图像隐写术中的HUGO算法基本原理

原创

江户川柯壮 于 2018-11-01 17:47:40 发布 4878 收藏 20
分类专栏： 图像处理

本文链接：https://blog.csdn.net/edogawachia/article/details/83618317


图像处理 专栏收录该内容
18 篇文章 7 订阅
订阅专栏

图像隐写术中的HUGO算法基本原理

HUGO的意思是**H**ighly **U**ndetectable ste**GO**，是一种当前常用的stegnography的方法。

文献出处

Tomas Pevny, Tomas Filler, Patrick Bas. Using High-Dimensional Image Models to Perform Highly Undetectable Steganography. Information Hiding, Jun 2010, Calgary, Canada. pp.2010, 2010. <hal-00541353>

基本原理

文章先简单介绍了LSB replacement，即least significant bit replacement。基于LSB最朴素的方案就是这个，即把信息直接embedding进最末尾的bit，这个过程由于其asymmetry使得很容易被检出。然后对LSB replacement的改进就是LSB matching，即将LSB进行随机＋1或者－1，这样导致过程变成balanced，因此可以很难检出。LSB matching的一个假设是图像中pixel之间是independent的，但是自然图像并非如此，导致这个漏洞可以被用来做detector。

之所以上面的方法容易检出，是因为没有考虑到自然图像的统计特性，因此，这里的作者提出了一个新的基于自然图像的各种各样的dependency的通用模型，用来进行隐写。

模型basic principle叫做 minimal impact embedding，这个principle是之前就有的。这个principle将隐写算法设计分成了**image model 设计** 和 **coder 设计** 两个步骤。这里提出的模型基于SPAM features （subtractive pixel adjacency matrix）。

Minimizing Embedding Impact

这个是上面提到的HUGO的basic principle。首先考虑：

$$D(\mathbf{X},\mathbf{Y}) = \sum_{i=1}^{n} \rho_i |x_i - y_i|.$$

这里的D是一个非负数的distortion measure，代表y（stego）和x（cover）的差异，rho是系数，如果rho = inf，那么这个点叫做**wet pixel** ，指的是不允许被改变的点。由于逐像素sum的形式，说明像素间没有Dependency，这个只有在隐写的数据量比较少且embedding的位置距离较远的时候有用。

下面的定理介绍了，当我们的要传输的信息量是m bit 时，那么对于binary的embedding方法，可以计算出最优的distortion：

Theorem 1. *Let $\rho = (\rho_i)_{i=1}^n$, $0 \leq \rho_i < \infty$, be the set of constants defining the additive distortion measure (1) for $i \in \{1, \dots, n\}$. Let $0 \leq m \leq n$ be the number of bits we want to communicate by using a binary embedding operation. The minimal expected distortion has the following form*

$$D_{min}(m, n, \rho) = \sum_{i=1}^{n} p_i \rho_i,$$

where

$$p_i = \frac{e^{-\lambda \rho_i}}{1 + e^{-\lambda \rho_i}} \tag{2}$$

is the probability of changing the ith pixel. The parameter λ is obtained by solving

$$-\sum_{i=1}^{n} \Big(p_i \log_2 p_i + (1 - p_i) \log_2 (1 - p_i) \Big) = m. \tag{3}$$

这里的p的分布是由求解最后的cross entropy的方程得到的，该theorem的大概意思就是说，embedding至少要在理论上（信息论的意义上）能够足以传输m个bit 的信息量。

根据上面的定理，optimal coding可以通过按照上述求出的pi对每个pixel进行flip得到。rho_i的求解是根据image model得到的。所以下面的工作内容就是怎样求解rho。

从隐写分析方法到隐写术

邻近pixel的dependency一方面来自于inherent smoothness，也就是自然图像低频成分多的内在属性，另一方面还来自于预处理如de-mosaicking或者sharpening等造成的noise的相关性（未经处理的noise一般是independent的）。其中noise的相关性对于steganalysis比较重要，因为stego信息一般隐藏在高频噪声里。

SPAM features 是利用了高阶马尔科夫链（high-order Markov chains）来模拟邻近像素点的依赖关系。计算了从一个点到8邻域的八个方向的probability。基本过程如下：

Let $\mathbf{I} \in \mathcal{X}$ be an image of size $n_1 \times n_2$. The calculation starts by computing the difference array $\mathbf{D}^{\bullet}$, which is for a horizontal left-to-right direction

$$\mathbf{D}_{ij}^{\rightarrow} = \mathbf{I}_{ij} - \mathbf{I}_{i,j+1},$$

for $i \in \{1, \ldots, n_1\}$, $j \in \{1, \ldots, n_2 - 1\}$. Depending on the desired order of the features, either the first-order Markov process is used,

$$\mathbf{M}^{\rightarrow}_{d_1 d_2} = Pr(\mathbf{D}^{\rightarrow}_{i,j+1} = d_1 | \mathbf{D}^{\rightarrow}_{ij} = d_2), \tag{4}$$

or the second-order Markov process is used,

$$\mathbf{M}^{\rightarrow}_{d_1 d_2 d_3} = Pr(\mathbf{D}^{\rightarrow}_{i,j+2} = d_1 | \mathbf{D}^{\rightarrow}_{i,j+1} = d_2, \mathbf{D}^{\rightarrow}_{ij} = d_3), \tag{5}$$

