创新创业实践

Project 2

实验报告

姓名 迟曼

学号 202200460070

学院 网络空间安全

专业 网络空间安全

目录

一、实验目的

Project 2:基于数字水印的图片泄露检测，编程实现图片水印嵌入和提取（可依托开源项目二次开发），并进行鲁棒性测试，包括不限于翻转、平移、截取、调对比度等。

二、文件说明

cmwatermark.py	代码文件
test_image.jpg	测试图像文件
Project2 说明-202200460070-迟曼.pdf	说明文档

三、实验原理

数字水印的本质是将特定信息（如版权标识、用户 ID）不可见地嵌入图像数据中，利用人类感知冗余（如对高频信号或细微像素变化不敏感）实现隐蔽性。根据目标不同，水印分为两类：

1.鲁棒水印：用于版权保护与泄露溯源，需抵抗压缩、裁剪、滤波等攻击，核心是保障攻击后水印可提取。

2.脆弱水印：用于内容认证，对篡改敏感（如像素修改、裁剪），可定位篡改区域。水印嵌入域分为空域（直接修改像素）与变换域（修改频域系数）。空域方法简单高效但抗攻击弱；变换域方法通过修改频域系数平衡不可见性与鲁棒性。

（一）量化索引调制模型：

实验代码采用改进的量化索引调制模型（Quantization Index Modulation, QIM），是一种基于量化思想的非线性信息隐藏技术，主要用于数字水印、隐写术等领域。其核心目标是在载体信号（如图像、音频）中嵌入隐藏信息（如水印），并实现盲提取（即无需原始载体即可提取信息），同时平衡不可见性、鲁棒性和嵌入容量。

嵌入公式：

$$F'(u,v)=\begin{cases}\Delta\cdot round\left(\dfrac{F(u,v)}{\Delta}\right)+\dfrac{\Delta}{4} & w_m=1\\ \Delta\cdot round\left(\dfrac{F(u,v)}{\Delta}\right)-\dfrac{\Delta}{4} & w_m=0\end{cases}$$

其中$F(u,v)$是原始频域系数，Δ是自适应量化步长$\Delta=\alpha\cdot\sigma_{block}$，$\alpha$是强度因子，$\sigma_{block}$代表图像块

标准差，w_m是水印比特。

（二）频域系数选择策略：

采用 DCT 分块变换，将输入图像分割为 8×8 像素块，对每块进行二维 DCT 变换，将空域像素转换为频域系数矩阵。频域系数选择策略：

$$P(u,v) = \frac{1}{\sqrt{\left(u-\frac{N}{2}\right)^2 + \left(v-\frac{N}{2}\right)^2}} \cdot e^{-\frac{(u-v)^2}{2\sigma^2}}$$

优先选择 DCT 系数矩阵的中频区域：

$$\alpha_{final} = \alpha(1 + 0.2 \cdot \log(1 + \sigma_{global}))$$

该区域既避开高频（易受压缩影响）又避开低频（人眼敏感）。通过公式修改系数：

$$C_{mod} = C_{orig} + \alpha \cdot w$$

其中$w \in \{-1,1\}$表示水印比特（二值化后），α为嵌入强度因子，控制不可见性与鲁棒性平衡。

（三）抗攻击能力数学建模

DCT 分块特性使局部几何变形不影响全局水印分布。水平翻转通过系数对称性天然抵抗。平移攻击因分块独立性，未移动的块仍可提取水印。

1.几何攻击（翻转/平移）

攻击模型：

$$I' = T_{affine} \cdot I$$

鲁棒性来源于频域系数的幅度不变性：

$$|F_{transformed}| \approx |F_{original}|$$

2.噪声攻击

椒盐噪声模型：

$$I'(x,y) = \begin{cases} 0 & p < prob/2 \\ 255 & p > 1 - prob/2 \\ I(x,y) & otherwise \end{cases}$$

生存条件：

$$prob < \frac{\alpha}{10}$$

3.JPEG 压缩

压缩主要丢弃高频分量，中频水印得以保留。光度调整（亮度/对比度），量化过程：

$$F_Q(u,v) = round(\frac{F(u,v)}{Q(u,v)})$$

水印存活条件：

$$\frac{\Delta}{4} > \frac{\| Q \|_2}{2}$$

4.光度调整（亮度/对比度）

线性变换$g(x) = \alpha x + \beta$在频域表现为系数缩放:

$$\mathcal{F}\{g(x)\} = \alpha \cdot \mathcal{F}\{x\} + \beta \cdot \delta(f)$$

水印嵌入系数 α"缩放后仍保持相对关系,提取不受影响。"

