Stéganalyse : détection d'information cachée dans des contenus multimédias

Rémi Cogranne, Patrick Bas, Marc Chaumont

1

STÉGANALYSE : Détection d'information cachée dans des contenus multimédias

Rémi COGRANNE[1], Patrick BAS[2], Marc CHAUMONT[3]
[1]Université de Technologie de Troyes
[2]CRIStAL, Lille
[3]LIRMM, Univ Montpellier, CNRS, Univ Nımes

Dans le chapitre précédent, nous avons présenté la stéganographie, c'est-à-dire les méthodes permettant de dissimuler des informations dans des images numériques. Nous avons notamment insisté sur le fait que les méthodes de stéganographie se construisent dans un contexte où un adversaire cherche à identifier parmi une liste d'image celles servant à véhiculer des informations dissimulées. Dans ce chapitre, nous verrons donc comment procéder à une analyse d'une image numérique en vue d'obtenir des informations sur les données qui peuvent y avoir été cachées.

1.1. Introduction, enjeux et contraintes (2 pages)

Avant de rentrer dans le vif du sujet, nous rappellerons brièvement que le but de la stéganographie est de dissimuler des informations secrètes dans des médias

STÉGANALYSE : Détection d'information cachée dans des contenus multimédias, coordonné par William PUECH. © ISTE Editions 2019.

numériques afin que ces derniers demeurent visuellement et statistiquement, « aussi proche possible » que les médias originaux anodins. L'exemple du problème de prisonnier **??**, permet d'illustrer ce contexte. Le but de la stéganalyse peut sembler, à première vue, assez simple : il s'agit « seulement » de détecter les médias contenant des informations cachées afin d'empêcher leur transmission. En pratique, le sujet est beaucoup plus vaste que cela. En réalité, comme très souvent dans le domaine de la sécurité, le but de la stéganalyse dépend largement du *scénario* considéré et notamment des connaissances *a priori* dont dispose Ève.
Nous verrons dans ce chapitre que la stéganalyse a été très largement étudiée comme outil d'évaluation des méthodes de stéganographie. Cette approche pose un contexte très particulier dans lequel Wendy, la gardienne procédant à la stéganalyse, est supposée *omnisciente* : elle peut accéder à (presque) toutes les informations relatives aux données dissimulées. Dans ce scénario on considère généralement que Wendy ignore uniquement (1) quel est le message dissimulé, bien que sa distribution statistique soit connue (2) quelle est la clé d'insertion utilisée et (3) si un message est effectivement inséré. Ce scénario est très pratique pour la stéganographie Alice, car d'une part cela lui permet de se focaliser sur le problème d'intérêt sans tenir compte des « difficultés techniques » annexes ; par ailleurs, dans ce scénario Alice se place dans le pire cas, elle peut donc être certaine que son évaluation est pessimiste et que, en pratique, un adversaire plus réaliste ne concevoir une stéganalyse aussi précise.

Les différents objectifs de la stéganalyse

De nombreux travaux ont été proposés pour faire différents types de stéganalyse, nous allons brièvement les décrire ci-dessous.
D'une part, différentes méthodes de détection peuvent être envisagées selon le média de couverture : les images, les sons et les vidéos étant des médias différents et compressés de façon différente, ces derniers devraient être traités différemment pour y déceler des informations cachées. Plus largement la stéganographie dans les textes ou dans les réseaux informatiques sont de nature si différente qu'il apparait difficile de les analyser des façons analogues.
D'autres travaux, plus proches de ce qui se fait en cryptanalyse, se focalisent sur la recherche de la clé d'insertion à partir d'images contenant des informations cachées. Ces travaux requièrent souvent des connaissances importantes de la part de la stéganalyste Wendy.
Dans la stéganalyse *quantitative* le but est d'estimer la taille d'un possible message dissimulé. On comprendra aisément qu'estimer la taille du message peut être rapproché d'un problème de détection binaire. Une première vision, assez naïve, consiste à estimer la taille du message caché et de décider que le média est stéganographie lorsque cette estimation dépasse un seuil donné.
Enfin on peut mentionner la stéganalyse active dans laquelle Wendy modifie légèrement le média transmis afin de préserver son aspect visuel tout en rendant l'exctration du message caché impossible.

Il serait difficile d'aborder dans ce seul chapitre l'ensemble des problèmes qui constituent la stéganalyse. Aussi, bien que ces questions sont intéressantes et difficiles, nous reviendrons au cadre classique qui s'intéresse à détecter des informations cachées dans les images numériques.
Comme nous l'avons précédemment expliqué, la stéganalyse s'est largement développée pour permettre l'évaluation des méthodes de stéganographie. Aussi, la plupart des travaux dans ce domaine invoquent le principe de Principe de Kerckhoffs [FIXEME ?add footnote si premier usage], reformulé par Claude. E. Shannon comme le fait que « l'adversaire connaît le système utilisé ». En vertu de ce principe, on suppose que Wendy, la stéganalyse, ignore uniquement ce qui concerne le message inséré et la clé d'insertion utilisée. Nous reviendrons plus longuement dans la section 1.3 en détail sur ce scénario de stéganalyse *ciblée*, mais cela suppose que la stéganalyse, Wendy, connaît le type de média, généralement une image, l'algorithme d'insertion, la taille du message et ses propriétés statistiques, etc. ... La principale conséquence de l'utilisation de la stéganalyse pour l'évaluation de l'efficacité est son manque de réalisme en pratique.Nous verrons dans la dernière partie de ce chapitre 1.6, que les conditions expérimentales sont également très éloignées des conditions opérationnelles dans lesquelles la stéganalyse pourrait être réalisée.

Les différentes méthodes pour faire de la stéganalyse

Les méthodes de détection d'informations cachées les plus efficaces sont indubitablement celles basées sur des signatures. Dans ce cas il de détecter la stéganographie indirectement en identifiant une propriété particulière que l'on retrouve systématiquement lorsque l'insertion d'un message a été effectuée avec un certain outil et uniquement dans ce cas. Il s'agit donc de détecter une spécificité liée à l'utilisation d'un outil précis plus que liée à la présence d'une information cachée elle-même. Cette approche peut être rapprochée de ce qui se fait pour la sécurité informatique (détection d'attaque réseaux, de virus, etc. ...) et sera abordée dans la section 1.2 au travers de quelques exemples.
Une seconde famille de méthodes de stéganalyse, plus générale, basée sur la modélisation et la détection statistique sera abordée dans la section 1.3. Ce type de méthodes de détection est très complexe à mettre en œuvre en pratique, car elle nécessite de disposer d'un modèle statistique très précis d'une image de couverture et d'une image stéganographiée afin de pouvoir évaluer, *statistiquement* si une image donnée provient plus vraisemblablement d'un de ces deux modèles. Dans cette section nous définirons précisément le cadre de la stéganalyse *ciblée* qui est nécessaire pour la construction de tels modèles statistiques.
La section 1.4 traitera de l'utilisation de méthodes d'apprentissage statistiques. Les méthodes de stéganalyse les plus efficaces reposent sur cette approche et c'est donc naturellement celle qui a été le plus largement étudiée. Nous verrons plus précisément qu'avec ces approches la stéganalyse est subdivisée en deux problèmes distincts ; le premier concerne l'extraction de *caractéristiques* pertinentes et le second

est d'apprendre à distinguer, sur la base d'un vaste lot d'exemples, les *caractéristiques* provenant d'images de couverture et d'images stéganographiées.
Depuis peu, on assiste à un développement spectaculaire des méthodes d'apprentissage profond, ou *Deep Learning*, qui, au contraire, procèdent en une seule phase qui regroupe caractérisation et classification. L'utilisation de ces méthodes pour la stéganalyse sera abordée dans la partie 1.5.
Ce chapitre sera terminé par une brève liste non exhaustive des sujets qui nous ont semblé les plus intéressants et qui demeurent portant largement ouverts.

1.2. Détection par signature (2 pages)

Dans le domaine de la sécurité informatique, la détection par signature se définit généralement comme la « détection basée sur la recherche de motifs spécifiques, de traces caractéristiques indiquant l'utilisation d'un logiciel particulier ». Ici l'aspect logiciel est important, car, une signature ne dépend pas de la méthode d'insertion considérée, mais est bien spécifique à une implémentation donnée.
Un exemple simple et peut-être un peu caricatural aidera certainement à y voir plus clair. L'algorithme de stéganographie F5 a été proposé par A. Westfeld en 2001 (Westfeld 2001) associé à un démonstrateur afin d'en permettre une utilisation au sein de la communauté scientifique. A. Westfeld ne souhaitait pas programmer entièrement la partie compression JPEG et a utilisé un encodeur libre, ... mais ce dernier inséré systématiquement dans l'entête du fichier le commentaire suivant « JPEG ENCODER COPYRIGHT 1998, JAMES R. WEEKS AND BIOELECTROMECH ». Cet encodeur étant en réalité très peu utilisé, il a été possible de détecter les images contenant des informations cachées avec F5 non pas en détectant directement les données dissimulées, mais en détectant ce commentaire très particulier. Ici il s'agit en quelque sorte de détecter une « erreur d'implémentation » et, généralement, un logiciel de stéganographie ne doit pas laisser de signatures qui permet de l'identifier. En ce sens une signature est toujours une faille liée à une implémentation bien précise.

Un second example de détection par signature a été présenté dans (Goljan and Fridrich 2015) et concerne les images couleurs. Afin d'être le plus clair possible dans nos explications, rappelons qu'un appareil photographique est par nature insensible à la couleur. Pour relever la couleur, un micro filtre rouge, vert ou bleu, est placé devant chaque pixel qui enregistre alors uniquement l'information couleurs correspondant à la couleur du filtre (voir Chapitre 1[FIXME]). Il faut ensuite « reconstituer les couleurs manquantes » ce qui se fait à partir des pixels voisins. Il a été observé dans (Goljan and Fridrich 2015) que la stéganographie peut aller à l'encontre des règles élémentaires de reconstruction de la valeur des couleurs manquantes ; cela est particulièrement flagrant si, par exemple, la composante verte d'un pixel estimé à partir de ses voisins devient, à cause de la stéganographie, plus importante que tous ses pixels voisins. Il semble en effet impossible qu'une estimation fournisse une valeure supérieure au maximum des voisins ?

Les auteurs de (Goljan and Fridrich 2015) ont alors proposé d'utiliser 7 caractéristiques qui compte le nombre de pixels dont la valeur n'est pas en accord avec des règles de reconstruction des couleurs à partir des pixels voisins ; ces valeurs sont toujours supposées être nulles pour une image naturelle et permettent très efficacement de détecter par signature la stéganographie dans les images couleurs.

Enfin, un dernier exemple est celui des images utilisées dans le domaine spatial pour cacher un message, mais qui ont été préalablement compressées au format *JPEG* (voir ....[FIXME]). Cette approche peut sembler intéressante, car une image non compressée peut être vecteur d'une plus grande quantité de données pour la stéganographie. Cependant, si une image a été compressée en utilisant le standard *JPEG*, les blocs de 8×8 pixels pourront être représentés par une somme des composantes de la base de représentation *DCT* pondérée par des coefficients entiers. Plus exactement notons $\mathbf{X}$ un bloc de 8×8 pixels ce dernier peut s'écrire :

$$\mathbf{X} = \text{round}\left(\sum_{k=0}^{7}\sum_{l=0}^{7} c_{k,l} q_{k,l} \mathbf{D}_{k,l}\right), \qquad [1.1]$$

avec $q_{k,l}$ les éléments de la matrice de quantification, $c_{k,l}$ les coefficients (inconnus) des composantes de la base DCT notés $\mathbf{D}_{k,l}$ et round$(\cdot)$ la fonction d'arrondie. Si l'on modifie certains pixels de ce bloc $\mathbf{X}$ après compression JPEG, il deviendra impossible de trouver des coefficients $c_{k,l}$ permettant de représenter le bloc $\mathbf{X}$. Le stéganalyste, Alice, pourra alors déduire que cette image semble avoir été compressée au format JPEG, mais que certains pixels vont à l'encontre de ce qui aurait dû être obtenu lors de la décompression. Si l'on ajoute à cela que ces pixels sont souvent modifiés de $+1$ ou -1 cela représente une signature d'une image JPEG décompressée puis utilisée pour la stéganographie.

Plus récemment, une signature assez similaire, basée également sur ce qui a été appelé la « compatibilité avec la compression JPEG » a été proposée dans (Butora and Fridrich 2019). Il s'agit dans ce cas de détecter la stéganographie dans les images compressées au format JPEG en utilisant le fait que modifier les coefficients DCT peut amener à produire des valeur de pixels qui ne sont pas possible pour une image naturelle. Nous décrirons plus en détail cette méthode de détection en fin de section, 1.3.

Les méthodes de détection par signature sont en général très fiables dans le sens où les erreurs sont peu fréquentes. Cependant, ce type de détection repose généralement sur une erreur d'implémentation (comme dans l'exemple de l'insertion du commentaire avec F5). Chaque version de chaque logiciel doit donc être minutieusement analysée afin de trouver des traces caractéristiques qui relèvent son utilisation. Ce type d'analyse est très chronophage et peu généralisable.

1.3. Détection par méthode statistique (5 pages)

Les méthodes de détection que nous verrons dans la suite de ce chapitre sont généralement moins fiables que les méthodes de détection par signature, mais présentent l'avantage d'être beaucoup plus générale dans le sens où elle vise à détecter les modifications liées à la dissimulation d'information dans le contenu même d'un média : il n'est pas possible d'y échapper. Enfin, elles sont plus intéressant sur le plan méthodologique.
Nous commencerons par décrire les méthodes statistiques, qui sont simples et permettent de comprendre comment améliorer la stéganographie à l'aide de la stéganalyse. Considérons une image $\mathbf{X}$ de taille $M \times N$, que l'on considérera comme une matrice de pixels codés sur 8 bits $x_{m,n} \in \{0; \ldots; 255\}$, la stéganalyse consiste à choisir entre les deux hypothèses suivantes : « Les pixels $x_{m,n}$ sont issus d'une image de couverture » et « Les pixels $x_{m,n}$ sont issus d'une image stéganographiée ». La principale difficulté réside alors de définir précisement ce qui caractéristent et ce qui différencient images de couvertures (naturelles, issues d'un appareil photographique) et aux images stéganographiées.