where $d_i \in \{-T, \ldots, T\}$. The calculation of the features is finished by separate averaging of the horizontal and vertical matrices and the diagonal matrices to form the final feature sets. With a slight abuse of notation, this averaging can be written as

$$\begin{aligned} \mathbf{F}^{\bullet}_{1,\ldots,k} &= \frac{1}{4}\left[\mathbf{M}^{\rightarrow}_{\bullet} + \mathbf{M}^{\leftarrow}_{\bullet} + \mathbf{M}^{\downarrow}_{\bullet} + \mathbf{M}^{\uparrow}_{\bullet}\right], \\ \mathbf{F}^{\bullet}_{k+1,\ldots,2k} &= \frac{1}{4}\left[\mathbf{M}^{\searrow}_{\bullet} + \mathbf{M}^{\nwarrow}_{\bullet} + \mathbf{M}^{\swarrow}_{\bullet} + \mathbf{M}^{\nearrow}_{\bullet}\right], \end{aligned} \tag{6}$$

where $k = (2T+1)^2$ for the first-order features and $k = (2T+1)^3$ for the second-order features. In [22], the authors used $T = 4$ for the first-order features (leading to 162 features) and $T = 3$ for the second-order features (leading to 686 features).

这里采用的是更高维度的features，原因在于：**对于stegnography，即编码的隐写来说，更高的维度不会带来太大的drawback，但是对于隐写分析，即解码来说，由于要提取的特征较多，从而训练ML的模型的时候会遇到 dimensionality curse，即所谓的维数灾难，也就是由于数据量较少导致的过拟合，从而降低了steganalysis的performance**。这是HUGO的基本出发点。

HUGO的基本流程如下：首先，把simulated maximum payload 嵌入到图像中去，这里指的就是以1/2概率randomly increase或者decrease pixel value。 然后，用Fisher LDA的方法进行evaluate，然后用来确定rho（直接改变cost或其他更复杂的启发式方法）。

HUGO的实际操作

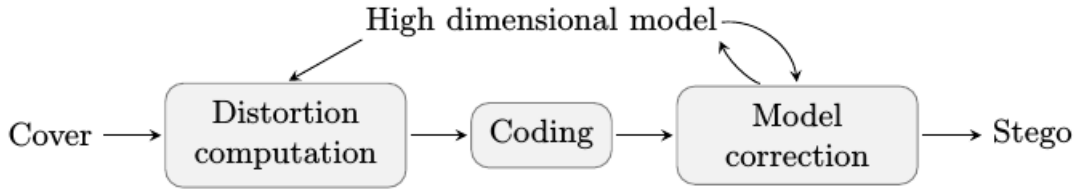

Fig. 2: High-level diagram of HUGO.

首先，evaluation相关信息：模型评估用的是precision和recall的直接平均折中。分类器用的是高斯核函数的soft-margin SVM。

对于image model，用的是SPAM features， 即co-occurence matrices。

伪代码如下：

HUGO embedding algorithm

```
for (i,j) in PIXELS {              //function D is taken from (10)
   Yp = X; Yp(i,j)++; rho_p(i,j) = D(X,Yp);  //calculate emb. impact
   Ym = X; Ym(i,j)--; rho_m(i,j) = D(X,Ym);  //for each pixel
}
rho_min = min(rho_p, rho_m);     //elementwise; use minimum for embedding
PIXELS_TO_CHANGE = minimize_emb_impact(LSB(X), rho_min, message)
Y = X;                           //start making changes in cover image
for (i,j) in PIXELS_TO_CHANGE { //order given by the MC visit. strategy
   if ( model_correction_step_enabled ) {
      Yp = Y; Yp(i,j)++; dp = D(X,Yp); Ym = Y; Ym(i,j)--; dm = D(X,Ym);
      if ( dp<dm ) { Y(i,j)++; } else { Y(i,j)--; }
   } else {
      if ( rho_p(i,j)<rho_m(i,j) ) { Y(i,j)++; } else { Y(i,j)--; }
   }
}
```

Fig. 3: Pseudo-code of the HUGO embedding algorithm as described in Section 4.3.

可以看到，对于cover， 对plus和minus两种操作进行一个对比，计算出操作后的rho的变化，取最小的rho变化对应的操作，然后确定需要改变的pixel，仍然比对plus和minus对于函数D带来的改变的影响，这个函数D指的是distortion error，也就是尽量减小隐写带来的distortion。

总结

HUGO这种隐写术的目标就是通过增加features的维度，使得stego难以被检测。distortion function用的是weighted difference of extended SOTA feature vectors already used in steganalysis，就是把 feature 的差异进行了加权求和，就是在上面提到的第一个公式。image model 是源于SPAM features，权重是用Fisher LDA得到的。对于隐写术的security，作者也进行了分析，相对于LSB在40%的错误率上的payload 0.04bpp， HUGO在同样错误率下能达到0.3bpp，得到了显著提高。

慧极必伤，情深不寿，强极则辱，谦谦君子，温润如玉。——《书剑恩仇录》