Fader Networks : Manipuler des images en faisant glisser des attributs

Guillaume Lample[1,2], Neil Zeghidour[1,3], Nicolas Usunier[1], Antoine Bordes[1], Ludovic Denoyer[2], Marc'Aurelio Ranzato[1]

{gl, neilz, usunier, abordes, ranzato}@fb.com

ludovic.denoyer@lip6.fr

[1]Facebook AI Research

[2]Sorbonne Universités, UPMC Univ Paris 06, UMR 7606, LIP6

[3]LSCP, ENS, EHESS, CNRS, PSL Research University, INRIA

Résumé

Cet article introduit une nouvelle architecture encodeur-décodeur qui est entraînée à reconstruire des images en dissociant les informations saillantes de l'image et les valeurs des attributs directement dans l'espace latent. En conséquence, après l'entraînement, notre modèle peut générer différentes versions réalistes d'une image d'entrée en faisant varier les valeurs des attributs. En utilisant des valeurs d'attributs continues, nous pouvons choisir à quel point un attribut spécifique est perceptible dans l'image générée. Cette propriété pourrait permettre des applications où les utilisateurs peuvent modifier une image en utilisant des curseurs, comme les faders d'une console de mixage, pour changer l'expression faciale d'un portrait ou pour mettre à jour la couleur de certains objets. Comparée à l'état de l'art, qui repose principalement sur l'entraînement de réseaux adversariaux dans l'espace des pixels en modifiant les valeurs des attributs pendant l'entraînement, notre approche aboutit à des schémas d'entraînement beaucoup plus simples et s'adapte élégamment à de multiples attributs. Nous présentons des preuves que notre modèle peut modifier de manière significative la valeur perçue des attributs tout en préservant le caractère naturel des images.

1 Introduction

Nous nous intéressons au problème de la manipulation d'images naturelles en contrôlant certains attributs d'intérêt. Par exemple, étant donné une photographie du visage d'une personne décrite par son genre, son âge et son expression, nous voulons générer une version réaliste de cette même personne paraissant plus âgée ou plus heureuse, ou bien l'image d'un jumeau hypothétique du sexe opposé.

Cette tâche, ainsi que le problème connexe du transfert de domaine non supervisé, ont récemment suscité beaucoup d'intérêt [17, 24, 9, 26, 21, 23], à la fois comme étude de cas pour les modèles génératifs conditionnels et pour des applications telles que l'édition automatique d'images. Le principal défi est que les transformations sont mal définies et que l'entraînement est non supervisé : le jeu d'entraînement contient des images annotées avec les attributs d'intérêt, mais aucun exemple de transformation n'est fourni. Dans de nombreux cas, comme dans l'exemple de « changement de genre » ci-dessus, il n'existe pas de paires d'images représentant la même personne en tant qu'homme et en tant que femme. Dans d'autres cas, la collecte d'exemples nécessite un processus d'annotation coûteux, comme prendre des photos de la même personne avec et sans lunettes.

Notre approche repose sur une architecture encodeur–décodeur où, étant donnée une image d'entrée x avec ses attributs y, l'encodeur projette x dans une représentation latente z, et le décodeur est entraîné à reconstruire x à partir de (z, y). Lors de l'inférence, une image de test est encodée dans l'espace latent, et l'utilisateur choisit les valeurs d'attributs y qui sont fournies au décodeur. Même avec des valeurs d'attributs binaires pendant l'entraînement, chaque attribut peut être considéré comme une variable continue pendant l'inférence afin de contrôler à quel point il est perçu dans l'image finale. Nous appelons notre architecture *Fader Networks*, par analogie avec les curseurs d'une console de mixage audio, puisque l'utilisateur peut choisir la quantité de chaque attribut qu'il souhaite incorporer.

La caractéristique fondamentale de notre approche est de contraindre l'espace latent à être invariant par rapport aux attributs d'intérêt. Concrètement, cela signifie que la distribution des représentations latentes des images doit être identique pour toutes les valeurs possibles des attributs. Cette invariance est obtenue en utilisant une procédure similaire à l'entraînement adversarial de domaine (voir par exemple [20, 6, 14]). Dans ce processus, un classifieur apprend à prédire les attributs y à partir de la représentation latente z pendant l'entraînement, tandis que l'encodeur–décodeur est entraîné selon deux objectifs simultanés. Le premier objectif est l'erreur de reconstruction du décodeur, c'est-à-dire que la représentation latente z doit contenir

FIGURE 1 – Interpolation entre différents attributs (zoomez pour une meilleure résolution). Chaque ligne montre les reconstructions du même visage avec différentes valeurs d'attributs, où chaque attribut est contrôlé comme une variable continue. Il est ainsi possible de faire paraître une personne âgée plus vieille ou plus jeune, un homme plus masculin ou d'imaginer sa version féminine. Les images de gauche sont les originales.

suffisamment d'informations pour permettre la reconstruction de l'entrée. Le second objectif consiste à tromper le classifieur d'attributs, c'est-à-dire que la représentation latente doit l'empêcher de prédire correctement les valeurs des attributs. Dans ce modèle, atteindre l'invariance revient à filtrer ou à masquer les propriétés de l'image liées aux attributs d'intérêt. Une seule représentation latente correspond ainsi à différentes images partageant une structure commune mais avec des valeurs d'attributs différentes. L'objectif de reconstruction contraint alors le décodeur à utiliser les valeurs des attributs pour choisir, à partir de la représentation latente, l'image souhaitée.

