RISC Lab

2ème Edition de la journée des doctorants
07 MARS 2018
Thème : Recherche et développement : contribution du chercheur en sciences de l'ingénieur

Stéganalyse d'images via un réseau de neurones convolutifs

Rabii Elbeji , Marwa Saidi , Houcemeddine Hermassi , Rhouma Rhouma

Université de Tunis El Manar, Ecole Nationale d'Ingénieurs de Tunis, Unité de recherche, BP 37, le Belvédère, 1002 Tunis, Tunisie

Contexte

La stéganalyse d'images représente l'ensemble des techniques et des méthodes utilisées pour détecter la présence d'information incorporée dans l'image et l'extraire si cela est possible.
Au cours des dernières années, avec le progrès des techniques d'apprentissage automatique (machine learning) et des modèles de réseaux de neurones (deep learning) dans le domaine de l'imagerie, la stéganalyse de l'image suit cette évolution.
Depuis l'année 2015, plusieurs architectures de réseaux neuronaux convolutifs RNC sont apparues. Ces architectures, inspirées du modèle de *Qian et al* [1], sont généralement complexes et coûteuses en termes de calcul et de temps d'apprentissage ou de formation.

Objectif

Les différentes architectures RNC proposées récemment en stéganalyse d'image sont implémentées sur des cartes graphiques (GPUs) très performantes. Notre objectif était de concevoir une nouvelle architecture simple (minimale en termes de nombre de couches convolutives, de nombre de neurones et de taille d'image d'entrée), et de l'implémenter sur des processeurs de performance faible ou moyenne.

Notre méthode

Nous avons construit une architecture RCN simple illustrée dans la *Figure 1*. L'architecture contient trois parties principales. Tout d'abord, un prétraitement est appliqué, par quatre filtres (voir Figure 2) pour diminuer la similarité entre le deux images (*cover* et *stego*) par l'extraction de bruit résiduel [2]. Ensuite, deux couches convolutives sont appliqué. Les poids ou le filtres de ce deux couches sont mis a jour après chaque itération par rétropropagation. Le nouveaux filtres sont calculés par un algorithme d'optimisation spécifié en se basant sur le taux d'erreur d'une fonction de perte donnée. Nous avons choisi l'entropie croisée binaire (*binary cross entropy*) comme fonction de perte car nous avons entrain de faire une classification binaire et *RMSProp Optimizer* [3] qui est un optimiseur très populaire dans la communauté de *deep learning* connu par sa capacité de bien gérer les objectifs stochastiques, ce qui le rend applicable à l'apprentissage en mini-lot (*mini batch size*).

