

图像隐写分析-DCT特征编程实现

原创

[iamsongyu](#) 于 2018-09-12 10:14:56 发布 5780 收藏 17

分类专栏: [编程实践](#) [理论知识](#) 文章标签: [图像隐写分析](#) [DCT特征](#) [jpeg图像隐写](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/iamsongyu/article/details/82657164>

版权



[编程实践](#) 同时被 2 个专栏收录

19 篇文章 0 订阅

订阅专栏



[理论知识](#)

109 篇文章 7 订阅

订阅专栏

在图像隐写分析中, 这几个特征是比较经典的

图像隐写分析中DCT特征与Markov特征展现出了极大的潜力, 小波变换的奇异值分解 (Wavelet Singular Value Decomposition, WSVD) 特征也有奇效, 本文实现前人论文的特征提取编程代码, 基于matlab

先说说理论知识

扩展DCT统计特征提取

大多数的隐密算法都是对JPEG图像的DCT系数进行操作，以此来嵌入秘密信息。DCT系数统计特征，旨在捕捉DCT系数的统计量的特征，以此来区分载体图像和隐密图像。

DCT系数统计算法由Fridrich【1】提出，其中包含了DCT系数直方图，共生矩阵，空域块间相关性等部分。首先用DCT系数替换相同位置的原始图像像素，使用 $d_{ij}(k)$ 来表示DCT系数矩阵，其中 $i, j=1, \dots, 8, k=1, \dots, nB$ 。而 $d_{ij}(k)$ 则代表的是在第 k 个 8×8 DCT块中处于 (i, j) 位置的DCT系数，而DCT块一共有 nB 块。为了减少计算量和特征维度，在计算特征之前需要进行预处理，将所有DCT系数值范围限定在 $[-5, 5]$ 之间，大于和小于该范围内的值全部变换为 -5 到 $+5$ 之间。

第一个特征是全局所有 $64 \times nB$ 个 DCT 系数的统计值 H ：

$$H = (H_L, \dots, H_R) \quad (2.6)$$

其中 $L = \min_{i,j,k} d_{ij}(k), R = \max_{i,j,k} d_{ij}(k)$ 。

第二个特征是 5 个直方图特征：

$$h^d = (h_L^d, \dots, h_R^d) \quad (2.7)$$

上式计算了五个局部位置的 DCT 系数统计量，即 $(i, j) \in \{(1, 2), (2, 1), (3, 1), (2, 2), (1, 3)\}$ 。

接下来的 11 个特征是用 8×8 矩阵 $g_{ij}^d, i, j=1, \dots, 8, d=-5, \dots, 5$ 代表的双柱状图：

$$g_{ij}^d = \sum_{k=1}^{nB} \delta(d, d_{ij}(k)) \quad (2.8)$$

其中， $\delta(x, y)$ 表示仅当 $x=y$ 时取值为 1，其余的情况下为 0。

接下来的 6 个特征捕捉了 DCT 系数的块间相关性，第一个特征是 V ：

$$V = \frac{\sum_{i,j=1}^8 \sum_{k=1}^{U_i-1} |d_{ij}(I_r(k)) - d_{ij}(I_r(k+1))| + \sum_{i,j=1}^8 \sum_{k=1}^{U_i-1} |d_{ij}(I_c(k)) - d_{ij}(I_c(k+1))|}{|I_r| + |I_c|} \quad (2.9)$$

其中， lr 和 lc 表示图像DCT系数块的两种排列方式，分别是行扫描顺序和列扫描顺序。

接下来的两个特征 B_α 是从解压的JPEG图像中计算，也是一种块间相关性的特征：

$$B_\alpha = \frac{\sum_{i=1}^{[(M-1)/8]} \sum_{j=1}^N |C_{8i,j} - C_{8i+1,j}|^\alpha + \sum_{j=1}^{[(N-1)/8]} \sum_{i=1}^M |C_{i,8j} - C_{i,8j+1}|^\alpha}{N[(M-1)/8] + M[(N-1)/8]} \quad (2.10)$$

