_model.trainable = False
```

Figure 4.11 : *Le modèle de base utilisé*

C'est un modèle intégré à Keras développer par Google entrainer sur la base de données d'images ImageNet qui a plus de 1.4 million d'images avec plus de 1000 classes, on va utiliser que les dernières couches du modèle ceux qu'on appelle les "bottleneck layers" car les couches intermédiaires sont spécialisées au travail qu'on leur a demandé d'apprendre donc ils ne sont pas utiles à un nouvel entrainement, c'est pour ça qu'on a mis le paramètre trainable à false.

Après que le modèle de base est configuré correctement on y ajoute quelques couches convolutifs pour personnaliser la détection générale d'objet en détection de visage, les couches sont comme suit :

- **Couche convolutif 2D** : C'est la première couche avec comme paramètres 32 nœud, un noyau de taille 3 et une fonction d'activation relu, les nœuds dans cette fonction agisse comme des filtres pour extraire les features maps quant à la fonction de relu, elle sert à éliminer les valeurs négatives des sorties des nœuds.
- **Couche dropout :** La méthode du dropout consiste à désactiver des sorties de neurones aléatoirement pendant la phase d'apprentissage pour éliminer les neurones potentiellement inutiles avec un taux spécifique, dans notre cas on a choisi d'utilisé 20%.
- **Couche average pooling 2D :** Comme expliqué dans les chapitres précédents cette couche extrait des valeurs des feature map selon le type utilisé, dans notre cas on utilise l'average pooling qui a pour avantage de renvoyer le moyen pourcentage de présence des traits du visage, donc le modèle peut mieux identifier le propriétaire du visage que dans le cas d'utiliser un max pooling ou un min.
- **Couche dense :** Aussi appeler la couche entièrement connectée, c'est la couche finale de n'importe quel modèle CNN, elle consiste essentiellement en des poids (paramètres) que nous multiplions par l'entrée, nous ajoutons des biais b et appliquons une fonction d'activation, le principal rôle de cette couche est de connecter tous les nœuds vers les classes finales voulue, dans notre cas le nombre de classe est de 2 et on a utilisé une fonction d'activation softmax qui donne la probabilité d'appartenance d'un élément à une certain classe.

```
In [14]: model = tf.keras.Sequential([
             base_model,
             tf.keras.layers.Conv2D(32, 3, activation='relu'),
             tf.keras.layers.Dropout(0.2),
             tf.keras.layers.GlobalAveragePooling2D(),
             tf.keras.layers.Dense(2, activation='softmax')
         ])

In [15]: model.compile(optimizer=tf.keras.optimizers.Adam(),
                       loss='categorical_crossentropy',
                       metrics=['accuracy'])
```

Figure 4.12 : *Les couches du modèle CNN utilisé*

Le compiler utilisé pour le modèle est le compiler de Keras Adam, il permet de réduire le temps d'entrainement et le coup de VRAM, c'est le compiler le plus utilisé ces dernières années.

Le modèle final crée est un modèle séquentiel avec plus de 2.6M paramètres avec 368 milles qu'on va entrainer, la figure suivante est un résumé de ce qu'on a parlé dans cette partie :

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
mobilenetv2_1.00_224 (Functi (None, 7, 7, 1280)        2257984
_________________________________________________________________
conv2d (Conv2D)              (None, 5, 5, 32)          368672
_________________________________________________________________
dropout (Dropout)            (None, 5, 5, 32)          0
_________________________________________________________________
global_average_pooling2d (Gl (None, 32)                0
_________________________________________________________________
dense (Dense)                (None, 2)                 66
=================================================================
Total params: 2,626,722
Trainable params: 368,738
Non-trainable params: 2,257,984
```