Notre motivation est d'apprendre un espace latent désentrelacé dans lequel nous avons un contrôle explicite sur certains attributs d'intérêt, sans supervision du résultat attendu lors de la modification des valeurs d'attributs. Avec une motivation similaire, plusieurs approches ont été testées sur les mêmes tâches [17, 24], sur des problèmes de traduction d'image à image apparentés [9, 26], ou pour des applications plus spécifiques comme la création d'avatars paramétrés [23]. En plus d'une perte de reconstruction, la grande majorité de ces travaux s'appuient sur un entraînement adversarial dans l'espace des pixels, qui compare pendant l'entraînement des images générées avec une modification

2 Travaux connexes

Il existe une littérature importante sur la génération d'images basée sur des attributs et/ou conditionnelle, que l'on peut diviser selon le niveau de supervision requis, avec trois niveaux distincts.

À une extrémité se trouvent les approches entièrement supervisées, développées pour modéliser des transformations connues, où les exemples prennent la forme de triplets *(entrée, transformation, résultat de la transformation)*. Dans ce cas, le modèle doit apprendre la transformation souhaitée. Ce cadre a été exploré auparavant pour apprendre des transformations affines [8], des rotations 3D [25], des variations d'éclairage [11] et des animations de jeux vidéo 2D [19]. Cependant, les méthodes développées dans ces travaux reposent sur un cadre supervisé et ne peuvent donc pas être appliquées dans notre configuration.

À l'autre extrémité du spectre de supervision se trouvent les méthodes entièrement non supervisées, qui visent à apprendre des réseaux neuronaux profonds capables de désentrelacer les facteurs de variation dans les données, sans spécification préalable des attributs. Des exemples de telles méthodes sont InfoGAN [4], ou le cadre de *minimisation de la prédictibilité* proposé dans [20]. L'éditeur de photos neuronal [3] désentrelace les facteurs de variation dans des images naturelles pour l'édition d'images. Ce cadre est considérablement plus difficile que celui que nous considérons, et il peut être ardu pour ces méthodes de découvrir automatiquement des concepts de haut niveau tels que le genre ou l'âge.

Notre travail se situe entre ces deux cadres précédents. Il est lié à l'approche basée sur l'information de [15]. Les méthodes développées pour le transfert de domaine non supervisé [9, 26, 21, 23] peuvent également être appliquées à notre cas : étant donnés deux domaines d'images différents tels que « dessins » et « photographies », on cherche à mapper une image d'un domaine vers l'autre sans supervision ; dans notre cas, un domaine correspondrait à une valeur d'attribut. Ces correspondances sont apprises à l'aide d'un entraînement adversarial dans l'espace des pixels, comme mentionné dans l'introduction, en utilisant des encodeurs et/ou des décodeurs séparés par domaine, ce qui ne s'adapte pas bien à de multiples attributs. Dans cette lignée de travaux, mais plus spécifiquement pour le problème de la modification d'attributs, le *Invertible conditional GAN* [17] entraîne d'abord un GAN conditionné sur les valeurs d'attributs, puis apprend dans une deuxième étape à mapper les images d'entrée vers l'espace latent du GAN — d'où le nom de GAN inversible. Ce modèle est utilisé comme référence dans nos expériences. Antipov *et al.* [1] utilisent un système de reconnaissance faciale pré-entraîné à la place d'un GAN conditionnel pour apprendre l'espace latent,

et se concentrent uniquement sur l'attribut d'âge. L'approche *Attribute-to-image* [24] est un autoencodeur variationnel qui désentrelace le premier plan et l'arrière-plan afin de générer des images à partir des seules valeurs d'attributs. La génération conditionnelle est effectuée en inférant l'état latent correspondant aux bons attributs, puis en modifiant ces attributs.

De plus, notre travail est lié aux travaux sur l'apprentissage d'espaces latents invariants à l'aide de l'entraînement adversarial dans le cadre de l'adaptation de domaine [6], de la classification équitable [5] et de l'inférence robuste [14]. Le critère d'entraînement que nous utilisons pour imposer l'invariance est similaire à celui employé dans ces travaux ; la différence étant que le but final de ces derniers est uniquement de filtrer les variables parasites ou les informations sensibles. Dans notre cas, nous apprenons des modèles génératifs, et l'invariance est utilisée comme un moyen de forcer le décodeur à utiliser les informations d'attributs dans sa reconstruction.