1.3.1. *Test statistique du χ^2*

Historiquement, la première approche pour aborder ce problème a été proposée dans (Westfeld and Pfitzmann 1999). Ne pouvant décrire précisément ce qu'est une image naturelle, il a été proposé de modéliser les pixels après utilisation de la stéganographie. Pour expliquer le fonctionnement de ce test, il nous faut rappeler comment procède l'insertion d'un message ; pour insérer dans le pixel $x_{m,n}$ le i-ième bit du message $m_i \in \{0; 1\}$ la stéganographie par substitution des bits de poids faibles modifie le pixel de la façon suivante

$$z_{m,n} = x_{m,n} - \text{LSB}(x_{m,n}) + m_i, \qquad [1.2]$$

avec $z_{m,n}$ le pixel de l'image stéganographiée et $\text{LSB}(x_{m,n})$ le bit de poids faible de $x_{m,n}$. On considère généralement en stéganalyse que le message inséré $\mathbf{m} = (m_1, \ldots, m_I)$ est une suite de bits tous indépendants et identiquement distribués (i.i.d) suivant une loi uniforme : $\mathbb{P}[m_i = 0] = \mathbb{P}[m_i = 1] = 1/2$.
La conséquence de l'insertion du bit m_i dans le pixel $x_{m,n}$ est que le pixel stéganographié peut se modéliser par la distribution de probabilité suivante :

$$\mathbb{P}[z_{m,n} = x_{m,n}] = \frac{1}{2} = \mathbb{P}[z_{m,n} = \bar{x}_{m,n}], \qquad [1.3]$$

avec $\bar{x} = x + (-1)^x$ la valeur entière x dont le bit de poids faible a été inversé.
Si l'on considère, en outre, que la totalité des pixels est utilisée pour dissimuler

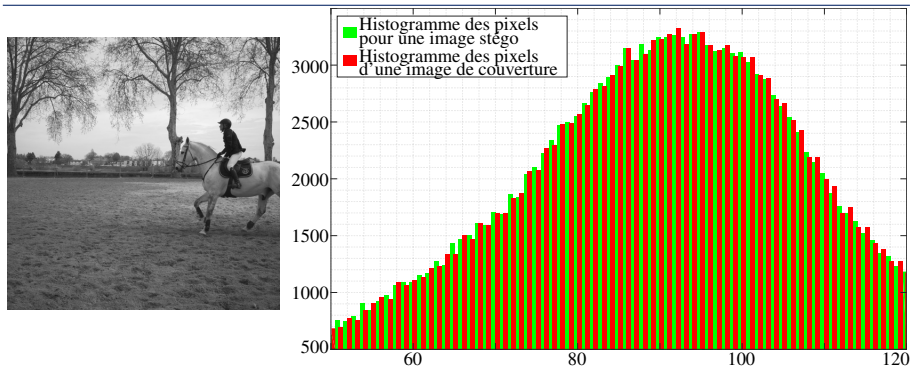


Figure 1.1: Illustration de l'impact de la stéganographie par LSBR et sa détection par le test du χ^2.

un message secret (très) volumineux, on peut modéliser la distribution des pixels stéganographiés de la façon suivante :

$$\forall k \in \{0;\ldots;255\}\,,\ \forall (m,n)\,,\ \mathbb{P}\left[z_{m,n}=k\right]=\mathbb{P}\left[z_{m,n}=\bar{k}\right]. \qquad [1.4]$$

On peut alors tester statistiquement si une image est stéganographiée en mesurant la différence entre la distribution théorique [1.4] et celle observée sur une image analysée. C'est justement l'objet du test du χ^2, dans le présent cas, mesure la différence entre distribution théorique [1.4] et observations de la façon suivante :

$$\chi^2=\sum_{k=0}^{255}\frac{(N_k-N_k^\star)^2}{N_k^\star} \quad \text{avec} \quad N_k^\star=\frac{N_k+N_{\bar{k}}}{2} \qquad [1.5]$$

où N_k représente le nombre de pixels dont la valeur est k et $N_k^\star$ représente le nombre attendu de pixels avec la valeur k ou $\bar{k}$. On reconnaitra dans la formule [1.5] une mesure de la différence entre distribution théorique et empirique : $(N_k-N_k^\star)^2$. La Figure 1.1 illustre le modèle de distribution [1.4] et [1.5] pour une image stéganographiée ; on constate en effet un nombre de pixels k et $\bar{k}$ qui se trouvent égalisés par la stéganographie.
Lorsque la valeur du χ^2 est supérieure à un seuil donné τ, l'image sera jugée comme étant de couverture, ou statistiquement trop différente de la distribution théorique pour une image stéganograhiée [1.4]. Comment doit-on alors fixer le seuil τ ? Les auteurs dans (Westfeld and Pfitzmann 1999) proposent de choisir ce seuil afin d'assurer que, théoriquement, une image stéganographiée soit jugée comme naturelle avec une probabilité p_0 (probabilité dite de « détection manquée » ou de

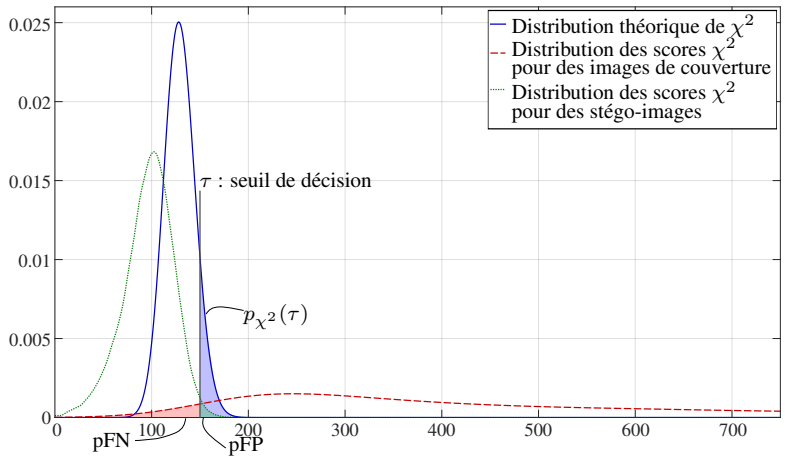


Figure 1.2: Illustration des distributions de probabilités (empiriques et théoriques) pour le résultats du test dû χ^2 [1.6] et probabilité d'erreur qui en résultent.

« faux-négatif »). Pour cela les auteurs utilisent la distribution du χ^2 qui permet de calculer cette probailité avec la relation suivante :

$$p_{\chi^2}(\tau) = \mathbb{P}\left[\chi^2 > \tau\right] = 1 - \int_0^{\tau} \frac{t^{(\nu-2)/2} e^{-t/2}}{2^{\nu-2}\Gamma(\nu)} dt, \qquad [1.6]$$

avec ici $\nu = 128 - 1$ le « nombre de degrés de liberté » ; On notera que $(N_k - N_k^\star)^2 = (N_{\bar{k}} - N_k\star)^2 = 1/4(N_k - N_{\bar{k}})^2\star$; le nombre de degrés de liberté est donc défini par le nombre de « paires de valeurs » qui peuvent être permutées entre elles.
La figure 1.2 illustre cette façon de fixer le seuil de décision, en fonction d'une probabilité p_0 préalablement fixée de « faux-négatif ». Elle permet également de comparer la distribution théorique de la statistique χ^2 [1.5] avec la distribution théorique du χ^2. La différence entre les observations et le modèle théorique est dû au fait que la très grande majorité des images ne possèdent pas des pixels avec tous les valeurs entre 0 et 255 et ont donc un nombre de degré de liberté inférieur à $\nu = 128 - 1$.

La force du test dû χ^2 est de proposer un test statistique sans avoir à résoudre le problème (très délicat) de modéliser la distribution statistique des pixels d'une image de couverture. En effet le test propose essentiellement de vérifier si les pixels d'une image analysée répondent au modèle d'une image stéganographiée ; dans le cas contraire l'image est jugée, par défaut, comme étant de « de couverture ». Un autre intérêt de ce test est de tenter de fixer le seuil de détection τ sur la base d'une probabilité de détection manquée $p_{\chi^2}(\tau)$ [1.6].
Malheureusement, ce test n'est pas très performant, notamment, car il ne modélise par une image de couverture. De façon générale en détection statistique, lorsque seule une

des deux hypothèses peut être caractérisée, on procède à un test du type « goodness-of-fit », adéquation des observations à ce modèle et cela est généralement moins fiable que l'on peut caractériser exactement les deux hypothèeses en concurrence.

1.3.2. *Test du rapport de vraisemblance*

Avant d'aller plus loin, formalisons les choses : un test est une fonction δ qui, à partir d'un ensemble d'observations $\mathbf{X}$ retourne une décision binaire : $\delta : \mathbf{X} \to \{0, 1\}$ de sorte que l'hypothèse $\mathcal{H}_0$ est acceptée si $\delta(\mathbf{X}) = 0$. Rappelons brièvement qu'un test n'est jamais parfait et que deux types d'erreurs sont possibles : faux-positif et faux-négatif (ou fausse-alarme et non-détection). En général, l'hypothèse $\mathcal{H}_0$ est dite « nulle » et correspond au cas « normal » alors que l'hypothèse alternative $\mathcal{H}_1$ correspond à la situation problématique qu'il est souhaité détecter. Le faux-positif correspond donc au cas où le test décide d'accepter l'hypothèse $\mathcal{H}_1$ alors qu'en vérité les observations sont issues de l'hypothèse $\mathcal{H}_0$. Dans le cas qui nous intéresse, l'image analysée est une image de couverture qui est classée comme stéganographiée. À l'inverse, un faux-négatif correspond au cas où le test accepte l'hypothèse $\mathcal{H}_0$ alors que les observations sont réellement issues de l'hypothèse alternative $\mathcal{H}_1$; pour la stéganalyse, c'est le cas où la gardienne Wendy laisse passer une image stéganographiée.

Afin de monter la faiblesse du test du χ^2 nous devons disposer d'un modèle statistique des images de couverture ; ce dernier est construit en supposant les pixels sont statistiquement indépendant et tous issus d'une distribution Gaussienne :

$$x_{m,n} \sim \mathcal{N}\left(\mu_{m,n}, \sigma^2_{m,n}\right), \qquad [1.7]$$

où $\mu_{m,n}$ représente l'espérance mathématique du pixel (*i.e.* sa « moyenne » ou sa valeur théorique) et $\sigma^2_{m,n}$ représente la variance du bruit. C'est un modèle couramment utilisé en traitement des images qui suppose qu'une image peut être décomposée sous la forme d'un « contenu » théorique et d'un bruit liée aux imperfections de l'appareil photographie. Pour être plus précis, les pixels sont représentés par des valeurs entières et, par souci de simplicité, on supposera que cela n'a pas d'impact sur la distribution de probabilité des pixels :

$$\mathbf{p}_0 = \mathbb{P}\left[x_{m,n} = k\right] = \left(p_0(k)\right)_{k\in\mathcal{Z}} \propto \left(\frac{1}{\sigma_{m,n}\sqrt{2\pi}} \exp\left(\frac{(k-\mu_{m,n})^2}{2\sigma^2_{m,n}}\right)\right)_{k\in\mathbb{Z}} \qquad [1.8]$$

Nous allons considérer dans un premier temps le cas d'un stéganographie qui utilise la stéganographie par substitution (non adaptative) de LSB. Chaque pixel peut

Résultat \ Vérité	Hypothèse 0 : (image de couverture)	Hypothèse 1 : (stégo-image)
Accepter l'hypothèse 0	Décision correcte	Faux-négatif (détection manquée)
Accepter l'hypothèse 1	Faux-positif (PFA : $\alpha = \mathbb{P}[\delta(\mathbf{X}) = \mathcal{H}_1 \| \mathcal{H}_0]$)	Décision correcte ($\mathbb{P}[\delta(\mathbf{X}) = \mathcal{H}_1 \| \mathcal{H}_1]$)

Figure 1.3: Les différentes possibilités de bonnes et de mauvaises détection.

être modifié avec la même probabilité de $\frac{\beta}{2} = \frac{I}{2MN}$ qui correspond au ratio du nombre de bits du message inséré (I) par pixel (MN) :

$$p_1(k) = \mathbb{P}[x_{m,n} = k|\mathcal{H}_1] = \left(1 - \frac{\beta}{2}\right) p_0(k) + \frac{\beta}{2} p_0(\bar{k}). \quad [1.9]$$

Comment utiliser ces modèles statistiques pour décider si une image inspectée $\mathbf{X}$ provient plutôt du modèle définissant une hypothèse $\mathcal{H}_0$ [1.8] ou une autre, $\mathcal{H}_1$ [1.9]. Il existe plusieurs réponses qui ont en commun un élément central, le rapport de vraisemblance (RV) qui s'exprime :

$$\Lambda(\mathbf{X}) = \frac{p_1(\mathbf{X})}{p_0(\mathbf{X})} = \prod_{m,n} \frac{p_1(x_{m,n})}{p_0(x_{m,n})} \quad [1.10]$$

avec p_0 et p_1 les modèles statistiques de distribution pour des images de couverture [1.8] et stéganographiées [1.9]. La seconde partie de cette égalité découle directement du modèle d'indépendance statistique entre les pixels qui, même si cela n'est pas totalement exact, est très largement utilisé, car il simplifie beaucoup les choses.
Clairement ; le rapport de vraisemblance est d'autant plus grand que la probabilité d'observée les données à analyser est plus grande sous $\mathcal{H}_1$ que sous $\mathcal{H}_0$; à l'inverse, si la probabilité d'observée ces données est beaucoup plus grande sous $\mathcal{H}_0$ que sous $\mathcal{H}_1$, le rapport de vraisemblance [1.10]. Basé sur cette observation, le test du rapport de vraisemblance consiste simplement à seuiller ce rapport de vraisemblance : $\delta(\mathbf{X}) = \mathcal{H}_0$ si $\Lambda(\mathbf{X}) < \tau$ et $\delta(\mathbf{X}) = \mathcal{H}_1$ si $\Lambda(\mathbf{X}) \geq \tau$.
Le test du rapport de vraisemblance est notamment utilisé, car le Lemme de Neyman-Pearson stipule qu'il permet d'atteindre la plus grande puissance ς pour une probabilité de faux-positif α fixée à $\alpha = \mathbb{P}[\Lambda(\mathbf{X}) > \tau]$; inversement, cette dernière relation permet de fixer le seuil afin de respecter un taux de faux-positif préalablement établi.

