Apprentissage par réseaux de neurones profonds
GLO-7030
Session hiver 2020

Rapport final de projet

Destinataire

Département d'informatique et de génie logiciel
M. Philippe Giguère

Date de remise : 12 Mai 2020

Nantenaina Rabemiafara Mamiharimalala 111 251 821
AKOBI Israël 111 252 705

Détection des deepfakes : contenus trompeurs rendus profondément crédibles

Nantenaina Rabemiafara et Israël Akobi

Département d'informatique et de génie logiciel
Université Laval, Québec, Qc, Canada

***Abstract*— L'apprentissage profond a été appliqué avec succès pour résoudre divers problèmes complexes allant de l'analyse des mégadonnées à la vision par ordinateur. Cependant, des avancées en matière d'apprentissage profond ont également été utilisées pour créer des logiciels susceptibles de menacer la vie privée et la sécurité de tout en chacun. L'une de ces applications propulsées par l'apprentissage profond a récemment vu le jour : le «deepfake». Les algorithmes deepfakes peuvent créer de fausses images et vidéos que nous les humains ne pouvons pas distinguer à l'oeil nu des images authentiques. La proposition de technologies capables de détecter et d'évaluer automatiquement l'intégrité des supports visuels numériques est donc devenu indispensable. Dans ce présent travail, nous abordons le problème de la détection des manipulations de visage dans des séquences vidéos. Nous présentons un aperçu des algorithmes utilisés pour créer des deepfakes et des méthodes proposées pour détecter les deepfakes dans la littérature à ce jour. Et nous proposons notre approche pour résoudre ce problème en utilisant des architectures récentes en apprentissage profond, plus spécifiquement en réseau de neurones profond à convolution.**

I. INTRODUCTION

Au cours des dernières années, d'énormes progrès ont été accomplis dans le domaine des techniques de montage vidéo automatique. En particulier, un grand intérêt a été montré pour les méthodes de manipulation faciale. Pour ne citer qu'un exemple, il est aujourd'hui possible de transférer facilement les expressions faciales d'une vidéo à une autre. Cela permet de changer l'identité d'un locuteur avec très peu d'effort.

Les systèmes et outils de manipulation faciale sont désormais si avancés que même les utilisateurs sans expérience préalable en retouche photo et en arts numériques peuvent les utiliser. En effet, le code et les bibliothèques qui fonctionnent de manière quasi automatique sont de plus en plus souvent mis gratuitement à la disposition du public. D'une part, cette avancée technologique ouvre la porte à de nouvelles possibilités artistiques (par exemple, la réalisation de films, les effets visuels, les arts visuels, etc.). D'autre part, malheureusement, cela facilite également la génération de contrefaçons vidéo par des utilisateurs malveillants.

Ainsi est apparu dans les médias le mot "deepfake", c'est une contraction entre "Deep Learning" et "Fake" que l'on pourrait traduire par "faux profond". En effet, il s'agit de contenus trompeurs, rendus "profondément" crédibles grâce à l'intelligence artificielle. Plus précisément au Deep Learning ou apprentissage profond.

La diffusion de fausses nouvelles et la vengeance ne sont que quelques-unes des applications malveillantes possibles de la technologie avancée de deepfake entre de mauvaises mains. Étant donné que la distribution de ces types de vidéos manipulées entraîne incontestablement des conséquences graves et dangereuses, la capacité de détecter si un visage a été manipulé dans une séquence vidéo est devenue d'une importance capitale.

Détecter si une vidéo a été modifiée n'est pas un problème nouveau en soi. Les chercheurs en criminalistique multimédia ou Multimedia forensics researchers travaillent sur ce sujet depuis de nombreuses années, proposant différents types de solutions à différents problèmes [1], [2], [3]. Le principe commun entre ces solutions est assez simple : chaque opération non réversible laisse une empreinte particulière qui peut être dévoilée pour détecter le changement spécifique. Cependant, les empreintes criminalistiques sont souvent très subtiles et difficiles à détecter. C'est le cas des vidéos subissant une compression excessive, plusieurs opérations de montage à la fois ou un fort sous-échantillonnage. C'est également le cas des contrefaçons très réalistes opérées par des méthodes difficiles à modéliser formellement. Pour cette raison, les techniques modernes de manipulation faciale sont très difficiles à détecter du point de vue criminalistique. En fait, il existe de nombreuses techniques différentes de manipulation du visage c'est-à-dire qu'il n'y a pas de modèle unique expliquant ces contrefaçons. De plus, ils ne fonctionnent souvent que sur de petites régions de la vidéo : le visage ou une partie de celui-ci, et non le plein cadre. Enfin, ces types de vidéos manipulées sont généralement partagées via les plateformes sociales (Facebook, Instagram, ...) qui appliquent des étapes de redimensionnement et de codage, ce qui entrave davantage les performances des détecteurs de deepfake.

Dans ce présent travail, nous abordons le problème de la détection de la manipulation faciale opérée par des solutions modernes. En particulier, nous nous concentrons sur toutes les techniques de manipulation rapportées dans [12]. Dans ce contexte, nous étudions la possibilité d'utiliser le modèle EfficientNet pour être en mesure de classifier les vidéos deepfakes.

L'évaluation est effectuée sur l'ensembles de données FaceForensics++ [12], publié par Google en collaboration avec Jigsaw : un grand ensemble de données de deepfakes visuels.

Fig. 1: Extraits de visages de l'ensemble de données FaceForensics++, rapportant des échantillons vierges et manipulés

La figure 1 illustre quelques exemples de visages extraits de l'ensembles de données, rapportant des échantillons vierges et manipulés.

Le reste du document est structuré comme suit. La section II présente une analyse documentaire des derniers travaux connexes. La section III présente tous les détails de la méthode proposée. La section IV détaille la configuration expérimentale. La section V rassemble tous les résultats obtenus. Enfin, la section VI conclut le document.

II. DESCRIPTION DU PROJET EN LIEN AVEC LA LITTERATURE

Nous allons dans un premier temps décrire quelques concepts clés dans la création de deepfake avant de faire une analyse documentaire des derniers travaux de détection de ce dernier.