在该公式中， M 和 N 表示的是图像尺寸的宽度和高度， C_{ij} 表示 JPEG 图像被解压到空域之后的像素灰度值，其中 $\alpha=1, 2$ 。

其余的特征是从相邻 DCT 系数的共生矩阵计算而来：

$$\begin{aligned} N_{00} &= C_{0,0}(J_1) - C_{0,0}(J_2) \\ N_{01} &= C_{0,1}(J_1) + C_{0,1}(J_2) - C_{1,0}(J_2) + C_{-1,0}(J_1) - C_{-1,0}(J_2) + C_{0,-1}(J_1) - C_{0,-1}(J_2) \\ N_{11} &= C_{1,1}(J_1) + C_{1,1}(J_2) - C_{1,-1}(J_2) + C_{-1,1}(J_1) - C_{-1,1}(J_2) + C_{-1,-1}(J_1) - C_{-1,-1}(J_2) \end{aligned} \quad (2.11)$$

其中，功能 C 的定义如下：

$$C_x = \frac{\sum_{i,j=1}^8 \sum_{k=1}^{U_i-1} \delta(s, d_{ij}(I_r(k))) \delta(t, d_{ij}(I_r(k+1))) + \sum_{i,j=1}^8 \sum_{k=1}^{U_i-1} \delta(s, d_{ij}(I_c(k))) \delta(t, d_{ij}(I_c(k+1)))}{|I_r| + |I_c|} \quad (2.12)$$

在DCT系数统计的隐密分析中，Fridrich首次提出了用于隐密分析的“校准”概念和计算原理：特征计算函数 F ，训练或测试图像 J_1 ，将图像 J_1 解压到空域并沿各个方向裁剪四个像素，然后使用同 J_1 相同的量化表压缩得到的图像 J_2 。 f 表示最终获取的特征，而最后的特征由 $f=F(J_1)-F(J_2)$ 计算得到。

采用如此计算方式的原理如下：裁剪之后的图像和原始图像内容上大体上完全一致，虽然裁剪之后的图像失去了原来的DCT块，但是其统计特征应与原来相差不多。而这个过程会对嵌入的信息十分敏感，使裁剪前后的特征差别较大。经过实验证明，如此提取特征的方法非常有效果。

总结来说，DCT系数统计特征对DCT系数全局和局部进行了统计分析，并且捕获DCT系数的块间相关性和空域像素的相关性等特征。对于JPEG图像来说，所有隐密算法都是针对DCT系数进行修改，该算法确实是有一定的效果。实验中，该特征集展现了不错的分析效果，在0.2的嵌入率情况下可以达到平均95%的准确率，但是对MB算法的效果一般，尤其是MB2。

原始DCT统计特征已经有一定的检测效果，本文先对其进行扩展，加强特征的检测效果。对于全局直方图函数H，可以得到范围在[-5, +5]中的元素个数的差异，包括全局直方图和局部直方图，局部直方图选择的位置为{(1, 2), (2, 1), (3, 1), (2, 2), (1, 3)}。因此，直方图特征是：

$$H_l(J_1) - H_l(J_2), l \in \{-5, +5\} \quad (3.1)$$

$$h_l^j(J_1) - h_l^j(J_2), l \in \{-5, +5\} \quad (3.2)$$

接下来的双直方图，选择这样九个位置：

$$g_{ij}^d(J_1) - g_{ij}^d(J_2), (i, j) \in \{(2,1), (3,1), (4,1), (1,2), (2,2), (3,2), (1,3), (2,3), (1,4)\} \quad (3.3)$$