En utilisant les modèles des images de couvertures [1.8] et stéganographiées [1.9], le rapport de vraisemblance s'écrit :

$$\Lambda(\mathbf{X}) = \prod_{m,n} (1 - \beta) + \beta \frac{p_0(x_{m,n}) + p_0(\bar{x}_{m,n})}{2p_0(x_{m,n})}.$$

Par simplicité on utilise généralement le logarithme du RV, afin de remplacer le produit par une somme dans [1.11] ; par ailleurs, en utilisant la définition de $p_0(x_{m,n})$ [1.8], un calcul (un peu fastidieux) permet de simplifier la relation précédente comme suit (Cogranne 2011) :

$$\log\big(\Lambda(\mathbf{X})\big) = \sum_{m,n} \beta \log\big(\Lambda(x_{m,n})\big) = \sum_{m,n} \beta \frac{(x_{m,n} - \mu_{m,n})(x_{m,n} - \bar{x}_{m,n})}{2\sigma^2_{m,n}} \quad [1.11]$$

Dans cette approche, le plus intéressant n'est pas pas tant de disposer d'une relation simple permettant de calculer le rapport de vraisemblance (bien que quelques simplifications aient dû être faites), mais davantage de pouvoir caractériser la distribution statistique de ce dernier et donc, *in fine*, de maîtriser les probabilités de fausses-alarmes et de non-détection. En particulier, on peut montrer que pour une image de couverture, l'espérance mathématique et la variance du terme $\log\big(\Lambda(\mathbf{X})\big)$ sont données par :

$$\mathbb{E}\left[\log\big(\Lambda(\mathbf{X})\big)|\mathcal{H}_0\right] = 0 \quad \text{et} \quad \mathbb{V}ar\left[\log\big(\Lambda(\mathbf{X})\big)|\mathcal{H}_0\right] = \frac{\beta^2}{4\sigma^2_{m,n}}. \quad [1.12]$$

Il est possible de normaliser le RV comme suit :

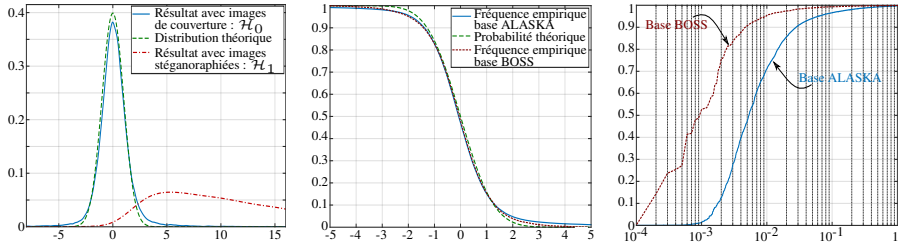
$$\log\left(\Lambda(\mathbf{X})\right) = \frac{1}{\varrho}\sum_{m,n} \log\left(\Lambda(x_{m,n})\right) \quad \text{avec} \quad \varrho = \left(\sum_{m,n} \frac{\beta^2}{4\sigma^2_{m,n}}\right)^{1/2}, \quad [1.13]$$

de sorte que, en vertu du théorème de la limite centrale (Lehmann and Romano 2005, Theorem 11.2.5) il est possible de calculer la distribution statistique du log-RV pour une image donnée :

$$\begin{cases} \text{sous } \mathcal{H}_0 : & \log\left(\Lambda(\mathbf{X})\right) \sim \mathcal{N}\left(0,1\right), \\ \text{sous } \mathcal{H}_1 : & \log\left(\Lambda(\mathbf{X})\right) \sim \mathcal{N}\left(\varrho,1\right). \end{cases} \quad [1.14]$$

La distribution Gaussienne étant assez simple à « manipuler » on peut aisément calculer le seuil de décision τ afin d'assurer une probabilité de faux-positif, PFA, α_0 fixée : $\tau(\alpha_0) = \Phi^{-1}(1-\alpha_0)$ de sorte que $\mathbb{P}\left[\log\left(\Lambda(\mathbf{X})\right) > \tau(\alpha_0)|\mathcal{H}_0\right] = \alpha_0$; de même la puissance du test ς en calculant la probabilité $\mathbb{P}\left[\log\left(\Lambda(\mathbf{X})\right) > \tau|\mathcal{H}_0\right] = 1 - \Phi\left(\tau - \varrho\right)$.

De nombreuses remarques sont nécessaires pour comprendre ces différents résultats. D'une part, notons qu'il est ici très « spécifique » que l'espérance mathématique du RV pour une image stéganographiée soit identifique à sa variance. C'est pour cette



(a) Distribution du log-RV pour des images de couvertures et stéganographiées.

(b) Proba. de faus-positif (PFA) α en fonction du seuil de décision ; théorie et résultats empiriques.

(c) Courbe COR représentant la puissance ς en fonction de la PFA α.

Figure 1.4: Résultats de l'application du test du RVG [1.13]

raison spécifique que le facteur de normalisation ϱ dans [1.13] correspond à la fois à l'espérance dans [1.14].

On notera également, par souci de clarté, certaines simplifications sont omises, le lecteur intéressé pour trouver des détails dans (Cogranne 2011). Mais surtout, tous les calculs précédents nécessitent de connaître l'espérance mathématique $\mu_{m,n}$ et la variance $\sigma^2_{m,n}$ de chacun des pixels, or, en pratique, ces variables sont inconnues. Il est donc proposé de les remplacer par des estimations, ces estimations sont assez délicates à obtenir de façon « précise », et c'est là que l'application de cette approche devient beaucoup plus compliquée et notamment que la garantie des probabilités d'erreur devient très difficile. Quelques résultats sont présentés dans les figures 1.4 en utilisant la base d'images ALASKA (Cogranne *et al*. 2019). La figure 1.4a montre que la distribution du log-RV obtenu pour des images de couvertures est plutôt cohérente avec la théorie ; pour les images stéganographiées, cela dépend du facteur ϱ qui varient pour chaque image créant cet « étalement ». La figure 1.4b compare la probabilité de faux-positif en fonction du seuil de décision τ en théorie et en pratique en utilisant deux bases d'images différentes. On on constate que cela fonctionne pour des probabilités « assez élevées » ; si l'on souhaite typiquement obtenir de très faibles taux de faux-positif, les estimations ne seront pas assez précises pour offrir des garanties pertinentes. Enfin, la figure 1.4c montre les performances obtenues au travers d'une courbe COR (Caractéristiques Opérationelles de Réception) qui présente la probabilité de faux-positif $\alpha_0(\tau)$ en fonction de la puissance de détection $\varsigma(\tau)$. On constate que pour un taux d'insertion $\beta \approx 0.09$ (24 kilobits insérés dans des images de 512x512 pixels) les performances sont plutôt très bonnes sur les deux bases utilisées.

Enfin, on notera que le test du RV peut être comparé avec le test du WS initialement proposés en 2004 par (Fridrich and Goljan 2004) pour estimer la taille du message inséré par moindres-carrés. Là encore dans ce test on n'utilise pas directement la valeur des pixels, mais une estimation $\widehat{\mu}_{m,n}$ pour finalement estimer β ainsi :

$$\widehat{\beta} = \frac{2}{N} \sum_{m,n} w_{m,n}(x_{m,n} - \widehat{\mu}_{m,n})(x_{m,n} - \bar{x}_{m,n}) \qquad [1.15]$$

avec $w_{m,n}$ un poids qui permet, selon les auteurs, de « moduler l'importance » donnée à chaque pixel en fonction de la précision de l'estimation $\mu_{m,n}$.On remarque clairement que cette estimation est très similaire au test du RV tel qu'écrit dans [1.11] ; cependant les performances du WS n'ont jamais été étudiées autrement qu'empiriquement.

1.3.3. *Détection de correspondance de LSB*

Fondamentalement, les détecteurs présentés dans la section 1.3.2 montrent que la stéganographie par substitution des bits de poids faibles (LSBR) introduit clairement un biais en incrémentant les valeurs paires et en décrémentant les valeurs impaires. Cela explique pourquoi cette méthode d'insertion est clairement à éviter au profit de la correspondance (LSBM ou LSB$\pm$1).
Après avoir présenté avec détail l'application d'un test statistique pour la stéganalyse, nous pouvons aborder la détection de LSBM ou LSB$\pm$1 qui a été largement moins étudiée dans la littérature au travers de tests « simples ». Par ailleurs, nous considérerons également le cas d'un schéma de stéganographie adaptatif, c'est-à-dire que la probabilité d'utiliser le pixel $x_{m,n}$ est différente pour chaque pixel (et cela en fonction d'une fonction de coût, voir [FIXME]) et sera donc notée $\beta_{m,n}$. Le lecteur notera qu'il est très facile de remplacer le « taux d'insertion moyen » β par un taux d'insertion pour chaque pixel $\beta_{m,n}$ sans aucune modification supplémentaire dans les équations [1.9] et [1.11]-[1.13]

Nous ne détaillerons pas tous les détails des calculs qui peuvent être trouvés dans, (Cogranne 2011 ; Sedighi, Cogranne and Fridrich 2016), mais le calcul du log-RV pour la correspondance de bit de poids faibles amène, après quelques simplifications, à l'expression suivante :

$$\Lambda^{\pm}(\mathbf{X}) = \sum_{m,n} \beta_{m,n} \Lambda^{\pm}(x_{m,n}) = \sum_{m,n} \beta_{m,n} \left(\frac{(x_{m,n} - \mu_{m,n})^2 - 1/12}{\sigma^4_{m,n}} - \frac{1}{\sigma^2_{m,n}} \right). \qquad [1.16]$$

Là encore, le plus intéressant est de calculer les probabilités d'erreurs de ce test et, en utilisant comme précédemment le théorème de la limite centrale, cela nécessite de connaître les deux premiers moments qui sont donnés par :

$$\begin{aligned} &\mathbb{E}\left[\Lambda^{\pm}(x_{m,n})|\mathcal{H}_0\right] = 0 \ , \ \mathbb{E}\left[\Lambda^{\pm}(x_{m,n})|\mathcal{H}_1\right] = \frac{2\beta_{m,n}^2}{\sigma_{m,n}^4}, \\ &\mathbb{V}ar\left[\Lambda^{\pm}(x_{m,n})\right] = \frac{2\beta_{m,n}^2}{\sigma_{m,n}^4}. \end{aligned} \quad [1.17]$$

La comparaison entre le RV pour la détection substitution de LSB [1.11] et de LSB$\pm$1 [1.16] permet de constater que la détection de LSBR (substitution) repose essentiellement sur un écart à l'espérance, au travers du terme $(x_{m,n}\mu_{m,n})$; à l'inverse la détection de LSB$\pm$1 repose essentiellement sur un écart entre la variance théorique et celle observée au travers du terme $(x_{m,n} - \mu_{m,n})^2$. Or si estimer l'espérance mathématique des pixels est délicat (c'est un problème de « débruitage » qui a un intérêt majeur pour l'amélioration des images et a donc été largement étudié), l'estimation précise de la variance des pixels est beaucoup plus difficile et a été nettement moins étudié. En outre, on souhaite détecter un écart à une variance qui n'est pas connue et doit être estimée, et cela, de façon indépendante de la présence d'informations cachées.
Si l'application de la détection statistique explique pourquoi la stéganalyse de LSB$\pm$1 est plus difficile, il faut noter que l'étude des performances montre également que pour un pixel donné, la « détectabilité » dépend essentiellement du « rapport insertion-sur-bruit » définit par $\beta_{m,n}^2/\sigma_{m,n}^4$ dans les relations [1.17].
Une application intéressante de la théorie des tests d'hypothèses a été d'utiliser ce résultat concernant la « détectabilité statistique » de chacun des pixels afin de concevoir un algorithme d'insertion(Sedighi, Cogranne and Fridrich 2016) qui, au lieu de minimiser une distorsion heuristique, minimise la détectabilité théorique. Bien que cela nécessite d'estimer l'espérance mathématique et la variance de chaque pixel (qui reste un problème ouvert), cela a montré son efficacité [FIXME].

Enfin, un dernier de détection statistique concerne le LSB$\pm$1 dans les images compressées avec le standard JPEG. En effet, la théorie des tests d'hypothèses peut être utilisée pour détecter la stéganographie dans les coefficients DCT d'une image JPEG. Cependant, on utilise généralement, on utilise un modèle de distribution des coefficients DCT différent, voir par exemple (Thai *et al.* 2014) qui présente un modèle statistique de ces coefficients et une application à la stéganalyse.
Récemment il a été proposé (Butora and Fridrich 2019) d'exploiter le fait, qui était resté assez peu connu, que les pixels sont quantifiés avant utilisation de la transformée en cosinus discrète (DCT). Pour une image compressée, il est donc possible de la décompresser (c'est-à-dire recalculer la valeur des pixels, à partir des coefficients DCT quantifiés) et de mesurer a variance du bruit de quantification par :

$$\frac{1}{MN}\left\|\mathbf{X} - \text{round}\big(\mathbf{X}\big)\right\|_2^2 = \frac{1}{MN}\sum_{m,n}\big(x_{m,n} - \text{round}\big(x_{m,n}\big)\big)^2 \quad [1.18]$$

avec, M et N le nombre de lignes et de colonnes de l'image $\mathbf{X}$, round$\big(\,\cdot\,\big)$ la fonction d'arrondie à la plus proche valeur entière. Si des informations ont été cachées dans les coefficients DCT, l'erreur d'arrondie dans le domaine spatial [1.18] aura tendance a augmentée.
Ce test est en fait très efficace (il a été légèrement amélioré dans (Cogranne 2020)) avec une détection de quelques centaines de bits « quasi-parfaite »), car il repose sur un modèle très précis et qui ne dépend pas de paramètres à estimer, mais uniquement du bruit de quantification qui, lui, est le même quel que soit l'image analysée. Cependant, il s'agit d'un cas très spécifique que l'on peut assimiler à de la détection par signature.