A. Création de deepfake

Les deepfakes sont devenus populaires grâce à la qualité des vidéos trafiquées et à la facilité d'utilisation de leurs applications à un large éventail d'utilisateurs ayant des compétences informatiques variées, du professionnel au novice. Ces applications sont pour la plupart développées sur la base de techniques d'apprentissage profond. L'apprentissage profond est bien connu pour ses capacités de représenter des données complexes et de grande dimension. Une variante des réseaux profonds avec cette capacité est celle des autoencodeurs profonds, qui ont été largement appliqués pour la réduction de la dimension et la compression d'image [4], [5], [6]. La première tentative de création de deepfake a été FakeApp, développée par un utilisateur de Reddit utilisant la structure d'appariement autoencodeur-décodeur [7], [8]. Dans cette méthode, l'autoencodeur extrait les caractéristiques latentes des images de visages et le décodeur est utilisé pour reconstruire les images de visages. Pour échanger des visages entre les images sources et des images cibles, il faut deux paires d'encodeurs-décodeurs, chaque paire étant utilisée pour s'entraîner sur un ensemble d'images et les paramètres de l'encodeur sont partagés entre les deux paires de réseaux. Cette stratégie permet à l'encodeur commun de trouver et d'apprendre la similarité entre deux séries d'images de visages, qui sont relativement peu exigeantes car les visages ont normalement des caractéristiques similaires telles que les yeux, le nez, la position de la bouche.

Pour l'ensemble de données FaceForensics++, deux approches ont été utilisées pour générer du contenu Deepfake. La première basée sur les graphiques à savoir Face2Face et FaceSwap et deux autres basées sur l'apprentissage DeepFakes et NeuralTextures. Les quatre méthodes ont nécessité des paires de vidéo d'acteurs source et cible en entrée. Le résultat final de chaque méthode est une vidéo composée d'images générées.

- FaceSwap fait référence à la tâche d'échanger des visages entre des images ou dans une vidéo, tout en conservant le reste du corps et le contexte de l'environnement [7].
- Face2Face est un système de reconstitution faciale qui transfère les expressions d'une vidéo source vers une vidéo cible tout en conservant l'identité de la personne cible [11].
- DeepFakes, désigne ici la méthode de manipulation, pour ne pas confondre avec le terme générique deefake pour désigner les contenus manipulés, et fait référence à ce qu'on a expliqué plus haut, une méthode utilisant les encodeurs-décodeurs.
- NeuralTextures fait référence à l'article publié par Thies et al. dans [13] où ils montrent une reconstitution faciale comme exemple pour leur approche de rendu basée sur NeuralTextures.

B. Détection de deepfake

Cette sous-section se concentre sur les méthodes de détection des vidéos deepfakes et les classe en deux groupes : les méthodes qui utilisent des caractéristiques temporelles (temporal features) et celles qui explorent les artefacts visuels dans les images par CNN.

1) Méthode basée sur les caractéristiques temporelles: Partant du constat que la cohérence temporelle n'est pas appliquée efficacement dans le processus de synthèse des deepfakes : Sabir et al. [14] ont exploité l'utilisation des caractéristiques spatio-temporelles des flux vidéo pour détecter les deepfakes (Fig. 2-a). La manipulation vidéo est effectuée image par image, de sorte que les artefacts de bas niveau produits par les manipulations de visages sont censés se manifester davantage sous la forme d'artefacts temporels avec

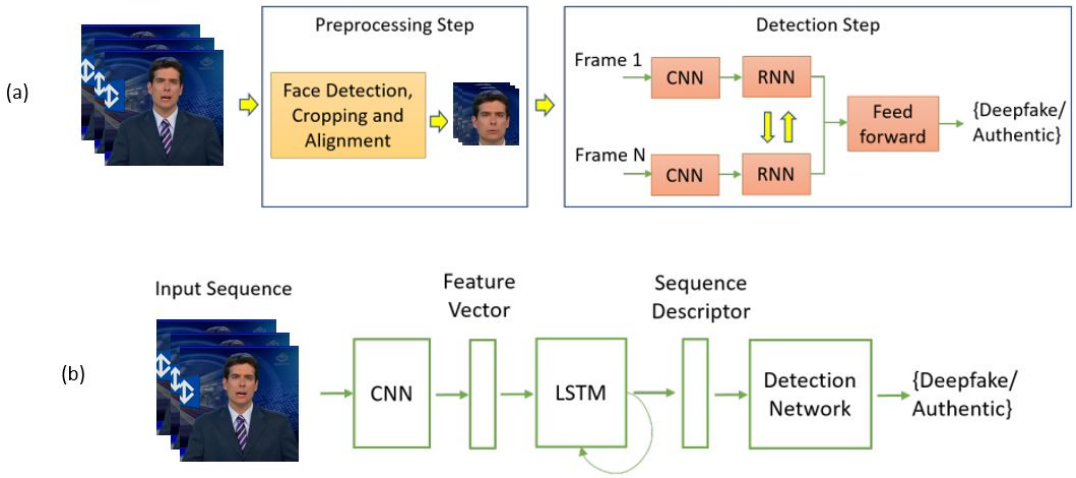

Fig. 2: (a) Un processus en deux étapes pour la détection de la manipulation des visages où l'étape de prétraitement vise à détecter, recadrer et aligner les visages sur une séquence d'images et la deuxième étape distingue les images de visages deepfakes et authentiques en combinant le réseau neuronal convolutif (CNN) et le réseau neuronal récurrent (RNN) [14]. (b) Une méthode de détection basée sur le CNN et la LSTM (Long Short Term Memory) pour extraire les caractéristiques temporelles d'une séquence vidéo donnée, qui sont représentées par le descripteur de séquence. Le réseau de détection constitué de couches entièrement connectées est utilisé pour prendre le descripteur de séquence comme entrée et calculer les probabilités de la séquence d'images appartenant à la classe authentique ou à la classe deepfake [15].

des incohérences entre les images. Un modèle convolutionnel récurrent (Recurrent Convolutionnal Network) a été proposé, basé sur l'intégration du modèle DenseNet et des cellules unitaires de GRU (Gated Recurrent Unit), afin d'exploiter les divergences temporelles entre les images.