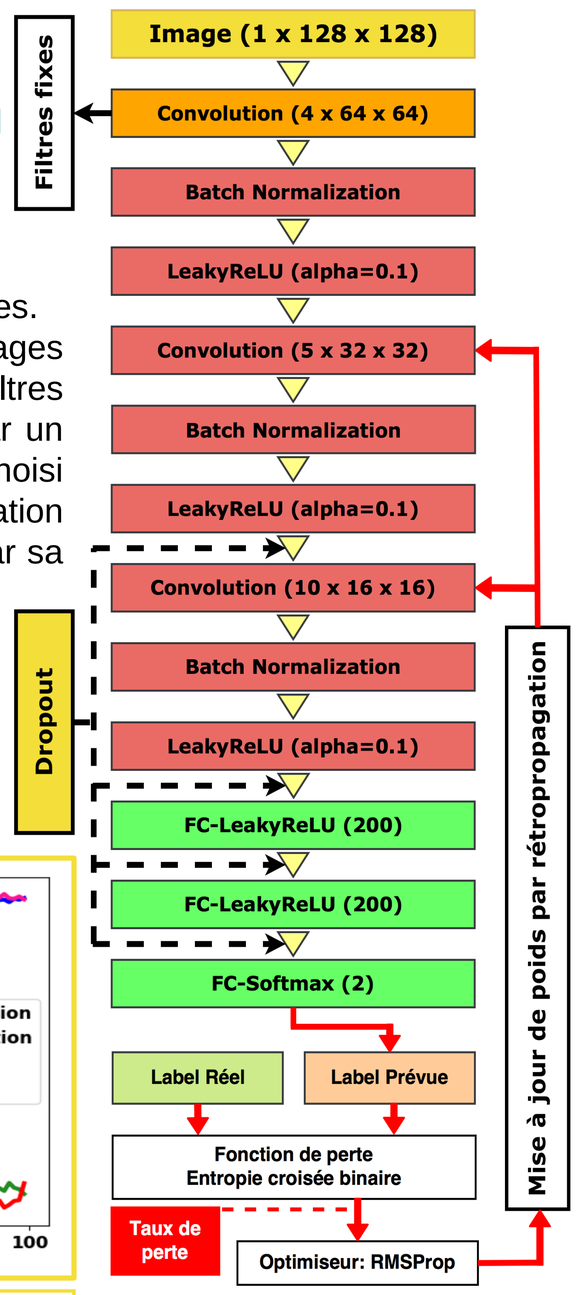


Figure.1: Architecture RNC proposée

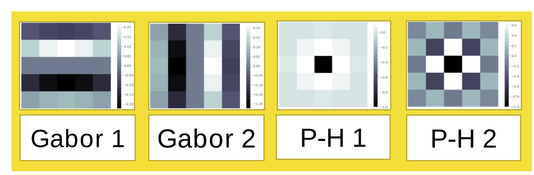


Figure.2: 4 filtres (Gabor et passe-haut (P-H) fixes) pour le prétraitement

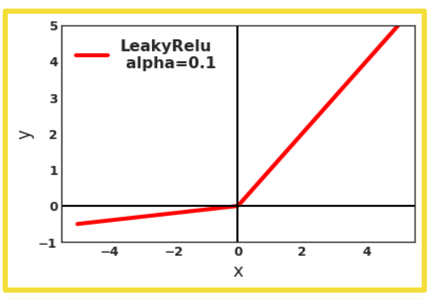


Figure.3: Tracé de la fonction LeakyReLU

Chaque convolution est suivie d'une fonction d'activation (*LeakyReLu* dans notre cas, voir Figure 3). En générale, le rôle d'une fonction d'activation dans un réseau de neurones est de produire une limite de décision non linéaire via des combinaisons des entrées pondérées. Nous avons utilisé des techniques telles que *Batch Normalization*, *Dropout*, et *data augmentation* pour minimiser le risque de surestimation durant l'apprentissage.

Pour une capacité d'insertion (CI) élevée de 1bpp (bit par pixel), l'apprentissage est effectué sur 100 epoch afin d'atteindre le maximum de précision comme le montre la Figure 4.

Alors que pour les charges inférieures à 1bpp, nous utilisons la technique de transfert de connaissances (*Transfer Learning*) pour accélérer la convergence.

Enfin une classification binaire est effectuée à travers trois couches entièrement connectées (*Fully-Connected*).

