Stéganographie Adaptative par Oracle (ASO)

Sarra Kouider[2] Marc Chaumont[1,2] William Puech[2]

[1] UNIVERSITE DE NIMES, F-30021 Nîmes Cedex 1, France

[2]LIRMM, UMR 5506, CNRS, Université de Montpellier II,
161 rue Ada, 34095 Montpellier Cedex 05, France

{sarra.kouider, marc.chaumont, william.puech}@lirmm.fr

Résumé

Les deux algorithmes les plus sûrs en 2011 sont les algorithmes HUGO [1] et MOD[2]. Dans la continuité de ces travaux, nous proposons un algorithme de stéganographie adaptative par oracle (ASO : Adaptive Steganography by Oracle). Pour cela, nous définissons la carte de détectabilité en exploitant les connaissances d'un oracle construit à partir de l'ensemble de classifieurs de Kodovský [3]. Lors de l'insertion du message dans l'image, l'oracle permet d'obtenir une carte de détectabilité qui prend en compte à la fois la distribution de l'image de couverture et à la fois la distribution de la base d'images de l'émetteur. Par ailleurs, notre approche permet de définir un nouveau paradigme en stéganographie : la stéganographie par base. Les résultats expérimentaux montrent que notre approche présente de bonnes performances en terme de sécurité puisque la probabilité de détection de présence d'un message, obtenue par un stéganalyseur, est beaucoup plus forte pour l'approche paramétrée HUGO que pour notre approche non-paramétrée ASO.

Mots clefs

Stéganographie, Stéganalyse, Minimisation d'impact d'insertion, Carte de détectabilité, Ensemble de classifieurs.

1 Introduction

La stéganographie est l'art de communication secrète. L'objectif est de dissimuler une information secrète dans un médium anodin sans qu'elle ne puisse être détectée. Avec la généralisation d'Internet, plusieurs philosophies de conception de schéma stéganographique ont été proposées. Parmi les méthodes actuelles appliquées aux images numériques complexes [1], nous trouvons la stéganographie par minimisation d'impact d'insertion.

Soit $\mathbf{x} = (x_1, ..., x_n)$ un support de couverture composé de n éléments. L'objectif de la stéganographie par minimisation d'impact d'insertion [2] est d'apporter le minimum d'altérations au support hôte $\mathbf{x}$, afin de produire le *stégo* objet $\mathbf{y} = (y_1, ..., y_n)$ qui communiquera le message $\mathbf{m} = (m_1, ..., m_m)$. Pour cela, nous nous munissons d'une fonction de distorsion $D(\mathbf{x}, \mathbf{y})$ que nous minimisons sous la contrainte d'un *payload* fixe. Cette fonction de distorsion se base généralement sur l'utilisation d'une carte de détectabilité $\rho \in \mathbb{R}_+^n$ qui attribue à chaque élément de couverture x_i avec $i \in \{1, ..., n\}$, un coût de détectabilité $\rho_i \in \mathbb{R}_+$ modélisant l'impact sur la sécurité dû à la modification de cet élément.

L'algorithme HUGO [1] proposé lors de la compétition BOSS [3] [5] utilise une carte de détectabilité variable, qui à chaque pixel de l'image de couverture, attribue un niveau de détectabilité $\rho_i \in [0, \infty]$ comme suggéré dans [6]. Le calcul du coût de détectabilité repose sur l'utilisation de caractéristiques de haute dimension calculées à partir de l'image de couverture. Ces caractéristiques correspondent aux probabilités conditionnelles en chaque pixel de l'image filtrée. L'algorithme MOD [4] [7] proposé début 2011, étend la proposition HUGO en définissant un coût de détectabilité $\rho_i \in [0, \infty]$ paramétré par un nombre élevé de paramètres. La recherche des paramètres menant au meilleur niveau de sécurité est effectué itérativement en répétant insertion puis modification des paramètres, grâce à l'algorithme d'optimisation *downhill simplex*. Le niveau de sécurité est évalué à chaque itération en utilisant comme critère la taille de la marge d'un SVM [5].

Dans cet article, nous proposons un nouveau schéma stéganographique adaptatif dans le domaine spatial basé sur l'utilisation d'un oracle pour le calcul de la carte de détectabilité. La principale originalité de ce travail est que lors de l'insertion, l'approche ne cherche pas uniquement à préserver le modèle de distribution de l'image de couverture courante que l'on souhaite stéganographier, mais également à préserver le modèle de toute la base de données de l'émetteur. Contrairement à l'approche proposée par Filler *et al.* [7] qui utilise une méthode paramétrique pour ré-

1. images digitale réelles non synthétiques.

2. Le principe de minimisation d'impact d'insertion a été défini dans [4] en 2007. Il repose sur l'adaptativité de l'insertion à travers l'utilisation d'une carte de détectabilité.