De même, Guera et Delp [15] ont souligné que les vidéos truquées contiennent des incohérences intra-image et des incohérences temporelles entre les images. Ils ont ensuite proposé la méthode du pipeline temporel qui utilise le CNN et la LSTM pour détecter les vidéos truquées (Fig. 2-b). Le CNN est utilisé pour extraire les caractéristiques au niveau des images, qui sont ensuite introduites dans la LSTM pour créer un descripteur de séquence temporelle. Un réseau entièrement connecté est ensuite utilisé pour classer les vidéos par rapport aux vidéos réelles sur la base du descripteur de séquence.

D'autre part, l'utilisation d'un signal physiologique, le clignement des yeux, pour détecter les deepfakes a été proposée dans [16] sur la base de l'observation qu'une personne en deepfake clignote beaucoup moins fréquemment que celle qui se trouve dans des vidéos non trafiquées.

2) Méthode basée sur les features apprises par CNN: Les vidéos deepfakes sont normalement créées avec des résolutions limitées, ce qui nécessite une approche affine de déformation du visage (c'est-à-dire mise à l'échelle, rotation et cisaillement) pour correspondre à la configuration des vidéos originales. En raison de l'incohérence de la résolution entre la zone du visage déformé et le contexte environnant, ce processus laisse des artefacts qui peuvent être détectés par les modèles CNN. Dans [17], ils proposent une méthode d'apprentissage profond pour détecter les deepfakes basée sur les artefacts observés pendant l'étape de déformation du visage des algorithmes de génération de deepfake. La méthode proposée est évaluée sur deux ensembles de données de deepfake, à savoir l'UADFV [18] et DeepfakeTIMIT [19]. Pour cela, ils ont entrainé quatre modèles CNN : VGG16, ResNet50, ResNet101 and ResNet152.

MesoInception-4 [29] est un réseau CNN inspiré d'InceptionNet qui permet de détecter la manipulation du visage dans les vidéos. Le réseau comporte deux modules d'inception et deux couches de convolution classiques intercalées avec des couches de max-pooling. Ensuite, il y a deux couches entièrement connectées. Au lieu de la perte d'entropie croisée (Cross Entropy Loss) classique, les auteurs proposent l'erreur quadratique moyenne.

Récemment, Nguyen et al [20] ont proposé l'utilisation de réseaux de capsules (capsule networks) pour détecter les images manipulées et les vidéos. La méthode proposée donne les meilleurs résultats par rapport à ses méthodes concurrentes. Cela montre le potentiel du réseau de capsules dans la construction d'un système de détection général qui peut fonctionner efficacement pour diverses attaques de fausses images et vidéos.

Afin de surmonter la limitation de l'analyse des pixels, d'autres techniques sont basées sur une analyse sémantique des trames. Dans [9], une technique qui apprend à distinguer la pose naturelle de la tête. D'autre, dans [21] rapporte une méthodologie basée sur l'analyse des yeux clignotants. En effet, la première génération de vidéos deepfakes montrait des artefacts oculaires qui pouvaient être capturés avec cette méthode. Malheureusement, plus les techniques de manipulation produisent des résultats réalistes, moins les

méthodes sémantiques fonctionnent.

Inspiré par l'état de l'art et notamment par les travaux réalisés dans [12], dans ce présent travail, nous nous proposons d'attaquer au problème de détection des vidéos deepfakes en utilisant une approche d'analyse basée sur les features apprises par CNN avec le modèle EficientNet.

III. Approche proposée

Dans cette section, nous décrivons la méthode que nous proposons pour la détection de vidéos deepfakes, plus spécifiquement la détection de la manipulation des visages, c'est-à-dire, à partir d'une image vidéo, détecter si les visages sont réels ou faux.

Pour ce faire, nous considérons comme point de départ la famille de modèles EfficientNet, proposée dans [22]. Cet ensemble d'architectures permet d'obtenir une meilleure précision et efficacité par rapport à d'autres CNN de pointe et est considéré comme l'état de l'art actuel. Dans ce qui suit, nous donnons plus de détails sur l'architecture EfficientNet.

A. *EfficientNet*

Parmi la famille des modèles EfficientNet, nous choisissons EfficientNetB4 comme base de référence pour notre travail, motivé par le bon compromis offert par cette architecture en termes de nombre de paramètres, de durée d'exécution et la performance de classification. Comme indiqué dans [22], avec 19 millions de paramètres et 4,2 milliards de FLOPS, EfficientNetB4 atteint un top-1 accuracy de 83% sur ImageNet. Sur le même ensemble de données, XceptionNet, utilisé comme modèle pour la détection de vidéos deefakes par les auteurs de [12], atteint un top-1 accuracy de 79% au détriment de 23 millions de paramètres et 8,4 milliards de FLOPS. L'architecture EfficientNetB4 est représentée dans la Fig. 3, où toutes les couches sont définies en utilisant le même nomenclature introduite dans [22].

L'entrée dans le réseau est une image couleur carrée, c'est-à-dire, le visage extrait de nos vidéos. En effet, les auteurs de [12] recommandent de travailler sur le visage au lieu d'utiliser le cadre complet de l'image comme entrée pour augmenter la précision de la classification. En plus, les visages peuvent être facilement extraits des cadres à l'aide de l'un des des détecteurs de visage largement disponibles proposés dans la littérature [23], [24].

Les variantes qu'on propose pour l'architecture standard EfficientNetB4 s'inspirent de plusieurs contributions dans les domaines du traitement du langage naturel et de la vision par ordinateur qui utilisent des mécanismes d'attention. Des travaux tels que le Transformer [25] montrent comment il est possible pour un réseau de neurones d'apprendre quelle partie de son entrée (une image ou une séquence de mots) est la plus pertinente pour accomplir la tâche à accomplir. Dans le contexte de la détection de vidéos deepfakes, il serait très utile de découvrir quelle partie de l'entrée donne au réseau plus d'informations pour son processus de prise de décision.