Figure 4.13 : *Résumé sur le modèle CNN*

6. L'entrainement et les résultats

6.1 Modèle de la création

Le modèle de la création des deepfakes a été long à entrainer vu la quantité d'informations à traiter et l'utilisation d'une architecture assez complexe, le résultat reste très prometteur, la figure ci-dessous montre l'entrainement à travers les différentes itérations :

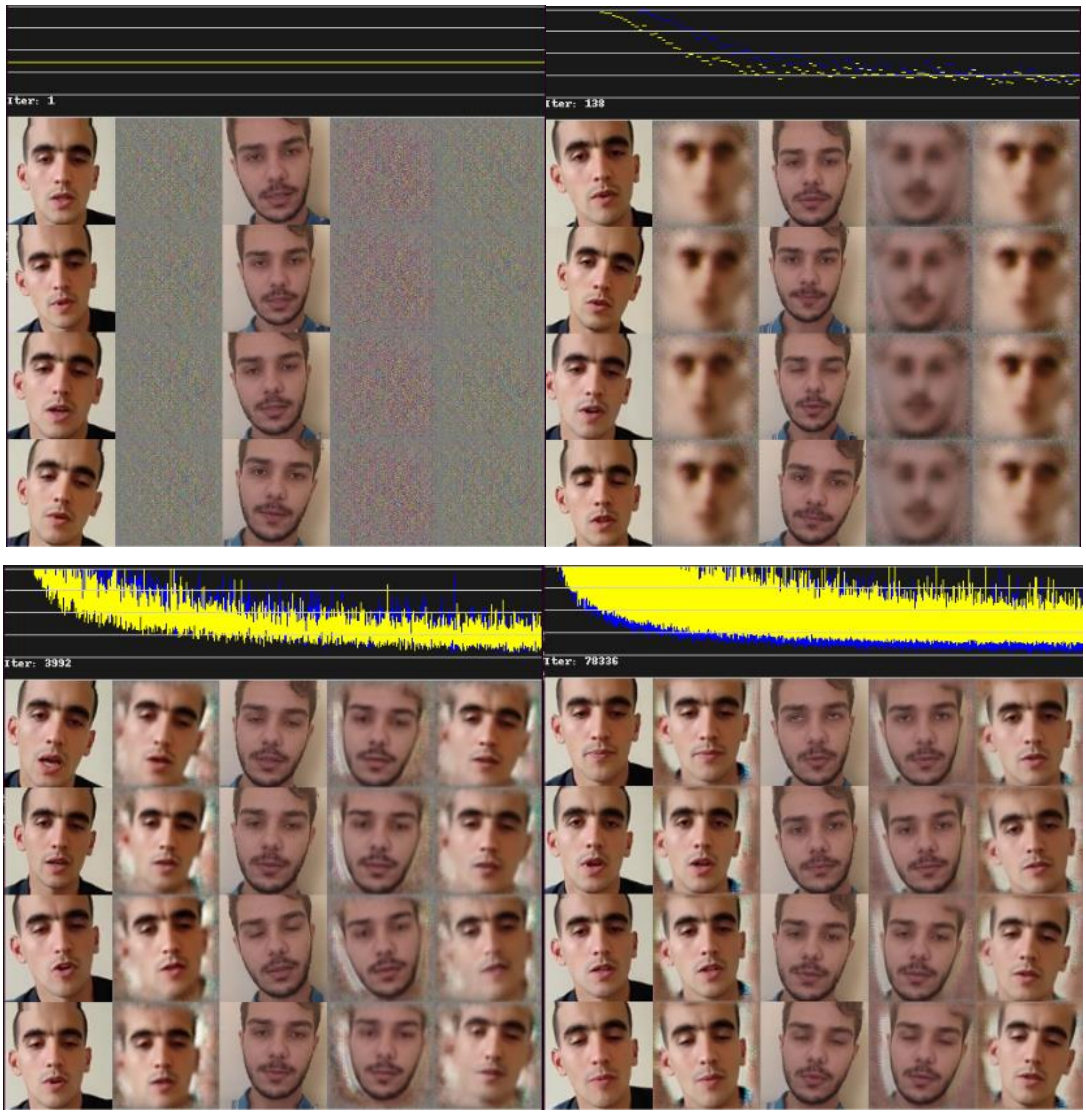

Figure 4.14 : *Les résultats du modèle à différente itérations*

Le nombre d'itérations finale est de 78336 avec un temps total de 2 heures d'entrainement, nous mentionnons que la courbe dont la couleur jaune représente le spectrum du taux de précision d'encoudage-decoudage de chaque frame de la vidéo source et la courbe bleu représente celle du destinataire.

Le taux d'erreur du modèle se calcule via le jeu de somme nulle du GAN, en terme plus simple on calcule le taux de rejet des images par le discriminateur.

```
Starting. Press "Enter" to stop training and save model.

Trying to do the first iteration. If an error occurs, reduce the model parameters.

[14:00:28][#000002][0098ms][5.2409][5.1163]
[14:15:24][#009214][0104ms][0.4692][0.4492]
[14:30:24][#018928][0091ms][0.2363][0.2447]
[14:45:24][#028732][0100ms][0.1892][0.1995]
[15:00:24][#038510][0118ms][0.1648][0.1766]
[15:15:24][#048284][0117ms][0.1503][0.1628]
[15:30:24][#058084][0104ms][0.1403][0.1534]
[15:45:24][#067924][0087ms][0.1330][0.1470]
[16:00:24][#077799][0133ms][0.1274][0.1419]
[16:01:14][#078343][0088ms][0.1189][0.1534]
```

Figure 4.15 : *Score du système GAN du modèle durant l'entrainement*

Afin de mieux montrer l'efficacité des résultats de notre travail on a utilisé l'application Reface avec les mêmes personnes, la figure suivante parle d'elle-même :

