背景

## 基于LSB的图像隐写技术

### 隐写术

隐写是指把一个文件、消息、图像或者视频隐藏到另一个文件、消息、图像或者视频的行为。与密码学不同的是，隐写术旨在隐藏消息或其他形式的信息本身的存在，不引起发送方和接收方以外的人的怀疑而完成信息的交流，而密码学则用于隐藏这些信息的内容，使得非发送方或接收方即使截获消息也无法得到所交流的信息的真实内容。隐写术的主要任务是使发生在公共信道上的秘密信息交流不被察觉，隐藏了秘密信息的图片或其他格式的载体与隐藏之前的原始数据在视觉上以及其他几个重要特征一致。

隐写术是一种秘密通信的艺术，这个术语最早在从几千年前开始，人类就着迷于密文书写，并出于多种原因和动机学习这种和研究这种技术[1]，早期密写通常是指密码学，而随着时代发展现在密写也涵括了隐写术，两者用不同的方式实现密写的目的。无论在哪个时代，隐写术广泛应用于各个领域。早期的隐写实践是使用不可见的墨水在信件中书写消息，显而易见地，军事和政治中，在不被敌方察觉的前提下向友方传递信息的能力十分关键。在数字时代，这种思想发展为在多媒体文件中隐藏其他数字消息。现代隐写术在1985年以后随着个人计算机的推广而问世，随着计算机科学技术和数学的发展以及深入研究而迅速发展并投入更广泛的应用，例如电子通信包含了传输层内的隐写码以传送如文档文件或图片文件等多媒体数据。

与密码学相似，隐写术的使用场景决定了它必须满足以下三个要求：

* 保密性：不容易被探查到隐藏消息的存在
* 可获得性：不会出现由于修改数据的载体导致秘密消息的丢失，秘密消息可以被恢复
* 完整性： 其他人无法伪造出错误信息

隐写系统可以被视为加密系统的一个特例[2]，在这个系统中我们要求密文与明文对于其他人来说难以区分。值得注意的是，隐写加密方必须首先合成一个与秘密消息无关的无害的文件。载体合成具有很大的挑战性，高效合成载体的隐写机制很少。对于这个问题，使用通过修改实现隐写的方法很好地避免了这个挑战。这种系统使用一个现有的与秘密消息无关的文件，也就是载体，作为隐写系统的输入的一部分，接下来秘密消息转换为以让人难以觉察的载体的修改，得到与载体极其相似的结果，即伪装（stego）。理论上，载体合成是最自由且强大的隐写方法，因为它可以不受限制地适用于任意的秘密消息。但在实践中，无害文件的合成是一个非常复杂且低效的过程，所以大多数众所周知的实用系统实用的都是修改载体的方法。当然除了合成和修改两种模式以外，载体选择[3]也是可行的方案，可以类比为传统密码学中使用的编码本，有多个载体或密文分别对应特定的秘密消息。然而，这种模式需要的可用载体数量太大，并不是在所有场景下都实用。所以我们在大多数情况下会选择通过修改进行隐写的模式，也就是说隐写的过程需要使用已经存在的文件作为原始的输入，而多媒体文件（如图像、音频和视频等）往往较大，包含了大规模的数据，可以找到足够的空间隐藏消息同时在不表达出可以被察觉的异常效果，是理想的载体。其中，数字图像的应用场景广泛，修改方便，且容易在互联网快速传播，成为了应用最多的载体。本文也将围绕图像隐写技术展开。

早期的隐写安全完全依赖于隐写算法，只将载体和秘密消息作为输入而不使用秘钥，被称为纯隐写系统，一旦隐写算法泄露则整个系统被破解。将秘钥引入作为输入的隐写系统则被称为秘钥隐写系统。在密码学中，加密者和解密者共享秘钥，关于算法的知识对双方区分伪装和正常消息没有帮助，这个结论即kerckhoffs原则，在隐写系统中并不总是成立[4]。符合kerckhoffs原则的系统在使用一个秘钥的实例被攻破后，使用其他秘钥的实例的安全性不受影响，与其他系统相比这样的系统具有巨大的优势，因为生成不同的秘钥恨容易，而重新设计一个算法却很困难。但事实上，基于kerckhoffs原则安全的隐写系统很难设计，同时大多数应用场景下，除了发送方和接收方外其他人对于他们的隐写系统通常一无所知，因此，不遵循kerckhoffs原则的系统也是可行的。

### LSB隐写算法

对于一个二进制整数来说，最低有效位（LSB）是最低的比特位（即第0位），决定了这个数是奇数还是偶数，这个比特位相比于其他位置的变化对于整个数值变化影响是最小的。

LSB嵌入方法是一种经典的图像隐写算法。这种方法最早被用于像素图像，在像素图像中，每个像素都是代表该点颜色强度的整数。在灰度图像中，每个点的像素值表达了该点色彩介于黑白之间的程度，而在具有三个色彩信道的RGB图像中，每个像素点由三个独立个代表红、绿、蓝三种颜色强度的值合成单元，这些像素值的取值范围通常是8bit的整数值，也就是。在颜色强度上的微小改动被察觉的可能性很小，LSB隐写算法正是利用了像素图像的这个特性，舍弃每个像素原来的最低有效位（LSB），并替换为需要隐藏的消息。接收者得到伪装完成的图像后可以通过模2操作提取新的LSB并将之还原为完整的消息。

