Stéganalyse groupée en JPEG : comment gérer la stratégie d'étalement?

Ahmad ZAKARIA[1], Marc CHAUMONT[1,2], Gérard SUBSOL[1]

[1]LIRMM, Univ Montpellier, CNRS, 161 rue Ada 34095 Montpellier Cedex 5 - France

[2]Univ Nîmes, Place Gabriel Péri 30000 Nîmes Cedex 01 - France

ahmad.zakaria@lirmm.fr, marc.chaumont@lirmm.fr, gerard.subsol@lirmm.fr

Résumé – Dans la Stéganalyse Groupée d'images (SG), une stéganalyste (Eve) vise à détecter si un ensemble d'images, envoyées par une stéganographe (Alice) à un pair (Bob) via un réseau, contient un message caché. Nous pouvons raisonnablement supposer qu'Eve ne connaît pas la stratégie exacte utilisée pour insérer et étaler le message à travers les images, mais elle devine qu'Alice et Bob utilisent un certain algorithme stéganographique. Au meilleur de notre connaissance, dans ce cas, la solution la plus appropriée pour la SG est d'utiliser un détecteur d'image unique (SID) pour estimer si une image est stéganographiée ou non et de faire la moyenne des scores obtenus sur un ensemble d'images. Et si Eve pouvait prendre en compte la stratégie d'étalement? Pourrait-elle alors utiliser un algorithme de SG meilleur que la moyenne des scores? Dans cet article, nous proposons une architecture générale de SG prenant en compte différentes stratégies d'étalement. Les résultats expérimentaux obtenus avec six stratégies et un détecteur d'image unique montrent que, si Eve discrimine la stratégie d'étalement, elle peut améliorer la précision de la SG.

Abstract – In image Pooled Steganalysis (PS), a steganalyst (Eve) aims to detect if a set of images sent by a steganographer (Alice) to a peer (Bob) through a network contains a hidden message. We can reasonably assess that Eve does not know the strategy used to spread the payload across images, but she guesses that Alice and Bob use a certain steganography algorithm. To the best of our knowledge, in this case, the most appropriate solution for PS is to use a *Single-Image Detector* (SID) to estimate if an image is steganographied or not and to average the scores obtained on a bag of images. What now if Eve could discriminate the spreading strategy? Could she use a PS algorithm better than averaging the scores? In this paper, we examine how it is possible to discriminate the spreading strategy. Empirical results made with six different embedding strategies and a state-of-the-art SID show that, if Eve discriminate the spreading strategy, she can improve the accuracy of the PS.

1 Introduction

La *Steganographie* consiste à modifier un objet numérique (appelé *cover*), d'une manière anodine, pour cacher un message. L'objet modifié résultant est appelé *stego*. La science de détection de la présence du message caché, étant donné un objet, est appelée *Stéganalyse*. Dans cet article, nous nous concentrons sur la stéganalyse des images numériques. Plus précisément, nous utilisons des images codées en JPEG.

La stéganographie se concentre traditionnellement sur l'insertion d'un message dans une seule image à la fois, mais il est beaucoup plus réaliste pour Alice de cacher le message en l'étalant sur plusieurs images. C'est ce qu'on appelle la *Stéganographie par lots* qui peut être opposée à la *Stéganalyse groupée* (SG) où un ensemble d'images est analysé afin de conclure à la présence d'un message ou non.

La stéganographie par lots et la SG ont été introduites dans [1] et sont devenues l'un des problèmes ouverts les plus difficiles dans ce domaine ces dernières années [2].

Comme présenté dans la Fig. 1, la stéganographie par lots consiste à étaler le message m dans un ensemble (appelé *sac*) d'images *cover* en utilisant la stratégie d'étalement s. Dans [3, 4, 5], les chercheurs ont comparé l'efficacité de certaines stratégies d'étalement et ils montrent que la stratégie optimale consiste à concentrer le message sur le moins d'images *cover* possible ou, au contraire, à étaler le message aussi finement que possible dans toutes les images. D'autres stratégies pratiques peuvent être trouvées dans [6, 7].