3. Break Our Steganography System : premier challenge en stéganalyse, visant à évaluer la sécurité de l'algorithme de HUGO. Une base de 1000 images était mise à disponibilité de la communauté de stéganalyse pour être analysée. L'objectif était de distinguer les images *stégo* des images *cover*. http ://www.agents.cz/boss/BOSSFinal/

4. MOD : Model Optimized Distortion [2].

5. SVM : Support Vector Machine.

duire la marge SVM séparant la classe *cover* de la classe *stégo*, nous proposons une méthode de calcul de carte de détectabilité non paramétrique qui utilise l'ensemble de classifieurs FLD [6] proposé par Kodovský *et al.* [3] comme oracle. Nous exploitons à la fois les informations de la l'image de couverture courante et les informations tirées de la base entière des images de couvertures.
Ce papier est organisé de la façon suivante : la section 2 présente quelques notions préliminaires. L'algorithme de stéganographie par oracle est décrit en section 3. La section 4 est consacrée à la présentation et à la discussion des résultats obtenus. Enfin, nous concluons en section 5.

2 Prérequis

Dans ce qui suit, les symboles : $\mathbf{x} = (x_1, ..., x_n) \in \mathcal{X} = \{0, ..., 255\}^n$ et $\mathbf{y} = (y_1, ..., y_n) \in \mathcal{Y} = \{0, ..., 255\}^n$ désigneront respectivement l'image *cover* et l'image *stégo* composée de n pixels en niveau de gris.

2.1 Minimisation d'impact d'insertion

Tout algorithme pratique de stéganographie par minimisation d'impact d'insertion a pour but d'insérer un message $\mathbf{m} = \{0, 1\}^m$ donné dans un support hôte $\mathbf{x}$, en essayant de réduire au minimum l'impact dû à l'insertion [6]. Afin d'atteindre cet objectif, il est important d'établir une mesure de distorsion D capable de modéliser la détectabilité statistique due à l'insertion.

Filler *et al.* [6] proposent de modéliser l'impact d'insertion par une fonction additive et positive $D : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+$ définie par :

$$D(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} \rho_i |x_i - y_i|, \tag{1}$$

avec $0 \leq \rho_i < \infty$ le coût de modification du $i^{\text{ème}}$ pixel, et telle que l'additivité de cette fonction suppose que toutes les modifications effectuées sont totalement indépendantes les unes des autres. Autrement dit, la modification d'un élément de couverture n'affecte pas la détectabilité des sites voisins.

Pour une fonction de distorsion additive D telle que (1), et une insertion binaire ($|x_i - y_i| \leq 1$), la solution au problème de minimisation d'impact d'insertion sous la contrainte d'un *payload* fixe est de la forme suivante [4] :

$$\min D(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} p_i \rho_i, \tag{2}$$

avec p_i la probabilité de modification du $i^{\text{ème}}$ pixel définie par [4] :

$$p_i = \frac{e^{-\lambda \rho_i}}{1 + e^{-\lambda \rho_i}}. \tag{3}$$

Le paramètre λ est obtenu en résolvant l'équation suivante :

$$-\sum_{i=1}^{n} \Big(p_i \log_2 p_i + (1 - p_i) \log_2 (1 - p_i) \Big) = m. \tag{4}$$

6. FLD : Fisher Linear Discriminant.

Cette formalisation de la stéganographie adaptative par minimisation d'impact d'insertion permet de découper le processus d'insertion en deux étapes successives : 1) le calcul d'une carte de détectabilité et 2) l'insertion par un algorithme adaptatif pratique. L'avantage majeur qui découle de cette séparation est que l'évaluation de la sécurité d'une carte de détectabilité ne nécessite pas d'utiliser un algorithme de l'état de l'art pour l'insertion. En pratique, si l'on souhaite insérer un message en minimisant l'impact d'insertion (la carte de détectabilité ρ est donc connue) avec la contrainte d'un *payload* fixe, il est possible de simuler l'insertion optimale en cherchant le paramètre λ (résolution de l'équation (4)), puis en modifiant chaque pixel x_i selon la probabilité p_i définie à l'équation (3).

2.2 L'ensemble de classifieurs FLD

Les schémas stéganographiques modernes tel que HUGO [1] ont tendance à utiliser des espaces de caractéristiques de plus en plus large, ce qui constitue un vrai problème pour la stéganalyse. Afin de résoudre ce problème, Kodovský *et al.* [3] proposent un nouvel outil d'apprentissage et de classification alternative aux outils classiques tel que les SVM, ou les réseaux de neurones. Leur classifieur est basé sur l'utilisation d'un ensemble de classifieurs de faible complexité. Ils utilisent pour l'apprentissage et la classification un ensemble $\mathcal{F} = \{F_1, ..., F_L\}$ de classifieurs FLD binaires.

Soit $\mathcal{A} = \{\mathbf{f}_i, c_i\}_{i=1}^{i=N}$ une base d'apprentissage composée de N observations appartenant aux deux classes d'images *cover* et *stégo*, avec $\mathbf{f}_i \in \mathbb{R}^d$ un vecteur caractéristique de grande dimension d, caractérisant la $i^{\text{ème}}$ image, et $c_i \in \{0, 1\}$ la classe qui lui est associée (0 pour une image *cover*, et 1 pour une image *stégo*).