最基础的基于LSB的隐写系统从图像的左上角开始逐位嵌入秘密消息，推进的顺序是由发送者和编程语言决定的最自然的方向。为了方便接受者确定隐藏的消息在何处结束，我们可以将消息的大小作为头部隐藏在前像素中，这里的是一个双方已经约定好的整数。通过这个简单的python程序可以演示这个隐写的过程（在python中最自然的方向是逐行嵌入）：

def convert\_message\_to\_bit(string\_text):  
 '''convert the secret message from string to bytearray'''  
 bit\_string=[]  
 for char in string\_text:  
 bit\_string.extend([int(d) for d in bin(ord(char))[2:].zfill(8)])  
 return np.array(bit\_string)  
  
def hide\_side\_information(carrier,length,pro\_text\_size):  
 '''hide some side information(i.e. the length of secret message here) in the front of the image'''  
 bit=bin(length)[2:].zfill(pro\_text\_size)  
 length\_data=np.array([int(d) for d in bit ])  
 side\_info=carrier[:pro\_text\_size]+length\_data  
 carrier[:pro\_text\_size]=side\_info  
 return carrier  
  
def binary\_array\_to\_int(arr):  
 '''convert a 0-1 array into a decimal integer'''  
 bit\_string=''.join(arr.astype(np.str))  
 return int(bit\_string,2)  
  
def hide(cover\_pic,secret\_text,pro\_text\_size):  
 '''hide secret information in a given picture(cover), return a image object'''  
 length=len(secret\_text)  
 cover=np.array(cover\_pic)  
 size=cover.shape  
 cover\_data=cover.flatten()  
 carrier\_data=cover\_data & 0b11111110 #truncated cover without LSB plane  
 l=hide\_side\_information(carrier\_data,length,pro\_text\_size)  
 secret\_bytes=convert\_message\_to\_bit(secret\_text)  
 secretbytes\_length=length\*8  
 carrier\_data[pro\_text\_size:pro\_text\_size+secretbytes\_length]=carrier\_data[pro\_text\_size:pro\_text\_size+secretbytes\_length]+secret\_bytes  
 new\_data=carrier\_data.reshape(size)  
 new\_iamge=Image.fromarray(new\_data)  
 return new\_iamge  
  
def reveal(secret\_iamge,pro\_text\_size):  
 '''extract secret message from a given picture(stego) return a secret string'''  
 im=np.array(secret\_iamge).flatten()  
 lsb\_plane=im & 0b00000001  
 length=binary\_array\_to\_int(lsb\_plane[0:pro\_text\_size])  
 secret\_bytes=lsb\_plane[pro\_text\_size:pro\_text\_size+length\*8].reshape((length,8))  
 secret\_bit=[binary\_array\_to\_int(x) for x in secret\_bytes]  
 return ''.join(chr(c) for c in secret\_bit)

这种方法使得隐藏的消息总是在同样的位置。显然，作为最经典的图像隐写算法，LSB嵌入模式许多的变体，针对这个问题，如果随机选择用于嵌入消息的像素点的位置则可以让伪装变得更加隐蔽。在伪装系统中引入伪随机数生成器（PRNG），依据其产生的序列选择嵌入消息的像素位置，可以使伪装消息的分布更加随机。这个伪随机数生成器的种子作为系统的秘钥必须在之前就由双方完成交换。

除此之外，还有一种经典的基于LSB嵌入的改进，被称作LSB匹配，体现在随机化了每个样本的篡改。不同于经典LSB嵌入直接舍弃最低有效位的操作，LSB匹配将观察该像素的最低有效位与需要嵌入的信息单元的关系，如果一致则保持最低有效位不变，否则将等概论地对该像素值，具体来说对于偶数值像素而对于奇数值像素。消息的提取同样可以使用模2运算直接完成。

基于LSB的多种隐写方法被广泛应用在空域（如像素图像）并推广到了其他整数信号发生幅度的变化难以被觉察到的场景。对于LSB隐写算法进行在多个方面进行一些微小的改进，就可以获得在隐藏效果上得到巨大的提升，因此现有的工作往往针对于LSB具体实现参数的调整而非设计LSB以外的隐写算法。

## 隐写分析

隐写分析是指对于使用隐写术隐藏的秘密消息的探查。隐写分析的主要任务是识别给定的文件中是否隐藏了可疑的秘密消息，最基础的隐写分析算法以可疑文件作为输入，而将二元分类的“是”和“否”作为输出结果。在此基础之上，有些隐写分析也加入了恢复隐藏信息的功能，但是作为隐写分析最根本的任务是探查是否存在隐藏的消息，一旦探查到隐藏消息的存在，甚至不需要恢复出具体的消息，就可以认为该隐写系统被攻破。

图像隐写分析方法根据手段不同主要分为四种：视觉隐写分析，结构隐写分析，统计隐写分析和学习隐写分析。

视觉隐写分析是最容易的分析手段，即观察图像是否存在视觉上的异常痕迹，最常见的实现方式即提取图像的LSB平面。通过修改图像的LSB而达到隐藏效果的隐写图像相比于自然图像在LSB平面体现出一定的异常，特别是在与高位比特平面的图像对比时，不自然的人工痕迹更为明显。

结构隐写分析则寻找在媒体表达或文件格式上泄露的痕迹。由于人为限制，某个版本的隐藏工具会对图像的大小有所要求，比如Seek and Hide 4.1[5]只会使用像素的图像作为载体，那么我们可以根据这个特性重点探查像素的图像。再如JPEG兼容压缩攻击[6]：JPEG图像使用的是有损压缩，不是每个像素组都会成为可能的一个给定的JPEG解压实现的可能输出，因此这种攻击方法可以针对载体之前使用过JPEG压缩的像素图像的嵌入模式进行分析。

统计隐写分析借鉴统计学的方法探查隐写。这种方法通常需要一个统计模型来描述伪装和/或载体的概率分布，因此我们需要预先通过分析相关载体和隐写系统建立一个这样的数据模型。统计假设测验是最典型的统计方法，用以判定一个可疑的文件是可信的载体还是伪装。不仅如此，一些量化的分析使用了参数估计来估测嵌入消息的长度。对于载体的数据建模被证明是困难的，但是对于给定的隐写系统，却往往容易建立良好的关于伪装数据的模型，因此可以有针对地应用于探查某种隐写方法。