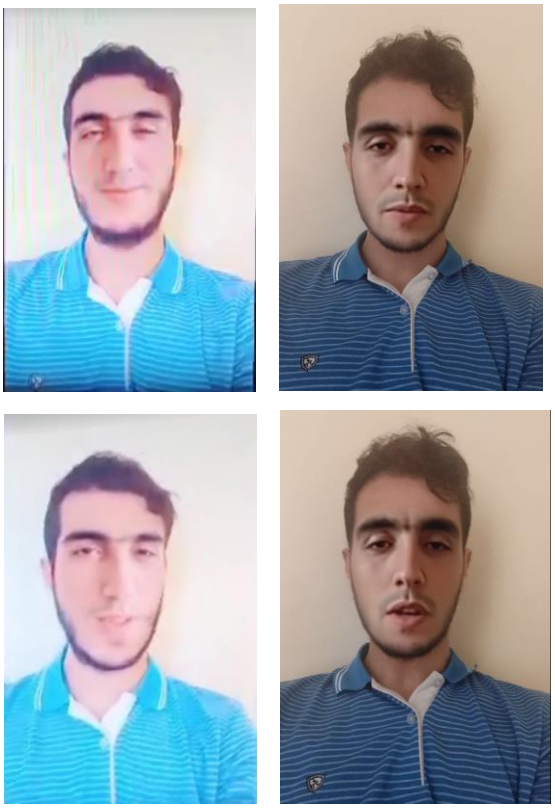

Figure 4.16 : *Comparaison des faceswap avec l'application Reface*

6.2 Modèle de la détection

Le modèle de la détection a été plus optimisé sur l'efficacité du traitement et de l'entrainement au niveau du temps consumé vu le grand nombre des vidéo générées, on a opté pour 3 epoches avec un temps d'exécution total d'une minute et demi avec une précision de 100% et un loss presque nulle, la figure ci-dessus montres ces statistiques :

```
In [16]: epochs = 3

In [17]: history = model.fit(train_generator,
                             epochs=epochs,
                             validation_data=val_generator)

Epoch 1/3
355/355 [==============================] - 36s 93ms/step - loss: 0.0673 - accuracy: 0.9813 - val_loss: 2.4367e-04
- val_accuracy: 1.0000
Epoch 2/3
355/355 [==============================] - 31s 87ms/step - loss: 0.0013 - accuracy: 1.0000 - val_loss: 7.2064e-08
- val_accuracy: 1.0000
Epoch 3/3
355/355 [==============================] - 31s 86ms/step - loss: 5.9236e-05 - accuracy: 1.0000 - val_loss: 2.4246e
-08 - val_accuracy: 1.0000
```

Figure 4.17 : *Historique d'entrainement du modèle de détection*