Lors de *la phase d'apprentissage*, chaque classifieur FLD apprend à associer correctement à chaque observation $\mathbf{f}_i$ le numéro de la classe c_i à laquelle elle appartient :

$$\begin{array}{rccc} F_l : & \mathbb{R}^d & \to & \{0, 1\} \\ & \mathbf{f}_i & \to & F_l(\mathbf{f}_i). \end{array}$$

Pour cela, chaque classifieur FLD utilise la base d'apprentissage $\mathcal{A}$, afin de calculer le vecteur $\mathbf{w}_l$ orthogonal à l'hyper-plan séparant les observations de la classe *cover* de ceux de la classe *stégo*.

Dans le but de réduire la complexité de calcul, l'apprentissage et la classification de chaque classifieur FLD est effectué sur un sous-espace de caractéristiques de dimension d_{red} avec ($d_{red} \ll d$). En pratique, avant la phase d'apprentissage chaque classifieur FLD choisit pseudo-aléatoirement un sous-ensemble de caractéristiques (d_{red} caractéristiques) à partir de chaque vecteur caractéristique $\mathbf{f}_i \in \mathbb{R}^d$.

Lors de *la phase test*, une observation $\mathbf{f}$ donnée en entrée de cet ensemble de classifieurs est classée par chaque classifieur. Chaque classifieur FLD retourne alors une décision binaire indiquant le numéro de la classe attribuée à cette

observation. La décision finale est obtenue en fusionnant par vote majoritaire les résultats des différents classifieurs FLD, à travers la formule suivante [3] :

$$\begin{array}{rccc} R: & \mathbb{R}^d & \rightarrow & \{0,1\} \\ & \mathbf{f} & \rightarrow & R(\mathbf{f}), \end{array}$$

$$\text{tel que :} \quad R(\mathbf{f}) = \begin{cases} 1 & si \quad \sum_{l=1}^{L} F_l(\mathbf{f}) > L/2, \\ 0 & sinon. \end{cases}$$

La performance de l'ensemble de classifieurs FLD en terme de détection peut être évaluée en utilisant la probabilité d'erreur (P_E) définie par :

$$P_E = \tfrac{1}{2}\left(P_{FP} + P_{FN}\right),$$

avec P_{FP} la probabilité de faux positif et P_{FN} la probabilité de faux négatif. Plus la probabilité d'erreur (P_E) est petite, plus la classification en deux classes est meilleure.

3 Le schéma ASO

3.1 Calcul de la carte de détectabilité

Notre stratégie de dissimulation de données s'appuie sur le principe de minimisation d'impact d'insertion (section 2.1). Il s'agit d'une technique de stéganographie adaptative par oracle (ASO [7]) qui repose sur l'adaptativité de l'insertion à travers l'utilisation d'une carte de détectabilité $\rho = \{\rho_i \in [0, \infty[\}_{i=1}^{n}$ calculée par un oracle. Les fonctionnalités de l'ensemble de classifieurs FLD de Kodovský [3], ainsi que les informations acquises lors de l'apprentissage sont exploitées lors du calcul de la carte de détectabilité. L'objectif est de tirer profit des informations de la base de données utilisée par l'émetteur afin d'augmenter la sécurité du processus d'insertion.

Considérons une image de couverture en niveau de gris $\mathbf{x} = (x_1, ..., x_n)$ composée de n pixels, un vecteur $\mathbf{f_x}$ caractérisant cette image, une fonction de distorsion D additive telle que (1), et une insertion en *LSB-matching* [8].

La carte de détectabilité $\rho \in \mathbb{R}^n$ est calculée indépendamment en chaque pixel x_i. La détectabilité ρ_i du pixel x_i est définie de la même façon que dans HUGO [1] par :

$$\rho_i = min\left(\rho_i^{(+)}, \rho_i^{(-)}\right), \tag{5}$$

avec $\rho_i^{(+)}$ (respectivement $\rho_i^{(-)}$) la détectabilité après modification $+1$ (respectivemennt -1) du pixel x_i.

Nous proposons de calculer la détectabilité $\rho_i^{(+)}$ (resp. $\rho_i^{(-)}$) grâce à un oracle formé de L classifieurs FLD de Kodovský *et al* [3]. Pour cela, nous définissons la détectabilité $\rho_i^{(+)}$ (resp. $\rho_i^{(-)}$) comme étant la somme sans pondération de la détectabilité $\rho_i^{(l)}$ de chaque classifieur F_l, avec $l \in \{1..,L\}$:

$$\rho_i^{(+)} = \sum_{l=1}^{L} \rho_i^{(l)(+)}, \qquad \text{et} \qquad \rho_i^{(-)} = \sum_{l=1}^{L} \rho_i^{(l)(-)}, \tag{6}$$

avec $\rho_i^{(+)}$ (resp. $\rho_i^{(-)}$) la détectabilité fournit par le $l^{ème}$ classifieur.