但在实际场景中很难单纯通过分析的方法建立一个统计模型，对此我们可以借助计算的方法，使用学习隐写分析，也就是使用穷举的思想得到经验模型。具体的解决方案可以使用模式识别、机器学习或人工智能等领域的方法。

针对基于LSB的图像隐写分析的方法主要有测验[7]，样本对分析[8]和RS隐写分析[9]，他们的主要特性如表所示。

|  |  |  |  |
| --- | --- | --- | --- |
| 方法 | 原则 | 特征 | 作为探查分类器 |
| 测验 | 将图像进行分片并对每个分片使用测试 | 系数 | 基于阈值 |
| RS隐写分析 | 使用翻转和掩码来识别R,S,U群并画出RS图进行估计 | 在不同的掩码中R,S,U群出现的频率 | 基于阈值 |
| 样本对分析 | 计算样本对的频率，解方程估计 | 每个特定样本对出现的频率 | 基于阈值 |

### 测验

测验，也就是值对（PoV）攻击，是统计隐写分析的先驱。攻击的设计基于图像直方图来检测LSB嵌入。

为了理解值对，我们可以先观察载体和使用LSB满容量嵌入得到的伪装的直方图，直方图统计了每个灰度值出现的次数，可以从图中看出伪装图像的直方图中有更多的成对出现的大致等高的条方。

回顾LSB嵌入的过程，载体图像中一个值为的像素可能在依然保留值或者改变为，同理，一个值为的像素也可能保持不变或者变为，因此对于每一个的都形成了一个值对，LSB嵌入可以改变一个值对内的两个像素值但无法将一个像素值从一个值对改变到另一个值对。如果嵌入的消息是一个随机消息或者加密过的消息，也就是需要嵌入的位近似于0和1的均匀分布，那么对于伪装图像的每一个来说，和也是近似于均匀分布的。

基于上述特性，我们可以使用统计假设测验来检查可疑的图像是否为包含嵌入消息的伪装图像。这里选择使用测验来检验像素值和出现的频率是否符合均匀分布。令表示颜色强度值出现的次数，在（满容量）伪装图像中我们期望一个值对中的两个值出现的频率符合均匀分布，因此对所有有，根据标准测试[10]有

查询分布表我们可以计算对应的值，用以度量图像为伪装图像的概率，根据值大小决定该图像是否包含隐藏消息。

这样的测验一开始只适用于探查是否存在连续的LSB替换，对于低嵌入率的情况效果不佳，且对于使用了伪随机数决定嵌入像素的隐写系统无效。随着现有研究的发展，测验的思想被推广，可以通过多次渐进迭代、从特定位置开始探查、增大样本大小等方法[11]改进用于随机位置的LSB嵌入以及其他LSB衍生系统的探查。

### 样本对分析

样本对分析（SPA）方法跟踪LSB嵌入前后的样本对构成的多个集合，并分析多个集合间的关系计算嵌入消息的长度。

假设图像用连续的样本表示，其中为划分的样本总数，一个样本对，，集合为图像中所有样本构成的集合，我们定义是包含类似于或的的子集，其中。对于，定义为前7比特的差值的样本对。我们再定义以及，对于且有。如果中的样本对离散均匀分布，那么对于任意的满足的都有

这也是SPA方法的关键理论基础。

显然集合中的样本对都形如或，中的样本对则形如或，如果我们考虑LSB嵌入中样本对的翻转，我们有四种修改模式：00,01,10,11，其中1表示样本对中LSB发生改变的样本，而0表示保持完好的样本，所以对于每一个，多元子集都被划分为，和，那么显而易见，这些在嵌入后都是相近的，而，和却不是。所以我们继续把分割为和，这时包含形如值为或的样本对，包含形如包含形如值为或的样本对，的多重集合通过上述操作可以被分为，，和，成为的变多重集合，那么在LSB嵌入后，我们可以获得一个描述变多重集合间转换的有限状态机。

根据[8]中提到的方法我们可以用如下的方程计算嵌入消息的长度:

我们令即假设只使用最低一位进行嵌入，则可以估计嵌入消息长度的最小值。

### RS隐写分析

RS方法挖掘伪装图像空间的相关性。相比于用于嵌入操作的原始载体，LSB嵌入后得到的伪装图像的空间相关性往往有所下降，我们可以利用这个特点探查隐写的存在。RS方法的主要思想如下：先将图像划分为大小相等的小图像块（群)，对得到的每个小图像块都分别进行非负翻转和非正翻转操作，再对比统计经过这些操作以后小图像块的空间相关性的变化，Fridrich等人经过统计分析认为，LSB嵌入前后空间相关性增加的小图像块数量和相关性减小的小图像块数量各自呈现一定的关系，可以根据这样的关系判断图像中是否有经过LSB嵌入的隐藏消息。

我们规定三种翻转操作：正翻转，负翻转和零翻转。对于任意的，正翻转是像素值在和间的转换，如,,,，是使用LSB嵌入秘密消息过程中发生的翻转操作。负翻转则是像素值在和间的转换，如,,,。零翻转则保持原像素值不变。原始的LSB嵌入过程可以描述为如下过程：当嵌入的位与像素的LSB位相同时对该像素进行零翻转，否则使用正翻转。

我们定义作用在群的函数为平滑度函数，表示为，图像的平滑度函数函数是每个像素与其邻接像素的绝对差值的和，反映了图像的空间相关性。平滑函数的值越大，图像的混乱程度越大，图像空间的相关性越小。