Comme vous pouvez le constater 2 epoches était largement suffisante pour avoir les mêmes résultats sois qu'une minute d'entrainement.

Pour montrer l'efficacité de notre modèle nous avons utilisé la vidéo deepfake crée sur l'outil web proposé par l'antivirus Zemana, leur modèle met un peu plus de temps à exécuter par contre il couvre d'autre aspect de détection comme le type d'encodage utilisé et la signature vidéo et s'améliore progressivement en s'entrainant à fur et à mesure grâce aux utilisateurs qui upload leur vidéo au site, la figure suivante montre les résultats obtenus avec leur modèle :

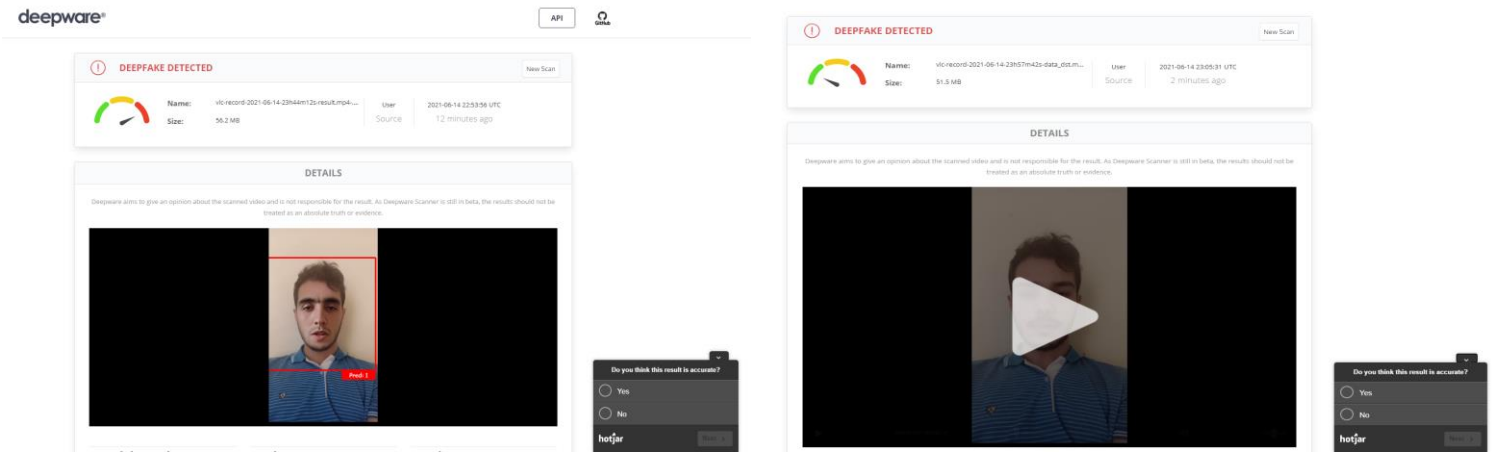

Figure 4.18 : *Résultat de l'outil web deepware sur 2 vidéos*

Comme vous pouvez le constater la précision de l'outil reste plus au moins relative car elle a même détecté la vidéo originale en étant une deepfake, voici le résultat donné :

```
In [34]: images = np.vstack(images)
         classes = model.predict_classes(images, batch_size=10)
         labels = ['not fake', 'fake']
         print([labels[i] for i in classes])

['not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not
fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
```