Pour un classifieur F_l, $l \in \{1,..L\}$, nous définissons la détectabilité $\rho_i^{(l)(+)}$, $l \in \{1..,L\}$, par :

$$\begin{aligned} \rho_i^{(l)(+)} &= \frac{\mathbf{w}^{(l)}.\mathbf{f_x}^{(l)(+)} - \mathbf{w}^{(l)}.\mathbf{f_x}^{(l)}}{s^{(l)}} \\ &= \frac{\mathbf{w}^{(l)}.\left(\mathbf{f_x}^{(l)(+)} - \mathbf{f_x}^{(l)}\right)}{s^{(l)}}, \end{aligned} \tag{7}$$

De même, la détectabilité $\rho_i^{(l)(-)}$ est définie par :

$$\rho_i^{(l)(-)} = \frac{\mathbf{w}^{(l)}.\left(\mathbf{f_x}^{(l)(-)} - \mathbf{f_x}^{(l)}\right)}{s^{(l)}}, \tag{8}$$

avec $s^{(l)} \in \mathbb{R}_+$ le facteur de normalisation (*le scaling*) du $l^{ème}$ classfieur F_l, $\mathbf{w}^{(l)}$ le vecteur orthogonal à l'hyper-plan séparateur des deux classes *cover* et *stégo* du classifieur F_l, $\mathbf{f_x}^{(l)}$ le vecteur caractéristique à classer par le classifieur F_l, et $\mathbf{f_x}^{(l)(+)}$ (resp. $\mathbf{f_x}^{(l)(-)}$) le vecteur caractéristique après modification +1 (resp. -1) du pixel x_i.

Notre objectif est d'obtenir une valeur $\rho_i^{(l)(+)}$ (resp. $\rho_i^{(l)(-)}$) faible, lorsque la modification +1 (resp. -1) entraine un déplacement $(\mathbf{f_x}^{(l)(+)} - \mathbf{f_x}^{(l)})$ (resp. $(\mathbf{f_x}^{(l)(-)} - \mathbf{f_x}^{(l)})$) vers la classe *cover*. L'hypothèse forte de notre schéma ASO est que la modification d'un pixel doit avoir tendance à rapprocher l'image *stégo* d'une image *cover*. Par construction, le vecteur $\mathbf{w}^{(l)}$ est toujours orienté dans le sens *cover* vers *stégo*. Ainsi, en calculant $\rho_i^{(l)(+)}$ et $\rho_i^{(l)(-)}$ par les équations (7) et (8), nous obtenons exactement ce comportement puisque les détectabilités $\rho_i^{(l)(+)}$ et $\rho_i^{(l)(-)}$ sont minimales lorsque le vecteur $\mathbf{w}^{(l)}$ et le vecteur $(\mathbf{f_x}^{(l)(+)} - \mathbf{f_x}^{(l)})$ (resp. $(\mathbf{f_x}^{(l)(-)} - \mathbf{f_x}^{(l)})$) sont colinéaires et de sens opposé, autrement dit, lorsque :

$$\mathbf{w}^{(l)}.\left(\mathbf{f_x}^{(l)(+)} - \mathbf{f_x}^{(l)}\right) < 0 \tag{9}$$

ou lorsque :

$$\mathbf{w}^{(l)}.\left(\mathbf{f_x}^{(l)(-)} - \mathbf{f_x}^{(l)}\right) < 0. \tag{10}$$

Par ailleurs, on comprend aisément la nécessité d'avoir un facteur de mise à l'échelle $s^{(l)}$ propre à chaque classifieur. En effet, cela permet d'avoir une mesure de détectabilité $\rho_i^{(l)(+)}$ et $\rho_i^{(l)(-)}$ d'un même ordre de grandeur entre les classifieurs. Puisque le calcul $\rho_i^{(+)}$ et $\rho_i^{(-)}$ s'obtient par une somme des $\rho_i^{(l)(+)}$ et $\rho_i^{(l)(-)}$, alors chaque classifieur doit donner un coût de détectabilité $\rho_i^{(l)(+)}$ ou $\rho_i^{(l)(-)}$ avec $l \in \{1..,L\}$, d'un même ordre de grandeur.

Nous définissons le facteur de normalisation $s^{(l)}$ par :

$$s^{(l)} = (\mu_1^{(l)} - \mu_0^{(l)})\mathbf{w}^{(l)} + 2(\sqrt{\mathbf{w}^{(l)T}\Sigma_0^{(l)}\mathbf{w}^{(l)}} + \sqrt{\mathbf{w}^{(l)T}\Sigma_1^{(l)}\mathbf{w}^{(l)}}), \tag{11}$$

7. ASO : Adaptive Steganography by Oracle.
8. *LSB-matching* : Modification des pixels par ± 1.

avec $\mu_0^{(l)}$ et $\Sigma_0^{(l)}$ le vecteur moyenne et la matrice de co-variance de la classe *cover*, et $\mu_1^{(l)}$ et $\Sigma_1^{(l)}$ le vecteur moyenne et la matrice de co-variance de la classe *stégo*.