Comme présenté dans la Fig. 1, étant donné un ensemble d'images $\mathcal{X}$, Eve prend un sac B de b images $\{x_1, ..., x_b\} \subset \mathcal{X}$ qui peuvent être *cover* ou *stego*, applique pour chaque image la fonction f du *Single-Image Detector* (SID) afin d'obtenir les scores $\{f(x_1), ..., f(x_b)\}$ et agrège ensuite ces scores en utilisant une fonction de pooling $g : R^b \mapsto R$ afin d'obtenir une seule sortie qui permet de classer le sac comme "*cover*" ou "*stego*".

Dans [1], les sorties d'un SID ont été combinées et il est montré que la fonction de pooling optimale g dépend de la stratégie utilisée par Alice pour étaler un message entre plusieurs objets.

Dans [8], il est montré que lorsque Eve possède peu ou pas d'informations sur les communications sécurisées, la solution la plus appropriée est de faire la moyenne des scores obtenus sur l'ensemble des images steganalysées. De [6, 7], nous concluons que les meilleurs résultats pour la SG sont obtenus dans le cadre du scénario *Clairvoyant* où la stratégie d'étalement

est considérée être connue.

Une préoccupation importante est alors de savoir s'il est possible de prendre en compte les stratégies d'étalement possibles afin d'obtenir des résultats plus proches du scénario Clairvoyant.

Pour ce faire, cet article décrit d'abord une liste de stratégies d'étalement de l'état de l'art en § 2 et propose ensuite une architecture générale de SG en § 3 qui permet de réaliser des expériences en § 4. Les résultats et la conclusion sont présentés dans § 5.

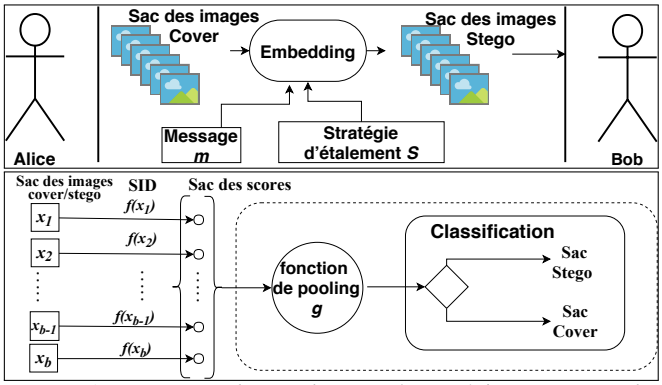

FIGURE 1. *Stéganographie par lots* avec la stratégie s, et *Stéganalyse groupée* avec la fonction de pooling g.

2 État de l'art

2.1 Stratégies d'étalement par lots

Nous avons compilé les stratégies d'étalement de lots proposées dans [6, 7] dans la liste suivante :

1. **Greedy :** Alice incorpore le message de charge utile α, dans aussi peu de *covers* que possible. Alice choisit au hasard une image *cover* et insère une partie de son message jusqu'à 1 bpc [1]. Elle répète le processus avec une autre image *cover* choisie au hasard, jusqu'à ce que le message entier soit inséré.
2. **Linéaire :** Alice distribue le message uniformément sur toutes les images *cover* disponibles.
3. **Uses-β :** Alice distribue le message uniformément sur une fraction β d'images *cover* disponibles. Dans cet article, β est fixé à 0.5 i.e. 50% des images dans un sac sont stegos.
4. **IMS :** pour un sac B d'images de taille b, Alice fusionne les b images en une seule et laisse l'algorithme d'insertion répartir le message dans cette grosse image.
5. **DeLS :** Alice répartit le message sur les images tel que chaque image du sac contribue avec la même divergence de Kullback-Leibler (KL) (coefficient de déviation) basée sur le schéma d'insertion MiPOD [9].
6. **DiLS :** Alice répartit le message afin que chaque image du sac contribue avec la même valeur de distorsion.

1. bpc signifie bits par coefficients, car nous utilisons à la fois des coefficients de haute fréquence AC et DC pour insérer le message. Pour le sac d'images, la taille du message est exprimée en bits per total coefficients (bptc).

