Suréchantillonage Actif pour Modérer l'Apprentissage de Biais Visuels

Alexandre Devillers, Benoît Alcaraz, Mathieu Lefort

Suréchantillonage Actif pour Modérer l'Apprentissage de Biais Visuels

A. Devillers[1], B. Alcaraz[2], M. Lefort[1]

[1] Univ Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, F-69622 Villeurbanne, France
[2] Université du Luxembourg

alexandre.devillers@liris.cnrs.fr, benoit.alcaraz@uni.lu, mathieu.lefort@liris.cnrs.fr

Résumé

Les techniques de reconnaissance d'image sont devenues particulièrement performantes lors de cette dernière décennie. Cependant l'apprentissage sur des données biaisées, par exemple, la présence régulière d'un fond bleu derrière un poisson, réduit drastiquement les performances des approches actuelles qui ont tendance à apprendre ces biais. Dans ce papier, nous proposons `ImRAN`*, une méthode d'apprentissage actif qui consiste à surreprésenter les données mal classifiées, et qui améliore l'état de l'art sur le jeu de données Biased MNIST lorsqu'il est très biaisé.*

Mots-clés

Apprentissage profond non-biaisé, apprentissage de raccourcis, apprentissage actif, classification d'image.

Abstract

Image recognition techniques have become particularly powerful in the last decade. However, learning on biased data, for example, the regular presence of a blue background behind a fish, drastically reduces the performances of current approaches that tend to learn these biases. In this paper, we propose `ImRAN`*, an active learning method that consists in over-representing misclassified data, which improves the state of the art on the Biased MNIST dataset when it is highly biased.*

Keywords

De-biased deep learning, shortcut learning, active learning, image classification.

1 Introduction

Les techniques de reconnaissance d'image sont devenues particulièrement performantes lors de cette dernière décennie [15], en particulier grâce à l'utilisation de réseaux de neurones convolutifs [10]. Ces approches statistiques dépendent principalement de la qualité des données d'apprentissage. Cependant, en pratique certains jeux de données peuvent présenter un ou plusieurs biais, c'est-à-dire une corrélation plus ou moins forte entre un élément récurrent des images et leurs classes, sans pour autant que cet élément soit prédictif. Or, l'apprentissage d'un modèle se basant sur ces caractéristiques se traduirait par une faible capacité de généralisation hors du domaine d'entraînement. Les réseaux de neurones, ayant tendance à converger vers ces solutions lorsqu'elles sont simples et à exploiter ces raccourcis d'apprentissage [8], sont possiblement inopérants sur certains de ces jeux de données. Par exemple, dans une tâche de classification d'animaux, les photos de poissons ont régulièrement un fond bleu. Un modèle d'apprentissage profond risque ainsi de converger vers un modèle de décision classifiant toute image avec un fond bleu comme étant un poisson. Cela limite la reconnaissance de poissons dans d'autres décors, et crée des erreurs de classification pour d'autres images avec un fond bleu n'étant pas des poissons.

Avoir un algorithme performant malgré la présence de biais est une propriété souhaitable, car il n'est pas toujours possible de connaître leur présence à l'avance dans le jeu de données. Une mauvaise gestion de ces derniers peut donc entraîner une perte importante de performances lors d'une utilisation applicative de ces algorithmes. Cela peut dans certains cas, comme les voitures autonomes, causer des dommages matériels ou physiques. Dans un autre contexte applicatif, une approche résiliente aux biais permettrait de limiter les biais sociétaux des bases de données, comme l'influence du genre, de la couleur de peau, ou autres facteurs pouvant dépendre des échantillons récoltés pour l'apprentissage [23]. Cependant, puisque la présence ou la nature des biais n'est pas toujours connue à l'avance, il est souhaitable qu'une méthode limitant l'apprentissage de biais ne dégrade pas les performances de base lorsqu'il n'y en a pas.

