ECE 271A Homework 3 Q4 Report

Meng-Chi Tsai

4-a

In this part, we construct the predictive distribution $P_{x|T}(x \mid D_i)$ for each subset D_i and use it to classify every 8×8 block in the cheetah image. We assume that the feature vector x of each block follows a multivariate Gaussian distribution with known covariance Σ and unknown mean μ. A Gaussian prior is placed on the mean:

$$p(\mu) = \mathcal{N}(\mu_0, \Sigma_0), \qquad \Sigma_0 = \alpha \cdot \mathrm{diag}(W_0),$$

where α controls the strength of the prior.

Given the training samples $D_i = \{x_1, \ldots, x_N\}$, the posterior of μ is also Gaussian:

$$p(\mu \mid D_i) = \mathcal{N}(\mu_1, \Sigma_1),$$

with

$$\Sigma_1 = (\Sigma_0^{-1} + N\Sigma^{-1})^{-1}, \qquad \mu_1 = \Sigma_1(\Sigma_0^{-1}\mu_0 + N\Sigma^{-1}\bar{x}).$$

Integrating out μ, the predictive distribution becomes

$$P_{x|T}(x \mid D_i) = \mathcal{N}(x; \mu_1, \Sigma + \Sigma_1).$$

Each block is then classified using

$$g_{\mathrm{FG}}(x) = \log P_{x|T}(x \mid D_i, \mathrm{FG}) + \log P_Y(\mathrm{FG}), \qquad g_{\mathrm{BG}}(x) = \log P_{x|T}(x \mid D_i, \mathrm{BG}) + \log P_Y(\mathrm{BG}),$$

and the predicted segmentation is compared with the ground-truth mask to compute the error rate P_e for each value of α. For D_1 under Strategy 1, the predictive error is lowest when α is small and increases as α becomes larger. This is because when the dataset is very small, a stronger prior yields a more stable posterior mean, whereas a larger α reduces the influence of the prior and leads to worse performance.

4-b

We compute the probability of error for the ML classifier on D_1, using the same Gaussian model and decision rule as in the previous homework. The ML curve in Fig. (1) is a horizontal line, because the ML estimates of μ and Σ do not depend on α. Therefore P_e^{ML} is constant for all values of α. Comparing with the predictive results in (a), we see that for small α the predictive classifier achieves a lower error than the ML solution. This happens because with very limited data in D_1, the ML estimates of the class means are noisy, while the Bayesian predictive model can use the prior to regularize these estimates. As α grows, the predictive error approaches the ML error.

4(c)

The MAP classifier replaces the ML mean with the posterior mean μ_1, but keeps the ML covariance Σ. Therefore its behavior lies between the predictive classifier and the ML classifier. From the curve in Fig. (1), the MAP error decreases when α is small, since the posterior mean is pulled toward the prior and becomes more stable than the noisy ML estimate obtained from D_1. However, the MAP error remains higher than the predictive error because the predictive classifier accounts for the additional uncertainty through Σ_1. As α increases, the influence of the prior declines and the posterior mean approaches the ML mean. Thus, the MAP error increases and eventually becomes nearly identical to the ML error.

4(d)

We repeat the procedure for $D_1, \ldots, D_4$ under strategy 1. As the dataset size increases, the overall probability of error decreases. Moreover, the three curves become much closer and less sensitive to α. For D_2, the predictive and MAP errors rise quickly as α increases, and eventually become slightly higher than the ML error. For D_3 and D_4, the predictive, MAP, and ML curves almost coincide for all α, indicating that the influence of the prior becomes negligible in larger datasets. This shows that as more training data are available, the posterior is dominated by the data, and all three classifiers converge in performance.

4(e)

The two strategies only differ in the prior parameters, so the effect appears mainly through the Bayesian methods. For D_1, Strategy 1 achieves lower error when α is very small, but its error increases as α grows. In contrast, Strategy 2 performs poorly for small α, but improves as α increases and eventually achieves the lowest error. The predictive one performs the best among them, which aligns with Strategy 1's result. A similar pattern can be observed in D_2, but the differences between Strategy 1 and Strategy 2 are smaller. As for D_3, D_4, the curves under the two strategies become almost identical, and the error rates are very close for all values of α, indicating that the influence of the prior diminishes as the data size increases. Overall, the prior choice can significantly affect performance when the dataset is very small, but its impact drops rapidly as the sample size increases.