Nous mettons donc explicitement en œuvre un mécanisme d'attention similaire à celui déjà exploité par EfficientNet lui-même, ainsi qu'aux mécanismes d'auto-attention présentés dans [26], [27].

La première variante consiste à :

— Sélectionner les feature maps extraites par EfficientNetB4 jusqu'à une certaine couche, choisie de telle sorte que ces caractéristiques fournissent suffisamment d'informations sur la trame d'entrée sans être trop détaillées ou, au contraire, trop peu explicites. À cette fin, nous sélectionnons les features de sortie au niveau du troisième bloc MBConv qui ont une taille de 28x28x56 ;

— Traiter les features map avec une seule couche convolutive avec un noyau de taille 1 suivie d'une fonction d'activation Sigmoïd pour obtenir une seule carte d'attention ;

— Multiplier la carte d'attention pour chacune des feature maps à la couche sélectionnée.

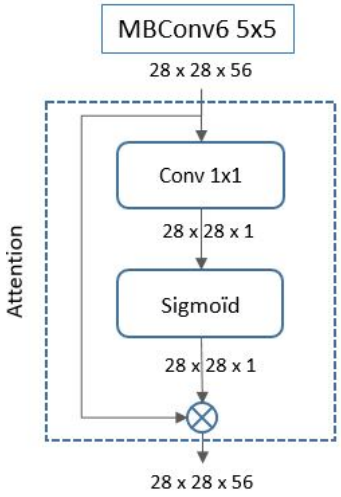

Fig. 4: Simple mécanisme d'attention

D'une part, ce mécanisme simple permet au réseau de se concentrer uniquement sur les parties les plus pertinentes des feature maps, d'autre part, il nous donne un aperçu plus précis des parties de l'entrée que le réseau considère comme les plus informatives. En effet, la carte d'attention obtenue peut être facilement mise en correspondance avec l'échantillon d'entrée, en mettant en évidence les éléments de celui-ci auxquels le réseau a accordé le plus d'importance. Le résultat du bloc d'attention est finalement traité par les couches restantes de EfficientNetB4.

La seconde variante que nous avons exploité consiste à intégrer dans le réseau le "Spatial Transformer Network (STN)" introduit dans [30]. Le but du STN est d'améliorer l'invariance spatiale (la translation, la rotation, l'échelle, etc.) en ajoutant un module ajustable (learnable) qui manipule explicitement les données dans le domaine spatial. Il est pleinement différentiable et peut être insérée partout dans une architecture préexistante, dans notre cas après le MBConvBloc 20.

Comme montré dans la Fig. 5, le mécanisme du STN est divisé en trois parties :

— le réseau de localisation (localisation net) prend

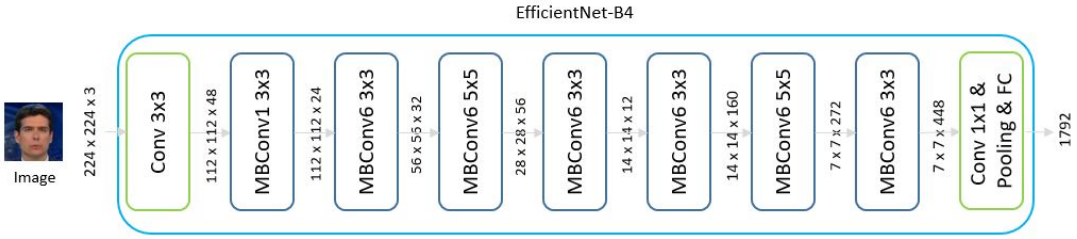

Fig. 3: Modèle EfficienNet-B4

d'abord la feature map en entrée, et à travers un CNN ou fully-connected avec couche finale de régression, puis produit les paramètres de la transformation spatiale qui doivent être appliqués à la feature map. Cela donne une transformation θ conditionnelle à l'entrée.

— Ensuite, les paramètres de transformation prédits sont utilisés pour créer une grille d'échantillonnage à travers le grid generator.

— Et enfin, la feature map et la grille d'échantillonnage sont pris comme entrées de l'échantilloneur pour finalement produire la sortie.

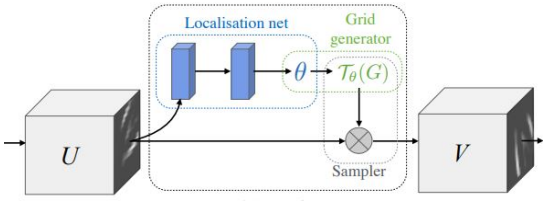

Fig. 5: Spatial Tansformer

IV. Protocole d'expérimentation

Dans cette section, nous présentons tous les détails concernant l'ensemble de données utilisé et le dispositif expérimental.

A. Ensemble de données

1) Description: Nous testons la méthode proposée sur l'ensemble de données FaceForensics++ [12] rendu disponible par Google AI en 2019.

FaceForensics++ est un ensemble de données de manipulation faciale à grande échelle généré à l'aide des technologies de deepfake comme vu dans la section II-A. Chaque méthode de deepfake est appliquée à 1000 vidéos de haute qualité téléchargées sur YouTube, sélectionnées manuellement pour présenter des sujets presque en face sans occlusion. Toutes les séquences contiennent au moins 280 images. Au final, une base de données de plus de 1,8 million d'images provenant de 4000 vidéos manipulées est constituée. Les vidéos sont compressées à l'aide du codec H.264 et sont disponibles en haute et basse qualité en utilisant un paramètre de compression égal à 23 ou 40.

Nous prenons comme base de référence de nos expérimentations, les résultats obtenus avec FaceForensics++ dans [12] où l'architecture XceptionNet présente les meilleurs résultats dans la détection de vidéos deepfakes dans ce jeu de données.

2) Enjeux de stockage: Les données vidéos de FaceForencis++ ammènent d'importants enjeux de stockage. Etant donné que nous avons travaillé sur la plateforme Google Colab et sur Google Drive pour les données, nous avons donc du augmenté la capacité de stockage de Google Drive pour accueillir les vidéos et également les visages à extraire de chaque video.