Dans ce papier, nous proposons la méthode *Image Representation Avoiding Naive learning* (`ImRAN`) qui modifie l'apprentissage d'une architecture de réseaux de neurones convolutionnels. Cette méthode consiste en un changement automatique et en ligne de la distribution du jeu de données d'apprentissage, ainsi qu'au bruitage des images afin de rajouter un peu de variabilité. Un élément sera d'autant plus vu que sa classification aux époques précédentes a été faible. En effet, si le modèle tend à apprendre le biais, les données mal classifiées seront principalement celles non-biaisées, et multiplier leurs occurrences dans le jeu d'apprentissage aura donc tendance à le débiaiser.

Le papier est divisé comme suit. Tout d'abord, la Section 2 recontextualise notre article dans l'état de l'art. En-

suite, nous introduisons notre méthode ImRAN dans la Section 3. Dans la Section 4, nous présentons le protocole et analysons les résultats obtenus. Enfin, nous discutons de notre approche et des travaux futurs dans la Section 5, puis concluons dans la Section 6.

2 Travaux antérieurs

La nature des réseaux de neurones à exploiter des raccourcis d'apprentissage, tel que les biais, réduit la confiance qui leur est accordée et peut limiter leur application pratique [8]. La forte présence de biais dans un jeu de données peut réduire à un niveau proche de l'aléatoire les performances des réseaux de neurones dans une tâche de classification binaire [16]. Un autre exemple où l'apprentissage de biais affecte les performances est dans les approches de *Visual Question Answering* (VQA), où les modèles ignorent la modalité visuelle pour reposer principalement sur des biais statistiques présents dans le langage [2]. Par exemple, une question comme "Quelle est la couleur de la banane présente à l'image ?" aura comme réponse "Jaune" indépendamment de la couleur de la banane dans l'image. Dans cet article, nous nous concentrons sur de la classification d'images, bien que notre méthode puisse éventuellement être adaptée à d'autres tâches ou modalités.

Parmi les approches existantes visant à réduire l'apprentissage de biais visuels, certaines demandent d'avoir un certain nombre de connaissances expertes vis-à-vis des biais présents. Une partie des méthodes repose sur le fait de connaître au préalable, pour chaque image du jeu d'entraînement, la présence ou non d'un biais [17, 21, 23]. Bien que ces méthodes puissent présenter des performances importantes, elles sont difficilement applicables en pratique. En effet, un jeu de données n'est pas toujours connu à l'avance comme comportant des biais. De plus, dans l'éventualité où l'information serait disponible, l'accès à une annotation des biais reste peu fréquent. D'autres méthodes demandent uniquement à connaître le type de biais, ou bien exigent qu'il soit de bas niveau sémantique [3, 6, 19, 22]. Cependant, le spectre d'application de ces méthodes reste limité.

Certaines approches de la littérature, qu'elles utilisent ou non des informations sur le biais, sont basées sur la modification de la fonction de coût qu'optimise le modèle, et plus particulièrement sur l'ajout d'un terme de régularisation dont l'objectif est d'empêcher la convergence vers les raccourcis d'apprentissage exploitant les biais présents dans les données d'apprentissage. C'est par exemple le cas de EnD [21] qui force les représentations des données partageant un même biais à être différentes (ce qui demande de connaître le biais), tout en rapprochant entre elles les représentations des données d'une même classe. Des travaux se sont concentrés sur l'utilisation de réseaux de neurones naïfs, peu profonds et avec un champ perceptif réduit, afin de converger vers l'apprentissage des biais présents dans les jeux de données [3, 6, 19, 22], pour ensuite pénaliser l'apprentissage d'un autre modèle, plus complexe, afin qu'il évite les biais appris par les modèles naïfs. Ces approches font généralement l'hypothèse forte, appuyée par des connaissances expertes, que les biais sont souvent des caractéristiques de bas niveau tandis que les caractéristiques prédictives des classes sont hautement sémantiques. C'est notamment le cas lorsque le biais relève de la présence d'une couleur ou d'une texture. Récemment, la fonction de coût *Generalized Cross Entropy* (GCE) [24], connue comme facilitant l'apprentissage des biais, a été utilisée pour remplacer les architectures naïves et accentuer l'apprentissage des biais du premier modèle [16, 19]. Enfin, l'approche RUBi [5] propose une fonction de coût qui force une rétro-propagation d'un gradient faible pour les exemples où le modèle a déjà un haut niveau de confiance dans la bonne réponse après *softmax*, et à l'inverse, rétropropage un gradient fort si le modèle avait un taux de confiance faible. L'idée sous-jacente est d'attribuer une plus grande importance aux exemples non-biaisés qu'aux exemples biaisés, en faisant l'hypothèse qu'un exemple biaisé donnera un haut taux de confiance pour la bonne réponse. La méthode que nous présentons dans cet article se rapproche de RUBi dans l'idée de pondérer plus fortement les exemples non-biaisés pendant l'apprentissage, cependant, là où RUBi applique cette pondération au niveau de la rétro-propagation, ImRAN joue sur la distribution du jeu de données.