2.2 SG et fonction de pooling

La fonction de pooling g agrège les scores $f(x_i)$ donnés par le SID. En supposant que $f : \mathcal{X} \mapsto R$ est un détecteur de stéganalyse quantitative non biaisé, ce qui signifie que le détecteur estime la longueur du message caché, il est prouvé dans [1], avec des algorithmes d'insertion non adaptatifs, que la fonction moyenne $g_{mean} = \frac{1}{b}\sum_{i=1}^{b} f(x_i)$, est optimale lorsqu'Alice utilise la stratégie $Linéaire$, et que la fonction maximale $g_{max} = \max_{i\in\{1,\dots,b\}} f(x_i)$, est optimale quand Alice applique la stratégie $Greedy$. Les hypothèses ci-dessus permettent de trouver analytiquement des stratégies optimales mais, en fait, elles ne sont guère satisfaites dans la pratique, car il existe bien d'autres stratégies d'étalement pour lesquelles aucune fonction optimale de pooling n'a été proposée.

Une autre fonction de pooling a été étudiée dans [7] où l'auteur utilise une approche d'apprentissage machine pour montrer, pour un stéganalyste connaissant la stratégie d'étalement, comment apprendre une fonction optimale combinant les sorties d'un SID sur un sac d'objets.

En outre, dans [6], l'auteur a étudié le cas de la SG pour un stéganalyste omniscient (c'est-à-dire qu'Eve connaît l'algorithme d'insertion, la distribution des *cover*/*stego* dans la base de test, la taille de la charge utile d'une image, la taille du sac, la taille des images et la stratégie d'étalement).

Nous concluons de tous ces résultats qu'il est important de connaître la stratégie d'étalement, car les meilleures fonctions de pooling dans toutes les approches sont pour une stratégie d'étalement donnée et connue. Une solution consisterait donc à créer une fonction de pooling optimale pour chaque stratégie d'étalement existante. Mais il faudrait pour cela être en mesure d'estimer d'abord la stratégie utilisée par Eve.

Une autre solution consiste à définir un système de SG, intégrant une fonction de pooling, qui serait à peu près générale, c'est-à-dire qui pourrait prendre en charge toutes les stratégies d'étalement.

3 Une architecture générale de SG

3.1 Description de l'architecture

Comme il existe de nombreuses stratégies d'étalement pour lesquelles aucune fonction de pooling optimale n'a été proposée, l'idée est d'apprendre la fonction de pooling g à partir d'une base de données d'entraînement d'images dont les messages ont été insérés avec différentes stratégies d'étalement.

La fonction SID f étant fixée, une image que Eve observe peut être représentée par un seul nombre réel qui est le score $f(x)$ donné par le SID, et les sacs d'images deviennent alors un vecteur de nombres réels $\{f(x_1), ..., f(x_b)\}$. Nous avons choisi comme SID l'algorithme de steganalyse quantitative proposé dans [10]. Celui-ci se fonde sur une architecture de régression par apprentissage machine qui assemble, via un processus de gradient boosting, un grand nombre de module d'apprentissage de base, plus simples, construits sur des sous-espaces

aléatoires de l'espace original qui est de dimension élevée. Chaque module de base est un arbre de régression adapté pour refléter la nature spécifique des espaces de caractéristiques d'images de haute dimension utilisés en stéganalyse. Le SID est appliqué sur les caractéristiques de Gabor résiduelles (GFR) [11].

Une difficulté est que le nombre d'images dans chaque sac peut être différent. Nous utilisons la méthode décrite dans [7] pour résoudre ce problème par un ré-échantillonnage uniforme basé sur les fenêtres de Parzen. Le sac, c'est-à-dire le vecteur $z = \{f(x_1), ..., f(x_b)\}$, est transformé en un histogramme grâce à l'estimation par fenêtres de Parzen. Etant donné une fonction du noyau $k : R \times R \mapsto R$ (par exemple un noyau gaussien, $k(x; y) = exp(-\gamma||x - y||^2)$), pour un sac z, l'histogramme résultant est :

$h = [\frac{1}{b}\sum_{f(x_i)\in z} k(f(x_i), c_1),\ \ldots,\ \frac{1}{b}\sum_{f(x_i)\in z} k(f(x_i), c_p)]$

avec $\{c_i\}_{i=1}^{p}$ un ensemble de points régulièrement espacés dans l'interval réel délimité par $min_{x\in\mathcal{X}} f(x)$ et $max_{x\in\mathcal{X}} f(x)$. Notez que cet histogramme est indépendant de l'ordre des images dans le sac.