我们对于每个组都使用掩码为的翻转操作，得到的翻转结果为其中。根据翻转前后像素组值大小的变化我们可以定义三类像素组,和：如果则为正则组，记为；如果则为奇异组，记为；除此以外，若则为不变组，记为。 掩码表示的非负翻转操作下的正则组的相对数量记为，奇异组的相对数量记作，同理对于掩码为的情况，即非正翻转操作得到的正则则的相对数量记为，奇异组的相对数量表示为。

Fridrich等人提出零假设是对于典型的未经过LSB嵌入的载体图像满足和的大小近似相等，同样和也是近似相等的关系，也就是经过非负翻转后空间相关性减小的小图像块和经过非正翻转后空间相关性减小的小图像块数量大致相等，并且对于相关性增大的小图像块这种类似的关系也成立。同时，对于未经LSB嵌入的图像还有和，这意味着对于自然图像，非零翻转必然使图像空间的像素相关性呈现下降趋势。对于经过LSB嵌入消息的伪装图像，应用非负翻转后的空间相关性相较于应用非正翻转有很大提升，所以有和。

由于LSB嵌入本质上应用的也是一种非负翻转的实例，所以在统计意义上，随着嵌入率的上升，和的差值会有所减小。然而在对伪装图片进行非正翻转的过程中，有些像素可能经历过正负两次翻转，致使其与原始值偏离得更远，因此和不会随着嵌入率的上升而发生显著变化。根据和的差值可以量化估计嵌入消息的长度，但是由于计算量较大这里不作介绍，RS隐写分析的主要实现是捕获两类翻转后图像空间相关性的非对称性变化的特点发现图像中LSB隐写痕迹。

## 支持向量机

通过以上对隐写分析方法的介绍，容易发现单纯的LSB替换或者LSB匹配等基于LSB嵌入的隐写方案在安全性方面存在一定的缺陷，易受到这些隐写分析方法的攻击。针对这个问题，我们可以引入学习的概念，寻找适合隐藏密码消息的像素的位置，相比按照顺序连续嵌入或使用伪随机数生成器寻找嵌入位置的隐写方法，通过学习现有样本和及对应的评估结果的关系，相比于传统的对抗隐写分析的方法可以省略复杂的分析过程，基于经验得到直观的选择嵌入位置的方法。这里的过程可以被简述为生成一些在不同图像上选择不同位置的LSB嵌入的伪装图像，将它们的隐写分析结果连同位置和图像的几个特征作为训练集，使用支持向量机（SVM）进行学习，完成有监督学习得到的分类器用于对需要嵌入消息的图像的位置（像素）进行二元分类，输出结果为适合隐藏消息的位置的集合和不适合隐藏消息的位置的集合。

能实现分类效果的机器学习模型算法数不胜数，每种算法都存在各自独有的优势和劣势。本文选择SVM是出于其能最大限度区分出每个像素点是否适合隐藏秘密数据的考虑，不仅如此，SVM理论和实践发展较为成熟，关于SVM本身的优化和参数调节的研究丰富，易于实现，在不同应用领域的适用性都比较理想，适合用于支持LSB隐写方法的改进优化。

对于一个给定的容量为的训练样本集，其中，如果存在可以根据的值线性划分整个样本空间的超平面，那这样的超平面往往不止一个，但是需要挑选最佳的超平面作为分类器。 SVM方法选择这个最佳的超平面的标准是使两类样本向量中距离超平面最近的向量到超平面的距离之和最小，这个距离之和也被称为间隔，同时，这些到每个类的样本中到超平面距离最近的向量则被称为是支持向量。

SVM的输出结果是超平面的法向量和样本空间中的截距，这两个系数可以唯一确定一个可用于样本分类的超平面，对于任意的向量，代入函数可以得到分类结果。

SVM的训练过程是找到这样一个可以分割不同类样本的超平面并使其中的间隔最大，可以表示为优化过程：

但是直接从此式入手求解过程有些复杂，使用Lagrange乘子法可以把这个问题转化为对应的对偶形式，即凸二次规划问题：

求解得到维的Lagrange乘子向量,系数，根据KKT条件中的约束，找到一个对应的Lagrange乘子为0的向量，也就是支持向量，代入求得的值。这样简化后的过程可以通过简单的编程完成，也可以使用SMO算法[12]等技巧提升计算速度。

原始的SVM方法如上所述，可以解决大部分的简单分类问题，但是依然存在一定的局限，如对于线性不可分的训练集无法得到闭解，以及对于最大间隔的追求可能导致出现过度拟合。所以在基本的SVM之上可以引入核技巧或者软间隔的概念提升SVM的适用性。

对于线性不可分的情况，我们可以将维向量变换为新的维特征空间的向量，一般来说低维无法解决的问题往往向高维寻求解决方法，因此，通常有。如果我们规定新的特征空间中两个向量的内积可以用函数来表示，即，这个函数被称为核函数。使用核函数代换原来的对偶问题中目标函数的低维内积操作为，以及最后的训练结果也就是分类函数

，这种计算技巧被称为是核技巧，在原特征向量空间无法解决或者解决效果不理解的SVM问题中，应用特征空间变换操作可以把样本映射到高维的特征空间寻找可行的解，但是由于我们很难直接计算高维内积，所有可以直接使用现有的核函数计算内积，也等价于应用了该核函数对应的空间变换操作。在设置SVM参数的过程中，选择核函数即选择空间变换的具体形式，如应用可以把原向量映射到无限维空间的高斯核（下文提到RBF核为其中的一种特例）就相当于选择训练和预测过程中把样本变换为无限维。所以，选择一个合适的核函数对于训练效果和预测性能是至关重要的，以下为常见的核函数：

|  |  |  |
| --- | --- | --- |
| 核函数 | 表达式 | 参数 |
| 线性核 |  | 无 |
| 多项式核 |  | 为多项式的最高次数，为系数，为常数项 |
| RBF核 |  | 系数 |
| sigmoid核 |  | 系数，常数项 |