Une autre catégorie d'approches a cherché à modifier directement les images des jeux de données en entrée afin d'en retirer les biais à l'échelle des pixels [1, 7, 9, 18, 20, 23]. Toutefois, cela présente une limitation majeure due au fait que retirer certains biais peut être complexe, voire impossible, par exemple lorsque le biais relève de la texture [9] ou bien du genre des personnes [12]. Pour contourner cette limitation, [16] propose de manipuler les biais à l'échelle des représentations et de croiser les biais des exemples entre eux afin qu'ils ne soient plus prédictifs des différentes classes.

Pour finir, un autre type d'approches vise à changer directement la distribution des données afin d'augmenter la proportion de données non-biaisées comparativement aux données biaisées. Cela a pour effet de réduire la prédictivité des biais vis-à-vis des classes, supprimant ainsi les biais dans la nouvelle distribution obtenue. C'est ce que propose la méthode REPAIR [17], qui attribue à chaque exemple du jeu de données un poids. Ce poids est ensuite appris via rétro-propagation dans l'objectif de réduire la probabilité de tirage des exemples avec des biais. Cependant, REPAIR demande d'avoir des informations expertes sur les biais.

Dans cet article, nous proposons une méthode qui change activement la distribution des données d'apprentissage pour tendre à augmenter la proportion d'exemples non-biaisés. De plus, notre méthode ne demande pas de connaissance experte sur les biais présents.

3 Méthode proposée

Nous nous plaçons dans le cadre d'un jeu de données biaisé, c'est-à-dire qu'il y est plus facile de reconnaître les classes dans le jeu d'apprentissage via la détection d'un élément (appelé biais) qui est non pertinent pour la classification sur

FIGURE 1 – Exemples d'images du jeu de données *Biased* MNIST dans lequel la couleur de fond est corrélée avec la classe.

FIGURE 2 – À gauche une image de *Biased* MNIST sans bruit, à droite, avec application d'un bruit gaussien de moyenne 0 et de déviation standard 1.

le jeu de test. Le jeu d'apprentissage n'est ainsi pas représentatif de celui du jeu de test. Pour pallier ce problème, nous proposons la méthode *Image Representation Avoiding Naive learning* (`ImRAN`) qui cherche à modifier l'échantillonnage des données dans le jeu d'entraînement pour essayer de la rendre plus similaire à celui du jeu de test, sans pour autant modifier l'algorithme d'apprentissage.
Le principe consiste à surreprésenter les exemples mal classifiés par le modèle dans le jeu d'apprentissage. En effet, dans le cas d'un jeu d'apprentissage biaisé, le modèle tend à apprendre le biais, ce qui implique que les données non-biaisées finissent généralement mal classifiée. La proportion de ces exemples est alors augmentée dans le jeu de données, permettant de réduire le biais dans le jeu d'apprentissage et donc le risque pour le modèle de l'apprendre. L'avantage de cette méthode par rapport à l'existant est double. Premièrement, nous n'avons pas besoin d'information ou d'annotation du biais, la détection se faisant uniquement sur la performance de classification de chaque exemple. Deuxièmement, si le jeu de données n'est pas biaisé, la remise des exemples mal classifiés ne devrait pas introduire de biais et donc affecter de façon de manière significative les performances issues de l'apprentissage.