Figure 4.19 : *Résultat de la detection de notre modèle de detection sur notre deepfake*

Le résultat est un tableau de tout les échantillons de test que le modèle à traiter trier selon la classe à qui il appartienne, pour avoir une idée sur le taux de detection de la deepfake on calcule la moyenne d'erreur de detection, le taux d'erreur detecter ici est 0.

```
In [48]: print(print(sum(classes)/len(classes)))

0.0
None
```

Figure 4.20 : *Taux de détection de la présence d'un deepfake*

Nous avons aussi testé notre modèle sur la deepfake crée par Reface et on a eu ces résultats :

```
In [41]: images = np.vstack(images)
         classes = model.predict_classes(images, batch_size=10)
         labels = ['not fake', 'fake']
         print([labels[i] for i in classes])

['not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not
fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not f
ake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake',
'not fake', 'not fake', 'fake', 'not fake', 'fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'n
ot fake', 'fake', 'not fake', 'not fake', 'fake', 'not fake', 'not fake', 'fake', 'not fake', 'not fake', 'not fak
e', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'n
ot fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fak
e', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'fake', 'fake', 'not fake', 'not fake', 'not fak
e', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'n
ot fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fak
e', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'n
ot fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fak
e', 'not fake', 'not fake', 'not fake', 'fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'fak
e', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'not fake', 'n
ot fake', 'not fake', 'not fake', 'fake', 'not fake']
```

Figure 4.21 : *Résultat de la détéction sur le deepfake de Reface*

Quant au taux de présence du deepfake elle est comme suit :

```
In [47]: print(sum(lbl)/len(lbl))

0.05319148936170213
```

Figure 4.22 : *Pourcentage de présence du deepfake*

Veuillez noter que comme la précision du modèle est de 100%, une présence de plus de 0.01 est automatiquement une deepfake.

7. Conclusion

Dans ce chapitre final on a présenté notre méthode sur deux aspects opposés mais qui reste relier, un peu comme les deux faces d'une pièce, le premier modèle concernant la détection avait un taux d'entrainement moyen mais produit des résultats indétectables à l'œil nue, quant au modèle de détection, les résultats reste relative car les deepfakes ne cesse d'évoluer mais est doté d'une efficacité pratique vu qu'il n'a pas vraiment besoin de machine puissante pour se compiler et s'entrainer.

Conclusion générale et Perspectives

Conclusion générale et perspectives

L'apparition des deepfake a été une grande révélation de l'extension des choses que l'intelligence artificiel peut achever, le fait de pouvoir truquer le visage des gens à faire des choses qu'ils n'ont pas fait reste fascinant mais en même temps dangereux, en effet grâce à cette technologie on pourrait facilement taché la réputation des célébrités, les politiciens et même les gens normaux sans passé par l'apprentissage de logiciel complexe tel que adobe Photoshop, notre modèle de création est un bon exemple d'un deepfake indétectable à l'œil nue avec une architecture GAN utilisant des auto-encodeurs à base d'architecture CNN pour la partie génératrice et un modèle discriminateur entrainer par des milliers d'exemple et reste très efficace avec un temps d'entrainement très négligeable par rapport aux autres logiciels commerciaux.

Heureusement plusieurs entreprises informatiques et boites de programmation ont commencé à développer des logiciels de détection des deepfakes, une grande majorité a été éliminée du web, le modèle que nous proposons offre de bons résultats avec un coût réduit, par conséquent un traitement rapide d'une grosse base de données de deepfake, ce qui se révèle important pour un triage initial lors d'une purification à grand échelle.