Une fois la fenêtre de Parzen appliquée, le vecteur h de dimension fixe p est donné à un classifieur SVM qui agrège le composantes de h dans l'espace de redescription, pour effectuer la classification $\Delta = \sum_{i=1}^{p} \omega_i\ \phi(h[i])$, avec ϕ, la fonction de redescription.

Nous pouvons voir en regardant Δ, que la fonction de pooling est une somme pondérée où les poids ω_i sont appris pendant l'entraînement SVM. Il est clairement plus subtil d'agréger l'ensemble des caractéristiques $\{\phi(h[i])\}_{i=1}^{p}$ du sac que d'utiliser une fonction simple comme g_{mean} ou g_{max}.

En utilisant l'histogramme de Parzen, l'espace de redescription SVM et la somme pondérée, nous espérons que lorsque notre architecture générale apprendra avec différentes stratégies d'étalement, elle sera capable de s'adapter à la stratégie d'étalement et, d'une certaine mesure, prédira la stratégie par la somme pondérée (au lieu d'une simple somme).

3.2 Apprentissage des stratégies

Le but principal de ce travail est d'étudier l'efficacité de la " discrimination " d'une stratégie s parmi un ensemble de stratégies S. Pour chaque stratégie s, nous classons les histogrammes de Parzen h en utilisant g_{clair} dans le scénario clairvoyant, g_{disc} dans le scénario discriminatif, g_{mean} et g_{max}. Cette évaluation nécessite les scénarios d'apprentissage spécifiques suivants :

1. g_{disc} **:** on entraîne la fonction de pooling g_{disc} sur les vecteurs h de toutes les stratégies de S, et on la teste sur le vecteur h de la stratégie s. g_{disc} apprendra donc les motifs des stratégies et essaiera de discriminer s pour améliorer la classification.
2. g_{max} **:** on choisit les vecteurs h de toutes les stratégies puis on calcule le maximum des scores pour chacune. On calcule le seuil τ_{max} qui minimise la probabilité d'erreur de classification totale $P_e = \frac{1}{2}(P_{fa} + P_{md})$, où P_{fa} et P_{md} sont la fausse alarme et la probabilité de détection manquée. Enfin, nous testons le vecteur h de s en seuilant leur score maximum avec τ_{max}.
3. g_{clair} **:** on entraîne et teste les vecteurs h à partir de la même stratégie s, en utilisant la fonction de pooling g_{disc} qui aura une connaissance parfaite sur la stratrégie s puisque l'entraînement est fait uniquement dans ce cas.
4. g_{mean} **:** on choisit les vecteurs h de toutes les stratégies puis on calcule la moyenne des scores pour chacune. On calcule le seuil τ_{mean} qui minimise P_e. Enfin, on teste les vecteurs h de s en seuilant leur score moyen avec τ_{mean}.

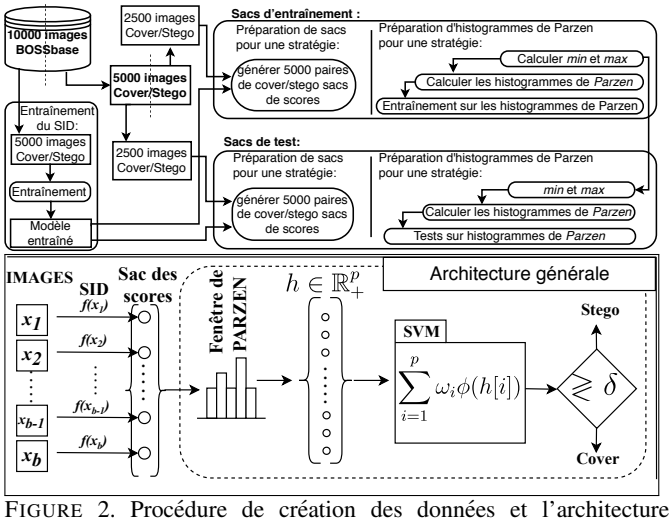

FIGURE 2. Procédure de création des données et l'architecture générale de Stéganalyse Groupée.

4 Évaluation expérimentale

Dans cette section, g_{disc} est comparé à g_{mean}, g_{max} et g_{clair}. La comparaison se fait sur 10 000 images de la base BOSSbase 1.01 [12], converties en JPEG avec un facteur de qualité de 75.