Algorithm 1 Boucle principale d'apprentissage.

$\mathcal{D}$: jeu de données initial.
$\mathcal{D}'_i$: jeu de données augmenté au pas de temps i.
$n \in \mathbb{N}$: nombre d'époques pour l'apprentissage.

$i \leftarrow 0$
$\mathcal{D}'_i \leftarrow \mathcal{D}$
while $i < n$ **do**
    `learn`$(\mathcal{D}'_i)$
    $\mathcal{D}'_i \leftarrow$ `duplique`$(\mathcal{D}, \mathcal{D}'_i)$
    $i \leftarrow i + 1$
end while

La boucle d'apprentissage est décrite dans l'Algorithme 1. À chaque itération, la fonction `duplique`$(\mathcal{D}, \mathcal{D}'_{i-1})$ génère un nouveau jeu de données $\mathcal{D}'_i$ à partir de la base de données originale $\mathcal{D}$, du nombre d'erreurs pour chaque duplication, et du nombre total de duplications pour chaque élément, du jeu de données $\mathcal{D}'_i$ issu de la fonction `duplique` au pas de temps précédent. Plus précisément, la fonction `duplique`$(\mathcal{D}, \mathcal{D}'_{i-1})$, décrite dans l'Algorithme 2, fonctionne comme suit. Pour chaque entrée d du jeu de données originel $\mathcal{D}$, on retrouve l'ensemble des augmentations de d, c'est-à-dire des copies bruitées ou non, dans le jeu de données $\mathcal{D}'_{i-1}$ généré au pas de temps précédent $i-1$ par `duplique`. On calcule ensuite le ratio d'erreurs de prédiction du réseau sur ces données, arrondi à l'entier supérieur. Enfin, on génère un nombre d'augmentations entre 1 et K, proportionnel au taux d'erreur de prédiction. La fonction `augmente` prend en paramètre un élément d, un nombre de copies à générer, et une déviation standard pour le bruit, et retourne un ensemble de copies, bruitées par un bruit gaussien de moyenne nulle et de déviation standard σ.

Algorithm 2 Fonction `duplique` : génère un dataset augmenté $\mathcal{D}'_i$ comportant 1 à K copies de chaque entrée du dataset originel $\mathcal{D}$ au pas de temps i.

$\mathcal{D}$: dataset initial.
$\mathcal{D}'_{i-1}$: dataset augmenté au pas de temps $i-1$.
$K \in \mathbb{N}$: nombre maximum de duplications.
$C \in]0;1]$: paramètre de la moyenne glissante exponentielle. (Pas de lissage pour une valeur de 1)

function DUPLIQUE($\mathcal{D}, \mathcal{D}'_{i-1}$) :
    $\mathcal{D}'_i \leftarrow \varnothing$
    for all $d \in \mathcal{D}$ **do**
        $A \leftarrow$ `getAugmentations`$(d, \mathcal{D}'_{i-1})$
        $tauxErreur \leftarrow$ `getTauxErreur`(A)
        $nbCopies \leftarrow tauxErreur \times (K-1) + 1$
        $nbCopies \leftarrow \lceil nbCopies \times C + |A| \times (1-C) \rceil$
        $\mathcal{D}'_i \leftarrow \mathcal{D}'_i \cup$ `augmente`$(d, nbCopies, \sigma)$
    end for
    return $\mathcal{D}'_i$
end function

Une inertie est appliquée à la variation du nombre de copies entre chaque itération. Ce lissage permet de limiter les changements trop brusques entre les jeux de données d'époques consécutives. Le nombre d'occurrence d'une entrée est calculé comme la moyenne glissante entre son ancienne valeur et la nouvelle, pondéré par C.