Rappelons, que pour un classifieur FLD, le vecteur $\mathbf{w}^{(l)}$, ainsi que le facteur de normalisation $s^{(l)}$ sont calculés lors de la phase d'apprentissage. Ce sont donc des valeurs pré-calculées qui ne sont pas recalculées lors du calcul de $\rho^{(l)(+)}$ et $\rho^{(l)(-)}$ (équations (7) et (8)).

Remarquons également, que lors de la phase de construction de la carte de détectabilité, il est très complexe, en coût de calcul, de calculer les vecteurs $\mathbf{f_x}^{(l)(+)}$ et $\mathbf{f_x}^{(l)(-)}$ pour chaque pixel et pour chaque classifieur. Pour réduire la complexité, il faut non pas calculer $\mathbf{f_x}^{(l)(+)}$ et $\mathbf{f_x}^{(l)(-)}$, mais calculer directement sur une zone réduite de l'image, la variation $(\mathbf{f_x}^{(l)(-)} - \mathbf{f_x}^{(l)})$ ou $(\mathbf{f_x}^{(l)(+)} - \mathbf{f_x}^{(l)})$ impliquée par la modification -1 ou +1 sur le pixel x_i.

Nous obtenons une carte de détectabilité $\rho \in \mathbb{R}$ composée de valeurs positives et négatives. Pour obtenir une carte de détectabilité positive $\rho = \{\rho_i \in [0, \infty[\}_{i=1}^{n}$, nous translatons l'ensemble des valeurs de notre carte par la valeur $\rho_{min} = min(\rho)$ où ρ_{min} est la plus petite détectabilité de la carte ρ. Une fois calculée, cette carte de détectabilité peut être utilisée pour l'insertion du message, soit par simulation comme expliqué en section 2.1 (équations 3 et 4), soit en utilisant l'approche STC [9] proposée dans [6].

3.2 Schéma d'insertion adaptatif proposé

À travers notre schéma de stéganographie adaptative par oracle (ASO), nous proposons un nouveau concept de stéganographie : *la stéganographie par base d'images*. La méthode d'insertion proposée permet non pas d'obtenir une seule image *stégo* en sortie du système, mais d'obtenir toute une base d'images *stégo*. Lors de la transmission, l'émetteur peut alors choisir les images les plus sûres pour la communication de son message.

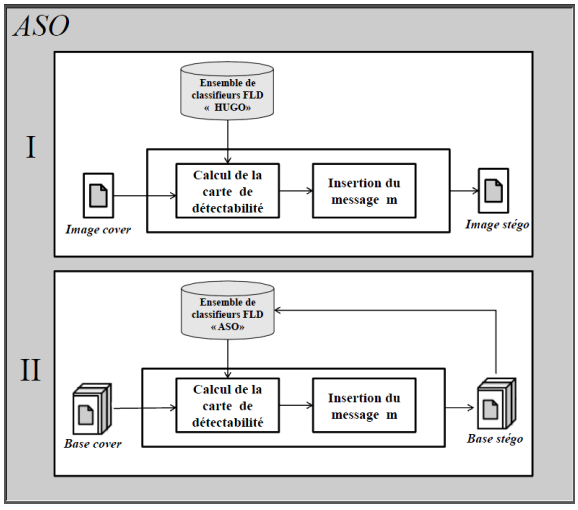

Figure 1 – *Schéma de Stéganographie Adaptative par Oracle (ASO).*

9. STC : Sydrome Trellis Codes.

Comme l'illustre la Figure 1, le schéma de stéganographie adaptative par oracle se décompose en deux phases :

La première phase (notée I sur la Figure 1), utilise pour le calcul de la carte de détectabilité (section 3.1), un ensemble de classifieurs FLD entrainé à distinguer les images *cover* des images *stégo* de l'algorithme HUGO. Une fois calculée, la carte de détectabilité est utilisée pour l'insertion du message. Cette opération est effectuée comme dans HUGO, en simulant l'insertion parfaite via l'algorithme optimal (section 2.1 : équations (3) et (4)). À la sortie de cette première phase nous obtenons une image *stégo* ASO.

La seconde phase (notée II sur la Figure 1) est une phase itérative, qui vise à augmenter l'indétectabilité du message (augmentation de la sécurité). Elle utilise, pour le calcul de la carte de détectabilité, à chaque itération, un ensemble de classifieurs FLD qui a appris à faire la différence entre les images *cover*, et les images *stégo* ASO obtenues lors de l'itération précédente. Le processus itératif est répété jusqu'à stabilisation de la probabilité d'erreur de la classification. Ainsi, à la fin de chaque itération, une stéganalyse par ensemble de classifieurs FLD de Kodovský *et al.* [3] est effectuée (section 2.2). Si la probabilité d'erreur ne varie plus, alors les itérations cessent. À la fin de cette seconde phase, nous obtenons une base d'images *stégo* ASO.