Nous n'avons pas de ce fait expérimenter tous les taux de compression différents proposés par le jeu de données, comme c'est le cas dans [12]. Il serait cependant simple d'appliquer différents taux de compression aux vidéos originales et de refaire les expériences avec notre méthodologie.

B. Configurations

Dans nos expériences, nous considérons le modèlele EfficientNetB4 comme base de référence et nous y avons effectués quatre (4) types de configuration (Section V pour les détails). Les modèles sont entrainés et testés avec le jeu de données FaceForensics++.

Nous utilisons une répartition similaire à celle dans [12] en sélectionnant 720 vidéos pour l'entrainement, 140 pour la validation et 140 pour le test à partir des séquences de vidéos originales tirées de YouTube. Les fausses vidéos correspondantes sont affectées à la même répartition.

Nous appliquons la même stratégie de prétraitement des données pour l'ensemble de données :

— Extraire les images de chaque vidéo ;

— Utiliser une technologie fiable de suivi des visages pour détecter les visages dans chaque image et recadrer l'image autour du visage.

Lors de la détection de vidéos créées avec une manipulation du visage, il est possible d'entraîner les détecteurs sur l'ensemble des images des vidéos, ou simplement de recadrer la zone entourant le visage, et d'appliquer le détecteur exclusivement sur cette zone recadrée. Nous avons appliqué cette dernière.

Nous avons utilisé une série de techniques d'augmentation des données différentes mais simples. En particulier, nous appliquons de manière aléatoire la réduction d'échelle, le retournement horizontal, le contraste de luminosité aléatoire, la saturation des teintes, le bruit et enfin la compression JPEG. Il convient de noter ici que l'augmentation des données n'est appliquée qu'aux images d'entrainement.

Nous avons avons entraîné les modèles en utilisant l'optimiseur Adam avec comme hyperparamètres : $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, un weight decay de 10^{-6} et un taux d'apprentissage initial de 10^{-4}.

Avec les moyens de calcul que nous utilisons (Google Colab), une seule epoch peut prendre 1h et puisque nous avons testé plusieurs configurations, nous avons donc décidé de réduire le nombre d'epoch pour l'entrainement de façon à ne pas nous ralentir dans nos expérimentations.

V. Résultats

Dans cette section, nous rassemblons tous les résultats obtenus au cours de nos expérimentations.

A. Courbe d'apprentissage

Nous présentons dans la Fig. 6 les courbes d'apprentissage de nos différentes expérimentations selon les quatre (4) configurations suivantes :

— Configuration 1 : En utilisant le modèle pré-entraîné EfficicentNetB4 et en freezant tous les paramètres de convolution.
— Configuration 2 : En utilisant le modèle pré-entraîné EfficicentNetB4 et en réentrainant les 8 derniers MBConvBloc avec les deux dernières couches (la couche de convolution et le classificateur).
— Configuration 3 : En utilisant le modèle pré-entraîné EfficicentNetB4 et en réentrainant les 16 derniers MBConvBloc avec les deux dernières couches (la couche de convolution et le classificateur).
— Configuration 4 : En utilisant le modèle pré-entraîné EfficicentNetB4 et en intégrant un Spatial Transformer Network après le MBConvBloc 20.

B. Performance

Nous voyons ci-dessous un tableau comparatif de la performance de nos résultats sur les différentes configurations.

Modèle	Acc(train)	Acc(valid)	Acc(test)
Configuration 1	83.64	83.64	83.44
Configuration 2	89.69	87.84	86.94
Configuration 3	92.99	90.34	90.08
Configuration 4	92.05	89.67	91.10

TABLE I: Tableau comparatif des résultats

C. Mécanisme d'attention

Notons que les résultats de la variante du modèle EfficicentNetB4 dans le lequel nous avons intégré un simple mécanisme d'attention (Fig. 4) ne sont pas présentés ici, ce modèle était tout simplement une façon pour nous de mieux comprendre le mécanisme d'attention.

Nous avons ainsi analysé la carte d'attention calculée sur les visages de notre ensemble de données. Nous sélectionnons la sortie de la couche Sigmoïd dans le bloc d'attention (Fig. 4), qui est une carte en 2D de taille 28 x 28. Ensuite, nous l'agrandissons à la taille de la face d'entrée (224 x 224), et superposer cela à la face d'entrée.

Il convient de noter à partir de la Fig. 8 que ce simple mécanisme d'attention a tendance à selectionner la partie la plus importante du visage, c'est-à-dire les yeux, la bouche, le nez et les oreilles. Au contraire, les régions plates (où les gradients sont faibles) ne sont pas informatifs pour le réseau.

En effet, les artefacts des méthodes de génération de deepfake sont principalement localisés autour des traits du visage. Ces résultats concordent bien à la réalité.

Fig. 8: Mécanisme d'attention

VI. Discussion des résultats

Dans cette avant-dernière partie, nous discutons des résultats présentés précédement, des problèmes rencontrés au cours du projet et les solutions qu'on a adoptées pour les résoudre, les améliorations que nous projetons de faire et enfin quelques perspectives de recherche sur le domaine que nous avons noté lors de l'élaboration de ce travail.

A. Interprétation des résultats

Notons d'abord les grandes lignes de différence entre nos différentes configurations de modèles :

— En utilisant le modèle pré-entraîné, mais en freezant tous les paramètres de convolution (configuration 1), nous veillons à ce que toutes les caractéristiques robustes apprises précédemment par le modèle EfficicentNetB4 ne soient pas détruites.
— En utilisant le modèle pré-entraîné, mais en freezant les paramètres de convolution dans certains blocs (configuration 2 et 3) : en plus du classifieur, nous entraînons sur les nouvelles images les couches non-fixées, qui correspondent ici aux plus hautes du réseau. Puisque nos nouvelles images n'ont pas beaucoup de points communs avec les anciennes : utiliser les features du réseau pré-entraîné pour les représenter n'est pas une très bonne idée. Les features des couches basses sont simples et génériques, tandis que celles des couches hautes sont complexes et spécifiques à notre problème. Nous faisons cela pour décourager le modèle de simplement mémoriser les différences entre les vrais et les faux visages, et de l'encourager à rechercher des artefacts utiles à la détection des faux visages.