除此之外，经过证明，当且仅当上对称函数对于任意的维向量组成的数据集的运算得到的维核矩阵是半正定矩阵时，这个函数有对应的空间变换映射的，可以作为SVM的核函数使用[13]。因此，除了这些常用的核函数，在实际应用中也可以通过对常用核函数进行线性组合、求直积和其他方法设计实用的核函数。

总结上文，在处理线性不可分的样本集时，将样本空间进行变换，也就是通过核技巧找到可以把异类样本完全分开的曲面。然而事实上训练过程中很难确定能完全正确分类训练集的核函数，即使找到了这样的核函数也有很大是过度拟合的结果，在预测时表现出的效果并不理想。所以，我们可以在训练分类器的时候，对于样本分类的正确性作出一定的妥协，以获得可行的分类函数以及更大的间隔，期望得到更健壮的SVM分类器，这就是软间隔SVM的思想。

为了表现违反分类规则的样本在该分类下的“偏离程度”，我们使用hinge损失函数来量化

并把这一项乘以一个系数与原最小化优化目标函数作和得到新的目标函数，经过与硬间隔SVM相似的求解方法得到一个几乎与硬间隔完全完全相同的凸二次规划表达式，除了的约束由原来的缩小为。参数的作用是调节训练过程中对于间隔大小和对训练集样本分类准确率的权重之比，一般来说，越大，训练过程中对于准确率的偏好越高，对于违反分类的情况的容忍度越低，最后得到的间隔越小，当C趋向于无穷大时，这样的SVM也就退化为了对分类错误零容忍的硬间隔SVM。

在下文的方法中，为了得到高性能的SVM分类器，我们的方法将选择引入核函数的软间隔SVM对图像样本以及其隐写效果进行分类和训练，再对给定图像预测出适合隐藏秘密消息的像素位置。

## 优化实现

通过前面的介绍，我们知道，如果通信双方在嵌入和提取之前已经完成了秘钥交换，那么可以引入伪随机数生成器，以秘钥作为种子确定随机的隐藏位置序列，可以提高的LSB嵌入的安全性。然而，需要注意的是这里的秘钥必须通过安全信道提前完全交换，实际应用中是并不是一定有条件实现这个要求的，因此在没有额外的资源的情况下，保证隐藏的位置完全随机且接收方可以准确提取是不可能的。

本文所做的工作是在与原始的LSB给定的资源条件下实现一个更安全高质量的隐写系统，如上文所述，我们假设在我们的应用场景中无法使用秘钥等开销较大的信息作为系统的基础，但允许双发约定前位作为隐藏信息长度的嵌入位置以及紧接着的位长度的像素嵌入一个坐标，这种假设基于一些无法完成秘钥交换和更新的情况，但和的大小决定了其容易通过约定俗成的方式让双方获得。同时这样的限制让我们无法随机地、离散地选择嵌入位置，只能按照像素的自然顺序嵌入消息，但相比于原始的LSB嵌入方法，新的方案可以选择从图像的一个特定位置开始顺序隐写，相当于从原图像中分割出了一个最小图像块进行顺序嵌入。

无论使用哪种隐写方案，当嵌入率上升到一定程度时，都无可避免地会被隐写分析方法探查出异常的存在；同理，当嵌入率接近0时，即使使用最简单的隐写算法也很难被探查出来。本文仅讨论像素值为8比特的灰度图像（实际对于RGB图像也同理），假设图像大小为像素，只在图像中嵌入位的辅助信息，包括秘密消息的长度和起始隐写像素的坐标，只要满足，已有的通用LSB隐写分析算法对于探查这些消息的存在没有帮助，所以在这里我们可以认为在图像的前位嵌入这些辅助消息是安全的。接下来需要完成的任务就是使用SVM分类器找到整个图像中可以安全嵌入消息的图像块，为了完成这个任务，需要先随机选择一些图片并在位置随机的图像块中嵌入消息，并记录每个图像块的相关特征，再用经典的隐写分析方法评估得到的伪装图像是否安全记为，其中，将作为训练集来训练SVM，多次训练，把这些SVM聚合为一个分类器，依据其判断在给定图像中的指定位置嵌入消息是否安全。

### 图像块的选择

首先为了确定图像块的大小，我们规定使用的图像块为的正方形，因为在同等面积的情况下使用正方形的像素块隐藏数据比较均匀，一般来说可以在一定程度上抗视觉攻击。假设8bit存储的秘密消息的长度为，载体图像大小为，那么正方形图像块的边长的必须满足

，容易求得其中，因此，这样的模式对嵌入率提出更多的限制。

在使用这个新的隐写系统时，首先根据消息长度计算图像块边长，确定图像块的大小，并将的信息嵌入图像的前位。接下来需要在整个图像中寻找适合隐藏消息的图像块，SVM训练完成后，可以使用启发式的方法或者单纯运用随机的方法找到几个图像块作为SVM的输入，从中挑选适合被SVM分类器判定为适合隐藏数据的图像块进行消息的嵌入，并将起始坐标嵌入图像前后紧接的位。

### 特征选择

为了评价一个给定的图像块是否适合隐藏给定的消息，本文选择了四个特征来描述这个图像块。这四个特征分别是方差、整体差异度、sc匹配度和平滑度，它们体现的特征包括图像块像素值的多样度、与整个图像在像素值分布上的相似性、图像块LSB平面与秘密消息的匹配程度以及图像块本身的连续程度，相对于上文提到的几种隐写分析方法针对图像某个单一特征做出评估，我们的这些特征更能完整评价图像块是否可以用于隐藏数据。