4 Expérimentations

4.1 Protocole expérimental

4.1.1 *Biased* MNIST

Le jeu de données *Biased* MNIST [4] est composé de 60000 images de taille 28×28 biaisées synthétiquement. Les images sont des chiffres écrits à la main en blanc sur fond noir issues du jeu de données MNIST [14] ; chaque chiffre représentant une classe. Pour y ajouter synthétiquement un biais, chaque classe se voit attribuer une couleur unique, et le fond noir présent sur les images est remplacé par la couleur correspondant au label associé à l'image (voire Figure 1). Les images non-biaisées ont quant à elles un fond

ρ	Vanilla	LearnedMixin	RUBi	ReBias	ImRAN (avec bruit)	ImRAN (sans bruit)
.999	10.4	12.1	13.7	22.7	**44.2 ± 2.8**	33.5 ± 4.1
.997	33.4	50.2	43.0	64.2	**74.8 ± 2.2**	69.4 ± 2.3
.995	72.1	78.2	**90.4**	76.0	82.7 ± 1.3	79.7 ± 1.0
.990	89.1	88.3	**93.6**	88.1	90.5 ± 0.5	90.5 ± 0.5
.100	99.2	54.6	**99.3**	**99.3**	99.0 ± 0.1	99.2 ± 0.1

TABLE 1 – **Résultats pour *Biased* MNIST.** *Accuracy* des différentes méthodes sur le jeu de données *Biased* MNIST, suivant le taux d'images biaisées ρ. Chaque valeur est la moyenne de 10 exécutions pour ImRAN et 3 exécutions pour les autres méthodes. La dernière colonne compare les performances entre ImRAN avec et sans bruit appliqué aux données dupliquées.

d'une couleur aléatoire parmi celle des autres classes. La proportion des images biaisées peut ainsi être contrôlée pour créer des variantes du jeu de données. Le fait que le biais ne modifie que la couleur du fond garantit que les caractéristiques prédictives des classes (i.e. les chiffres) soient identiques quel que soit la proportion choisie. De cette façon, la difficulté de la tâche de classification à partir des caractéristiques pertinentes reste la même. C'est l'identification et l'extraction de ces caractéristiques prédictives qui devient plus difficile quand la présence de biais augmente. Par ailleurs, le fait que le biais soit toujours localement au même endroit sur une large surface (i.e., le fond), et qu'il soit composé d'une caractéristique de bas niveau (i.e., la couleur), le rend simple à apprendre et donc dur à éviter.

La proportion des images ayant la couleur de leur fond associée à leur classe est contrôlée par le paramètre ρ. De cette façon, pour $\rho = 1$, toutes les images sont biaisées, alors qu'avec $\rho = 0.1$, les images ne sont pas du tout biaisées par la couleur, étant donné qu'il y a 10 classes. Pour nos expérimentations, nous avons utilisé les valeurs 0.999, 0.997, 0.995, 0.990, et 0.1 pour ρ sur le jeu d'entraînement, valeurs classiquement utilisées dans la littérature [3], à l'exception de 0.1 qui est utilisé ici pour comparer les méthodes sur un jeu de données non-biaisé. Par la suite, les modèles sont évalués sur un jeu de test non-biaisé, mais cependant modifié synthétiquement avec $\rho = 0.1$ pour avoir une couleur de fond autre que du noir.

Enfin, la séparation entre images biaisées et non-biaisées est faite avant de procéder à l'entraînement, et la proportion reste la même au sein de chaque classe, les exemples non-biaisés restant donc bien les mêmes à travers tout l'entraînement.

4.1.2 Hyperparamètres

Afin d'être comparable avec la littérature, dans nos expérimentations, nous avons utilisé la même architecture et les mêmes hyperparamètres que dans ReBias [3]. Cette méthode de l'état de l'art n'utilise pas directement l'information en lien avec les biais présents, ce qui est aussi le cas pour notre méthode, et a historiquement les meilleures performances sur *Biased* MNIST avec $\rho = 0.999$, qui représente la variante la plus difficile testée dans nos expérimentations.