Lors de la phase de transmission, l'émetteur choisit le ou les images les plus sûres pour la transmission de ses données secrètes. La transmission du message est effectuée de manière concrète en utilisant l'algorithme STC [6].

4 Résultats expérimentaux

4.1 Conditions expérimentales

Pour vérifier l'efficacité de notre schéma de stéganographie, nous avons procédé à deux types de tests : un test de validité, et un test comparatif de performance entre l'algorithme HUGO et l'algorithme ASO. Nos tests ont été effectués sur la base d'images BOSSBase v1.00 [10] contenant 10000 images *cover* en niveau de gris de taille 512×512 en format pgm.

L'algorithme ASO est appliqué à chaque étape (durant la phase I et la phase II) sur les 10000 images *cover* de la base BOSSBase v1.00. Autrement dit, à chaque fois les mêmes 10000 images sont stéganographiées par ASO. Par contre, l'apprentissage de l'oracle, et le test par le stéganalyseur nécessite à chaque fois de former deux bases composées chacune de 5000 *cover*, et de 5000 *stégo* ASO obtenues à l'étape précédente.

Lors de la phase I, les 10000 images *cover* de BOSSBase v1.00 sont stéganographiées par l'algorithme ASO. Dans cette étape l'oracle doit avoir préalablement appris sur une de base de 5000 images *cover* de BOSSBase v1.00 et 5000 images *stégo* HUGO. Le *payload* choisi par l'utilisateur

10. BOSSBase v1.00 : base d'images disponible sur http ://agents.cz/boss/BOSSFinal/.

impose l'utilisation d'images *stégo* de même *payload* durant toutes les étapes.

Lors de la phase II, les 10000 images *cover* et 10000 images *stégo* ASO de l'étape précédentes sont utilisées pour former une base (notée $\mathcal{A}$) de : 5000 *cover* et 5000 *stégo* ASO, et une autre base (notée $\mathcal{T}$) composée des 5000 autres *cover* et 5000 autres *stégo*. La base $\mathcal{A}$ est utilisée pour que l'oracle apprenne à distinguer les images *cover* des images *stégo* ASO. La base $\mathcal{T}$ permet au stéganalyseur de déterminer les performances de la stéganographie par ASO, et de juger s'il faut relancer le processus ou pas.

Chaque image est représentée par un vecteur caractéristique MINMAX [8] de taille $d = 5330$. Les paramètres des caractéristiques MINMAX utilisées, ainsi que leur dimension sont présentés dans le tableau 1. Nous fixons, par minimisation de P_E, après de nombreuses expérimentations sur des images HUGO à 0.4 *bpp*, le nombre de classifieurs FLD à $L = 30$, et la dimension $d_{red} = 250$. Pour HUGO, nous obtenons une probabilité d'erreur de détection $P_E = 23.67\%$.

Pour l'implémentation, nous avons choisi de travailler avec la librairie *OpenMP* [11] pour paralléliser le processus d'insertion. Les expérimentations ont été effectuées sur un ordinateur à 8 processeurs *Quad-Core AMD Opeteron(tm) Processor 8384* à 2.69 GHz ; et les 32 cœurs sont utilisés.

s	**q**	**m**	**T**	**dim**
3	2	3	3	686
3	2	4	2	1250
3	2	3	4	1458
4	2	3	3	686
4	2	4	2	1250

Tableau 1 – *Paramètres des caractéristiques MINMAX [8] utilisées.* ***s*** *: le nombre des pixels utilisés pour le calcul du résidu,* ***q*** *: le pas de quantification,* ***m*** *: l'ordre des matrices de co-occurrence,* ***T*** *: le seuil de la troncature, et* ***dim*** *: la dimension résultante.*

4.2 Test de validité

Pour vérifier la validité du schéma ASO[14] nous avons construit deux bases de 110 images. La première base $\mathcal{B}'$ est constituée de 100 images *stégo* HUGO à 0.4 *bpp* et de 10 images *cover*. La deuxième base $\mathcal{B}''$ est constituée de 100 images *stégo* ASO [12] à 0.4 *bpp* et de 10 images *cover* [13]. Une fois construites, nous avons procédé à la stéganalyse des deux bases ($\mathcal{B}'$ et $\mathcal{B}''$) en utilisant le stéganalyseur par Ensemble [3] composé de $L = 30$ classifieurs FLD entrainé à distinguer les images *cover* des images *stégo* HUGO.

En utilisant le même stéganalyseur (sans ré-apprentissage) pour la stéganalyse des deux bases ($\mathcal{B}'$ et $\mathcal{B}''$), nous cherchons à tester si une image qui était considérée comme étant *stégo* lorsqu'elle est stéganographiée par HUGO, est toujours considérée comme tel après insertion par l'algorithme ASO (phase I), ou si elle est considérée comme étant *cover*. Autrement dit, si l'utilisation d'un oracle entrainé à distinguer les images *cover* des images *stégo* lors de la phase I, permet de faire basculer une image de la *classe stégo* vers la *classe cover*.