1.4. Détection par apprentissage supervisé (7 pages)

Nous allons à présent décrire des approches de stéganalyse qui sont radicalement différentes de ce qui a été précédemment décrit et qui repose sur des modèles de distribution statistique et sur la méthodologie des tests d'hypothèses. Cependant, dans bien des cas, dont la stéganalyse, il n'est pas possible de pouvoir établir un modèle statistique précis pour les observations.
Dans de pareils cas, il est usuel de recourir aux méthodes d'apprentissage statistique supervisées dont le fondement peut être décomposé en deux phases. Dans un premier temps il s'agit d'extraire des "caractéristiques" à partir des objets de la base considérée, des images en général dans le cas de la stéganographie. Ces caractéristiques doivent être révélatrice de la présence d'informations cachées (ou de ce que l'on souhaite détecter en général) et permette de réduire un objet complexe et variable comme une image, de taille quelconque, à un vecteur de p valeurs réelles. Sur la base de ces caractéristiques, une méthode d'apprentissage est utilisée pour déterminer une règle de décision qui fournira un résultat binaire qui sera le résultat de stéganalyse. Cet apprentissage est supervisé, car pour chaque image on associe, durant cette phase d'apprentissage, un label qui indique si l'image est stéganographiée ou de couverture. Déterminer une règle de décision correspond en fait à la résolution d'un problème d'optimisation qui chercher à associée à chaque vecteur de caractéristique une valeur aussi proche que possible du label.

Nous allons brièvement décrire dans ce chapitre comment ces deux phases (1) d'extraction de caractéristiques et (2) d'apprentissage supervisé sont mises en œuvre usuellement en stéganographie.

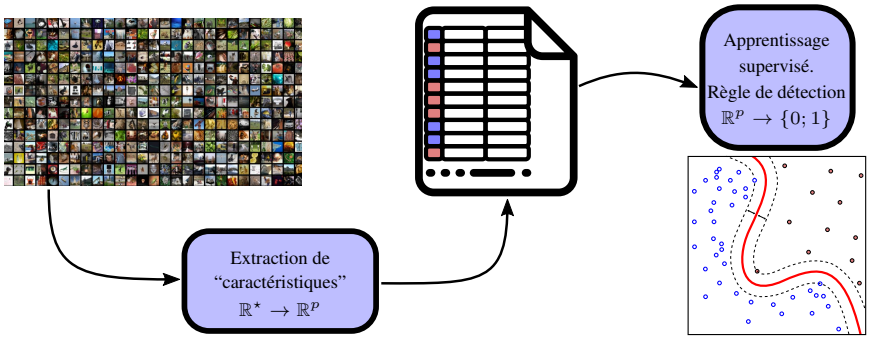


Figure 1.5: Illustration du principe "*l'apprentissage supervisé*" qui vise à déterminer une règle de détection à partir d'une base de données labélisée.

1.4.1. *Extraction de caractéristiques*

Il serait évidemment trop long de décrire en détail l'évolution de la stéganalyse pour aboutir aux techniques actuelles, mais notons globalement que une pierre angulaire dans méthodes actuelles trouve son origine (Pevný *et al*. 2010) qui exploités des différences entre pixels adjacents que nous noterons $\mathbf{D}^{\rightarrow}$:

$$d^{\rightarrow}_{m,n} = x_{m,n} - x_{m,n+1}, \qquad [1.19]$$

avec ici $\rightarrow$ qui représente la direction horizontale dans laquelle les différences sont calculées, parmi les 8 possibles et utilisées $\{\leftarrow, \rightarrow, \uparrow, \downarrow, \nwarrow, \searrow, \nearrow, \swarrow\}$. Sur la base de ces différences, il est proposé d'estimer la fréquence avec laquelle des différences successives se trouvent présentes dans une image. Représenter une fréquence empirique correspond à un dénombrement que l'on appelle, par un léger abus de langage, "histgramme" en stéganalyse. Ces dénombrements sont arrangés en vecteur :

$$f^{\rightarrow}_{k,l} = \sum_{m,n} \mathbb{1}\left[d^{\rightarrow}_{m,n} == k, d^{\rightarrow}_{m,n+1} == l\right] \ , \ (k,l) \in \{-T, \ldots, T\}\,, \qquad [1.20]$$

avec $\mathbb{1}[\cdot]$ la fonction indicatrice $\mathbb{1}[e] = 1$ si l'événement e est vraie et $\mathbb{1}[c] = 0$ dans le cas contraire et T le seuil de différence maximale considérée. Cette manière étonnante d'écrire le calcul d'un histogramme revient en effet à sommer des termes valant 1 uniquement sir des paires de "différences" adjacentes ont les valeurs recherchées ; il s'agit donc d'un dénombrement qui correspond à un "histogramme". Compte tenu que l'on dénombre les valeurs de différences adjacentes conjointement ces vecteurs sont appelés "co-occurrence" en stéganalyse. Ce concept peut être généralisé en utilisant plus que deux valeurs disjointes au prix d'une augmentation du

nombre de valeurs de co-occurence possibles. En effet en utilisant des co-occurrences de c différences adjacentes en utilisant un maximal de valeur de différence de T on obtient $(2T+1)^c$ valeur distincte de co-occurence possible. Enfin, une dernière étape proposée dans (Pevný *et al*. 2010) vise à limiter le nombre de caractéristiques en regroupant les valeurs calculées pour des directions opposées, par exemple $\leftarrow$ et $\rightarrow$ ou encore $\nwarrow$ et $\searrow$. Les auteurs proposent de regrouper directions "diagonales" et les directions horizontale et verticale :

$$f_{k,l} = \frac{1}{4}\left(f^{\rightarrow}_{k,l} + f^{\leftarrow}_{k,l} + f^{\uparrow}_{k,l} + f^{\downarrow}_{k,l}\right). \qquad [1.21]$$

Cette étape est appelée la "symétrisation" et a pour but de réduire le nombre de caractéristiques.
Les caractéristiques originales SPAM ("subtractive pixel adjacency matrix") (Pevný *et al*. 2010), utilisées des triples de 3 différences adjacentes, $c = 3$, $f_{k,l,m}$ (dite de second ordre) et 3 valeurs distinctes pour chacune $T = 3$ cela revient à dénombrer un nombre distinct de " co-occurrence" de $(3 \times 2 + 1)^3 = 7^3 = 343$. En ajoutant le fait que les co-occurrences diagonales sont distinguées des co-occurrences horizontales et verticales durant la phase de symétrisation on obtient un total de 686 caractéristiques. Clairement il s'agit là d'un nombre de caractéristiques (également appelées dimensions) plutôt élevées par rapport aux problèmes généralement étudiés dans le domaine de l'apprentissage statistique.

Cependant, il a été empiriquement constaté que, pour le cas particulier de la stéganalyse qui constitue essentiellement une détection d'un signal très faible dans un environnement (le média de couverture) de nature très hétérogène, généraliser cette approche permet d'améliorer la performance de détection.
Aussi, l'approche que nous avons brièvement décrit a été largement développée par la suite dans une méthodologie dont le principe de fonctionnement est illustré dans la figure 1.6. Cette figure illustre le fonctionnement des "modèles spatiaux riches" proposés dans (Fridrich and Kodovskỳ 2012). Il s'agit essentiellement d'un développement de la méthode proposée dans (Pevný *et al*. 2010). Ces modèles sont en outre assez standard et illustrent, eux aussi, le méthodes actuelles de stéganalyse qui reponsent globalement sur les 4 mêmes grandes étapes qui sont (1) le calcul de résidus (2) la quantification et le seuillage (3) le dénombrement de co-occurrence et enfin (4) la réduction de redondances par symétrisation.

Le calcul de résidus se fait par généralement par utilisation d'un filtre linéaire ; à partir d'une image $\mathbf{X}$ dont les pixels sont donnés par $x_{m,n}$ on applique la même relation entre pixels adjacents sur l'ensemble de l'image

$$r^{(n)}_{m,n} = \sum_{i,j} x_{m+i,n+j} k^{(n)}_{i,j} \qquad [1.22]$$

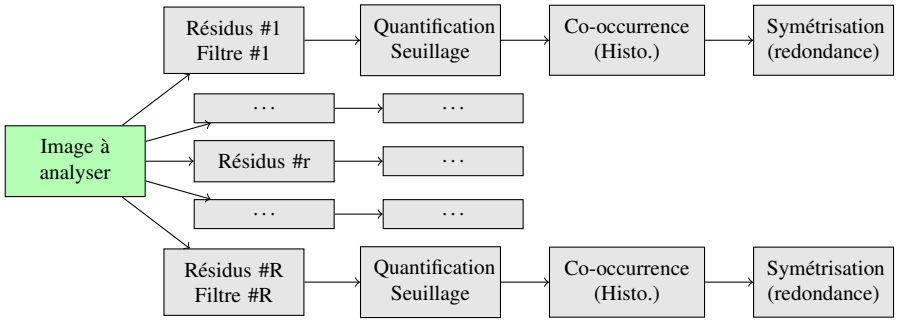


Figure 1.6: Illustration du principe de l'extraction des caractéristiques SRM "*Spatial Rich Model*".

Cette somme pondérée de pixels voisins est une opération de convolution $\mathbf{R}^{(n)} = \mathbf{X} \star \mathbf{K}^{(n)}$ dont le noyau $\mathbf{K}^{(n)}$ spécifie la valeur de ces facteurs de pondération. On parle généralement de filtre passe-bas lorsque $\sum_{i,j} k_{i,j} = 1$, il s'agit typiquement d'un "lissage" visant à diminuer le "bruit". En stéganalyse au contraire on parlera de "résidus" pour caractériser des filtrages passe-haut caractérisés par $\sum_{i,j} k_{i,j} = 0\,,\ k_{0,0} = -1$. En d'autres termes il s'agit "d'estimer" la valeur d'un pixel $x_{m,n}$ à partir de ces voisins $\widehat{x}^{(n)}_{m,n} = \sum_{(i,j)\neq(0,0)} x_{m+i,n+j} k^{(n)}_{i,j}$ puis de calculer la différence $r^{(n)}_{m,n} = \widehat{x}^{(n)}_{m,n} - x_{m,n}$ qui correspond à "l'erreur d'estimation" due au filtre $\mathbf{K}^{(n)}$.
Afin de pouvoir capturer toutes les traces possibles dues à la stéganographie on multiplie généralement les filtres ; à titre d'exemple, 78 résidus sont utilisés dans (Fridrich and Kodovskỳ 2012).

La seconde étape est la quantification et le seuillage. De façon générale, les facteurs de pondération utilisée dans les noyaux $\mathbf{K}^{(n)}$ ne sont pas des entiers et les résidus $\mathbf{R}^{(n)}$ doivent être regroupées en valeurs "proches" afin de pouvoir calculer des "histogrammes". Le principe général est alors de diviser les résidus $\mathbf{R}^{(n)}$ par un pas de quantification q et d'arrondir ensuite les résultats. Le pas de quantification q détermine alors la "granularité" de l'histogramme. Un pas très grand fournit des histogrammes grossiers ; par exemple un pas de $q = 10$ conduit à un arrondi à la dizaine la plus proche au contraire, un pas de $q = 1$ conduit à un arrondi à l'entier le plus proche. Le seuillage consiste simplement à limiter l'intervalle des valeurs possible (malgré la quantification) afin de limiter la dimension des "co-occurence" qui en découle. Il est simplement proposé de ne compter que les valeurs de résidus en dessous d'un certain seuil T et d'ignorer les valeurs au-delà.
Cette seconde étape peut finalement se représenter

$$\widetilde{r}^{(n)}_{m,n} = \text{Seuil}_T \left(\text{Round}\left(\frac{r^{(n)}_{m,n}}{q} \right) \right), \qquad [1.23]$$

avec Seuil_T l'opération de seuillage avec le seuil T et Round l'opération d'arrondie à l'entier le plus proche.
Dans les modèles riches spatial (Fridrich and Kodovskỳ 2012), 3 pas de quantification distincts sont utilisés ($q = \{1, 2, 3\}$), mais le seuil est toujours fixé à $T = 3$ cela afin de pouvoir représenter des valeurs de résidus plus ou moins grandes suivant le pas de quantification.

La troisième étape vise à représenter les "résidus" au travers d'histogramme à plusieurs dimensions ou de co-occurrence. Cela permet principalement de mieux représenter les propriétés statistiques globale des résidus, indépendamment de lors position dans une image, mais également de réduire le nombre de données et enfin d'avoir une représentation identique quelle que soit la taille de l'image analysée. Le calcul de co-occurrence pour un type de résidus donnés $\mathbf{R}^{(n)}$ se fait en procédant, comme expliqué précédemment, en dénombrant le nombre valeur adjacente dont les valeurs sont toutes égales à un certain motif recherché. Il s'agit donc d'une généralisation de la relation [1.20] à un tuple de c résidus adjacents. On notera que si l'on utilise c résidus adjacents, on peut également multiplier les direction (qui peuvent être bien plus générale que simplement les 8 directions considérées dans le cas de paires). On se limite généralement aux directions verticales et horizontales, car l'ensemble des parcours possibles deviendrait beaucoup trop important avec des co-occurrence de dimension $c = 4$ tel qu' utilisé dans (Fridrich and Kodovskỳ 2012).

Enfin la dernière étape, consiste à fusionner, au travers généralement d'un calcul de moyenne, les co-occurrences semblables, typiquement verticales vers la gauche ou vers la droite, Le principe sous-jacent étant qu'il n'y a pas raison objective de supposer que les traces de la stéganographie soient représentées différemment dans une direction. La symétrisation est une étape plus délicate à définir de façon générale, car cela dépend largement des filtres utilisés pour calculer pour les résidus ainsi que des co-occurrences utilisées. Généralement on agglomérera les différentes directions (co-occurrences verticales et horizontales) ainsi que les filtres qui peuvent se déduire les uns des autres par symétries.
Cette dernière étape permet de largement réduire le nombre de caractéristiques utilisées. En effet, considérons le cas classique des caractéristiques issues des modèles spatiaux riches (Fridrich and Kodovskỳ 2012). On dénombres 78 filtres distincts avec 3 pas de quantifications, un seuillage avec le seuil $T = 2$ et des co-occurrences calculées horizontalement et verticalement. On se retrouve donc au final (sans symétrisation) avec $78 \times 3 \times (2 \times 2 + 1)^4 \times 2 = 292\,500$ caractéristique (soit plus qu'il n'y a de pixels dans les images de la base de références BOSS (Bas *et al*. 2011) !).