L'architecture que nous avons utilisée dans nos expérimentations est composée d'un réseau de neurones convolutif avec 4 couches de convolutions, avec un noyau de taille 7, et respectivement 16, 32, 64, et 128 canaux. Chacune de ces couches de convolution est suivie d'une couche de *batch normalization* [11], ainsi que d'une fonction d'activation ReLU. La sortie de la dernière couche de convolution passe ensuite dans une couche d'*average pooling* pour produire un vecteur de dimension 128, avant de finir par une couche linéaire complètement connectée avec une taille de sortie à 10. Le classifieur est entraîné pendant 80 époques avec l'optimiseur Adam [13] et un taux d'apprentissage commençant à 0.001 qui est par la suite divisé par 10 toutes les 20 époques. Les images sont normalisées sur les 3 canaux avec 0.5 comme moyenne et 0.5 comme déviation standard. Pour ce qui est des hyperparamètres propres à notre méthode, nous les avons optimisé indépendamment afin d'avoir une bonne performance en un temps raisonnable sur $\rho = 0.999$. Ainsi, on fixe C le coefficient de la moyenne exponentiel mouvante à 0.001. Pour le nombre maximum de duplications K, nous avons choisis d'utiliser une valeur de 1000. Enfin, on fixe σ la déviation standard du bruit ajouté sur les images à 1 par défaut (e.g., Figure 2).

4.2 Résultats

Les résultats visibles dans la Table 4.1 montrent les performances de ImRAN, ainsi que les résultats de quatre autres méthodes dont l'approche standard (Vanilla) qui utilise la même architecture de réseau de neurones mais n'utilise aucun procédé particulier pour limiter l'effet des biais. Les méthodes utilisées pour la comparaison sont LearnedMixin [6], RUBi [5] et ReBias [3].

La première ligne contient les résultats des différentes méthodes pour $\rho = 0.999$, là où le biais est le plus fortement présent. Dans ce contexte, l'approche Vanilla n'obtient que 10.4% d'*accuracy* ce qui est comparable à un choix aléatoire parmi les 10 classes possibles. Cependant, on remarque que ImRAN performe mieux que n'importe quelle autre méthode que ce soit avec ou sans bruit, avec respectivement 44.2% et 33.5% d'*accuracy*. ImRAN double quasiment les performances de *ReBias* qui est la deuxième meilleure méthode pour cette valeur de ρ. Pour $\rho = 0.997$, on remarque que ImRAN est toujours la meilleure méthode avec une marge significative. Cela confirme l'efficacité de notre méthode dans un contexte où le jeu de données est fortement biaisé.

On observe que pour toutes les variantes biaisées du jeu de données ($\rho \neq 0.100$) ImRAN performe toujours mieux que Vanilla, LearnedMixin, et ReBias. De plus, en la présence de biais, ImRAN, ainsi que RUBi, sont les seules méthodes à toujours avoir une *accuracy* supérieure à l'ap-

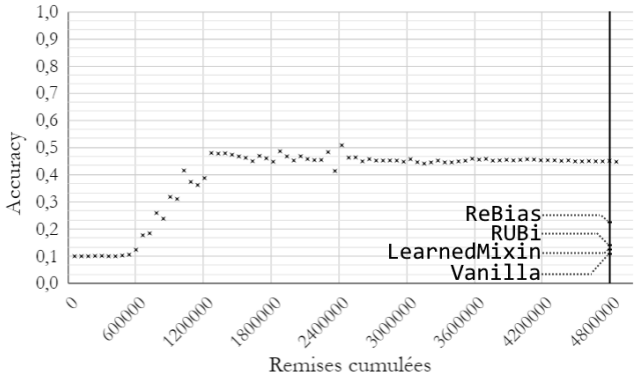

FIGURE 3 – Évolution de l'*accuracy* d'un apprentissage représentatif d'ImRANen fonction du nombre d'exemples parcourus. La barre à 4 800 000 exemples représente 80 époques pour un jeu de données sans remise.

proche Vanilla. La méthode que nous proposons semble donc être un mécanisme efficace pour réduire l'apprentissage de biais lorsqu'il y en a. Néanmoins, sur des valeurs de ρ comme 0.995 et 0.990, qui sont des valeurs relativement faibles, l'approche RUBi obtient les meilleurs résultats, alors que ImRAN se place deuxième. Toutefois, RUBi a de mauvaises performances sur les valeurs élevées de ρ, or la proportion de biais représenté par ρ n'étant généralement pas connu, ImRAN semble être un choix plus versatile.