Les résultats présentés dans le tableau 2 confirment la validité de notre algorithme de stéganographie adaptative par oracle (ASO). En effet, le stéganalyseur classe de manière erronée 17 images comme *cover* alors qu'elles sont *stégo* HUGO (FP = 17), et ce même stéganalyseur classe de manière erronée 100 images *cover* alors qu'elles sont *stégo* ASO (FP = 100). Autrement dit, la totalité des 100 images *stégo* ASO ont été classées comme étant des images *cover*. Ce test permet de valider l'hypothèse selon laquelle l'exploitation des informations tirées de la phase d'apprentissage par l'oracle, pour le calcul des $\rho_i^{(l)(+)}$ et $\rho_i^{(l)(-)}$ (équations 7 et 8) permet de faire basculer une image donnée de la *classe stégo* vers la *classe cover*.

On constate également, que sur les 10 images *cover* des deux bases ($\mathcal{B}'$ et $\mathcal{B}''$), 2 images sont classées *stégo* alors qu'elles sont *cover*. Cela confirme l'importance de la phase de sélection des images les plus sûres lors de la transmission des données secrètes. En effet, grâce au concept de *stéganographie par base d'images* proposé via notre schéma d'insertion ASO, nous offrons à l'émetteur la possibilité de choisir l'image ou les images les plus sûres pour la transmission de son message secret.

	HUGO $\mathcal{F}(\mathcal{B}')$	**ASO** $\mathcal{F}(\mathcal{B}'')$
FP	2	2
FN	17	100

Tableau 2 – *Test de validité de l'algorithme ASO, avec FP : le nombre de faux positifs, et FN : le nombre de faux négatifs.*

4.3 Test de performance

Afin de tester la performance de notre algorithme ASO en situation réelle, nous avons comparé la performance de l'algorithme ASO avec celle de HUGO [1]. Pour cela, nous avons procédé à la stéganalyse des algorithmes ASO et HUGO via l'ensemble de classifieurs FLD de Kodovský *et al.* [3] (section 2.2), à différents *payload*. Pour chaque *payload*, nous avons utilisé une base test et une base d'apprentissage constituées chacune de 5000 images *cover* et 5000 images *stégo*.

Les résultats obtenus pour ce test [14] sont présentés sur la Fi-

11. OpenMP : librairie pour la programmation parallèle disponible sur http ://openmp.org/wp/.

12. Notons que, les 100 images hôtes stéganographiées avec HUGO (base $\mathcal{B}'$) sont les mêmes que celles utilisées pour être stéganograhiées avec ASO (base $\mathcal{B}''$).

13. Les 10 images cover de la base $\mathcal{B}''$ sont les mêmes que celles utilisées dans $\mathcal{B}'$

14. Pour le test de validité et le test de performance, seule la phase I de ASO est utilisée. Autrement dit, pour l'algorithme ASO, nous avons effectué une seule itération en utilisant comme oracle un ensemble classifieurs FLD entrainé à distinguer les images *cover* des images *stégo* HUGO.

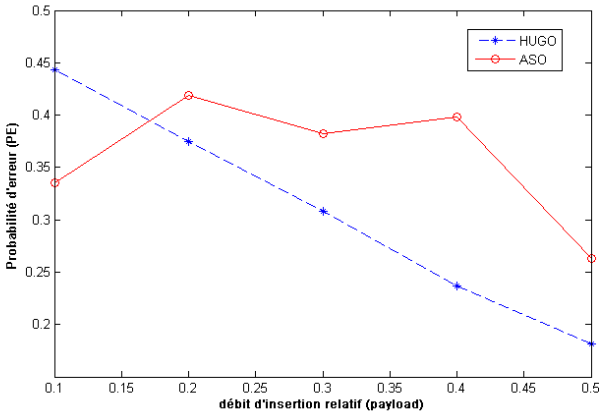

Figure 2 – *Comparaison du niveau de sécurité (ASO vs HUGO).*

gure 2. La performance de l'algorithme ASO en terme de sécurité est supérieure à celle de HUGO pour des *payload* de 0.2 *bpp* à 0.5 *bpp*. A titre d'exemple, pour un *payload* de 0.4 *bpp*, la probabilité d'erreur de détection (P_E) de l'algorithme ASO est de 40% contre 23,67% pour HUGO. Soit une différence de 16.33%. De même, pour un *payload* de 0.5 *bpp* la probabilité d'erreur de détection de ASO est supérieur à celle de Hugo. Elle est de 26.31% contre 18.13% pour HUGO, ce qui constitue une différence considérable. On peut dire que l'algorithme ASO offre à l'émetteur la possibilité d'envoyer de longs messages secrets avec une plus grande sécurité que celle de HUGO.