Listing 1: Code Section

```
load('./hw3Data/TrainingSamplesDCT_subsets_8.mat');
load('./hw3Data/Prior_1.mat');
load('./hw3Data/Alpha.mat');

Prior1.W0     = W0;
Prior1.mu0_BG = mu0_BG;
Prior1.mu0_FG = mu0_FG;

load('./hw3Data/Prior_2.mat');
Prior2.W0     = W0;
Prior2.mu0_BG = mu0_BG;
Prior2.mu0_FG = mu0_FG;

% Zig-zag pattern
zigzag = load('Zig-Zag␣Pattern.txt');
zigzag = zigzag + 1;

% Read image & ground truth
img  = im2double(imread('cheetah.bmp'));
gt   = im2double(imread('cheetah_mask.bmp'));

fprintf('img␣size␣=␣%d␣  ␣%d\n', size(img,1), size(img,2));
fprintf('mask␣size␣=␣%d␣  ␣%d\n', size(gt,1), size(gt,2));

% Zero padding and DCT (use block pipeline)
left   = zeros(size(img,1), 4);
right  = zeros(size(img,1), 3);
up     = zeros(4,  size(img,2) + 4 + 3);
bottom = zeros(3,  size(img,2) + 4 + 3);
img_pad = [up; [left img right]; bottom];

img_dct = dct_8(img, img_pad);

img_scan = blockproc(img_dct, [8 8], @(b) ZigZagScan(b.data, zigzag));

D_BG_all   = [D1_BG; D2_BG; D3_BG; D4_BG];
D_BG_index = [0, size(D1_BG,1), size(D2_BG,1), size(D3_BG,1), size(D4_BG,1)];

D_FG_all   = [D1_FG; D2_FG; D3_FG; D4_FG];
D_FG_index = [0, size(D1_FG,1), size(D2_FG,1), size(D3_FG,1), size(D4_FG,1)];

numSets = 4;
K       = length(alpha);
```

```
% p_pred(s, d, k): strategy s (1 or 2), dataset d (1..4), alpha index k
p_pred = zeros(2, 4, K);   % Predictive (full Bayesian)
p_map  = zeros(2, 4, K);   % MAP
p_ml   = zeros(2, 4, K);   % ML

for s = 1:2
    if s == 1
        prior = Prior1;
    else
        prior = Prior2;
    end

    for d = 1:4
        s_bg = 1;
        s_fg = 1;
        for i = 1:d
            s_bg = s_bg + D_BG_index(i);
            s_fg = s_fg + D_FG_index(i);
        end
        e_bg = s_bg + D_BG_index(d+1) - 1;
        e_fg = s_fg + D_FG_index(d+1) - 1;

        D_BG = D_BG_all(s_bg:e_bg, :);
        D_FG = D_FG_all(s_fg:e_fg, :);

        % Class priors (ML)
        Nbg = size(D_BG,1);
        Nfg = size(D_FG,1);
        P_bg = Nbg / (Nbg + Nfg);
        P_fg = 1 - P_bg;

        % ML mean & covariance for each class
        [mu_BG_ML, Sigma_BG] = ml_gaussian(D_BG);
        [mu_FG_ML, Sigma_FG] = ml_gaussian(D_FG);

        % ML baseline
        mask_ml = blockproc(img_scan, [1 64], ...
                  @(b) BDR(b.data, mu_BG_ML, mu_FG_ML, ...
                           Sigma_BG, Sigma_FG, P_bg, P_fg));
        p_ml(s, d, :) = P_Error(gt, mask_ml, P_bg, P_fg);

        % Do predictive & MAP for all alpha
        for k = 1:K
            a = alpha(k);
            % Posterior & predictive params for BG
            [mu_BG_post, Sigma_BG_post, mu_BG_pred, Sigma_BG_pred] = ...
                posterior_predictive(prior.mu0_BG, prior.W0, a, D_BG, Sigma_BG);
            % Posterior & predictive params for FG
            [mu_FG_post, Sigma_FG_post, mu_FG_pred, Sigma_FG_pred] = ...
                posterior_predictive(prior.mu0_FG, prior.W0, a, D_FG, Sigma_FG);
            % Predictive classifier: N(mu_pred, Sigma + Sigma1)
            mask_pred = blockproc(img_scan, [1 64], ...
                        @(b) BDR(b.data, mu_BG_pred, mu_FG_pred, ...
                                 Sigma_BG_pred, Sigma_FG_pred, P_bg, P_fg));
            p_pred(s, d, k) = P_Error(gt, mask_pred, P_bg, P_fg);
            % MAP classifier: N(mu_post, Sigma_ML)
            mask_map = blockproc(img_scan, [1 64], ...
                        @(b) BDR(b.data, mu_BG_post, mu_FG_post, ...
                                 Sigma_BG, Sigma_FG, P_bg, P_fg));
            p_map(s, d, k) = P_Error(gt, mask_map, P_bg, P_fg);
        end
        figure;
        semilogx(alpha, reshape(p_pred(s,d,:), [1 K]), '-o'); hold on;
        semilogx(alpha, reshape(p_map(s,d,:),  [1 K]), '-s');
        semilogx(alpha, reshape(p_ml(s,d,:),   [1 K]), '--');
        xlabel('\alpha'); ylabel('P_e');
        legend('Predictive', 'MAP', 'ML', 'Location', 'best');
        title(sprintf('Dataset␣D_%d,␣Strategy␣%d', d, s));
        saveas(gcf, sprintf('Pe_D%d_strategy%d.png', d, s));
    end
end
```

```
function [mu, Sigma] = ml_gaussian(X)
    mu = mean(X, 1);
    Xm = X - mu;
    N  = size(X,1);
    Sigma = (Xm' * Xm) / N;
end

function [mu1, Sigma1, mu_pred, Sigma_pred] = posterior_predictive(mu0, W0, alpha, X, Sigma)
    [N, ~] = size(X);
    xbar = mean(X, 1);
    % Prior covariance
    Sigma0 = alpha * diag(W0);
    % A =  0  +   /N
    A    = Sigma0 + Sigma / N;
    Ainv = inv(A);
    % Posterior mean  1
    mu1 = ( Sigma0 * Ainv * xbar' + Sigma * Ainv * (mu0') / N )';
    % Posterior covariance  1
    Sigma1 = Sigma0 * Ainv * (Sigma / N);
    % Predictive parameters
    mu_pred    = mu1;
    Sigma_pred = Sigma + Sigma1;
end

function vector = ZigZagScan(matrix, pattern)
    vector = zeros(1, numel(matrix));
    for i = 1:size(matrix,1)
        for j = 1:size(matrix,2)
            pos = pattern(i,j);
            vector(1,pos) = matrix(i,j);
        end
    end
end

function mask = BDR(feature, mu_bg, mu_fg, Sigma_bg, Sigma_fg, P_bg, P_fg)
    term_bg = (feature - mu_bg) / Sigma_bg * (feature - mu_bg)' ...
            + log((2*pi)^64 * det(Sigma_bg)) - 2*log(P_bg);
    term_fg = (feature - mu_fg) / Sigma_fg * (feature - mu_fg)' ...
            + log((2*pi)^64 * det(Sigma_fg)) - 2*log(P_fg);
    if term_bg < term_fg
        mask = 0;
    else
        mask = 1;
    end
end

function dct_img = dct_8(img, img_pad)
    [H, W] = size(img);
    dct_img = zeros(H*8, W*8);
    for i = 1:H
        for j = 1:W
            dct_img((8*i-7):(8*i), (8*j-7):(8*j)) = dct2(img_pad(i:i+7, j:j+7));
        end
    end
end

function p = P_Error(gt, mask, prob_bg, prob_fg)
    gt   = int8(gt > 0);
    mask = int8(mask);

    diff = gt - mask;
    detect = 1 - sum(diff(:)==1) / sum(gt(:)==1);
    fAlarm = sum(diff(:)==-1) / sum(gt(:)==0);
    p = fAlarm * prob_bg + (1 - detect) * prob_fg;
end
```