Cas particulier des images JPEG

En ce qui concerne les images compressées au format JPEG, l'extraction de caractéristiques pose un premier problème : faut-il utiliser directement les coefficients DCT, qui peuvent être modifiés, ou bien est-il préférable de "décompresser" l'image

afin d'analyser les pixels.
Les deux approches ont été étudiées et, aujourd'hui, la détection dans les images décompressées présente des performances plus intéressantes. Cela est notamment dû au fait que la modélisation et l'analyse de pixels contigus sont bien plus simples que les coefficients DCT qui, bien qu'étant voisins sont le résultat de filtres différent et présente donc des propriétés qui ne permettent pas une analyse aisément.

Un compromis entre ces deux domaines d'analyse, a été proposée récemment (Holub and Fridrich 2015) ; nous allons brièvement la décrire, car elle est désormais commune, efficace et a donné lieu à de nombreux travaux ultérieurs, notamment (Song *et al*. 2015).
Cette approche appelée DCTR (pour "DCT Résiuals") analyse une image en commençant par la décompresser puis applique aux pixels une transformation DCT similaire à celle utilisée dans la compression JPEG. Plus exactement, nous avons brièvement expliqué que la compression JPEG applique une série de 64 (8×8) filtre orthogonaux à des blocs disjoints de 64 (8×8) pixels. Le principe de DCTR repose sur l'utilisation de ces 64 filtres, mais ces filtres sur utilisés sur des blocs non-disjoints ; plus exactement ces filtres sont appliqués par convolution avec l'image décompressée. Cela revient à calculer des coefficients DCT en plus de ceux utilisés dans la compression JPEG (sur des portions de blocs 8×8 adjacents). Le résultat de cette opération est donc 64 images distinctes, de la même taille que l'image d'origine, chacune correspondant à l'application d'un des filtres de transformation DCT.
Cette étape peut être assimilée à la phase de calcul de résidus.

Ces images sont quantifiées, modifiées en valeurs absolues (les valeurs négatives sont reportées en valeurs positives) et souillées ; la quantification est d'autant plus grande que le facteur de qualité est petit (de façon similaire aux matrices de quantification JPEG) et le seuillage est fixé à 4.
Pour chacune de ces 64 "sous-images" plusieurs histogrammes sont calculés en fonction de la position des résidus. L'idée est clairement de donnée une "référence" comparable aux coefficients DCT qui sont utilisés dans la compression JPEG (et donc possiblement modifié par la stéganographie) avec ceux ne sont pas utilisés, mais qui correspondent à des images qui sont très similaires, mais décalées de quelques pixels.

Dans le cas DCTR il n'y a pas réellement de symétrisation, cependant certaines positions de "résidus DCT" sont rassemblées ; on se retrouve au final avec 25 histogrammes par filtres DCT et $4 + 1$ valeurs dans l'histogramme seuillé (soit au final $64 \times 25 \times 5 = 8000$ caractéristiques).

1.4.2. *Classification des caractétistiques*

Une fois les caractéristiques extraites, il reste une phase de classification à mettre en œuvre. Plus exactement, il faut extraies ces caractéristiques pour un grand nombre

d'images que l'on sait sans information cachée ; pus cacher des informations dans ces images à l'aide d'une méthode de stéganographie et re-extraire les mêmes caractéristiques à partir des mêmes images qui contiennent désormais des informations cachées. L'apprentissage d'une règle de classification, dans ce cadre, fait parti de l'apprentissage statistique supervisé : c'est-à-dire que nous avons deux bases de caractéristiques, la première pour des images sans informations cachées (de la classe 0) et la seconde pour des images contenant des informations cachées (de la classe alternative 1).
Il existe de nombreuses méthodes d'apprentissage statistique supervisées ; en général cela correspond mathématiquement à un problème d'optimisation afin de rechercher les paramètres d'une fonction permettant de maximiser la performance en détection.
En stéganalyse les principaux classifieurs adaptés sont linéaires, nous nous focaliserons donc ce type de méthodes.
Pour l'exposé, considérons deux bases de L caractéristiques calculés sur I images que nous noterons sur forme matricielle $\mathbf{C} = (\mathbf{c}^{(1)}, \ldots, \mathbf{c}^{(I)})$ pour les images originales (de la classe 0, sans information cachée) et $\mathbf{S} = (\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(I)})$ pour les images stéganographiées, avec des informations cachées.

Un classifieur linéaire repose sur une projection des caractéristiques sur un vecteur de discrimination $\mathbf{w}$: $< \mathbf{w}, \mathbf{c}^{(i)} >= \mathbf{w}^\top \mathbf{c}^{(i)} = \sum_{l=1}^{L} \mathbf{w}_l \mathbf{c}_l^{(i)}$. Cette opération est une somme pondérée des caractéristiques. Le but est donc de trouver un vecteur des coefficients de pondération $\mathbf{w}$ qui permet, après projection, de discriminer les données des images originales et stéganographiées.
Un critère simple pour cela consiste à chercher les coefficients de pondération $\mathbf{w}$ de sorte que les vecteurs $\mathbf{c}_i$ sont proches de la valeurs -1 alors que les vecteurs $\mathbf{s}_i$ sont proches de la valeur 1.
Si l'on créer des grands vecteurs $\mathbf{y}_0$ et $\mathbf{y}_1$ le premier contenant I fois la valeur -1 et le second contient I fois la valeur 1. Il s'agit alors de trouver les facteurs de pondération $\mathbf{w}$ minimisant la différence entre les valeurs $\mathbf{w}^\top \times \mathbf{C}$ et $\mathbf{y}_0$ (et réciproquement entre $\mathbf{w}^\top \times \mathbf{S}$ et $\mathbf{y}_1$) : $\left\|\mathbf{w}^\top \mathbf{C} - \mathbf{y}_0\right\|_2^2 + \left\|\mathbf{w}^\top \mathbf{S} - \mathbf{y}_1\right\|_2^2$.
Cette méthode revient en fait à un problème de minimisation au sens des moindres carrés. Cette approche est particulièrement intéressante, car une solution analytique est donnée par :

$$\mathbf{w} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{y} \qquad [1.24]$$

ou $\mathbf{X}$ est une matrice regroupant les caractéristiques $\mathbf{X} = (\mathbf{C}, \mathbf{S})$ et, de façon similaire, le vecteur $\mathbf{y}$ contient toutes les valeurs de labels $\mathbf{y} = (\mathbf{y}_0; \mathbf{y}_1)$.
Pour que cette méthode fonctionne correctement, il est généralement utile d'ajouter un facteur de régularisation, qui permet de trouver un compromis entre une solution "simple" et une solution adaptée aux données utilisées pour l'apprentissage. C'est cette méthode qui a été proposée dans (Cogranne *et al.* 2015) et qui permet d'obtenir des performances très proches de l'état de l'art en seulement quelques secondes (pour 10000 images et 40000 caractéristiques).

Une approche alternative intéressante reposant sur un autre classifieur linéaire, celui de Fisher, très similaire au précédent, mais qui vise à maximiser le critère de séparabilité :

$$\frac{\mathbf{w}^\top (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)}{\mathbf{w}^\top (\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1) \mathbf{w}}. \quad [1.25]$$

avec $\boldsymbol{\mu}_i$ la moyenne des caractéristiques pour les images de la classe i et $\boldsymbol{Sigma}_i$ la matrice de covariance des caractéristiques de la classe i. Cette méthode présente également l'avantage d'avoir une solution directement calculable :

$$\mathbf{w} = (\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1)^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0). \quad [1.26]$$

Le principal intérêt de la méthode proposée dans (Kodovskỳ *et al*. 2012) consiste a utiliser non pas un mais une multitude de classifieurs. Pour cela de nombreux classifieurs qui sont entraîner sur des "sous-ensembles" d'images et/ou sur des "sous-ensembles" de caractéristiques. L'intérêt repose principalement sur le fait que cet entrainement spécifique et cette procédure de multiplication des classifieurs permet de construire, au final, une classifieur "robuste" et performant.
Cette méthode a été améliorée dans (Cogranne and Fridrich 2015), mais, dans les deux cas, les résultats sont légèrement meilleurs que ceux obtenus avec un classifieur linéaire simple pour une complexité calculatoire beaucoup plus importante (de l'ordre de 20 à 100 fois supérieur).

1.5. Détection par Deep Learning (7 pages)

Les réseaux de neurones sont étudiés depuis les années cinquante. Initialement, ils ont été proposés pour modéliser le comportement du cerveau. En informatique, en particulier en intelligence artificielle, ils ont été utilisés pendant 30 ans à des fins d'apprentissage. Il y a encore une dizaine d'années (Hinton and Salakhutdinov 2006), les réseaux de neurones étaient considérés comme ayant un temps d'apprentissage trop long et comme étant moins efficaces que les classifieurs comme les SVMs ou bien les forêts aléatoires.

Grâce aux avancées récentes dans le domaine des réseaux de neurones (Bengio *et al*. 2013), grâce à la puissance de calcul fournie par les cartes graphique (GPUs), et enfin grâce à la profusion de données, les approches d'apprentissage en profondeur ont été proposées comme une extension naturelle des réseaux neuronaux. Depuis 2012, ces réseaux profonds ont marqué profondément les domaines du traitement du signal et de l'intelligence artificielle, car leurs performances ont permis de surpasser les méthodes les plus performantes de l'époque, mais aussi de solutionner des problèmes que les scientifiques n'arrivaient pas à résoudre jusqu'à maintenant (LeCun *et al*. 2015).

En stéganalyse, pendant les dix dernières années, la détection d'un message caché dans une image a été majoritairement réalisée par le calcul d'un modèle riche (RM) (Fridrich and Kodovskỳ 2012) suivi d'une classification par un ensemble classificateur (EC) (Kodovskỳ *et al*. 2012). En 2015, la première étude utilisant un réseau de neurones convolutionnel (CNN) a obtenu des premiers résultats de stéganalyse par "deep-learning" approchant les résultats des approches en deux étapes (EC+RM[1]) (Qian *et al*. 2015). Dès lors, sur la période 2015 - 2018, de nombreuses publications ont montrés qu'il était possible d'obtenir de meilleurs performances en stéganalyse spatiale, en stéganalyse JPEG, en stéganalyse informé, en stéganalyse quantitative, etc.

Dans la section 1.5.1 nous présentons de manière générique la structure d'un réseau de neurones profond. La lecture de cette section peut être complétée par des lecture sur l'apprentissage artificiel, et notamment sur la définition du Perceptron, la descente de gradient, la descente de gradient stochastique, l'extension au cas multi-classe. Ensuite, dans la sous-section **??** nous reviendrons sur les différents réseaux qui ont été proposés pendant sur la période 2015-2019 pour différents scénarios de stéganalyse.

Notez que ce chapitre est centré sur les apports du deep-learning pour la stéganographie et la stéganalyse du point de vue des publications dans la discipline. Ainsi, certaines avancées en deep-learning pouvant apporter à la discipline de la steganographie, sur la périodes 2015-2019, et proposés dans les conférences et journaux de machine-learning et de vision par ordinateur sont peut-être absentes.

1.5.1. *Les briques de bases d'un réseau de neuronne profond*

Dans les sections suivantes, nous rappelons les concepts majeurs d'un réseau de neurones convolutif (Convolutional Neural Network). Dans les sections suivantes, nous rappellerons les briques élémentaires d'un réseau en nous basant sur le réseau Yedroudj-Net qui a été publié en 2018 et qui reprend les idées présentes dans Alex-Net (Krizhevsky *et al*. 2012), ainsi que celles présentes dans des réseaux développés pour la stéganalyse dont le tout premier réseau de Qian *et al* ., et les réseaux de Xu-Net (Xu *et al*. 2016) et Ye-Net (Ye *et al*. 2017).

1.5.1.1. *Vue globale d'un réseau de neurones convolutif*

Avant de décrire la structure d'un réseau de neurones ainsi que ses briques élémentaire, il est utile de rappeler qu'un réseau de neurones appartient à la famille de l'apprentissage "machine". Dans le cas de l'apprentissage supervisé, qui est le cas qui

1. EC+RM désignera de manière générale les approches en deux étapes reposant sur le calcul d'un modèle riche (RM) puis l'utilisation d'un ensemble classifieur (EC).

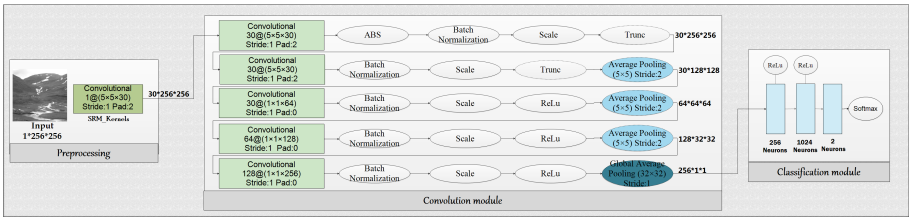


Figure 1.7: Le réseau Yedroudj-Net (Yedroudj, Comby and Chaumont 2018).

nous concerne, il est nécessaire de disposer d'une base de données d'images, avec, pour chaque image, son label, c'est-à-dire sa classe.

Les réseaux d'apprentissage profond sont de grands réseaux neuronaux qui peuvent directement prendre des données brutes en entrée. En traitement d'image, le réseau est directement alimenté par les pixels formant l'image. Un réseau d'apprentissage en profondeur apprend ainsi, de manière jointe, à la fois les caractéristiques intrinsèque de l'image (on parle de *feature map* ou bien *d'espace latent*), et à la fois la frontière de séparation permettant la classification (on parle également de *plans séparateurs*).