在共生矩阵特征中，控制元素的范围在[-2,+2]之间，这样共产生 25 个特征：

$$C_x(J_1) - C_x(J_2), s \in [-2, +2], t \in [-2, +2] \quad (3.4)$$

如此的DCT扩展特征共有193维，其特征组成见下表。

表 3.1 193 维 DCT 扩展特征组成

特征	维度
全局直方图统计	11
5个局部直方图统计	5×11
11个双柱状图特征	11×9
块间相关性 V	1
块间相关性 B _a	2
共生矩阵	25

实现代码

基于matlab的代码如下，需要安装jpegtoolbox工具箱，这个工具我记得需要编译，晚上就可以找到安装方法很方便：

```
function [y,DctinsteadIm,DctinsteadImJ2]=DCT193featureNEW(x,impath,quality)
%提取DCT特征 193维 因为需要获取图像的dct系数 也需要做裁剪 随意需要提供的参数如下:
%impath 图像的路径
%quality 图像的质量函数 例如aa

%输出
%y 193维特征
%DctinsteadIm 原始图像dct系数
%DctinsteadIm2 裁剪之后图像dct系数

% disp('----分配初始变量----');
blocknum=zeros(1,2); %长 宽
% Qtable=[16 11 18 16 24 40 51 61; %量化矩阵
%         12,12,14,19,26,58,60,55;
%         14,13,16,24,40,57,69,36;
%         14,17,22,29,51,67,80,62;
```

```

%      18,22,37,56,88,109,101,77;
%      24,35,55,64,81,104,111,92;
%      49,64,78,87,103,121,128,101;
%      72,92,95,98,112,100,103,98];

%保存数据
    blockn=zeros(1,2);
    %
    selectorder=[2,3,9,10,17]; %选择的dct系数所在的位置
    dorder=[-5,-4,-3,-2,-1,0,1,2,3,4,5]; %范围
    dualorder=[2,3,4,9,10,11,17,18,25]; %双正交图 特征选择
%需要跟据尺寸大小改变的存储dct系数和挑选dct系数的矩阵 这里用的512*512尺寸 故64*64块
    selectHj1dct=zeros(1,64*64); %每块里选择一个系数进行统计保存矩阵 块数
    selectHj2dct=zeros(1,64*64);
    alldct=zeros(1,512*512); %总体dct保存矩阵 块数*64
    finallrdct=zeros(2,512*512);
    finallcdct=zeros(2,512*512);

%保存特征的矩阵
    featureall=zeros(1,193);
    featureHj1=zeros(1,11);
    featureHj2=zeros(1,11);
    %blockn 存储块顺序 为nb-1

% disp('---提取图像---');
    for transform=1:2
        if transform==1 %此处循环两次 做出 H1和 H2
            x=double(x);
            %      im=(x-128);
            JPEGOBJ = jpeg_read(impath);
            dct = JPEGOBJ.coef_arrays{1};
            DctinsteadIm=dct; %保存DCT矩阵输出
            wh=size(dct); %亮度图像的尺寸 1为高 2为宽
        else
            origianwh=wh; %裁剪保存后读取 获得J2
            im=x(5:wh(1)-4,5:wh(2)-4);
            %图像保存需要uint8格式 计算需要double格式 需要转换
            im=uint8(im);
            if exist('tempcutim.jpg','file')
                delete('tempcutim.jpg');
            end
            imwrite(im,'tempcutim.jpg','Quality',quality);
            JPEGOBJ2 = jpeg_read('tempcutim.jpg'); %再次读取
            dct = JPEGOBJ2.coef_arrays{1};
            DctinsteadImJ2=dct;
            wh=size(dct);
        end
    end

%      disp('----开始分块----');
        blockn(transform)=-1;
        blocknum(1)=fix(wh(1)/8)-1; %取整计算分块数目
        blocknum(2)=fix(wh(2)/8)-1;
        for j=0:blocknum(1)
            for i=0:blocknum(2)

                blockn(transform)=blockn(transform)+1; %确定存储坐标
                block=dct(j*8+1:(j+1)*8,i*8+1:(i+1)*8); %截取相应大小
                DCTblock=block'; %转置一下 要不然的话是先行后行的
                alldct(blockn(transform)*64+1:(blockn(transform)+1)*64)=DCTblock(:);
            end
        end
    end
end