Enfin, pour l'application consistant à modifier automatiquement des visages à l'aide d'attributs, l'approche d'interpolation de caractéristiques proposée par [22] offre un moyen de générer des altérations d'images basées sur des attributs à l'aide d'un réseau pré-entraîné sur ImageNet. Bien que leur approche soit intéressante d'un point de vue applicatif, leur inférence est coûteuse et, comme elle repose sur des modèles pré-entraînés, elle ne peut pas naturellement intégrer des facteurs ou des attributs qui n'ont pas été anticipés lors du pré-entraînement.

3 Fader Networks

Soit X un domaine d'images et Y l'ensemble des attributs possibles associés aux images de X, où, dans le cas des visages humains, les attributs typiques sont lunettes/pas de lunettes, homme/femme, jeune/vieux. Pour simplifier, nous considérons ici le cas où les attributs sont binaires, mais notre approche pourrait être étendue à des attributs catégoriels. Dans ce cadre, $Y = \{0, 1\}^n$, où n est le nombre d'attributs. Nous disposons d'un jeu d'entraînement $D = \{(x_1, y_1), ..., (x_m, y_m)\}$ de m paires (image, attribut) telles que $x_i \in X$ et $y_i \in Y$. L'objectif final est d'apprendre à partir de D un modèle capable de générer, pour tout vecteur d'attributs y', une version d'une image d'entrée x dont les valeurs d'attributs correspondent à y'.

3.1 Architecture encodeur-décodeur

Notre modèle, décrit dans la Figure 2, repose sur une architecture encodeur–décodeur avec un entraînement adversarial de domaine dans l'espace

latent. L'encodeur $E_{\theta_{enc}} : X \to \mathbb{R}^N$ est un réseau de neurones convolutionnel avec paramètres θ_{enc}, qui mappe une image d'entrée vers sa représentation latente $E_{\theta_{enc}}(x)$. Le décodeur $D_{\theta_{dec}} : (\mathbb{R}^N, Y) \to X$ est un réseau de déconvolution avec paramètres θ_{dec}, qui produit une nouvelle version de l'image d'entrée donnée sa représentation latente $E_{\theta_{enc}}(x)$ et tout vecteur d'attributs y'. Lorsque le contexte est clair, nous utiliserons simplement D et E pour désigner respectivement $D_{\theta_{dec}}$ et $E_{\theta_{enc}}$.

La perte d'auto-encodage associée à cette architecture est une erreur quadratique moyenne (MSE) classique qui mesure la qualité de la reconstruction d'une image d'entraînement x donnée son vecteur d'attributs réel y :

$$L_{AE}(\theta_{enc}, \theta_{dec}) = \frac{1}{m} \sum_{(x,y)\in D} \|D_{\theta_{dec}}(E_{\theta_{enc}}(x), y) - x\|_2^2 \tag{1}$$

Le choix exact de la fonction de perte de reconstruction n'est pas fondamental dans notre approche, et des pertes adversariales telles que PatchGAN [12] pourraient être utilisées en plus de la MSE à ce stade pour obtenir de meilleures textures ou des images plus nettes, comme dans [9]. L'utilisation d'une erreur absolue moyenne ou quadratique moyenne reste néanmoins nécessaire pour garantir que la reconstruction corresponde à l'image originale.

Idéalement, modifier y dans $D(E(x), y)$ devrait générer des images avec des attributs perçus différents, mais similaires à x sous tous les autres aspects. Cependant, sans contrainte supplémentaire, le décodeur apprend à ignorer les attributs, et modifier y au moment du test n'a alors aucun effet.

3.2 Apprentissage de représentations latentes invariantes aux attributs

Pour éviter ce comportement, notre approche consiste à apprendre des représentations latentes invariantes par rapport aux attributs. Par invariance, nous entendons que, données deux versions d'un même objet x et x' qui ne diffèrent que par leurs valeurs d'attributs — par exemple, deux images de la même personne avec et sans lunettes —, les deux représentations latentes $E(x)$ et $E(x')$ devraient être identiques. Lorsque cette invariance est satisfaite, le décodeur doit utiliser les attributs pour reconstruire l'image originale. Comme le jeu d'entraînement ne contient pas de versions différentes d'une même image, cette contrainte ne peut pas être ajoutée directement à la perte.

Nous proposons donc d'incorporer cette contrainte via un entraînement adversarial dans l'espace latent. Cette idée s'inspire des travaux sur la *minimisation de la prédictibilité* [20] et sur l'entraînement adversarial pour l'adaptation de domaine [6, 14], où l'objectif est également d'apprendre une

représentation latente invariante à l'aide d'une formulation adversariale de la fonction de coût. Pour ce faire, un réseau de neurones additionnel appelé *discriminateur* est entraîné à identifier les vrais attributs y d'une paire (x, y) donnée $E(x)$. L'invariance est alors obtenue en apprenant l'encodeur E de manière à ce que le discriminateur soit incapable d'identifier correctement les attributs. Comme dans les GANs [7], cela correspond à un jeu à deux joueurs où le discriminateur cherche à maximiser sa capacité à identifier les attributs, tandis que E cherche à l'en empêcher.