#### 方差

本文对于图像块的方差理解和统计学中经典的方差概念一致，用以反映一个样本集中所有样本的离散程度。这里将图像块中每个像素作为一个样本，整个图像块作为一个样本集进行计算，像素的图像块的方差可以按照如下公式计算：

这里的表示坐标为的像素的值，为该图像块所有像素的值的平均值。

一般来说，越平滑的图像块方差越小，其中包含的像素之间的差异越小，越不适合隐藏数据。

#### 整体差异度

整体相似度反映了图像块与被提取的原图像的关系，我们用像素值的频率来描述图像和图像块的一些特征，使用它们之间的差值的平方和，也就是频率差值向量的二范数的平方来放大图像块与原图像之间的差异。对于大小为图像块和大小为图像，频率向量分别用如下公式计算：

其中函数是一个值域为的逻辑判断函数，当自变量表达式的值为真时其函数的值为1，否则为0。的值为一个256维向量，上式给出的是向量中每一个值的计算方法，这里的索引值从0开始计数。 接下来我们计算这两个向量中每个对应的像素值间的差值的平方和作为整体差异度

通过上述的计算，我们可以容易发现，对于这个可以反映像素值分布差异的特征，不同于其他特征仅考虑用于隐藏消息的位置自身的特性，整体差异度还考虑了选取的图像块在整个原图像中扮演的角色，也就是说与整体的融合程度。从概念来看，在图像块大小一定的情况下整体差异度的值越大说明该图像块在视觉上越突出，但不能简单地反映是否适合隐藏图像，只能作为一个用以辅助参考的特征。具体来说这样的“突出”需要分具体情况探讨在哪方面突出，隐藏数据的数据是否会因此在视觉攻击时暴力，都需要结合其他特征，如平滑度评价是否为不宜嵌入数据的突出的平滑区。另外，虽然在计算频率时有归一化的处理，但整体差异度的取值仍然很大程度地受到图像块大小的影响，这种计算方式的在图像块过小的时候将失去意义。

#### sc匹配度

sc(secret-cover)匹配度是指图像块本身的LSB平面与秘密消息的二进制表达之间相似程度，可以用来描述秘密消息和图像内容的关系。我们在介绍隐写术的时候已经提到为了不让人觉察到隐藏消息的存在，秘密消息的内容和图像内容往往是无关的，所以在这里说的内容并非其表达的内容，这样的内容因为多媒体文件的存储和表达的原因，很难在高层表现上让人察觉两种的联系，但可以通过底层实现探究图像和消息的关系，具体来说就是把它们都转换为二进制的表达形式，度量这些二进制数据间的相似程度。这种做法看似毫无意义，但是考虑到我们对秘密消息的隐写和恢复的过程也是通过将图像和消息都转换为二进制数据来实现的，以秘密消息的二进制形式来代替原来的二进制LSB平面，所以计算它们之间的相似性对于评价图像块是否适合隐藏特定的数据非常重要。

我们将长度为的秘密消息转换为8bit的二进制数据并以0-1向量表示为，同时以自然顺序取图像块中前像素的LSB位并构建0-1向量，这两个向量中对应位置的元素相同的个数与整个向量长度之比记为SC匹配度，按照如下公式计算

这里使用和分别表示和的第个元素，函数的定义与整体差异度中的函数一致，用以判定两个二进制表达中相同的位。

一般来说，对于性质良好的自然图像，sc匹配度越高，隐藏的数据就越能被察觉。在嵌入率不变的情况下，随着sc匹配度的提高，隐写过程对于载体图像的修改量越少，得到的图像越接近原图像，如果原图是典型的自然图像，也能很好地符合一些自然图像特征的图像，那么得到到伪装图像也接近自然图像，很难通过现有的隐写分析手段探查出隐藏消息的存在。我们可以想象一种极端的情况，要嵌入的秘密消息（包括消息长度等辅助信息）的二进制表达形式恰好与原图像的LSB平面完全一致，也就是sc为100%时，消息的嵌入过程不需要对图像做任何修改，载体图像与伪装图像完全一致，所有的隐写分析方法对于载体和伪装的检测结果都相同，显然无法发现隐藏消息。同理，sc匹配度为0时也相当于对图像用于嵌入消息所有像素的LSB位都进行了修改，这时伪装图像被检测出异常的概率就非常高。然而，这两种情况都太过理想。对于典型的隐写应用场景来说，秘密消息是特定的，图像以及可以选择的图像块也是有限的，且二者作为承载信息的数据都表现出一定的规律性，极少出现匹配率极高和极低的情况。为了模拟隐写场景，我们在实验中使用0~255间的随机整数构成数组作为秘密消息，经过测试，sc匹配度往往在0.4~0.6之间。

#### 平滑度

平滑度的概念是借鉴了前文RS隐写分析中提到的平滑度计算，两者有相似之处，在本文中使用的平滑度相对于RS隐写分析中提出的平滑度函数在定义上更为完善。原始的平滑度函数中将像素矩阵降维处理压平为向量，然后根据一个方向计算差值的绝对值之和，如随着的增大累加，这样的简单计算一个像素与前后像素的差值的模式不能完全反映该像素与周围像素的关系，因此，这里我们需要全面改进这种计算方式。

对于非边缘的像素，周围一共有8个像素，这8个像素位于该像素的8个不同方向，这种扩展的差值计算也被应用在一些隐写分析算法中[14]，可以根据方向表示为关于的集合

那么对于在边缘的像素点，相应的集合则会缺少一些方向的数据，仅仅保留周围存在的像素。那么我们可以使用以下的求和方式计算大小的图像块的平滑度

此时得到的平滑度函数充分考虑了像素点和各个方向的邻接像素的关系，但是，我们容易看出，有些差值也因此被重复计算，而且直接利用以上公式求值的计算量较大，因此我们可以使用矩阵偏移的方式计算平滑度。