En parallèle de ces expérimentations, les méthodes ont été testées sur une version non-biaisée (i.e., $\rho = 0.1$) de MNIST, pour cela nous avons utilisé le code source officiel de ReBias qui propose aussi une implémentation de RUBi et LearnedMixin. Le but est de s'assurer que les méthodes ne dégradent pas ou peu les performances de Vanilla lorsque le jeu de données n'a pas de biais. Les résultats, à la dernière ligne de la Table 4.1, montrent une *accuracy* très proche entre les différentes méthodes, à l'exception de LearnedMixin qui performe faiblement. Cela montre que ImRAN ne dégrade quasiment pas les performances lorsque appliquée à un jeu de données sans biais.

Nous avons également testé notre modèle sans utilisation du bruitage des données (i.e., $\sigma = 0$), afin de quantifier l'influence de ce mécanisme dans les résultats. Ces derniers sont reportés à droite de la Table 4.1. On constate une augmentation significative des performances sur les valeurs élevées de ρ lorsque du bruit est ajouté aux augmentations. Pour les valeurs plus modérées de ρ, les résultats avec et sans bruit sont équivalents, voire légèrement meilleurs pour $\rho = 0.1$, ne dégradant ainsi pas du tout les performances comparées à Vanilla. Le bruit peut donc s'avérer légèrement pénalisant pour des données non-biaisées, là où il est fortement utile pour des données très biaisées.

Enfin, le fait d'augmenter la taille du jeu de données dans ImRAN, fait qu'à nombre d'époque égal, notre méthode apprend sur légèrement plus de données. La Figure 3 montre l'évolution de l'*accuracy* en fonction du nombre d'exemples cumulés pour un apprentissage de ImRAN avec pour $\rho = 0.999$. On peut voir que pour un nombre d'exemples parcouru équivalent à 80 époques pour les autres approches, ImRAN a déjà convergé vers sa valeur finale. L'augmentation du nombre d'exemples vus n'a donc pas d'influence dans la hausse des performances obtenues par ImRAN.

5 Discussion et travaux futur

Le principal avantage de notre méthode, ImRAN, est qu'elle ne requiert ni de savoir si un biais est présent ou non, ni de disposer d'informations sur ce dernier comme son type (e.g., texture, couleur, forme) pour bien performer comme montré dans la Section 4.2. Comparée aux autres algorithmes de la littérature, notre méthode se montre très performante, allant jusqu'à presque doubler les performances de l'état de l'art dans les cas les plus extrêmes, et égaler les autres approches pour des valeurs de ρ plus faibles, à l'exception de RUBi. De surcroît, elle semble rester relativement efficace aussi bien dans ces cas extrêmes que sur des jeux de données à la proportion de biais plus modérée. Comme nous l'avons vu mentionné dans l'introduction, satisfaire ces deux cas est important puisque dans la plupart des jeux de données réels, il n'est pas possible de connaître à l'avance la présence de biais, ni dans quelles proportions. Cependant, ImRAN possède un léger défaut, le temps d'apprentissage est légèrement augmenté (entre 2% à 8% pour les variantes biaisées du jeu de données). Cela est dû à la duplication des éléments, et donc l'augmentation de la taille du jeu de données.

Ainsi, nos travaux futurs se focaliseront sur plusieurs points. Tout d'abord, l'expérimentation de notre approche sur plus de jeux de données afin de valider que la méthode est également efficace dans d'autres cas. En particulier avec des données plus réalistes et des biais plus complexes, comme par exemple avec le jeu de données bFFHQ qui contient des visages de personnes appartenant aux catégories jeune et âgée sachant que le genre de ces personnes est fortement corrélé avec leur tranche d'âge. Ensuite, la mise en place d'un mécanisme estimant la valeur de ρ pour ajuster la sélection des hyperparamètres, et notamment vis-à-vis du bruit. Enfin, nous souhaitons explorer des alternatives au bruit gaussien dans l'objectif de trouver une méthode générant une meilleure variance, et particulièrement au sein des exemples dupliqués.