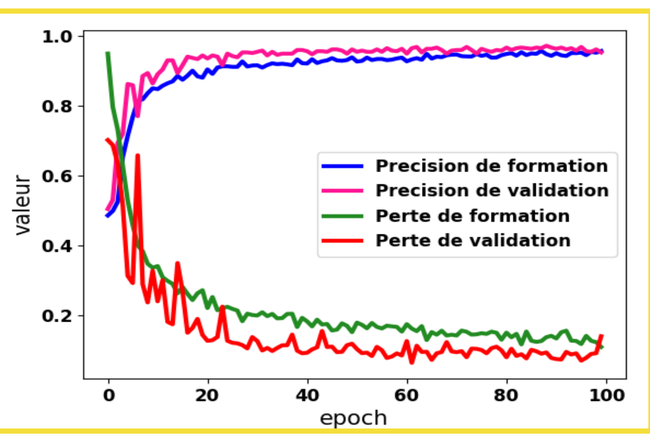


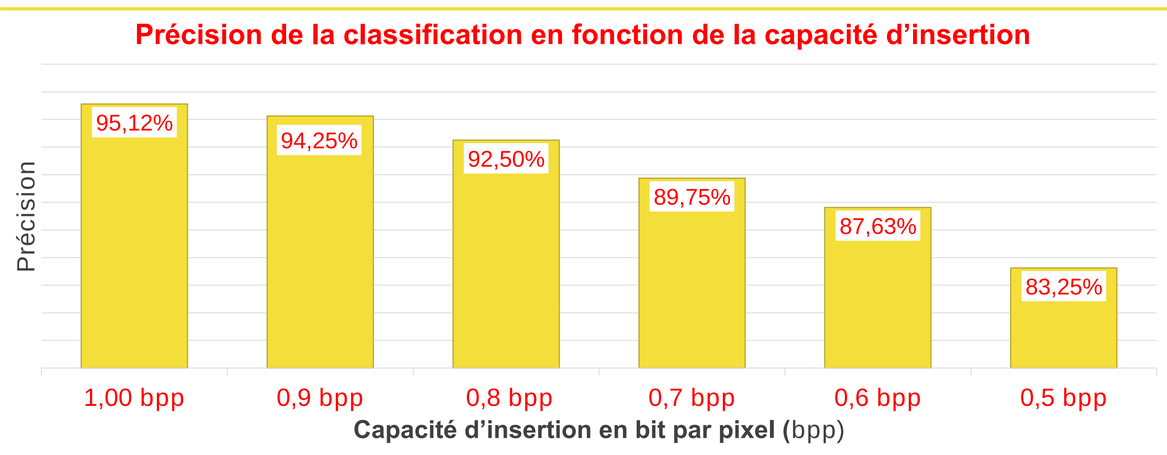
Figure.4: Précision et perte durant 100 epoch **d'apprentissage (pour CI = 1bpp)**

Résultats

Pour évaluer notre architecture RNC, nous avons utilisé une partie (2400 images) d'une base de données publique Bossbase [4]. Les nombres de paires d'échantillons (images de couverture et images stego) pour l'apprentissage, la validation et les tests sont présentés dans le tableau suivant. Les images stego sont générées par l'algorithme de WOW *(Wavelet Obtained Weights)* [5].

	Formation	Validation	Test
Nombre de paires d'images	1700	300	400

Les résultats des tests de la précision de la classification en fonction de la capacité d'insertion sont indiqués dans la figure ci-dessous.



Conclusion

Nous pouvons conclure que, malgré la simplicité de notre modèle et la qualité moyenne des processeurs, les résultats de la précision de la classification sont importants, en particulier pour les capacités d'insertion élevées. Ces résultats sont motivants pour créer des architectures plus complexes avec d'autres modèles de réseaux de neurones vue le succès de l'apprentissage en profondeur (*deep learning*) aujourd'hui.

Références

[1] Qian, Yinlong, Jing Dong, Wei Wang, and Tieniu Tan. "Deep learning for steganalysis via convolutional neural networks." Media Watermarking, Security, and Forensics 9409 (2015): 94090J-94090J.
[2] Chen Mo, Vahid Sedighi, Mehdi Boroumand, and Jessica Fridrich. "JPEG-phaseaware convolutional neural network for steganalysis of JPEG images." In Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, pp. 75-84. ACM, 2017
[3] Tieleman, T., and G. Hinton. Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning. Technical Report.
[4] Bas, Patrick, Toms Filler, and Toms Pevn. "Break Our Steganographic System: The Ins and Outs of Organizing BOSS." In International Workshop on Information Hiding, pp. 59-70. Springer, Berlin, Heidelberg, 2011.
[5] Pevn, Toms, Toms Filler, and Patrick Bas. "Using high-dimensional image models to perform highly undetectable steganography." In International Workshop on Information Hiding, pp. 161-177. Springer, Berlin, Heidelberg, 2010.