令表示以为起始坐标（左上角），为终止坐标（右下角）的像素矩阵，那么整个图像块可以表示为，我们可以使用如下公式计算每个方向上的像素绝对差矩阵

对这四个矩阵中每个元素作求和运算，则可以避免重复计算，而且直接对矩阵进行运算也大大提高了计算效率。

平滑度沿用了RS隐写分析中对于图像混乱程度的定义，与命名的直观含义相违背的是，平滑度越大，各个相邻像素间的差异越大，代表的图像的混乱程度越大。平滑度的含义与方差有一定的重合，都在一定程度上反映了图像各个像素间的差异性，但需要注意的是这两个特征体现了图像不同方面的离散程度，方差仅考虑图像作为一个无序像素集时各个像素之间的关系，而平滑度则考虑了像素在图像中的位置及其与周围像素的关系，两个特征存在一些联系但并不正相关，假设有一个图像块仅由同样大小的黑色色块（像素值0）和白色色块（像素值255）拼接而成，且交接的边界很小，对其计算两种特征，我们会发现它的方差很大但是平滑度不一定很大，尤其是对于图像本身较大的情况。

### SVM的训练和预测

在我们的优化方案中，已经完成训练的SVM连同秘密消息以及载体图像是作为嵌入消息过程的输入使用的，因此对于整个隐写系统来说，训练一个性能良好的SVM是预处理过程中应该完成的，且训练好的SVM是可以用于下一次隐写的。

首先我们需要以随机采样的方法构造训练集、检验集和测试集。应用场景假设在预处理过程中我们有足够多的图像资源和计算资源，生成长度已知的随机数组作为嵌入消息，那么构造训练集的过程可以随机的在每个已有的图像上选择若干个图像块，对于每个小图像按照前文提到的方法提取对应的四个特征作为该样本的特征向量，在小图像块嵌入消息后，使用基于阈值的RS隐写分析算法判断得到的整个伪装图像是否可以被认定为隐藏了消息的可疑图像，将隐写分析的结果作为样本的标签。

在预设的资源足够的预处理阶段，使用这样的采样方法可以构造出足够大的样本集。我们可以把这样的样本集简单分割为训练集、检验集和测试集，也可以按照一定的指标对样本分组进行测试，如嵌入率。

值得注意的是，采样过程中为了使样本尽可能多样化，我们对于每个样本使用的是随机数组，然而实际上消息隐写的实践中消息本身不可能完全随机无序，在二进制表达上会体现出一定的规律性，如字母的ASCII编码的二进制形式第一位为1，所以由字母构成的字符串在转化为字节流时也会每隔8位出现1，我们在这里不考虑这种规律给我们训练上带来的影响，因为我们的目的是训练出隐写技术适用范围更广泛的SVM。

## 实验与结果

### 数据集和实验平台设置

本文的实验中使用了v2版本的UCID图像数据集[15]作为图像资源。UCID数据集包含了1338张未经压缩的RGB彩色图像，每张图像的大小为或像素，格式为tif，图像的主题丰富，包括了风景、建筑、动植物和人物。为了简化操作突出实验的效果，我们把这些图像都转化为了灰度图像，实际上我们的机制对于像素图像的处理原理相同，使用RGB图像的效果也类似因此不再重复实验。

实验中使用的操作系统为win10，数据的部分预处理和收集工作使用Python3.5完成，大部分的运算则在Matlab2015b实现，其中SVM相关部分的训练和预测借助了LIBSVM3.21[16]工具箱完成。

### 实验设置

因为嵌入率对于隐写分析效果的影响巨大，而使嵌入率5%~50%是隐写的常见情况，为了在隐写系统中充分使用我们训练出来的模型，实验中我们设置了步长为5%的嵌入率从5%上升到50%的10组数据样本量为200的样本合成为一个样本量为2000的训练集，用另外2000个嵌入率分布相同的样本作为检验集。这些样本的采集过程则是迭代地在不同的200张灰度图像上随机选择与嵌入率相符的大小的图像块进行隐写，并提取特征和分析结果作为样本的数据。使用训练集训练不同参数的SVM后，我们通过它们在几个检验集上的表现调整训练参数，综合考虑支持向量的个数和准确率选择最佳的SVM分类器。

### 实验结果

我们使用了80组不同的参数对SVM进行训练，它们的关于训练集的错误率和检验集的错误率如散点图所示，我们选择其中比较性质较好的7组典型的参数如下表所示

接下来我们使用这7个SVM分类器对于嵌入率为5%~50%的11组在1338张图像中随机选择5个图像块嵌入消息得到的样本集进行分类测试，结果图所示

从图中可以明显观察得到，所有分类器的预测准确率都随着嵌入率的升高而表现出下降的趋势，但不同的是低次低代价的线性核和多项式对应的准确率的下降相对于高次高代价的多项式核以及RBF核更为平缓，虽然在低嵌入率时前者的表现较差，但即使嵌入率升高，前者的表现依然稳定。这是因为高次核以及高代价有很大概率会导致过度拟合，随着测试样本的变化，预测的准确率会发生很大的变化。同时，当嵌入率增大时，图像块的大小也随之增大，包含的信息量也随之增加，有限的选取特征对于图像块的描述力度减小，图像块对于用以隐藏消息是否适合的不确定度也有所增加，之前的分类器的决策水平随之降低，所以会出现准确度下降的现象。但是从整体上看，我们可以认为至少当嵌入率为5%~25%时，我们的分类器确实可以以高准确率帮助隐写者判定适合隐藏数据的位置。