3.3 Objectif du discriminateur

Le discriminateur renvoie des probabilités pour un vecteur d'attributs donné, noté $P_{\theta_{dis}}(y|E(x))$, où θ_{dis} sont les paramètres du discriminateur. En utilisant l'indice k pour se référer au k-ième attribut, on a :

$$\log P_{\theta_{dis}}(y|E(x)) = \sum_{k=1}^{n} \log P_{\theta_{dis},k}(y_k|E(x)).$$

Puisque l'objectif du discriminateur est de prédire les attributs de l'image d'entrée à partir de sa représentation latente, sa perte dépend de l'état courant de l'encodeur et s'écrit :

$$L_{dis}(\theta_{dis}|\theta_{enc}) = -\frac{1}{m} \sum_{(x,y)\in D} \log P_{\theta_{dis}}(y|E_{\theta_{enc}}(x)) \tag{2}$$

3.4 Objectif adversarial

L'objectif de l'encodeur est maintenant de calculer une représentation latente qui optimise deux critères : (1) le décodeur doit pouvoir reconstruire x à partir de $E(x)$ et y ; (2) le discriminateur ne doit pas pouvoir prédire correctement y à partir de $E(x)$. Nous considérons qu'une erreur est commise lorsque le discriminateur prédit $1 - y_k$ pour l'attribut k. Étant donnés les paramètres du discriminateur, la fonction de perte complète de l'architecture encodeur–décodeur est donc :

$$L(\theta_{enc}, \theta_{dec}|\theta_{dis}) = \frac{1}{m} \sum_{(x,y)\in D} \|D_{\theta_{dec}}(E_{\theta_{enc}}(x), y) - x\|_2^2 - \lambda_E \, \mathbb{E}\Big[\log P_{\theta_{dis}}(1-y|E_{\theta_{enc}}(x))\Big] \tag{3}$$

où $\lambda_E > 0$ contrôle le compromis entre la qualité de reconstruction et l'invariance des représentations latentes.

3.5 Algorithme d'apprentissage

Globalement, étant donné l'état courant de l'encodeur, les paramètres optimaux du discriminateur satisfont :

$$\theta^*_{dis}(\theta_{enc}) \in \arg\min_{\theta_{dis}} L_{dis}(\theta_{dis}|\theta_{enc}).$$

En ignorant les problèmes liés aux minima locaux ou multiples, la fonction objectif globale est :

$$(\theta^*_{enc}, \theta^*_{dec}) = \arg\min_{\theta_{enc},\theta_{dec}} L(\theta_{enc}, \theta_{dec}|\theta^*_{dis}(\theta_{enc})).$$

En pratique, il est irréaliste de résoudre $\theta^*_{dis}(\theta_{enc})$ à chaque mise à jour de θ_{enc}. Nous utilisons donc des mises à jour par gradient stochastique pour tous les paramètres, en considérant la valeur courante de θ_{dis} comme une approximation de $\theta^*_{dis}(\theta_{enc})$. La mise à jour au temps t s'écrit :

$$\begin{aligned}
\theta^{(t+1)}_{dis} &= \theta^{(t)}_{dis} - \eta\nabla_{\theta_{dis}} L_{dis}(\theta^{(t)}_{dis}|\theta^{(t)}_{enc}, x^{(t)}, y^{(t)}) \\
\left[\theta^{(t+1)}_{enc}, \theta^{(t+1)}_{dec}\right] &= \left[\theta^{(t)}_{enc}, \theta^{(t)}_{dec}\right] - \eta\nabla_{\theta_{enc},\theta_{dec}} L(\theta^{(t)}_{enc}, \theta^{(t)}_{dec}|\theta^{(t+1)}_{dis}, x^{(t)}, y^{(t)})
\end{aligned}$$

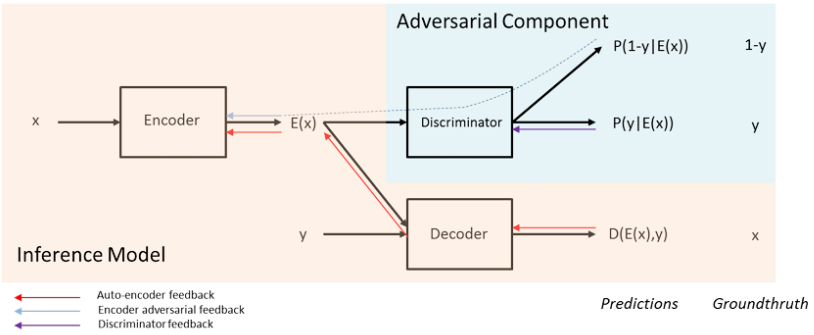

FIGURE 2 – Architecture principale. Une paire (image, attribut) (x, y) est donnée en entrée. L'encodeur mappe x vers la représentation latente z ; le discriminateur est entraîné à prédire y à partir de z, tandis que l'encodeur est entraîné pour rendre cette prédiction impossible. Le décodeur doit reconstruire x à partir de (z, y). Au moment du test, le discriminateur est écarté et le modèle peut générer différentes versions de x en lui fournissant différentes valeurs d'attributs.