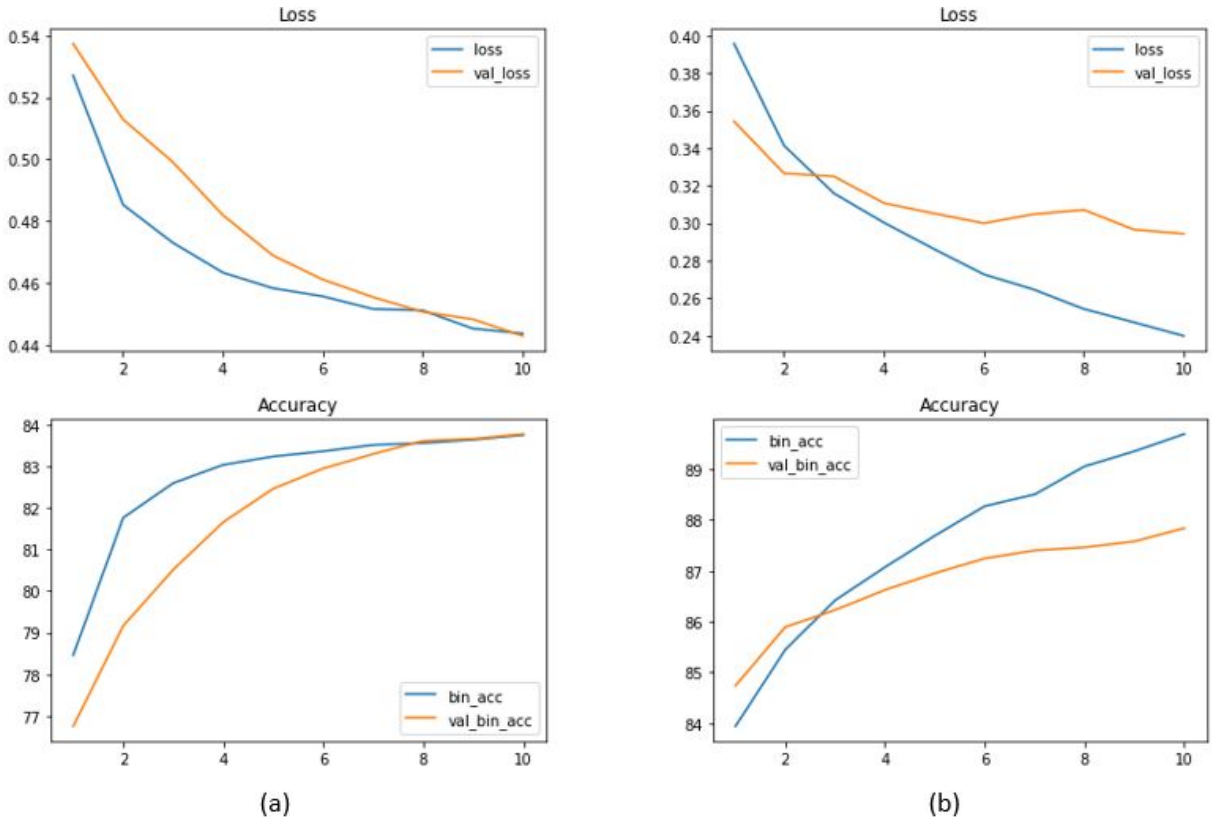

Fig. 6: Historique d'entrainement des configurations 1 (a) et 2 (b)

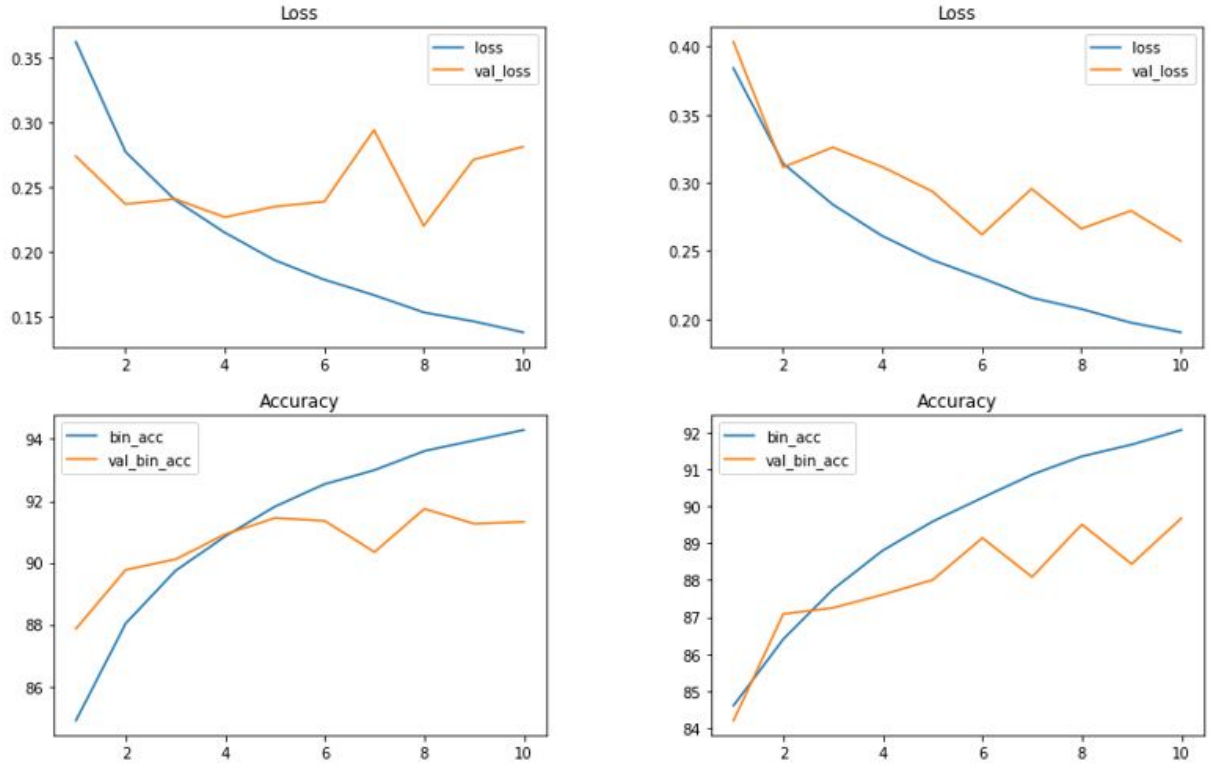

Fig. 7: Historique d'entrainement des configurations 3 (a) et 4 (b)

— Le dernier modèle où nous intégrons un Spatial Transformer apès le MBConvBloc 20 : le STN applique une transformation spatiale sur les features à son entrée lors de la propagation en avant et est pleinement différentiable (i.e compatible avec entraînement end-to-end via backprop).