接下来使用这6个SVM在5%~50%的11组样本量均为200的图像块上预测隐写的安全性，并使用样本对分析（SPA）的方法检验预测的准确率，可以作出图

从图中可以观察到，在随着嵌入率在5%-25%间升高，所有SVM分类器的准确率都有一定的上升趋势，当嵌入率到达25%以后，低次低代价的核在准确率上的表现开始出现回退，而高次核的表现出现很大的提高，这种结果正好与使用RS隐写预测的结果相反，也说明了高次核的使用对于对抗SPA攻击非常有效。通过第3章的研究，我们可以很容易发现RS隐写分析主要考虑特征变化自身单纯的变化，而SPA分析则将很多可能出现的特征结合考虑，分析的过程更为复杂，这种情况下使用高次核可以获得更好的预测效果。

低次核与高次核都有各自的优势和劣势，需要考虑具体的使用场景进行应用。当然，在以后的工作中，我们也可以考虑根据应用场景使用对多个低次核和高次核进行加权聚合得到一个更强大的SVM。[17] ###总结

本文对LSB隐写系统进行了深入的研究并探索了现有模式的局限，提出了一种适用于缺少安全信道交换秘钥时对LSB隐写在安全上的改进，但是这种改进的代价是需要寻找适合的数据集训练分类器，消耗一定的计算资源得到可以长期使用的选择隐写位置的SVM分类器，相较传统的顺序呢隐写方法，该分类器经过试验评估，确实可以在一定程度上提高隐写的安全性。[18–21][22][23]

[1] SCHAATHUN H G. Machine learning in image steganalysis[M]. Chichester West Sussex UK: Wiley-IEEE Press, 2012.

[2] BACKES M, CACHIN C. Public-key steganography with active attacks[G]//Theory of cryptography. Springer, 2005: 210–226.

[3] FILLER T, FRIDRICH J. Complete characterization of perfectly secure stego-systems with mutually independent embedding operation[C]//Acoustics, speech and signal processing, 2009. iCASSP 2009. iEEE international conference on. IEEE, 2009: 1429–1432.

[4] FRIDRICH J, GOLJAN M, SOUKAL D. Searching for the stego-key[C]//Electronic imaging 2004. International Society for Optics; Photonics, 2004: 70–82.

[5] WAYNER P. Disappearing cryptography: Information hiding: Steganography and watermarking (2Nd edition)[M]. 第2nd版. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002.

[6] FRIDRICH J, GOLJAN M, DU R. Steganalysis based on jPEG compatibility[C]//ITCom 2001: International symposium on the convergence of iT and communications. International Society for Optics; Photonics, 2001: 275–280.

[7] WESTFELD A, PFITZMANN A. Attacks on steganographic systems[C]//Information hiding. Springer, 1999: 61–76.

[8] DUMITRESCU S, WU X, MEMON N. On steganalysis of random lSB embedding in continuous-tone images[C]//Image processing. 2002. proceedings. 2002 international conference on. 2002, 3: 641–644 vol.3.

[9] FRIDRICH J, GOLJAN M, DU R. Reliable detection of lSB steganography in color and grayscale images[C]//Proceedings of the 2001 workshop on multimedia and security: New challenges. ACM, 2001: 27–30.

[10] JOHNSON R, BHATTACHARYYA G. Statistical concepts and methods[J]. Wiley series in probality and mathematical statistics., 1977.

[11] PROVOS N, HONEYMAN P. Hide and seek: An introduction to steganography[J]. Security & Privacy, IEEE, IEEE, 2003, 1(3): 32–44.

[12] PLATT J, OTHERS. Sequential minimal optimization: A fast algorithm for training support vector machines[J]. technical report msr-tr-98-14, Microsoft Research, 1998.

[13] SCHÖLKOPF B, SMOLA A J. Learning with kernels: Support vector machines, regularization, optimization, and beyond[M]. MIT press, 2002.

[14] 闫晓蒙, 张涛, 奚玲, 等. 一种针对 lSB 匹配隐写的负载定位新算法[J]. 数据采集与处理, 2016, 31(1): 145–151.

[15] SCHAEFER G, STICH M. UCID: An uncompressed color image database[C]//Electronic imaging 2004. International Society for Optics; Photonics, 2003: 472–480.

[16] CHANG C-C, LIN C-J. LIBSVM: A library for support vector machines[J]. ACM Transactions on Intelligent Systems and Technology, 2011, 2(3): 27:1–27:27.

[17] GE S, GAO Y, WANG R. Least significant bit steganography detection with machine learning techniques[C]//Proceedings of the 2007 international workshop on domain driven data mining. ACM, 2007: 24–32.

[18] HEMPSTALK K. Hiding behind corners: Using edges in images for better steganography[C]//Proc. computing women’s congress, hamilton, new zealand. 2006, 14.

[19] SINGH K M, SINGH L S, SINGH A B, 等. Hiding secret message in edges of the image[C]//2007 international conference on information and communication technology. 2007.

[20] SWANSON M D, ZHU B, TEWFIK A H. Robust data hiding for images[C]//Digital signal processing workshop proceedings, 1996., iEEE. IEEE, 1996: 37–40.

[21] WU D-C, TSAI W-H. A steganographic method for images by pixel-value differencing[J]. Pattern Recognition Letters, Elsevier, 2003, 24(9): 1613–1626.

[22] LUO W, HUANG F, HUANG J. Edge adaptive image steganography based on lSB matching revisited[J]. Information Forensics and Security, IEEE Transactions on, IEEE, 2010, 5(2): 201–214.

[23] PLATT J. Sequential minimal optimization: A fast algorithm for training support vector machines[R]. MSR-TR-98-14, Microsoft Research, 1998: 21.