Les détails d'entraînement et de modélisation sont fournis dans la section suivante.

4 Implémentation

Nous adaptons l'architecture de notre réseau à partir de [9]. Soit C_k une couche Convolution–BatchNorm–ReLU comportant k filtres. Les convolutions utilisent des noyaux de taille 4×4, avec un pas (*stride*) de 2 et un remplissage (*padding*) de 1, de sorte que chaque couche de l'encodeur divise par 2 la taille de son entrée. Nous utilisons des *leaky-ReLU* avec une pente de 0.2 dans l'encodeur, et de simples ReLU dans le décodeur.

4.1 Encodeur

L'encodeur est constitué des 7 couches suivantes :

$$C16 - C32 - C64 - C128 - C256 - C512 - C512$$

Les images d'entrée ont une taille de 256×256. En conséquence, la représentation latente d'une image est composée de 512 cartes de caractéristiques (*feature maps*) de taille 2×2. Dans nos expériences, utiliser 6 couches donnait des résultats similaires, tandis qu'utiliser 8 couches diminuait significativement la performance, même en augmentant le nombre de cartes de caractéristiques dans l'état latent.

4.2 Décodeur

Pour fournir au décodeur les attributs de l'image, nous concaténons le code latent à chaque couche donnée en entrée au décodeur, où le code latent d'une image est la concaténation des vecteurs *one-hot* représentant les valeurs de ses attributs (les attributs binaires sont représentés par $[1, 0]$ et $[0, 1]$). Nous ajoutons le code latent en tant que canaux d'entrée constants supplémentaires pour toutes les convolutions du décodeur. En notant n le nombre d'attributs (d'où un code de taille $2n$), le décodeur est symétrique à l'encodeur, mais utilise des convolutions transposées pour l'échantillonnage inverse (*up-sampling*) :

$$C_{512+2n} - C_{512+2n} - C_{256+2n} - C_{128+2n} - C_{64+2n} - C_{32+2n} - C_{16+2n}.$$

4.3 Discriminateur

Le discriminateur est une couche C_{512} suivie d'un réseau entièrement connecté de deux couches de taille 512 et n respectivement.

Modèle	Naturalité			Précision		
	Bouche	Sourire	Lunettes	Bouche	Sourire	Lunettes
Image réelle	92.6	87.0	88.6	89.0	88.3	97.6
IcGAN AE	22.7	21.7	14.8	88.1	91.7	86.2
IcGAN Swap	11.4	22.9	9.6	10.1	9.9	47.5
FadNet AE	88.4	75.2	78.8	91.8	90.1	94.5
FadNet Swap	79.0	31.4	45.3	66.2	97.1	76.6

TABLE 1 – Évaluation perceptuelle de la naturalité et de la précision de permutation (*swap*) pour chaque modèle. La *naturalité* correspond au pourcentage d'images jugées « réelles » par des évaluateurs humains à la question : « Cette image est-elle une photographie réelle ou une image générée ? ». La *précision* correspond à la justesse des évaluateurs humains à identifier les valeurs d'attributs.

4.4 Régularisation par Dropout

Nous avons constaté qu'il était extrêmement bénéfique d'ajouter du *dropout* dans notre discriminateur. Nous avons fixé le taux de *dropout* à 0.3 dans toutes nos expériences. En suivant [9], nous avons également essayé d'ajouter du *dropout* dans les premières couches du décodeur, mais cela a entraîné une diminution significative des performances dans nos expériences.

4.5 Planification du coût du discriminateur

De manière similaire à [2], nous utilisons un poids variable pour le coefficient de perte du discriminateur λ_E. Nous initialisons λ_E à 0, et le modèle est alors entraîné comme un autoencodeur classique. Ensuite, λ_E est augmenté linéairement jusqu'à 0.0001 au cours des 500 000 premières itérations, afin d'encourager progressivement le modèle à produire des représentations invariantes. Cette planification s'est révélée critique dans nos expériences. Sans elle, nous avons observé que l'encodeur était trop fortement affecté par la perte provenant du discriminateur, même pour de faibles valeurs de λ_E.

4.6 Sélection du modèle

La sélection du modèle a d'abord été effectuée automatiquement à l'aide de deux critères. Premièrement, nous avons utilisé l'erreur de reconstruction sur les images originales, mesurée par la MSE. Deuxièmement, nous voulions également que le modèle permute correctement les attributs d'une image. Pour ce second critère, nous avons entraîné un classifieur à prédire les attributs d'une image. À la fin de chaque époque, nous avons échangé (*swap*) les

attributs de chaque image du jeu de validation et mesuré les performances du classifieur sur les images décodées. Ces deux métriques ont été utilisées pour filtrer les modèles potentiellement performants. Le modèle final a été sélectionné sur la base d'une évaluation humaine d'images du jeu d'entraînement reconstruites avec des attributs échangés.