```

```

        end
    end

    finalldct(transform,1:(blockn(transform)+1)*64)=alldct(1:(blockn(transform)+1)*64); %所有系
    blockn(transform)=-1;
    % disp(size(finallldct));
    % disp(blockn);
    for i=0:blocknum(2)
        for j=0:blocknum(1)
            blockn(transform)=blockn(transform)+1;
            block=dct(j*8+1:(j+1)*8,i*8+1:(i+1)*8); %截取相应大小
            DCTblock=block'; %转置一下 要不然的话是先行后行的
            % disp(DCTblock');
            alldct(blockn(transform)*64+1:(blockn(transform)+1)*64)=DCTblock(:);
        end
    end
    finalldct(transform,1:(blockn(transform)+1)*64)=alldct(1:(blockn(transform)+1)*64); %所有系
end

% disp(blockn(1));
% disp(blockn(2));
%dct系数处理 范围[-5, +5]
finalldct(finallldct<-5)=-5;
finalldct(finallldct>5)=5;
finalldct(finallldct>5)=5;
finalldct(finallldct<-5)=-5;
% disp(finallldct);
% disp(finallcdct);
% disp(blockn);
% disp('----全局dct系数直方图----');
% 特征1 全局dct系数直方图 11维度
% Hs(J1)-Hs(J2), s ∈ {-5, . . . , 5}, 选择范围在-5到5的元素作为特征
% 将两个图像的对比
for k=1:11
    featureHj1(k)=length(find(finallldct(1,:)==dorder(k)));
    featureHj2(k)=length(find(finallldct(2,1:64*(blockn(2)+1))==dorder(k)));
end
% disp(featureHj1);
% disp(featureHj2);
featureall(1:11)=featureHj1-featureHj2;

% disp('----局部dct系数直方图----');
%特征2 dct系数直方图 11*5=55特征
%Hjjs(J1)-Hjjs(J2), s ∈ {-5, . . . , 5},(1,j)=(1, 2), (2, 1), (3, 1), (2, 2), (1, 3)
for secfeaturenum=1:5
    featureHj1= featureHj1*0;
    featureHj2= featureHj2*0;
    selectHj1dct=selectHj1dct*0;
    selectHj2dct=selectHj2dct*0;
    for n=1:blockn(1)+1
        selectHj1dct(n)=finalldct(1,(n-1)*64+selectorder(secfeaturenum));
    end
    for n=1:blockn(2)+1
        selectHj2dct(n)=finalldct(2,(n-1)*64+selectorder(secfeaturenum));
    end

    for k=1:11
        featureHj1(k)=length(find(selectHj1dct(1:blockn(1)+1)==dorder(k)));
        featureHj2(k)=length(find(selectHj2dct(1:blockn(2)+1)==dorder(k)));
    end
end
% disp(featureHj1);

```

```

% disp(featureHj2);
featureall(secfeaturenum*11+1:11+secfeaturenum*11)= featureHj1-featureHj2;
end

% disp('----双柱状图----');
%特征3 双柱状图的d*s矩阵 9*11=99维度
%g^d, d ∈ {-5, . . . ,+5},
%gij^d 所有块i, j位置与d相等为1 否为0的累加和
% (i, j) ∈ {(2, 1), (3, 1), (4, 1), (1, 2), (2, 2), (3, 2), (1, 3), (2, 3), (1, 4)}
featureHj1= featureHj1*0;
featureHj2= featureHj2*0;
for q=1:9 %9个位置
    for k=1:11 %11个特征
        for p=0:blockn(1)
            if finallrdct(1,p*64+dualorder(q))==dorder(k)
                featureHj1(k)=featureHj1(k)+1;
            end
        end
        for p=0:blockn(2)
            if finallrdct(2,p*64+dualorder(q))==dorder(k)
                featureHj2(k)=featureHj2(k)+1;
            end
        end
    end
end
% disp(featureHj1);
% disp(featureHj2);
% disp(featureHj1-featureHj2);
featureall((5+q)*11+1:(6+q)*11)= featureHj1-featureHj2;
featureHj1= featureHj1*0; %清0
featureHj2= featureHj2*0;
end

% disp(douhism);
% disp(featureall);
% disp(douhism);

% disp('----块间相关性----');
%特征4 块间相关性
%temp=zeros(1,64);
alladd=0;
for s=0:blockn-1
    temp=abs(finallrdct(1,(s+1)*64+1:(s+2)*64)-finallrdct(1,s*64+1:(s+1)*64))+abs(finallrdct(1,(s+
    alladd=alladd+sum(temp);
    %disp(temp);
    %disp('-----');
    % disp(tempdec);
    % disp(tempcdec);
    % disp(alldct);
end
%disp(alladd/(2*(blockn+1)));
featured=alladd/(2*(blockn(1)+1));
featureall(166)=featured;

% disp('----图像计算相关性----');
%特征5 图像计算相关性 a=1, 2
% 1