Le protocole d'apprentissage est similaire aux méthodes classique d'apprentissage "machine". Chaque image est donnée en entrée du réseau. Chaque valeur de pixel est transmise à un ou plusieurs neurones. Le réseau est constitué d'un nombre donné de *blocs*. Un bloc est constitué de neurones qui prennent des valeurs réelles en entrée, effectuent des calculs, puis transmettent au bloc suivant les valeurs réelles calculées. Un réseau de neurones peut donc être représenté par un graphe orienté où chaque nœud représente une unité de calcul. L'apprentissage est alors réalisé en fournissant au réseau des exemples composés d'une image et de son label, et le réseau modifie les paramètres de ces unités de calcul (il apprend) grâce au mécanisme de back-propagation.

Les réseaux de neurones convolutif utilisé pour la stéganalyse sont majoritairement construit en trois parties, que nous appelerons *modules* : le module de pre-processing, le module de convolution, et le module de classification. À titre d'illustration, la figure 1.7 schématise le réseau proposé par Yedroudj *et al.* en 2018 (Yedroudj, Comby and Chaumont 2018). Le réseau traite les images en niveau de gris de taille 256×256 pixels.

1.5.2. *Le module de préprocessing :*

$$F^{(0)} = \frac{1}{12}\begin{pmatrix} -1 & 2 & -2 & 2 & -1 \\ 2 & -6 & 8 & -6 & 2 \\ -2 & 8 & -12 & 8 & -2 \\ 2 & -6 & 8 & -6 & 2 \\ -1 & 2 & -2 & 2 & -1 \end{pmatrix} \quad [1.27]$$

Nous pouvons voir sur la figure 1.7 que dans le module de pré-processing, l'image est filtré par 30 filtres passe-haut. L'utilisation d'un ou plusieurs filtres passe haut comme pré-traitement est présente dans la majorité des réseaux utilisés pour la stéganalyse durant la période 2015-2018. À titre d'exemple, le noyau *square S5a* (Fridrich and Kodovskỳ 2012) est donné à l'équation 1.27.

Cette étape préliminaire de filtrage permet au réseau de converger plus rapidement et est probablement nécessaire pour obtenir de bonnes performances lorsque la base d'apprentissage est trop petite (Yedroudj, Chaumont and Comby 2018) (seulement 4 000 paires cover/stego de taille 256×256 pixels). Les images filtrées sont ensuite transmisent au premier bloc de convolution du réseau. Le réseau *Yedroudj-Net* possède 5 blocs de convolution (Yedroudj, Comby and Chaumont 2018), comme les réseaux de Qian *et al.* (Qian *et al*. 2015) et de Xu *et al.* (Xu *et al*. 2016).

Notez que le récent réseau SRNet (Boroumand *et al*. 2019) n'utilise pas de pré-filtres fixes, mais apprend les valeurs des filtres. Cela nécessite donc d'avoir une base de données beaucoup plus importante (plus de 15 000 paires cover/stego de 256×256 pixels), et éventuellement d'utiliser un apprentissage préliminaire pour partir d'une bonne initialisation. Il y a d'ailleurs un débat dans la communauté pour savoir si l'on doit utiliser des valeurs de filtres fixées, ou initialiser les valeurs des filtres avec des valeurs pré-sélectionnées et ensuite affiner ces valeurs par apprentissage, ou alors avoir une initialisation aléatoire et laisser le réseau apprendre les valeurs des filtres. Début 2019, dans la pratique, c'est à dire en situation opérationelle (**?**), le meilleur choix est probablement lié à la taille de la base de données d'apprentissage (qui n'est pas nécessairement BOSS (Bas *et al*. 2011)), et à la possibilité d'utiliser un apprentissage par transfert ou curriculum.

1.5.2.1. *Le module de convolution :*

Afin d'éviter toute ambiguïté sur les termes, nous éviterons d'utiliser le terme *couche* ; on préferera le terme de d'*opérations* pour une fonction élémentaire (convolution, activation, …) et le terme de *bloc* pour un ensemble de ces opérations qui peuvent se succéder.
Au sein du *module* de convolution, on retrouve plusieurs unités de calcul macroscopique que nous nommerons *blocs*. Un *bloc* est composé d'unités de calculs qui prennent des valeurs réels en entrée, effectuent des calculs, puis renvoient des valeurs

réelles, qui sont fournies au bloc suivant. Concrètement, un *bloc* prend un ensemble de *feature maps* (= un ensemble d'images) en entrée et retourne un ensemble de *feature maps* en sortie (= un ensemble d'images). À l'intérieur d'un bloc, on retrouve un certain nombre d'opérations dont les quatre suivantes : la *convolution*, l'*activation*, le *pooling*, et enfin la *normalisation*.

Notons que le concept de neurone, tel que défini dans la littérature, avant l'émergence des réseaux convolutifs, est toujours présent, mais il n'existe plus dans les librairies de réseau de neurones, en tant que structure de donnée. Dans les modules de convolution, il faut imaginer un neurone comme une unité de calcul qui, pour une position dans la *feature map* prise par le noyau de convolution lors de l'opération de convolution, effectue la somme pondéré entre le noyau et le groupe de pixels considéré. Le concept de neurone correspond au produit scalaire entre les données d'entrée que sont les pixels et des données propre au neurone, c'est-à-dire le noyau de convolution, suivi de l'application d'une fonction de $\mathbb{R}$ dans $\mathbb{R}$ appelée la fonction d'activation. Ensuite, par extension, on peut considérer que le pooling et la normalisation sont des opérations propres aux neurones.

En ne comptant pas le bloc de prétraitement, le réseau Yedroudj-Net (Yedroudj, Comby and Chaumont 2018) a un module de convolution composé de 5 blocs de convolution, comme les réseaux de Qian *et al.* (Qian *et al*. 2015) et de Xu *et al.* (Xu *et al*. 2016). Le réseau Ye-Net (Ye *et al*. 2017) possède un module de convolution composé de 8 blocs de convolution. Le réseau SRNet (Boroumand *et al*. 2019) contient un module de convolution fait de 11 blocs de convolution.
Sur la figure 1.7, représentant le réseau Yedroudj-Net, le premier bloc du module de convolution génère 30 *feature maps*, chacune de taille 256×256 pixels. Notez que cela signifie qu'il y a 30 filtres et donc 30 convolutions qui sont appliquées sur le groupe d'image données en entrée (les 30 images filtrées) de taille 256×256 pixels. Dans chacun des 5 blocs, on retrouve une convolution, de la normalisation (Batch + Scale), de l'activation (ABS, trunc, ReLU), et du pooling (average ou global pooling). Nous détaillerons plus tard la convolution, la normalisation, l'activation et le pooling.

1.5.2.2. *Le module de classification :*

Le dernier bloc du module de convolution (voir section précédente) est connecté au *module de classification* qui est généralement un réseau de neurones *entièrement connecté* composé de un à trois blocs. Ce *module de classification* est souvent un réseau de neurone traditionnel où chaque neurone est complètement connecté à la *couche* de neurones suivante et à *couche* de neurones précédente.

Après ce réseau de neurones complètement connecté, on trouve souvent une fonction softmax qui permet de normaliser les deux sorties fournies par le réseau de sorte que les valeurs emises appartiennent à $[0, 1]$ et que leur somme vaut 1. La fonction softmax retourne donc un score d'appartenance à une classe, c'est-à-dire un score par neurone de sortie. Par abus de langage, les scores sont appelés probabilités. Nous

conserverons cette dénomination. Dans le scénario habituel de steganalyse binaire, le réseau délivre donc deux valeurs en sortie : l'une donnant la probabilité de classement dans la première classe (par exemple la classe cover), et l'autre donnant la probabilité de classement dans la seconde classe (par exemple la classe stego). La décision de classification est alors obtenue en renvoyant la classe avec la plus forte probabilité. Le réseau Yedroudj-Net (voir figure 1.7) fournit effectivement deux valeurs en sortie. Notons que juste avant ce *module de classification*, nous pouvons trouver une opération de *pooling* particulière, telle qu'un *global average pooling*, un *Spatial Pyramid Pooling (SPP)* (He *et al*. 2014), ou un *extracteur de moments statistiques* (Fuji-Tsang and Fridrich 2018), etc. Une telle opération de "pooling" renvoie un vecteur de taille fixe, c'est-à-dire une carte de caractéristiques de dimension fixe, et cela, quelle que soit la dimension de l'image en entrée du réseau. Le bloc venant après cette opération de "pooling" est alors toujours connecté à un vecteur de taille fixe, il a donc un nombre fixe de paramètres d'entrée. Il est ainsi possible de présenter au réseau des images de toutes tailles sans avoir à modifier la topologie du réseau. Cette propriété est disponible dans le réseau Yedroudj-Net (Yedroudj, Comby and Chaumont 2018), le réseau Zhu-Net (Zhang *et al*. 2020), ou le réseau (Fuji-Tsang and Fridrich 2018).

Notons que (Fuji-Tsang and Fridrich 2018) est le seul papier, à la date d'écriture de ce chapitre, fin 2019, qui se soit sérieusement penché sur la viabilité d'un réseau invariant à la dimension des images qu'on lui donne en entrée. Le problème reste cependant ouvert. La solution proposée dans (Fuji-Tsang and Fridrich 2018) est une variante du concept d'average pooling. Pour le moment, il n'y a pas eu assez d'études sur le sujet pour déterminer quelle est la bonne topologie du réseau, comment construire la base de données d'apprentissage, dans quelle mesure le nombre de bits inséré influe sur l'apprentissage, et dans quelle mesure l'apprentissage doit prendre en compte la square root law, ...

Ici, s'achève la rapide présentation d'un réseau de neurones convolutif vue à travers les publications majeures en stéganographie/stéganalyse.

1.5.2.3. *La stéganalyse avec connaissance du canal adjacent (SCA)*

Fin 2018, deux approches combinent la connaissance du canal de sélection, le SCA-Ye-Net (qui est la version SCA de Ye-Net) (Ye *et al*. 2017), et le SCA-SRNet (qui est la version SCA du SRNet) (Boroumand *et al*. 2019). L'idée est d'utiliser un réseau utilisé pour la stéganalyse non informée (Not-SCA), et d'injecter non seulement l'image à stéganalyser, mais aussi la carte de probabilité de modification. On suppose donc qu'Eve connaît, ou peut obtenir une bonne estimation (Sedighi and Fridrich 2015) de la carte de probabilité de modification, c'est-à-dire qu'Eve a accès aux informations du canal adjacent (SCA).

La carte de probabilité de modification est donnée au bloc de prétraitement SCA-Ye-Net (Ye *et al*. 2017), et de manière équivalente au premier bloc de convolution pour SCA-SRNet (Boroumand *et al*. 2019), mais les valeurs du noyau sont remplacées

par leur valeur absolue. Après la convolution, chaque carte de caractéristiques est additionnée de point à point à la carte de probabilité de modification filtrée correspondante. Notez que la fonction d'activation de cette première convolution (bloc de prétraitement pour SCA-Ye-Net ou premier bloc pour SCA-SRNet) est (si ce n'est pas déjà le cas) remplacée par un ReLU. Dans SCA-Ye-Net, la fonction d'activation de troncature (*truncated linear unit (TLU)* dans l'article) est, en effet, remplacée par ReLU. Ceci permet de propager (forward pass) "virtuellement" dans tout le réseau les informations relatives à l'image, et une autre relative à la carte de probabilité de modification.

Notez que cette procédure de transformation d'un Not-SCA-CNNN en SCA-CNN s'inspire de la propagation de la carte de probabilité de modification proposée dans (Denemark *et al*. 2016). Ces deux articles constituent une amélioration par rapport au précédent maxSRM Rich Models (Denemark *et al*. 2014). Dans maxSRM, au lieu d'accumuler le nombre d'occurrences dans la matrice de cooccurrence, on utilise une accumulation du maximum d'une probabilité locale. Dans (Denemark *et al*. 2016), l'idée est de transformer la carte de probabilité de modification de la même manière que le filtrage de l'image, puis de mettre à jour la matrice de cooccurrence en utilisant la version modifiée de la carte de probabilité de modification, plutôt que la carte de probabilité de modification initiale. L'imitation de ce principe a été initialement intégrée dans Ye-Net pour la stéganalyse CNN, et ce concept est facilement transposable à la plupart des CNNs modernes.

1.5.2.4. *La stéganalyse JPEG*

Le meilleur CNN pour la stéganalyse JPEG à la fin de 2018 était SRNet (Boroumand *et al*. 2019). A cette époque, c'est le seul réseau pour la stéganalyse JPEG à disposer d'une version SCA (Side Channel Aware).

Il est intéressant d'énumérer et de discuter brièvement des CNNs précédents utilisés pour la stéganalyse JPEG. Le premier réseau, publié en février 2017, était le réseau de Zeng *et al.*. Il a été évalué avec un million d'images, et a été évalué également face au phénomène de stego-mismatch (Zeng *et al*. 2017) (Zeng *et al*. 2018). Puis, en juin 2017, à IH&MMSec'2017, deux réseaux ont été proposés : PNet (Chen *et al*. 2017) et Xu-Net-Jpeg (Xu 2017). Enfin, SRNet (Boroumand *et al*. 2019) a été mis en ligne en septembre 2018.

Dans le réseau de Zeng et al. (Zeng *et al*. 2017) (Zeng *et al*. 2018), le bloc de prétraitement prend en entrée une image déquantifiée (valeurs réelles), puis la convolue avec 25 bases DCT, puis quantifie et tronque les 25 images filtrées. Ce bloc de prétraitement, utilise des noyaux de filtres fabriqués à la main (bases DCT), les valeurs des noyaux sont fixes, et ces filtres sont inspirés des modèles riches DCTR (Holub and Fridrich 2015). Il y a trois quantifications différentes, donc, le bloc de prétraitement produit 3×25 images résiduelles. Le CNN est donc constitué de 3 sous-réseaux qui produisent chacun un vecteur caractéristique de dimension 512. Les sous-réseaux sont

inspirés de Xu-Net (Xu *et al*. 2016). Les trois vecteurs de caractéristiques, retournés par les trois sous-réseaux, sont ensuite donnés à une structure entièrement connectée, et le réseau se termine par une couche softmax.

À l'instar de ce qui a été fait pour la stéganalyse spatiale, ce réseau utilise un bloc de prétraitement inspiré des Rich Models (Holub and Fridrich 2015). Notez que les modèles riches les plus efficaces, aujourd'hui, sont les Gabor Filter Rich Models (GFR) (Song *et al*. 2015). Notez également que ce réseau tire parti de la notion d'ensemble, qui provient des trois sous-réseaux. Le réseau de Zeng et al. est moins efficace que Xu-Net-Jpeg (Xu 2017), mais donne une première approche intéressante guidée par les Rich Models.