5 Expériences

5.1 Expériences sur le jeu de données CelebA

Configuration expérimentale. Nous présentons d'abord des expériences sur le jeu de données CelebA [13], qui contient environ 200 000 images de célébrités de dimensions 178×218, annotées avec 40 attributs. Nous avons utilisé la division standard en ensembles d'entraînement, de validation et de test. Toutes les images présentées dans cet article ou utilisées pour l'évaluation proviennent de l'ensemble de test. Pour le prétraitement, nous avons recadré les images à 178×178 et les avons redimensionnées à 256×256, qui est la résolution utilisée dans toutes les figures de l'article. Les valeurs des pixels ont été normalisées dans l'intervalle $[-1, 1]$. Tous les modèles ont été entraînés avec l'optimiseur Adam [10], en utilisant un taux d'apprentissage de 0.002, $\beta_1 = 0.5$, et une taille de lot de 32. Nous avons effectué une augmentation de données en inversant horizontalement les images avec une probabilité de 0.5 à chaque itération. Comme modèle de référence, nous avons utilisé IcGAN [17], avec le modèle fourni par les auteurs et entraîné sur le même jeu de données. [1]

Évaluation qualitative. La Figure 3 montre des exemples d'images générées lors de l'échange de différents attributs. Les images générées ont une haute qualité visuelle et gèrent clairement les changements de valeurs d'attributs, par exemple en ajoutant des lunettes réalistes aux différents visages. Ces images générées confirment que la représentation latente apprise par les *Fader Networks* est à la fois invariante aux valeurs des attributs, tout en capturant les informations nécessaires pour générer n'importe quelle version d'un visage, pour n'importe quelle valeur d'attribut. En effet, lorsqu'on observe la forme des lunettes générées, on remarque que différentes formes et couleurs de lunettes ont été intégrées dans le visage d'origine selon la personne : notre modèle n'ajoute pas simplement des lunettes « génériques » à tous les visages, mais génère des lunettes plausibles dépendant de l'entrée.

Protocole d'évaluation quantitative. Nous avons effectué une évaluation quantitative des *Fader Networks* sur Amazon Mechanical Turk, en

1. Code disponible à https ://github.com/Guim3/IcGAN

FIGURE 3 – Échange des attributs de différents visages. Zoomez pour une meilleure résolution.

utilisant IcGAN comme référence. Nous avons choisi trois attributs : *Bouche* (ouverte/fermée), *Sourire* (avec/sans) et *Lunettes* (avec/sans), car ils étaient communs entre IcGAN et notre modèle. Nous avons évalué deux aspects différents des images générées : la *naturalité*, qui mesure la qualité des images générées, et la *précision*, qui mesure dans quelle mesure l'échange d'une valeur d'attribut est effectivement reflété dans la génération. Ces deux mesures sont nécessaires pour vérifier que nous générons des images naturelles et que le changement d'attribut est efficace.

Nous comparons les cas suivants : **REAL IMAGE**, qui fournit les images originales sans transformation ; **FADNET AE** et **ICGAN AE**, qui reconstruisent les images originales sans modification d'attributs ; et **FADNET SWAP** et **ICGAN SWAP**, qui génèrent des images avec un attribut échangé, par exemple *Avec lunettes* $\rightarrow$ *Sans lunettes*.

Avant d'être soumises à Mechanical Turk, toutes les images ont été recadrées et redimensionnées suivant le même traitement que IcGAN. Ainsi, les images de sortie ont été affichées en 64×64, empêchant les travailleurs de juger uniquement sur la netteté.

Techniquement, il faudrait aussi vérifier que l'identité d'une personne est préservée lors de l'échange d'attributs. Cela semblait être un problème pour les méthodes basées sur GAN, mais la qualité de reconstruction de notre modèle est très bonne (RMSE sur le test de 0.0009, contre 0.028 pour IcGAN), et nous n'avons pas observé ce problème. Par conséquent, nous n'avons pas évalué cet aspect.

Pour la *naturalité*, les 500 premières images du jeu de test, avec 250 images pour chaque valeur d'attribut, ont été montrées aux travailleurs de Mechanical Turk — 100 pour chacun des 5 modèles différents mentionnés ci-dessus.

Pour chaque image, nous avons demandé si elle semblait naturelle ou générée. La description donnée aux participants montrait 4 exemples d'images réelles et 4 exemples d'images générées (1 *FADNET AE*, 1 *FADNET SWAP*, 1 *ICGAN AE*, 1 *ICGAN SWAP*).

La précision de chaque modèle pour chaque attribut a été évaluée dans une tâche de classification distincte, résultant en un total de 15 expériences. Par exemple, l'expérience *FadNet/Glasses* consistait à demander si les personnes auxquelles le modèle *FADNET SWAP* avait ajouté des lunettes en portaient effectivement, et inversement. Cela permet d'évaluer la perceptibilité des changements pour l'œil humain. Dans chaque expérience, 100 images ont été présentées (50 par classe, dans l'ordre d'apparition dans le jeu de test).