四、实验过程

导入三个关键模块，其中 blind_watermark 实现数字水印的嵌入和提取功能；cv2 (OpenCV)：处理图像操作（如翻转、裁剪等）；numpy：提供数值计算支持，用于矩阵运算。

```
from blind_watermark import WaterMark
import cv2
import numpy as np
```

（一）水印嵌入功能

水印嵌入模块负责将文本信息"SECRET"不可见地植入原始图像中。通过创建 WaterMark 类实例并设置双重密钥 1（password_img 控制嵌入位置分布，password_wm 控制信息加密，相当于加解密密钥），首先读取原始图像并转换为频域处理格式。随后将字符串编码为二进制比特流，执行核心水印嵌入操作：将图像分块进行 8×8 像素的 DCT 变换，选择人眼不敏感的中频区域，按公式修改系数值后逆变换回空域，最终生成含水印图像 embedded.png。该过程保持视觉不可感知性，典型 PSNR 值超过 40dB。最后记录并输出水印的比特长度（后续提取需要此参数）

```
# ========== 嵌入水印 ==========
bwm1 = WaterMark(password_img=1, password_wm=1)
bwm1.read_img('test_image.jpg')
wm = 'SECRET'
bwm1.read_wm(wm, mode='str')
bwm1.embed('embedded.png')
len_wm = len(bwm1.wm_bit)
print(f'水印比特长度：{len_wm}')
```

（二）攻击模拟

```
    # 1. 水平翻转（几何攻击）[3,6](@ref)
    attacks['水平翻转'] = cv2.flip(img, 1)

    # 2. 垂直翻转（几何攻击）
    attacks['垂直翻转'] = cv2.flip(img, 0)

    # 3. 平移攻击（30像素偏移）[6](@ref)
    M = np.float32([[1, 0, 30], [0, 1, 30]])
    attacks['平移'] = cv2.warpAffine(img, M, (img.shape[1], img.shape[0]))

    # 4. 随机截取（裁剪50%区域）[1](@ref)
    h, w = img.shape[:2]
    attacks['截取'] = img[int(h / 4):int(3 * h / 4), int(w / 4):int(3 * w / 4)]

    # 5. 对比度调整（增强对比度）[6](@ref)
    attacks['高对比度'] = cv2.convertScaleAbs(img, alpha=1.5, beta=0)

    # 6. 亮度调整（增加亮度）
    attacks['高亮度'] = cv2.convertScaleAbs(img, alpha=1.0, beta=50)

    # 7. 椒盐噪声（5%噪声密度）[3](@ref)
    noise = np.copy(img)
    prob = 0.05
    rnd = np.random.rand(*noise.shape[:2])
    noise[rnd < prob / 2] = 0  # 椒噪声
    noise[rnd > 1 - prob / 2] = 255  # 盐噪声
    attacks['椒盐噪声'] = noise

    # 8. JPEG压缩（质量因子=30）[1](@ref)
    cv2.imwrite('temp_compressed.jpg', img, [int(cv2.IMWRITE_JPEG_QUALITY), 30])
    attacks['JPEG压缩'] = cv2.imread('temp_compressed.jpg')
```

apply_attacks 函数对输入的图像实施 8 种不同类型的攻击操作：

几何攻击：包括水平翻转（cv2.flip 实现图像镜像）和平移攻击（warpAffine 移动 30 像素）。

裁剪攻击：截取中心 50%区域验证局部水印存活率；

色彩攻击：光度调整类通过参数 alpha=1.5 增强对比度、beta=50 提升亮度，检查线性变换稳定性；

噪声攻击：添加 5%密度椒盐噪声模拟传输干扰；

压缩攻击：采用质量因子 30 的 JPEG 压缩检验频域抗性。

攻击模拟模块系统构建八类现实攻击场景以评估鲁棒性，构建完整的多场景测试集。

（三）应用攻击并提取水印

```
# ========== 应用攻击并提取水印 ==========
attacked_images = apply_attacks('embedded.png')
results = []

for attack_name, img in attacked_images.items():
    # 保存攻击后图像
    attack_path = f'attacked_{attack_name}.png'
    cv2.imwrite(attack_path, img)

    # 尝试提取水印
    bwm2 = WaterMark(password_img=1, password_wm=1)
    try:
        wm_extract = bwm2.extract(attack_path, wm_shape=len_wm, mode='str')
    except Exception as e:
        wm_extract = f"提取失败: {str(e)}"

    # 记录结果
    results.append({
        '攻击类型': attack_name,
        '提取结果': wm_extract,
        '图像路径': attack_path
    })

    print(f"{attack_name}攻击后提取结果: {wm_extract}")
```