Après la préparation et le prétraitement des données, le processus d'apprentissage se fait en deux étapes. Tout d'abord, les paramètres du SID sont appris pour différentes tailles de message et ensuite, la fonction g_{disc} est apprise sur des sacs d'images pour toutes les stratégies.

Les différentes hypothèses pour la SG sont de savoir si Eve connaît ou non l'algorithme d'insertion, la distribution des images de test, la taille de la charge utile de chaque image, la taille de la charge utile du sac, la taille du sac, la taille des images et la stratégie d'étalement.

L'architecture proposée peut être utilisée indépendamment de toutes ces hypothèses, à l'exception des problèmes de *cover source-mismatch* et *stego source-mismatch* que les SID ne sont pas en mesure aujourd'hui de résoudre. Nous faisons les expériences sur des images de taille fixe, avec le même schéma d'insertion (J-UNIWARD) [13] et nous faisons l'hypothèse raisonnable qu'avec le temps, Alice maintient une charge utile moyenne communiqué $\bar{R}$ = 0,1 bptc.

La fonction de pooling apprise g_{disc} utilise des vecteurs h, avec $p = 100$ centres régulièrement espacés dans la plage $[min_{x\in\mathcal{X}} f(x), max_{x\in\mathcal{X}} f(x)]$, où $min_{x\in\mathcal{X}} f(x), max_{x\in\mathcal{X}} f(x)$

ont été calculés sur les scores des images *cover* et *stego* sur la base d'apprentissage. Pour entraîner g_{disc}, nous utilisons le package SVM de la bibliothèque *Scikit-Learn*, avec *kernel='linear'*. Les autres paramètres sont ceux par défaut.

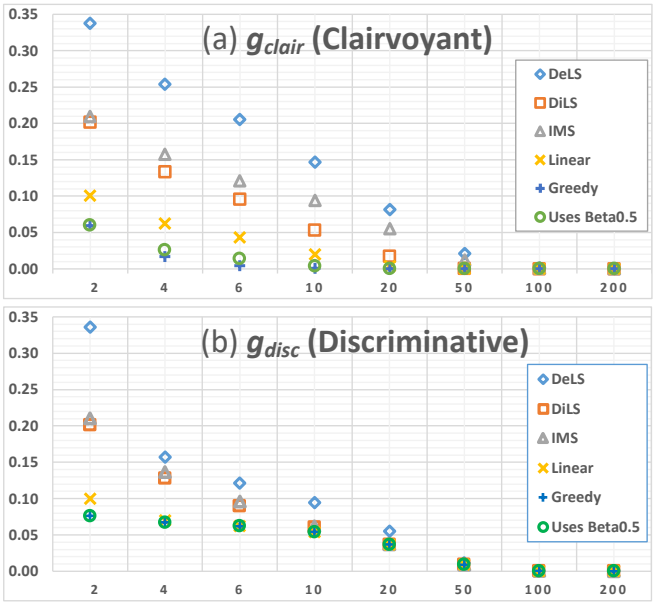

FIGURE 3. Comparaison des stratégies d'étalement ; Pe pour les fonctions de pooling g_{clair} et g_{disc} sur toutes les stratégies.

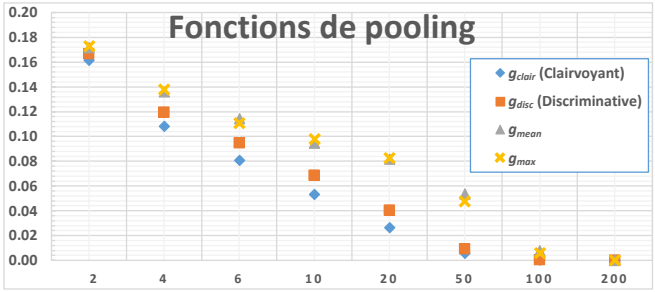

FIGURE 4. Comparaison des fonctions de pooling : moyenne de Pe sur toutes les stratégies pour chaque fonction de pooling.