Dans les deux évaluations quantitatives, chaque expérience a été effectuée par 10 participants, soit 5 000 échantillons par expérience pour la *naturalité*, et 1 000 échantillons par expérience de classification sur les attributs échangés. Les résultats de ces deux tâches sont présentés dans la Table 1.

FIGURE 4 – (Zoomez pour une meilleure résolution.) Exemples de permutations multiples d'attributs (genre / ouverture des yeux / lunettes) réalisées par le même modèle. Les images de gauche sont les originales.

Résultats quantitatifs. Dans les expériences de naturalité, seulement environ 90 % des images réelles ont été classées comme « réelles » par les évaluateurs, ce qui montre le haut niveau d'exigence pour produire des images naturelles. Notre modèle obtient de très bons scores de naturalité lors de la reconstruction sans échange d'attributs : 88.4 %, 75.2 % et 78.8 %, comparé aux reconstructions d'IcGAN dont la précision ne dépasse pas 23 %. Pour l'échange d'attributs, *FADNET SWAP* surpasse encore largement *ICGAN SWAP*. Cependant, la précision de naturalité varie selon l'attribut échangé : de 79.0 % pour l'ouverture de la bouche, à 31.4 % pour le sourire.

Les expériences de classification montrent que les reconstructions *FADNET AE* et *ICGAN AE* ont des scores de classification élevés, comparables à ceux des images réelles pour les attributs *Bouche* et *Sourire*. *FADNET*

SWAP obtient une précision de 66.2 % pour la bouche, 76.6 % pour les lunettes et 97.1 % pour le sourire, indiquant que notre modèle peut échanger ces attributs avec une grande efficacité. En revanche, avec des précisions de 10.1 %, 47.5 % et 9.9 % pour ces mêmes attributs, *ICGAN SWAP* ne parvient pas à produire des échanges convaincants.

Permutation multiple d'attributs. Nous présentons des résultats qualitatifs sur la capacité de notre modèle à échanger plusieurs attributs à la fois dans la Figure 4, en modifiant conjointement le genre, l'ouverture des yeux et les lunettes. Même dans ce cadre plus difficile, notre modèle peut générer des images convaincantes avec plusieurs échanges simultanés.

5.2 Expériences sur le jeu de données Flowers

Nous avons également mené des expériences sur le jeu de données Oxford-102, qui contient environ 9 000 images de fleurs classées en 102 catégories [16]. Comme ce jeu de données ne contient pas d'autres étiquettes que les catégories de fleurs, nous avons construit une liste d'attributs de couleur à partir des légendes textuelles fournies par [18]. Chaque fleur est décrite par 10 légendes différentes. Pour une couleur donnée, nous avons attribué à une fleur l'attribut correspondant si cette couleur apparaissait dans au moins 5 des 10 légendes. Bien que naïve, cette approche s'est révélée suffisante pour générer des étiquettes précises. Nous avons redimensionné les images à 64×64.

FIGURE 5 – Exemples de fleurs reconstruites avec différentes valeurs de l'attribut *rose*. La première ligne montre les images originales. Augmenter la valeur de cet attribut rend les fleurs plus roses, tandis que la diminuer sur des fleurs initialement roses les rend jaunes ou orangées.

La Figure 5 représente des fleurs reconstruites avec différentes valeurs de l'attribut *rose*. On peut observer que la couleur des fleurs change dans la direction souhaitée, tout en gardant le fond proprement inchangé.

6 Conclusion

Nous avons présenté une nouvelle approche pour générer des variations d'images en modifiant les valeurs des attributs. Cette approche repose sur l'imposition de l'invariance de l'espace latent par rapport aux attributs. Un avantage clé de notre méthode, comparée à de nombreux modèles récents [26, 9], est qu'elle génère des images réalistes de haute résolution sans nécessiter l'application d'un GAN sur la sortie du décodeur. En conséquence, elle pourrait facilement être étendue à d'autres domaines comme la parole ou le texte, où la rétropropagation à travers le décodeur peut s'avérer très difficile à cause du processus de génération non différentiable du texte, par exemple. Cependant, les méthodes couramment utilisées en vision pour évaluer la qualité visuelle des images générées, comme PatchGAN, pourraient tout à fait être appliquées en complément de notre modèle.

Remerciements

Les auteurs tiennent à remercier Yedid Hoshen pour les discussions initiales concernant les idées principales de l'article, ainsi que Christian Pursch et Alexander Miller pour leur aide dans la mise en place des expériences et des évaluations sur Mechanical Turk. Les auteurs sont également reconnaissants envers David Lopez-Paz et Mouhamadou Moustapha Cisse pour leurs retours utiles et leur soutien sur ce projet.