Comme attendu, nos modèles performent bien dans l'ensemble et nous observons des gains de précision en utilisant le Spatial Transformer Network dans la dernière configuration. Ceci dit, en entraînant nos modèles sur un nombre d'époch plus conséquent et en variant en conséquence nos hyperparamètres, nous serons en mesure d'obtenir de meilleures performances.

B. Problèmes rencontrés et réglés

Un des premiers problèmes que nous avons rencontré était le stockage du jeu de données pour y accéder depuis Google Colab. Puisque Google Colab pouvait se relier avec Google Drive, l'option de stockage était claire mais la version gratuite n'était pas suffisante pour contenir l'ensemble de données.

Puis le prétraitement du jeu de données représentait aussi un aspect assez long de notre travail. Même si les étapes qu'on a mentionnées précédent semblent à première vue simple, les détails nous a quand même donnés du fil à retordre. Notre objectif était de rendre le code assez générique pour nous permettre de relever par la suite d'autres défis entourant la détection des vidéos deepfake notamment le Deepfake Challenge lancé par Facebook l'année dernière. Pour se faire, pour la technologie de détection de visages, nous avons eu l'occasion de tester trois d'entre eux, celle disponible avec openCV, le détecteur MTCNN (Multi-task Cascaded Convolutional Neural Networks) [23] et le détecteur Blazeface [24], dans lesquels des implémentations open source existent. Nous avons retenu des trois, Blazeface.

Lors de nos premières expérimentations, notre modèle était sujet à du suraprentissage. Il collait parfaitement au jeu de données d'entrainement et avait vraiment du mal à généraliser. En effet, dans l'extraction des images dans les vidéos, nous ne considérons qu'un nombre limité d'images à extraire par vidéo, i) lorsqu'on a utilisé une quantité vraiment faible d'images par vidéo à hauteur de 5 images par vidéo, le modèle avait une forte tendance au suraprentissage ; (ii) augmenter le nombre d'images n'améliore pas les performances de manière justifiable mais nous permet d'éviter ce problème. En choisissant un nombre d'images à extraire au juste millieu nous avons pu résoudre ce problème, le choix s'est donc porté à 15 images par vidéo.

C. Améliorations

Une des améliorations que nous projétons de réaliser dans ce travail est l'utilisation des méthodes d'ensemble. Une méthode d'ensemble est un ensemble de modèles dont les prédictions sont combinées d'une certaine manière afin d'obtenir une prédiction généralement plus stable. Il est bien connu que la méthode d'ensemble peut conduire à de meilleures performances de prédiction. Il s'agit ici d'étudier la possibilité de comment il est possible de former différents classificateurs basés sur le CNN pour saisir différentes informations sémantiques de haut niveau qui se complètent les unes les autres, donc contribuant positivement à l'ensemble pour ce problème spécifique.

De même, les auteurs dans [15] ont proposé la méthode du pipeline temporel qui utilise le CNN et la LSTM pour détecter les vidéos truquées. Il serait intéressant de combiner notre approche actuelle pour extraire les caractéristiques au niveau des images, pour ensuite les introduire dans la LSTM pour créer le descripteur de séquence temporelle.

D. Discussions et perspectives de recherche

La qualité des deepfakes a augmenté et les performances des méthodes de détection doivent être améliorées en conséquence. L'idée est que ce qui a été cassé par l'IA peut être réparé par l'IA également [27]. Une approche visant à améliorer les performances des méthodes de détection voit peu à peu le jour, qui consiste à créer un ensemble de données de référence de plus en plus actualisé de deepfakes pour valider le développement continu des méthodes de détection, comme pour le cas actuellement du Deepfake Challenge Dastaset composé de plus de 119 000 séquences vidéos, créées spécifiquement pour ce défi, représentant à la fois des vidéos réelles et des fausses vidéos.. Cela facilitera le processus de formation des modèles de détection, en particulier ceux basés sur l'apprentissage profond, qui nécessite un vaste ensemble de données d'entrainement. D'autre part, nous avons remarqué les méthodes de détection actuelles se concentrent principalement sur les inconvénients des pipelines de génération de deepfake, c'est-à-dire sur la détection des faiblesses des concurrents pour les attaquer. Ce type d'informations et de connaissances n'est pas toujours disponible dans les environnements conflictuels où les attaquants tentent généralement de ne pas révéler ces technologies de création de deepfake. Il s'agit d'un véritable défi pour le développement de méthodes de détection et ainsi les futurs recherches doivent se concentrer sur l'introduction de méthodes plus robustes, plus évolutives et plus généralisables.

Un axe de recherche, qui nous intéresse réellement et qui nous a poussé à faire ce travail, consiste à intégrer les méthodes de détection dans les plateformes de médias sociaux afin d'accroître leur efficacité pour faire face à l'impact généralisé des deepfakes. Cette approche peut également être combiné à des outils de tatouage numérique afin de créer des métadonnées immuables pour stocker des détails d'originalité tels que l'heure et l'emplacement des contenus multimédia ainsi que leur attestation non altérée. Ce qui nous ammène à l'utilisation éventuelle de la technologie de blockhain. Cette technologie a été utilisée efficacement dans de nombreux domaines mais d'après nos recherches, il n'existe que très peu d'études sur les problèmes de détection des deepfakes basés sur cette technologie.