On constate également que le niveau de sécurité de ASO pour des petits *payload* est plus faible que celui de HUGO. En effet, pour un *payload* de 0.1 *bpp*, la probabilité d'erreur de détection de ASO est de 33.52% contre 44.34% pour HUGO. Cela peut s'expliquer par le fait que pour un tel *payload*, l'oracle utilisé pour le calcul de la carte de détectabilité (section 3.1) a du mal à distinguer les images *cover* des images *stégo* HUGO, ce qui influence automatiquement l'insertion avec ASO. Autrement dit, l'algorithme ASO n'arrive pas à distinguer les régions sûres de celles qui ne le sont pas.

5 Conclusion

Dans cet article, nous avons présenté un nouveau schéma de Stéganographie Adaptatif par Oracle (ASO), utilisant une carte de détectabilité calculée à partir de l'ensemble de classifieurs FLD de Kodovský *et al.* [3]. L'exploitation de l'ensemble de classifieurs comme oracle lors du calcul de la carte de détectabilité, permet d'augmenter la sûreté du processus d'insertion. En effet, nous préservons à la fois le modèle de distribution de l'image de couverture courante, et le modèle de la base d'images utilisée par l'émetteur. Par ailleurs, le concept de stéganographie par *base d'images* offre à l'émetteur la possibilité de choisir les images les plus sûres lors de la transmission. Comme l'illustre les résultats obtenus, le schéma d'insertion proposé présente de bonnes performances en termes de sécurité. Avec seulement une itération, l'algorithme ASO offre à l'émetteur la possibilité d'envoyer de longs messages secrets avec une plus grande sécurité que celle de HUGO. Les travaux futurs porteront sur : (1) l'étude du nombre d'itérations nécessaires pour l'amélioration de la sécurité de l'algorithme ASO, (2) la possible extension du schéma ASO via l'utilisation d'autres caractéristiques pour le calcul de la carte de détectabilité, et (3) l'étude de la complétude du modèle utilisé [2].

Remerciements.

Nous tenons à remercier le Ministère de l'Enseignement Supérieur et de la Recherche Scientifique de la République Algérienne Démocratique et Populaire pour le financement de ce travail.

Références

[1] T. Pevný, T. Filler, et P. Bas. Using High-Dimensional Image Models to Perform. Highly Undetectable Steganography. Dans *Information Hiding - 12th International Conference*, volume 6387 de *Lecture Notes in Computer Science, IH'10*, pages 161–177, Berlin, Heidelberg, October 01 2010. Springer-Verlag.

[2] J. Kodovský, T. Filler, J.J. Fridrich, et V. Holub. On Dangers of Overtraining Steganography to Incomplete. Cover Model. Dans *workshop on Multimedia and security, part of MM&Sec '11 Proceedings of the thirteenth ACM multimedia*, pages 69–76, Buffalo, NY, USA, September 29 - 30 2011. ACM.

[3] J. Kodovský et J.J. Fridrich. Steganalysis in High Dimensions: Fusing Classifiers Built on Random Subspaces. Dans *Media Watermarking, Security, and Forensics XIII, part of IS&T SPIE Electronic Imaging Symposium*, volume 7880, paper. 21, pages L 1–12, San Francisco, CA, January 23-26 2011.

[4] J.J. Fridrich et T. Filler. Practical Methods for Minimizing Embedding Impact in Steganography. Dans *Security, Steganography, and Watermarking of Multimedia Contents IX, part of IS&T SPIE Electronic Imaging Symposium*, volume 6505, pages 02–03, San Jose, CA, January 29-February 1 2007.

[5] P. Bas, T. Filler, et T. Pevný. Break Our Steganographic System — the ins and outs of organizing BOSS. Dans *Information Hiding - 13th International Workshop*, volume 6958 de *Lecture Notes in Computer Science, IH'11*, pages 59–70, Prague, Czech Republic, May 18-20 2011. Springer-Verlag.

[6] T. Filler, J. Judas, et J.J. Fridrich. Minimizing Embedding Impact in Steganography using Trellis-Coded Quantization. Dans *Media Forensics and Security II, part of IS&T SPIE Electronic Imaging Symposium*, volume 7541, paper. 05, San Jose, CA, USA, January 18-20 2010.

[7] T. Filler et J.J. Fridrich. Design of Adaptive Steganographic Schemes for Digital Images. Dans *Media Watermarking, Security, and Forensics XIII, part of IS&T SPIE Electronic Imaging Symposium*, volume 7880, paper. 13, pages F 1–14, San Francisco, CA, January 23-26 2011.

[8] J.J. Fridrich, J. Kodovský, V. Holub, et M. Goljan. Breaking hugo – the process discovery. Dans Tomás Filler, Tomás Pevný, Scott Craver, et Andrew D. Ker, éditeurs, *Information Hiding - 13th International Conference*, volume 6958 de *Lecture Notes in Computer Science, IH'11*, Prague, Czech Republic, May 18-20 2011. Springer.