Cependant jusqu'aujourd'hui aucun logiciel de détection s'est avéré efficace à plus de 80%, le système GAN est un système très difficile à battre car il laisse la machine à se surpasser avec chaque itération et chaque entrainement est malheureusement le travail que nous avons proposé reste de même, c'est pour ça que nous envisageons pour le travail future d'améliorer le modèle de détection pour prendre en compte plusieurs autres paramètres de détection (tel que la signature vidéo, les formats d'encodage d'enregistrement etc…) afin de restreindre encore plus la tricherie posée par les deepfakes et limiter leur utilisation à des fins de divertissement.

Bibliographie

[1] Yann Le Cun (16 octobre 2019) dans Quand la machine apprend: La révolution des neurones artificiels et de l'apprentissage profond (Français) Broché

[2] Esling, P., & Devis, N. (2020). Creativity in the era of artificial intelligence.

[3] https://fr.wikipedia.org/wiki/Intelligence_artificielle

[4] Arend Hintze (November 14, 2016) in Understanding the four types of AI, from reactive robots to self-aware beings.

[5] Russell, Stuart and Peter Norvig (2009). Artificial intelligence: A modern approach. 3rd edition. Prentice Hall

[6] https://www.24pm.com/117-definitions/423-systeme-multi-agents

[7] Negrello, L. (1991). Systèmes experts et intelligence artificielle. Schneider Electric España SA.

[8] Marc Parizeau (2004). Réseaux de neurones GIF-21140 et GIF-64326 LAVAL UNIVERSITY

[9] James D.Miller IBM (29 septembre 2018) in Watson Projects: Eight exciting projects that put artificial intelligence into practice for optimal business performance Broché.

[10] Jessica Burton dans La surveillance et l'intelligence artificielle et leur renforcement de la sécurité des banques par Chronique.

[11] Priyadharshini. (March 8, 2018). Machine Learning: What it is and Why it Matters, "simplilearn,".

[12] Frédéric SUR (2020-2021) dans Introduction à l'apprentissage automatique,Tronc commun scientifique FICM 2A, École des Mines de Nancy.

[13] CALEB GARLING dans une Interview de la magazine WIRED de Andrew NG.

[14]https://www.stemmer-imaging.com/fr-ch/conseil-techniqu/apprentissage-automatique-et-apprentissage-profond/

[15] https://www.intellipaat.com

[16] Ludovic L (22 December 2016) in Machine learning et deep learning, comment ça marche?

[17] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code and recognition. Neural computation.

[18] LeCun, Y., Haffner, P., Bottou, L., & Bengio, Y. (1999). Object recognition with gradient-based learning. In Shape, contour and grouping in computer vision (pp. 319-345). Springer, Berlin, Heidelberg.

[19] Valentin. B. (2015) approaches « deep learning» appliqué aux signaux audio : paroles et musique (Soutenance le 04/09/15).

[20] Rohrer, B. (2016). How do Convolutional Neural Networks work?. End-to-End Machine Learning, 18.

[21]https://colab.research.google.com/drive/1ZZXnCjFEOkp_KdNcNabd14yok0BAIuwS#forceEdit=true&sandboxMode=true.

[22] Ian Goodfellow, Yoshua Bengio, Aaron Courville (2016) in Deep Learning, MIT press.

[23] https://www.lemagit.fr/definition/Reseau-antagoniste-generatif-GAN

[24] https://www.lebigdata.fr/deepfake-tout-savoir

[25] Jason Brownlee (June 17, 2019) in : A Gentle Introduction to Generative Adversarial Networks (GANs).

[26] Akhil Santha (2019/2020) in Deepfake Generation using LSTM based Generative Adversarial Networks.

[27] Van Rossum, G. (1990). The python language. See http://www. python. org.

[28] Chollet, F. (2017). Deep learning with python. Manning Publications Co.

[29] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Ghemawat, S. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467.

[30] Tutorialspoint (20-Oct-2015). Jupyter Tutorial. Simply Easy Learning. https://www.tutorialspoint.com/