6 Conclusion

Dans ce papier, nous avons présenté une version préliminaire de la méthode ImRAN, qui modifie activement la distribution de ses données d'apprentissage pour limiter l'apprentissage de représentations biaisées aussi bien sur des jeux de données peu ou grandement biaisés. Nous avons ensuite comparé ses résultats aux approches actuelles, puis discuté des résultats obtenus, ainsi que des potentielles pistes d'amélioration.

Remerciements

Cette recherche est supportée par le fond national de recherche luxembourgeois (FNR) :

IPBG2020/IS/14839977/C21, et a été réalisé à l'aide de GPU offerts par la société NVIDIA. Nous remercions sincèrement ces soutiens.

Références

[1] Vedika Agarwal, Rakshith Shetty, and Mario Fritz. Towards causal vqa : Revealing and reducing spurious correlations by invariant and covariant semantic editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9690–9698, 2020.

[2] Aishwarya Agrawal, Dhruv Batra, and Devi Parikh. Analyzing the behavior of visual question answering models. *arXiv preprint arXiv :1606.07356*, 2016.

[3] Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. Learning de-biased representations with biased representations. In *International Conference on Machine Learning*, pages 528–539. PMLR, 2020.

[4] Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. Learning de-biased representations with biased representations. In *International Conference on Machine Learning*, pages 528–539. PMLR, 2020.

[5] Remi Cadene, Corentin Dancette, Matthieu Cord, and Devi Parikh. Rubi : Reducing unimodal biases for visual question answering. *Advances in neural information processing systems*, 32, 2019.

[6] Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. Don't take the easy way out : Ensemble based methods for avoiding known dataset biases. *arXiv preprint arXiv :1909.03683*, 2019.

[7] Luke Darlow, Stanis\law Jastrzębski, and Amos Storkey. Latent adversarial debiasing : Mitigating collider bias in deep neural networks. *arXiv preprint arXiv :2011.11486*, 2020.

[8] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11) :665–673, 2020. Publisher : Nature Publishing Group.

[9] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. ImageNet-trained CNNs are biased towards texture ; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv :1811.12231*, 2018.

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

[11] Sergey Ioffe and Christian Szegedy. Batch normalization : Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[12] Eungyeup Kim, Jihyeon Lee, and Jaegul Choo. Biaswap : Removing dataset bias with bias-tailored swapping augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14992–15001, 2021.

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

[14] Yann LeCun. The mnist database of handwritten digits. *http ://yann. lecun. com/exdb/mnist/*, 1998.

[15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553) :436–444, 2015.

[16] Jungsoo Lee, Eungyeup Kim, Juyoung Lee, Jihyeon Lee, and Jaegul Choo. Learning debiased representation via disentangled feature augmentation. *Advances in Neural Information Processing Systems*, 34 :25123–25133, 2021.

[17] Yi Li and Nuno Vasconcelos. Repair : Removing representation bias by dataset resampling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9572–9581, 2019.

[18] Matthias Minderer, Olivier Bachem, Neil Houlsby, and Michael Tschannen. Automatic shortcut removal for self-supervised representation learning. In *International Conference on Machine Learning*, pages 6927–6937. PMLR, 2020.

[19] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure : Debiasing classifier from biased classifier. *Advances in Neural Information Processing Systems*, 33 :20673–20684, 2020.

[20] Rakshith Shetty, Bernt Schiele, and Mario Fritz. Not Using the Car to See the Sidewalk–Quantifying and Controlling the Effects of Context in Classification and Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8218–8226, 2019.

[21] Enzo Tartaglione, Carlo Alberto Barbano, and Marco Grangetto. End : Entangling and disentangling deep representations for bias correction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13508–13517, 2021.

[22] Haohan Wang, Zexue He, Zachary C. Lipton, and Eric P. Xing. Learning robust representations by projecting superficial statistics out. *arXiv preprint arXiv :1903.06256*, 2019.

[23] Tianlu Wang, Jieyu Zhao, Mark Yatskar, Kai-Wei Chang, and Vicente Ordonez. Balanced datasets are not enough : Estimating and mitigating gender bias in deep image representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5310–5319, 2019.

[24] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.