5 Résultats et conclusion

Nous entraînons le SID sur les caractéristiques GFR pour 5 000 images de BOSSbase 1.01 compressées en JPEG avec un facteur de qualité 75 et inserées en utilisant J-UNIWARD. Les 5 000 images restantes sont utilisées pour créer des sacs de taille $b \in \mathcal{B} = \{2, 4, 6, 10, 20, 50, 100, 200\}$ insérés aussi en utilisant J-UNIWARD avec une stratégie $s \in S$, pour générer un vecteur de taille fixe $h \in R_+^p$, avec $\bar{R} = 0.1$ bptc.

Dans toutes les expériences, Eve connaît $\bar{R}$, b et le schéma d'insertion. Dans le cas *Clairvoyant*, Eve connaît aussi s, donc elle peut entraîner g avec la stratégie utilisée par Alice. g_{disc} et g_{clair} sont entraînées avec un SVM, tandis que pour g_{mean} et g_{max} on entraîne un seuil qui minimise Pe.

À la Fig. 3 (a), nous comparons les stratégies d'étalement. Dans les expériences clairvoyantes, on peut remarquer que - $DeLS$ est la meilleure stratégie alors que $Greedy$ est la pire. On peut regrouper les stratégies en 3 groupes. $Greedy$ et $Uses\text{-}\beta$ sont hautement détectables, leurs détectabilités commencent à coïncider pour les tailles de sacs ≥ 10 avec $Pe \approx 0$. $DeLS$, IMS et $DiLS$ sont bien plus sûres que $Greedy$ et $Uses\text{-}\beta$, tandis que $Linear$ se situe entre les deux groupes. $DeLS$ et IMS restent résistantes jusqu'à $b = 100$ où leurs probabilités d'erreur Pe commencent à coïncider et deviennent presque nulle.

À la Fig. 3 (b), nous rapportons la détection de chaque stratégie avec la fonction g_{disc}. Les meilleures stratégies sont, dans l'ordre décroissant c.à.d. dans le sens de la capacité à résister à la fonction de pooling qui tente de la discriminer : $DeLS$, IMS, $DiLS$, $Linear$, $Uses\text{-}\beta$, $Greedy$. Dans le cas de l'approche non-clairvoyante, $DeLS$ se comporte encore bien et demeure résistante jusqu'à $b = 100$ où Pe commence à coïncider avec celles des autres stratégies et devient ≈ 0. Un résultat étonnant est que la Pe de $Uses\text{-}\beta$ et de $Greedy$ coïncident $\forall\, b \in \mathcal{B}$.

La Fig. 4 fournit un aperçu utile sur la précision des méthodes de pooling. Cette figure montre la Pe moyenne en fonction de la taille du sac $b \in \mathcal{B}$, pour chaque fonction de pooling sur toutes les stratégies. Nous notons que g_{disc} surpasse g_{mean} et g_{max} avec une différence moyenne de $Pe \approx 2\%$ et est plus proche de g_{clair} avec une différence moyenne de $Pe \approx 0.8\%$.

Nous avons ainsi étudié la capacité d'Eve à discriminer la stratégie d'étalement. Les résultats empiriques obtenus avec 6 stratégies d'étalement différentes et un SID de l'état de l'art montrent que, si Eve discrimine la stratégie d'étalement, elle peut améliorer la précision de la Stéganalyse Groupée.

Références

[1] Ker. Batch steganography and pooled steganalysis. 2006.

[2] Ker et al. Moving steganography and steganalysis from the laboratory into the real world. 2013.

[3] Ker. Batch steganography and the threshold game. 2007.

[4] Ker. Perturbation hiding and the batch steganography problem. 2008.

[5] Ker Pevný. Batch steganography in the real world. 2012.

[6] Cogranne et al. Practical strategies for content-adaptive batch steganography and pooled steganalysis. 2017.

[7] Pevný Nikolaev. Optimizing pooling function for pooled steganalysis. 2015.

[8] Cogranne. A sequential method for online steganalysis. 2015.

[9] Sedighi et al. Content-adaptive steganography by minimizing statistical detectability. 2016.

[10] Kodovský Fridrich. Quantitative steganalysis using rich models. 2013.

[11] Song et al. Steganalysis of Adaptive JPEG Steganography Using 2D Gabor Filters. 2015.

[12] Bas et al. "break our steganographic system" : The ins and outs of organizing BOSS. 2011.

[13] Holub et al. Universal distortion function for steganography in an arbitrary domain. 2014.