L'idée principale du PNet (et aussi du VNet qui est moins efficace, mais qui prend moins de mémoire) (Chen *et al*. 2017) est d'imiter les Rich Models sensible à la phase, tels que DCTR (Holub and Fridrich 2015) ou GFR (Song *et al*. 2015), et d'obtenir une image en entrée décomposée en 64 cartes des caractéristiques représentant les 64 phases d'une image JPEG. Le bloc de prétraitement prend en entrée une image dé-quantifiée (valeurs réelles), la convolue avec quatre filtres, le "SQUARE5$\times$5" provenant du Spatial Rich Models (Fridrich and Kodovskỳ 2012), un filtre passe-haut "point" (appelé "catalyst kernel") qui complète le "SQUARE5$\times$5", et deux filtres directionnels de Gabor (angles 0 et $\pi/2$).

Juste après le deuxième bloc de convolution, un module de "division de phase" (*PhaseSplit Module*) divise l'image résiduelle en 64 cartes de caractéristiques (une carte = une phase), de la même façon que dans les modèles riches. Certaines procédures intéressantes ont été utilisées telles que : (1) l'enchainement de convolutions à valeurs figés dans le bloc de prétraitement, avec une seconde convolution dont les poids sont appris, (2) une mise à jour intelligente des paramètres de la BN, (3) l'utilisation de "Filter Group Option" qui construit virtuellement des sous-réseaux, (4) le bagging sur 5-cross validations, (5) l'utilisation des 5 dernières évaluations afin de donner l'erreur moyenne d'un réseau, (6) le brassage de la base de données au début de chaque époque, pour avoir un meilleur comportement de la BN, et aider à la généralisation, et (7) éventuellement utiliser un Ensemble. Avec un tel savoir-faire, PNet a battu les approches classiques d'apprentissage machine en deux étapes dans les scénarios Not-SCA et SCA (Ensemble Classifier + GFR).

Le Xu-Net-Jpeg (Xu 2017) est d'autant plus attractif que l'approche est légèrement meilleure que PNet, et ne nécessite pas une forte inspiration de domaine comme pour PNet. Le Xu-Net-Jpeg est fortement inspiré de ResNet (He *et al*. 2016), un réseau bien établi de la communauté d'apprentissage machine. ResNet permet l'utilisation de réseaux plus profonds grâce à l'utilisation de shortcuts. Dans Xu-Net, le bloc de prétraitement prend en entrée une image déquantifiée (valeurs réelles), puis convolues l'image avec 16 bases DCT (dans le même esprit que le réseau de Zeng et al. (Zeng *et al*. 2017) (Zeng *et al*. 2018)), puis applique une valeur absolue, une troncature, et

un ensemble de convolutions, BN, ReLU jusqu'à obtenir un vecteur caractéristique de dimension 384, qui est donné à un bloc entièrement connecté. On peut noter que le max pooling ou l'average pooling est remplacée par des convolutions. Ce réseau est vraiment simple et était, en 2017, la méthode la plus performante. D'une certaine manière, les réseaux proposés par la communauté de l'apprentissage machine sont très compétitifs, et il y a peu de connaissances de domaine à intégrer à la topologie d'un réseau pour obtenir un réseau très efficace.

En 2018, le CNN à l'état de l'art pour la stéganalyse JPEG (qui peut également être utilisé pour la stéganalyse spatiale) était SRNet (Boroumand *et al*. 2019). Ce réseau a été précédemment présenté dans la section **??**. Notez que pour la version de SRNet qui a connaissance du canal adjacent (SCA), la probabilité de modification par coefficient DCTs est d'abord ré-exprimée dans le domaine spatial en appliquent une DCT inverse, et en utilisant les valeurs absolues pour la base DCTs. La *carte d'information adjacente* ainsi obtenue, entre ensuite dans le réseau, et est convoluée avec chaque noyau (cette première convolution est équivalente au bloc de prétraitement). Notez que les convolutions dans ce premier bloc de cette *carte d'information adjacente* sont telles que les noyaux des filtres sont modifiés à leurs valeurs absolues. Après avoir passé la convolution, les feature maps sont additionnées avec avec la racine carrée des valeurs de la *carte d'information adjacente* précédemment convoluée. Notez que cette idée reprend celle qui est exposée dans la version SCA Ye-Net (SCA-TLU-CNN) (Ye *et al*. 2017) avec intégration d'une information adjacente, et la récente proposition pour la steganalyse avec connaissance du canal adjacent (SCA) dans JPEG avec des Rich Models (Denemark *et al*. 2016), où la construction de la *carte d'information adjacente*, et en particulièrement la quantité $\delta_{uSA}^{1/2}$[2].

1.6. Pistes de recherches actuelles (5 pages)

Nous terminerons ce chapitre introductif à la stéganalyse par un bref inventaire des problèmes ouverts qui nous semble les plus intéressants et les plus importants dans ce domaine.

Le problème du Cover-Source Mismatch

Le premier problème que nous allons décrire est celui du « *Cover Source Mismatch* » (CSM) ; ce problème est en fait un cas assez général dans le domaine de l'apprentissage statistique où on y fait référence sous le terme de robustesse. En pratique il s'agit de l'inadéquation entre la base d'apprentissage, sur lequel le classifieur est entraîné à prendre une décision pour distinguer média de couverture et stégo, et la base de test sur laquelle on souhaite détecter la présence d'information cachée.

2. uSA signifie Upper bounded Sum of Absolute values.

La particularité de la stéganalyse est double par rapport au CSM. D'une part en stéganalyse on souhaite faire de la détection de signaux très faible. D'autre part on travaille généralement avec des caractéristiques de (très) grandes dimensions. Conjointement ces deux particularités font que les méthodes de détection seront très sensibles aux bases d'apprentissages et difficilement généralisables.
Une étude assez exhaustive concernant l'évaluation des facteurs occasionnant les CSM a été menée dans (Giboulot *et al*. 2020) montrant par exemple le rôle prépondérant du traitement des images. Malheureusement il n'existe pas encore de travaux permettant d'en comprendre les causes profondes et, en conséquence, il reste très difficile de passer outre ce problème.

Le problème de la stéganalyse en conditions réelles

De façon assez similaire, la stéganalyse *en condition réelle* a été assez peu explorée (Ker and Pevný 2014). En fait, comme nous l'avons évoqué, la stéganalyse a été développée dans le but principal d'évaluer les différentes méthodes de stéganographie. Pour cela, la communauté travailler généralement en utilisant le scénario de stéganalyse ciblée et avec des bases d'images spécifiques, notamment la base BOSS (Bas *et al*. 2011) constituée d'images de 7 appareils photographiques reflex (pas de smartphone ni de compact) toutes traitées, à partir du RAW, de la même façon.
Des travaux ont montré que développer ces images raw de façon différente peut amener des résultats très différents (Sedighi, Fridrich and Cogranne 2016). Un premier concours a été lancé dans ce but avec une base d'image beaucoup plus hétérogène (Cogranne *et al*. 2019) et les propositions montrent la difficulté de procéder à de la stéganalyse sur des images hétérogènes. Les gagnants se sont notamment appuyés sur une multiplication de l'apprentissage en fonction des paramètres de compression JPEG (Yousfi *et al*. 2019), mais de nombreuses questions sur l'application en conditions réelles restent ouvertes.

La stéganalyse fiable

Un problème assez proche des deux précédents concerne la stéganalyse fiable. En effet comme nous l'avons évoqué, un test est nécessairement entaché d'erreurs (faux-positif et faux-négatif) et la conséquence dépend grandement du contexte applicatif. Il semble important en stéganalyse de minimiser les faux-positifs compte tenu du nombre important d'images que l'on peut avoir à analyser. Au contraire les méthodes basées sur l'apprentissage statistique visent généralement à minimiser le taux d'erreur global P_E. Or, à supposer que l'on soit capable de concevoir une méthode de détection dont la probabilité de faux-positif et de faux-négatif soit toutes les deux d'environ 1%, il est clairement illusoire d'appliquer cette méthode en pratique. Quelle est la valeur d'une détection d'information cachée obtenue avec cette méthode ? On pourrait, avec une approche Bayésienne, opposer notamment qu'une image jugée comme étant stéganographiée par un tel détecteur serait en fait plus probablement issue d'un faux-positif étant donné que, sur un site de partage d'images tel FlickR, il

est vraisemblable que moins de 1% des images contiennent effectivement des informations cachées ...
Se pose donc la question cruciale, comment concevoir une méthode de stéganalyse *fiable* dans le sens où elle garantirait une très faible probabilité de faux-positif ; typiquement une probabilité de faux-positif inférieure à 10^{-6} (soit moins de 1 sur 1 million). On a vu que la théorie des tests d'hypothèses permet de concevoir de telles méthodes de stéganalyse, mais cela repose sur un modèle statistique des images qui lui n'est pas nécessairement exact, et *a fortiori* lorsque certains paramètres doivent être estimés ... Une étude très intéressante dans (Pevný and Ker 2015) en étudiant la possibilité de faire de l'apprentissage avec pour critère de minimiser la probabilité de faux-positif si la puissance ς est fixée à 50%. Cependant ce travail préliminaire mérite d'être approfondi et des méthodes de stéganalyse fournissant une « p-valeur » (c'est-à-dire une probabilité que cette détection corr esponde à un faux-positif) précise manquent cruellement.

La stéganalyse d'images couleur

Pour terminer dans cette idée de travaux académiques différents de leurs utilisations pratiques, il faut noter que (Cogranne *et al*. 2019) la très grande majorité des portent sur des images en noir et blanc (ou plus exactement en « niveaux de gris »). La détection d'informations cachées dans les images couleur semble au contraire plus intéressante en pratique … ce sujet demeure largement ignoré. En effet, d'une part une part importante des chercheurs considère qu'une image couleur peut être considérée comme trois images en niveaux de gris (une rouge, une verte et une bleue) qui peuvent être analysés séparément.
Il semble intuitif de considérer cependant que les canaux de couleurs sont très statistiquement corrélés (notamment en raison du dématriçage lors de l'acquisition et du traitement des images, voir Chapitre 1[FIXME]). Cependant les travaux qui ont été menés dans ce domaine n'ont pas montré une grande amélioration en analysant les composantes conjointement (Goljan *et al*. 2014 ; Abdulrahman *et al*. 2016). Le premier travail dans ce domaine (Goljan *et al*. 2014) a par exemple proposé d'ajouter des histogrammes obtenus à partir des valeurs de pixels dans différentes couleurs. Cela a permis, certes d'améliorer légèrement la détection, mais en deçà de ce qui pouvait être attendu.
Enfin ce domaine d'étude n'a pas du tout été étudié pour les images couleur compressées avec le standard JPEG. Cela est plus dommageable, car, d'une part, les images couleur JPEG ne représentent pas le vert, le rouge et le bleu, mais des composantes dont les corrélations statistiques sont encore plus importantes, voir Chaptitre1 [FIXME]. D'autre part, les canaux de couleurs sont traités de façon distincte dans le standard JPEG, il faudrait donc les analyser de façon différenciée. Malheureusement, la détection de la stéganographie dans les canaux couleur du JPEG n'a pratiquement jamais été étudiée (voir notamment les articles (Cogranne *et al*. 2019) et (Yousfi *et al*. 2019) sur le challenge de stéganalyse ALASKA portant sur des images couleur).

Prise en compte de l'adaptativité de la stéganographie

Un problème très différent qui reste ouvert concerne la prise en compte du « canal de sélection » pour la stéganalyse. Le canal de sélection désigne les probabilités d'utiliser chaque pixel $\beta_{m,n}$. Clairement il semble intéressant (comme le montre d'ailleurs les expressions [1.13] et [1.16]) qu'un détecteur puisse utiliser une estimation, même erronée, de ces probabilités d'utilisation, mais cela très peu efficace avec les méthodes de détection actuelle. L'une des premières avancées qui a été proposée dans ce domaine (Denemark *et al*. 2014) consiste par exemple à remplacer le calcul d'histogrammes des co-occurrences (voir partie 1.4.1) par l'ajout, pour chacun des blocs considérés, du maximum de la probabilité d'utilisation $\beta_{m,n}$. Dans cette étude, indiquent avoir essayé plusieurs fonctions de pondération et avoir finalement constaté que le *maximum* permet d'obtenir les meilleurs résultats en termes de probabilité d'erreurs. Par ailleurs, les auteurs indiquent que cette fonction est également robuste à une mauvaise estimation du canal de sélection $\beta_{m,n}$. Cela est particulièrement important, car cette quantité qui n'est pas directement accessible et il faut donc l'estimer ce qui nécessite de connaître la méthode d'insertion et la taille du message. Or il ne faudrait pas le gain obtenu par l'utilisation du canal de sélection ne soit valable que si ce dernier est estimé de façon exacte, car cela est impossible ; une robustesse par rapport à une erreur d'estimation du canal est donc nécessaire.
On notera que pour la stéganalyse basée sur le *deep learning* deux approches coexistent ; d'une part certaines méthodes de détection reposent sur un réseau qui utiliserait deux entrées : l'image à analyser et le canal de sélection. D'autre part, certaines approches reposent sur l'hypothèse qu'un réseau de neurones suffisamment complexe peut apprendre ce canal de sélection et la meilleure façon de s'en servir pour peu que la quantité de données utilisées pour l'apprentissage soit suffisante.