VII. Conclusion

Etant donné l'impact significatif des vidéos dans notre vie quotidienne et dans les communications de masse, il est d'une importance capitale de détecter si une vidéo contient du contenu manipulé ou trompeur. Dans cette optique, dans ce travail, nous nous sommes attaqués à la détection de manipulations faciales dans les séquences de vidéo communément appelé les deepfakes. La méthode proposée est basée sur un modèle d'apprentissage profond, le modèle EfficientNet. L'idée derrière est que "ce qui a été cassé par l'IA peut être également réparé par l'IA". Les résultats évalués sur l'ensembles de données FaceForensics++ [12] révèlent que les modèles qu'on a proposés sont des solutions valables pour l'objectif de détection des manipulations faciales. Des améliorations sont encore bien évidemment à prévoire, nous projetons en effet de rendre nos codes disponibles en ligne pour permettre à d'autres étudiants intéressés par le domaine de prendre en main assez rapidement le jeu de données [12] et de tester par la suite sur différents ou combinaison de modèles.

References

[1] A. Rocha, W. Scheirer, T. Boult, and S. Goldenstein, *Vision of the unseen : Current trends and challenges in digital image and video forensics*, ACM Computing Surveys, vol. 43, no. 26, pp. 1–42, 2011.

[2] S. Milani, M. Fontani, P. Bestagini, M. Barni, A. Piva, M. Tagliasacchi, and S. Tubaro, *An overview on video forensics," APSIPA Transactions on Signal and Information Processing*, vol. 1, p. e2, 2012.

[3] M. C. Stamm, Min Wu, and K. J. R. Liu, *Information forensics : An overview of the first decade*, IEEE Access, vol. 1, pp. 167–200, 2013.

[4] Punnappurath, A., and Brown, M. S. (2019). *Learning raw image reconstruction-aware deep image compressors. IEEE Transactions on Pattern Analysis and Machine Intelligence*, DOI : 10.1109/TPAMI.2019.2903062.

[5] Cheng, Z., Sun, H., Takeuchi, M., and Katto, J. (2019). *Energy compaction-based image compression using convolutional autoencoder. IEEE Transactions on Multimedia*, DOI : 10.1109/TMM.2019.2938345

[6] Chorowski, J., Weiss, R. J., Bengio, S., and Oord, A. V. D. (2019). *Unsupervised speech representation learning using wavenet autoencoders*, IEEE/ACM Transactions on Audio, Speech, and Language Processing. 27(12), pp. 2041-2053.

[7] Faceswap : Deepfakes software for all. `https://github.com/deepfakes/faceswap`

[8] FakeApp 2.2.0, `https://www.malavida.com/en/soft/fakeapp/`

[9] DeepFaceLab, `https://github.com/iperov/DeepFaceLab`

[10] DFaker, `https://github.com/dfaker/df`

[11] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner. *Face2face : Real-time face capture and reenactment of rgb videos*, CVPR 2016.

[12] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, FaceForensics++ : *Learning to detect manipulated facial images, in International Conference on Computer Vision*, ICCV 2019.

[13] J. Thies, M. Zollhofer, and M. Nießner, Deferred neural rendering, *Image synthesis using neural textures, ACM Transactions on Graphics (TOG)*, vol. 38, no. 4.

[14] Sabir, E., Cheng, J., Jaiswal, A., AbdAlmageed, W., Masi, I., and Natarajan, P. (2019). *Recurrent convolutional strategies for face manipulation detection in videos*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops.

[15] Guera, D., and Delp, E. J. (2018, November). *Deepfake video detection using recurrent neural networks*. In 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) IEEE.

[16] Li, Y., Chang, M. C., and Lyu, S. (2018, December). *In ictu oculi : Exposing AI created fake videos by detecting eye blinking*. In 2018 IEEE International Workshop on Information Forensics and Security (WIFS).

[17] Li, Y., and Lyu, S. (2019). *Exposing deepfake videos by detecting face warping artifacts*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops.

[18] Yang, X., Li, Y., and Lyu, S., *Exposing deep fakes using inconsistent head poses*. In 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).

[19] Korshunov, P., and Marcel, S. *Speaker inconsistency detection in tampered video. In 2018 26th European Signal Processing* Conference (EUSIPCO) (pp. 2375-2379). IEEE.

[20] Nguyen, H. H., Yamagishi, J., and Echizen, I. *Capsule-forensics : Using capsule networks to detect forged images and videos*. In 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).

[21] DeepFake-tf : Deepfake based on tensorflow. Retrieved from `https://github.com/StromWine/DeepFaketf`

[22] M. Tan and Q. V. Le, "Efficientnet : *Rethinking model scaling for convolutional neural networks*, in International Conference on Machine Learning, (ICML) 2019, ser. Proceedings of Machine Learning Research, vol. 97. PMLR, 2019.

[23] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, *Joint face detection and alignment using multitask cascaded convolutional networks*, IEEE Signal Processing Letters, vol. 23, no. 10, pp. 1499–1503, 2016.

[24] V. Bazarevsky, Y. Kartynnik, A. Vakunov, K. Raveendran, and M. Grundmann, *Blazeface : Sub-millisecond neural face detection on mobile gpus*, CoRR, vol. abs/1907.05047, 2019. [Online]. Available : http ://arxiv.org/abs/1907.05047

[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, *Attention is all you need*, in Advances in Neural Information Processing Systems (NIPS).

[26] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. Jain, *On the detection of digital face manipulation*, 2019.Associates, Inc., 2017.

[27] J. Hu, L. Shen, and G. Sun, *Squeeze-and-excitation networks*, in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition, 2018, pp. 7132–7141.

[28] Floridi, L. (2018). *Artificial intelligence, deepfakes and a future of ectypes*, Philosophy and Technology.

[29] Darius Afchar, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. *Mesonet : a compact facial video forgery detection network*, arXiv preprint arXiv :1809.00888, 2018.

[30] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. *Spatial transformer networks*, In Advances in Neural Information Processing Systems, pages 2017–2025, 2015.