水印提取模块针对攻击后图像进行信息复原。通过循环处理各攻击结果，使用相同密钥同步嵌入位置，对攻击图像重新计算 DCT 系数，采用自适应阈值进行比特判决。当提取系数大于阈值时判定为 1，否则为 0，最终将比特流解码还原为 ASCII 字符串。异常处理机制捕获几何攻击导致的同步失败等情况，所有提取结果被结构化存储为包含攻击类型、提取结果和图像路径的三元组。

（四）结果分析模块

```
# ========== 输出最终报告 ==========
print("\n=== 鲁棒性测试最终报告 ===")
for res in results:
    status = "成功" if res['提取结果'] == wm else "失败"
    print(f"{res['攻击类型']}: 提取{status} -> {res['提取结果']}")

# ========== 完整提取代码（无攻击） ==========
bwm_final = WaterMark(password_img=1, password_wm=1)
wm_extract_final = bwm_final.extract('embedded.png', wm_shape=len_wm, mode='str')
print(f"\n最终提取的水印信息: {wm_extract_final}")
```

在代码的末尾，生成易读的测试报告，显示每种攻击类型、水印是否成功提取（与原始水印比对）、实际提取出的内容，从而直观展示水印算法对各种攻击的抵抗能力。

```
Welcome to use blind-watermark, version = 0.4.4
Make sure the version is the same when encode and decode
Your star means a lot: https://github.com/guofei9987/blind_watermark
This message only show once. To close it: `blind_watermark.bw_notes.close()`

[ WARN:0@6.740] global loadsave.cpp:848 cv::imwrite_ Unsupported depth image for selected encoder is fallbacked to CV_8U.
水印比特长度：47
水平翻转攻击后提取结果: )◆◆◆'
垂直翻转攻击后提取结果: H◆◆◆k◆
平移攻击后提取结果: ,◆◆*▫3
截取攻击后提取结果: ▫!▫T◆◆
高对比度攻击后提取结果: SECRET
高亮度攻击后提取结果: SECRET
椒盐噪声攻击后提取结果: SECRET
JPEG压缩攻击后提取结果: SECRET

=== 鲁棒性测试最终报告 ===
水平翻转: 提取失败 -> )◆◆◆'
垂直翻转: 提取失败 -> H◆◆◆k◆
平移: 提取失败 -> ,◆◆*▫3
截取: 提取失败 -> ▫!▫T◆◆
高对比度: 提取成功 -> SECRET
高亮度: 提取成功 -> SECRET
椒盐噪声: 提取成功 -> SECRET
JPEG压缩: 提取成功 -> SECRET

最终提取的水印信息: SECRET
```

根据最终结果，可以看出，该水印算法具有一定的鲁棒性，可以成功提取遭受高对比度，高亮度，椒盐噪声以及 JPEG 压缩攻击的图片水印。

五、总结与思考

本次实验围绕数字水印技术在图片泄露检测中的应用展开，通过编程实现了基于频域的水印嵌入与提取流程，并系统性地测试了其抗攻击能力。在实验过程中，依托开源库 blind_watermark 进行二次开发，重点验证了量化索引调制（QIM）模型在 DCT 域中频系数嵌入水印的可行性。

实验结果表明，所实现的水印系统对部分攻击展现出良好的鲁棒性。在高对比度调整、高亮度调整、椒盐噪声（5%密度）以及低质量 JPEG 压缩（质量因子 30）等攻击下，水印信息"SECRET"仍可被完整提取，说明算法在应对光度变换、噪声干扰和频域压缩方面具备一定优势。这得益于 QIM 模型的自适

应量化步长设计以及中频系数的选择策略——该区域既避开了人眼敏感的低频，又避开了易被压缩去除的高频，同时光度线性变换对频域系数的缩放特性未破坏水印的量化关系。

然而，水印系统对几何攻击和裁剪攻击的抵抗能力存在明显不足。水平翻转、垂直翻转和平移攻击导致水印提取完全失效或出现乱码，截取 50%区域后水印亦无法恢复。分析原因在于：DCT 分块处理虽具备局部独立性，但几何变换破坏了块的空间结构关系，导致水印嵌入位置与提取时的块索引失配；而裁剪攻击直接移除了大量嵌有水印的数据块，严重损失信息量。这反映了当前算法在全局同步机制上的局限性，未来需考虑引入模板同步或特征点匹配等策略增强几何鲁棒性。

通过本次实践，深刻认识到数字水印技术需在不可见性、鲁棒性与容量三者间寻求平衡。实验中调整量化步长参数 α 的经验表明，过高的 α 虽提升抗攻击能力，但会引入可见失真；过低则导致水印信号易被噪声淹没。