La stéganalyse groupée (par lot)

Une problématique qui est restée également largement peu étudiée concerne la stéganalyse « groupée » qui répond à la stéganographie « par lots » (Ker and Pevný 2012). Il s'agit d'un scénario plus général où Alice peut « étaler » son message dans plusieurs images distinctes, qui seront ensuite envoyées à Bob. La stéganalyste Wendy devra alors observer un lot d'images q et prendre une décision « groupée » concernant l'ensemble de ces images : elles peuvent être toutes de couverture, ou bien certaines, mais pas toutes, peuvent contenir une partie variable des informations cachées.
Pour la stéganographie, ce cas peut se modéliser comme un problème d'allocation d'un message qui doit être dispersé dans plusieurs images afin de minimiser la probabilité de détection. Pour la stéganalyste Wendy, le travail qui consiste à analyser un lot d'image est nettement plus délicat. De nombreuses questions restent ouvertes notamment, comment analyser un nombre d'images a priori inconnu ? Est-il préférable d'utiliser une méthode adaptée à l'analyser de plusieurs images conjointement ou bien faut-il procéder à q analyses d'images indépendantes ? Enfin, faut-il ne pas tenir compte de l'ordre des ? Quelques travaux ont été proposés,

notamment (Cogranne *et al*. 2017 ; Pevny and Nikolaev 2015) qui reposent tous les deux sur un détecteur adapté à l'analyse de chaque image individuellement ; le premier étudie comment pondérer les « scores (continus) » obtenus pour chaque image. Le second travail (Pevny and Nikolaev 2015) opère une détection en deux étapes, un histogramme des mêmes « scores » est d'abord construit puis un second classifieur est entraîné à identifier les lots d'images de couverture. Cependant, et malgré ces travaux préliminaires, de nombreuses questions demeurent sans réponse voire même pas étudiée du tout.

La stéganalyse universelle

Le dernier problème que nous souhaitons aborder est celui de la stéganalyse universelle. Là encore il s'agit d'un cadre plus général que celui de la décision binaire dans un scénario ciblé (qui comme nous l'avons exposé permet davantage d'évaluer la stéganographie). En pratique, il serait intéressant de concevoir des méthodes de stéganalyse opérationnelle si ni la taille du message ni l'algorithme d'insertion ne sont connus. La stéganalyse sans connaissance de la taille du message fait référence à des approches dites « quantitatives », c'est-à-dire visant à estimer la taille du message inséré (Pevný *et al*. 2009). L'étude de la performance du classifieur d'ensemble (Cogranne and Fridrich 2015) dans ce contexte où la taille du message est inconnue montre notamment que ce dernier (Kodovskỳ *et al*. 2012) ne peut être utilisé directement, car le seuil de détection est fixé en fonction des images stéganographiées utilisées durant l'apprentissage. L'approche proposée dans (Pevný *et al*. 2009) consiste essentiellement à utiliser des images contenant des messages de tailles différentes durant l'apprentissage.

De façon analogue, la stéganalyse mutli-classe a été très peu étudiée. Là encore l'extension du classifieur d'ensemble a été envisagée à cette fin (Cogranne and Fridrich 2015) en utilisant deux approches simples et relativement efficaces ; la première dite « media de couverture *vs.* le reste » consiste à créer une hypothèse alternative regroupant l'ensemble des algorithmes d'insertion possible et donc, au final, de réaliser un apprentissage statistique visant à détecter n'importe laquelle des méthodes de stéganographie. La seconde approche consiste au contraire à considérer $H + 1$ plusieurs hypothèses, correspondant aux H algorithmes d'insertion considérés auxquels est ajouté le cas d'images de couverture. Pour détecter la stéganographie et identifier la méthode d'insertion il est possible de créer de nombreux classifieurs spécifiquement entraîner à distinguer deux hypothèses. L'analyse d'une image nécessite alors d'utiliser tous les classifieurs pour ensuite choisir l'hypothèse qui regroupe le plus de « votes » de la part de l'ensemble de tous les classifieurs.

1.7. Bibliographie

Abdulrahman, H., Chaumont, M., Montesinos, P., Magnier, B. (2016), Color image steganalysis based on steerable gaussian filters bank, *in* Proc. ACM Workshop on Information Hiding and Multimedia Security, ACM, pp. 109–114.

Bas, P., Filler, T., Pevný, T. (2011), Break our steganographic system — the ins and outs of organizing boss, *in* Proc. Information Hiding, Lecture Notes in Computer Science, LNCS, Prague, Czech Republic, pp. 59–70.

Bengio, Y., Courville, A. C., Vincent, P. (2013), Representation Learning : A Review and New Perspectives, *IEEE Transaction on Pattern Analysis and Machine Intelligence, PAMI*, 35(8), 1798–1828.

Boroumand, M., Chen, M., Fridrich, J. (2019), Deep Residual Network for Steganalysis of Digital Images, *IEEE Trans. on Information Forensics and Security*, 14(5), 1181 – 1193.

Butora, J., Fridrich, J. (2019), Reverse jpeg compatibility attack *(available as Early Access)*, *IEEE Trans. on Information Forensics and Security*, pp. 1–1.

Chen, M., Sedighi, V., Boroumand, M., Fridrich, J. (2017), JPEG-Phase-Aware Convolutional Neural Network for Steganalysis of JPEG Images, *in* Proc. ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2017, Drexel University in Philadelphia, PA, pp. 75–84.

Cogranne, R. (2011), DÃ©tection statistique d'informations cachÃ©es dans une image naturelle Ã partir d'un modÃ¨le physique, PhD thesis, Troyes University of Technology (UTT).

Cogranne, R. (2020), Selection-channel-aware reverse jpeg compatibility for highly reliable steganalysis of jpeg images, *in* (*submitted*).

Cogranne, R., Fridrich, J. (2015), Modeling and extending the ensemble classifier for steganalysis of digital images using hypothesis testing theory, *Information Forensics and Security, IEEE Trans. on (in press)*, .

Cogranne, R., Giboulot, Q., Bas, P. (2019), The alaska steganalysis challenge : A first step towards steganalysis *into the wild*, *in* Proc. ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'19, ACM, New York, NY, USA, pp. 125–137.

Cogranne, R., Sedighi, V., Fridrich, J. (2017), Practical strategies for content-adaptive batch steganography and pooled steganalysis, *in* 2017 IEEE Intl' Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, pp. 2122–2126.

Cogranne, R., Sedighi, V., Fridrich, J., Pevný, T. (2015), Is ensemble classifier needed for steganalysis in high-dimensional feature spaces ?, *in* Information Forensics and Security (WIFS), IEEE 7th Intl' Workshop on, pp. 1–6.

Denemark, T., Boroumand, M., Fridrich, J. (2016), Steganalysis Features for Content-Adaptive JPEG Steganography, *IEEE Trans. on Information Forensics and Security*, 11(8), 1736–1746.

Denemark, T., Sedighi, V., Holub, V., Cogranne, R., Fridrich, J. (2014), Selection-channel-aware rich model for steganalysis of digital images, *in* Information Forensics and Security (WIFS), 2014 IEEE 6th Intl' Workshop on, pp. 48–53.

Fridrich, J., Goljan, M. (2004), On estimation of secret message length in LSB steganography in spatial domain, *in* Security, Steganography, and Watermarking of Multimedia Contents VI, vol. 5306 of *Proc. SPIE*, pp. 23–34.

Fridrich, J., Kodovskỳ, J. (2012), Rich models for steganalysis of digital images, *Information Forensics and Security, IEEE Trans. on*, 7(3), 868 –882.

Fuji-Tsang, C., Fridrich, J. J. (2018), Steganalyzing Images of Arbitrary Size with CNNs, *in* Proc. IS&T Electronic Imaging, Burlingame, California, USA, pp. 121(1)–121(8).

Giboulot, Q., Cogranne, R., Borghys, D., Bas, P. (2020), Effects and solutions of cover-source mismatch in image steganalysis, *To be Published*, .

Goljan, M., Fridrich, J. (2015), Cfa-aware features for steganalysis of color images, *in* Media Watermarking, Security, and Forensics 2015, vol. 9409, Intl' Society for Optics and Photonics, p. 94090V.

Goljan, M., Fridrich, J., Cogranne, R. (2014), Rich model for steganalysis of color images, *in* 2014 IEEE Intl' Workshop on Information Forensics and Security (WIFS), IEEE, pp. 185–190.

He, K., Zhang, X., Ren, S., Sun, J. (2014), Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition, *in* Proc. of the European Conference on Computer Vision, ECCV'2014, Zurich, Switzerland, pp. 346–361.

He, K., Zhang, X., Ren, S., Sun, J. (2016), Deep Residual Learning for Image Recognition, *in* Proc. of IEEE Conference on Computer Vision and Pattern Recognition, CVPR'2016, Las Vegas, Nevada, pp. 770–778.

Hinton, G. E., Salakhutdinov, R. R. (2006), Reducing the Dimensionality of Data with Neural Networks, *Science*, 313(5786), 504–507.

Holub, V., Fridrich, J. (2015), Low-complexity features for jpeg steganalysis using undecimated dct, *Information Forensics and Security, IEEE Trans. on*, 10(2), 219–228.

Ker, A. D., Pevný, T. (2012), Batch steganography in the real world, *in* Proc. of the on Multimedia and Security, IH&MMSec 12, ACM, New York, NY, USA, pp. 1–10.

Ker, A. D., Pevný, T. (2014), The steganographer is the outlier : realistic large-scale steganalysis, *Information Forensics and Security, IEEE Trans. on*, 9(9), 1424–1435.

Kodovskỳ, J., Fridrich, J., Holub, V. (2012), Ensemble classifiers for steganalysis of digital media, *Information Forensics and Security, IEEE Trans. on*, 7(2), 432–444.

Krizhevsky, A., Sutskever, I., Hinton, G. E. (2012), ImageNet Classification with Deep Convolutional Neural Networks, *in* Proceeding of Advances in Neural Information Processing Systems 25, NIPS'2012, Curran Associates, Inc., Lake Tahoe, Nevada, USA, pp. 1097–1105.

LeCun, Y., Bengio, Y., Hinton, G. (2015), Deep learning, *Nature*, 521(7553), 436–444.

Lehmann, E., Romano, J. (2005), *Testing Statistical Hypotheses, Second Edition*, 3rd edition, Springer.

Pevný, T., Bas, P., Fridrich, J. (2010), Steganalysis by subtractive pixel adjacency matrix, *IEEE Trans. Inform. Forensics and Security*, 5(2), 215–224.

Pevný, T., Fridrich, J., Ker, A. D. (2009), From blind to quantitative steganalysis, *in* Proc. IS&T Electronic Imaging.

Pevný, T., Ker, A. D. (2015), TowÂ²ysis, *in* Proc. SPIE, vol. 9409, pp. 94090I–94090I–14.

Pevny, T., Nikolaev, I. (2015), Optimizing pooling function for pooled steganalysis, *in* Proc. IEEE Intl' Workshop on Information Forensics and Security, pp. 1–6.

Qian, Y., Dong, J., Wang, W., Tan, T. (2015), Deep Learning for Steganalysis via Convolutional Neural Networks, *in* Proc. IS&T Electronic Imaging, vol. 9409, San Francisco, California, USA.

Sedighi, V., Cogranne, R., Fridrich, J. (2016), Content-adaptive steganography by minimizing statistical detectability, *IEEE Trans. on Information Forensics and Security*, 11(2), 221–234.

Sedighi, V., Fridrich, J. (2015), Effect of Imprecise Knowledge of the Selection Channel on Steganalysis, *in* Proc. ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2015, Portland, Oregon, USA, pp. 33–42.

Sedighi, V., Fridrich, J. J., Cogranne, R. (2016), Toss that bossbase, alice !, *in* Media Watermarking, Security, and Forensics, pp. pp. 1–9.

Song, X., Liu, F., Yang, C., Luo, X., Zhang, Y. (2015), Steganalysis of adaptive jpeg steganography using 2d gabor filters, *in* Proc. ACM Workshop on Information Hiding and Multimedia Security, ACM, pp. 15–23.

Thai, T. H., Cogranne, R., Retraint, F. (2014), Statistical model of quantized DCT coefficients : Application in the steganalysis of jsteg algorithm, *Image Processing, IEEE Trans. on*, 23(0), 1–14.

Westfeld, A. (2001), F5–a steganographic algorithm, *in* Proc. Information Hiding, vol. 2137 of *Lecture Notes in Computer Science*, LNCS, pp. 289–302.

Westfeld, A., Pfitzmann, A. (1999), Attacks on steganographic systems, *in* Proc. Information Hiding, pp. 61–76.

Xu, G. (2017), Deep Convolutional Neural Network to Detect J-UNIWARD, *in* Proc. ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2017, Drexel University in Philadelphia, PA, pp. 67–73.

Xu, G., Wu, H. Z., Shi, Y. Q. (2016), Structural Design of Convolutional Neural Networks for Steganalysis, *IEEE Signal Processing Letters*, 23(5), 708–712.

Ye, J., Ni, J., Yi, Y. (2017), Deep Learning Hierarchical Representations for Image Steganalysis, *IEEE Trans. on Information Forensics and Security, TIFS*, 12(11), 2545–2557.

Yedroudj, M., Chaumont, M., Comby, F. (2018), How to Augment a Small Learning Set for Improving the Performances of a CNN-Based Steganalyzer ?, *in* Proc. IS&T Electronic Imaging, Burlingame, California, USA, p. 7.

Yedroudj, M., Comby, F., Chaumont, M. (2018), Yedrouj-Net : An Efficient CNN for Spatial Steganalysis, *in* Proc. of IEEE Intl' Conference on Acoustics, Speech and Signal Processing, Calgary, Alberta, Canada, pp. 2092–2096.

Yousfi, Y., Butora, J., Fridrich, J., Giboulot, Q. (2019), Breaking alaska : Color separation for steganalysis in jpeg domain, *in* Proc. ACM Workshop on Information Hiding and Multimedia Security, ACM, pp. 138–149.

Zeng, J., Tan, S., Li, B., Huang, J. (2017), Pre-Training via Fitting Deep Neural Network to Rich-Model Features Extraction Procedure and its Effect on Deep Learning for Steganalysis, *in* Proceedings of Media Watermarking, Security, and Forensics 2017, MWSF'2017, Part of IS&T Symposium on Electronic Imaging, EI'2017, Burlingame, California, USA, p. 6.

Zeng, J., Tan, S., Li, B., Huang, J. (2018), Large-Scale JPEG Image Steganalysis Using Hybrid Deep-Learning Framework, *IEEE Trans. on Information Forensics and Security*, 13(5), 1200–1214.

Zhang, R., Zhu, F., Liu, J., Liu, G. (2020), Depth-wise separable convolutions and multi-level pooling for an efficient spatial cnn-based steganalysis, *IEEE Trans. on Information Forensics and Security*, 15, 1138–1150.