```

```

rb=(origianwh(1)-1)/8;
cb=(origianwh(2)-1)/8;
sumgrayd=0;
for rbi=1:rb
    sumgrayd=sumgrayd+sum(abs(x(rbi*8,:)-x(rbi*8+1,:)));
end

for cbi=1:cb
    sumgrayd=sumgrayd+sum(abs(x(:,cbi*8)-x(:,cbi*8+1)));
end
% disp(sumgrayd);
sumgrayd=sumgrayd/(rb*origianwh(2)+cb*origianwh(1));
featureall(167)=sumgrayd;

%      % 2
sumgrayd=0;
for rbi=1:rb
    sumgrayd=sumgrayd+sum((abs(x(rbi*8,:)-x(rbi*8+1,:)).^2);
end

for cbi=1:cb
    sumgrayd=sumgrayd+sum((abs(x(:,cbi*8)-x(:,cbi*8+1)).^2);
end
% disp(sumgrayd);
sumgrayd=sumgrayd/(rb*origianwh(2)+cb*origianwh(1));
featureall(168)=sumgrayd;

%      for rbi=1:rb
%          for rbj=1:wh(2)
%              te1=im(rbi*8,rbj);
%              te2=im(rbi*8+1,rbj);
%              sumgrayd=sumgrayd+(te1-te2)*(te1-te2);
%          end
%      end
%      for cbi=1:cb
%          for cbj=1:wh(1)
%              te1=im(cbj,cbi*8);
%              te2=im(cbj,cbi*8+1);
%              sumgrayd=sumgrayd+(te1-te2)*(te1-te2);
%          end
%      end
%      sumgrayd=sumgrayd/(rb*wh(2)+cb*wh(1));
%      featureall(168)=sumgrayd;

%特征56 Cst 相关性特征
asd=0;
sumadd=0;
sumaddJ=0;
%两个系数的组合
for s=-2:2
    for t=-2:2
        asd=asd+1; %循环次数记录 25次 5*5=25
        %具体系数的计算关系
        for choose=1:64
            for kuai=0:blockn(1)-1
                if finalrdct(1,kuai*64+choose)==s&&finalrdct(1,(kuai+1)*64+choose)==t
                    sumadd=sumadd+1;
                end
                if finalcdct(1,kuai*64+choose)==s&&finalcdct(1,(kuai+1)*64+choose)==t
                    sumadd=sumadd+1;
                end
            end
        end
    end
end

```

```

        end
    end
end
sumadd=sumadd/((blockn(1)+1)*2);
for choose=1:64
    for kuai=0:blockn(2)-1
        if finalrdct(2,kuai*64+choose)==s&&finalrdct(2,(kuai+1)*64+choose)==t
            sumaddJ=sumaddJ+1;
        end
        if finalcdct(2,kuai*64+choose)==s&&finalcdct(2,(kuai+1)*64+choose)==t
            sumaddJ=sumaddJ+1;
        end
    end

    end
end
sumaddJ=sumaddJ/((blockn(2)+1)*2);
featureall(168+asd)=sumadd-sumaddJ;
% disp(sumadd-sumaddJ);
sumadd=0;
sumaddJ=0;
end
end
%返回结果
y=featureall;
end

```

【1】FRIDRICH J. Feature-Based steganalysis for JPEG images and its implications for future design of steganographic schemes[C].International Conference on Information Hiding, 2004:67-81.



[创作打卡挑战赛](#) >

[赢取流量/现金/CSDN周边激励大奖](#)