Références

[1] Grigory Antipov, Moez Baccouche et Jean-Luc Dugelay. *Face aging with conditional generative adversarial networks.* arXiv preprint arXiv :1702.01983, 2017.

[2] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz et Samy Bengio. *Generating sentences from a continuous space.* arXiv preprint arXiv :1511.06349, 2015.

[3] Andrew Brock, Theodore Lim, J. M. Ritchie et Nick Weston. *Neural photo editing with introspective adversarial networks.* arXiv preprint arXiv :1609.07093, 2016.

[4] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever et Pieter Abbeel. *InfoGAN : Interpretable representation learning by information maximizing generative adversarial nets.* In Advances in Neural Information Processing Systems, pages 2172–2180, 2016.

[5] Harrison Edwards et Amos Storkey. *Censoring representations with an adversary.* arXiv preprint arXiv :1511.05897, 2015.

[6] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand et Victor Lempitsky. *Domain-adversarial training of neural networks.* Journal of Machine Learning Research, 17(59) :1–35, 2016.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville et Yoshua Bengio. *Generative adversarial nets.* In Advances in Neural Information Processing Systems, pages 2672–2680, 2014.

[8] Geoffrey Hinton, Alex Krizhevsky et Sida Wang. *Transforming auto-encoders.* Artificial Neural Networks and Machine Learning–ICANN 2011, pages 44–51, 2011.

[9] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou et Alexei A. Efros. *Image-to-image translation with conditional adversarial networks.* arXiv preprint arXiv :1611.07004, 2016.

[10] Diederik Kingma et Jimmy Ba. *Adam : A method for stochastic optimization.* arXiv preprint arXiv :1412.6980, 2014.

[11] Tejas D. Kulkarni, William F. Whitney, Pushmeet Kohli et Josh Tenenbaum. *Deep convolutional inverse graphics network.* In Advances in Neural Information Processing Systems, pages 2539–2547, 2015.

[12] Chuan Li et Michael Wand. *Precomputed real-time texture synthesis with markovian generative adversarial networks.* In European Conference on Computer Vision, pages 702–716. Springer, 2016.

[13] Ziwei Liu, Ping Luo, Xiaogang Wang et Xiaoou Tang. *Deep learning face attributes in the wild.* In Proceedings of International Conference on Computer Vision (ICCV), 2015.

[14] Gilles Louppe, Michael Kagan et Kyle Cranmer. *Learning to pivot with adversarial networks.* arXiv preprint arXiv :1611.01046, 2016.

[15] Michael F. Mathieu, Junbo Jake Zhao, Aditya Ramesh, Pablo Sprechmann et Yann LeCun. *Disentangling factors of variation in deep representation using adversarial training.* In Advances in Neural Information Processing Systems, pages 5041–5049, 2016.

[16] Maria-Elena Nilsback et Andrew Zisserman. *Automated flower classification over a large number of classes.* In Computer Vision, Graphics & Image Processing, 2008. ICVGIP'08. Sixth Indian Conference on, pages 722–729. IEEE, 2008.

[17] Guim Perarnau, Joost van de Weijer, Bogdan Raducanu et Jose M. Álvarez. *Invertible conditional GANs for image editing.* arXiv preprint arXiv :1611.06355, 2016.

[18] Scott Reed, Zeynep Akata, Honglak Lee et Bernt Schiele. *Learning deep representations of fine-grained visual descriptions.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 49–58, 2016.

[19] Scott E. Reed, Yi Zhang, Yuting Zhang et Honglak Lee. *Deep visual analogy-making.* In Advances in Neural Information Processing Systems, pages 1252–1260, 2015.

[20] Jürgen Schmidhuber. *Learning factorial codes by predictability minimization.* Neural Computation, 4(6) :863–879, 1992.

[21] Yaniv Taigman, Adam Polyak et Lior Wolf. *Unsupervised cross-domain image generation.* arXiv preprint arXiv :1611.02200, 2016.

[22] Paul Upchurch, Jacob Gardner, Kavita Bala, Robert Pless, Noah Snavely et Kilian Weinberger. *Deep feature interpolation for image content changes.* arXiv preprint arXiv :1611.05507, 2016.

[23] Lior Wolf, Yaniv Taigman et Adam Polyak. *Unsupervised creation of parameterized avatars.* arXiv preprint arXiv :1704.05693, 2017.

[24] Xinchen Yan, Jimei Yang, Kihyuk Sohn et Honglak Lee. *Attribute2image : Conditional image generation from visual attributes.* In European Conference on Computer Vision, pages 776–791. Springer, 2016.

[25] Jimei Yang, Scott E. Reed, Ming-Hsuan Yang et Honglak Lee. *Weakly-supervised disentangling with recurrent transformations for 3D view synthesis.* In Advances in Neural Information Processing Systems, pages 1099–1107, 2015.

[26] Jun-Yan Zhu, Taesung Park, Phillip Isola et Alexei A. Efros. *Unpaired image-to-image translation using cycle-consistent adversarial networks.* arXiv preprint arXiv :1703.10593, 2017.