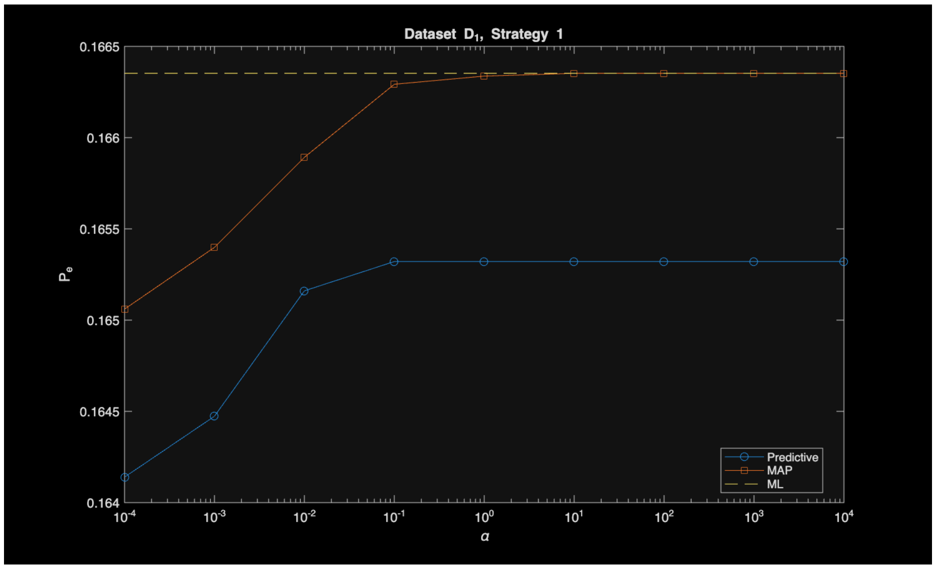

Figure 1: Histogram of DCT coefficients for background (cheetah).

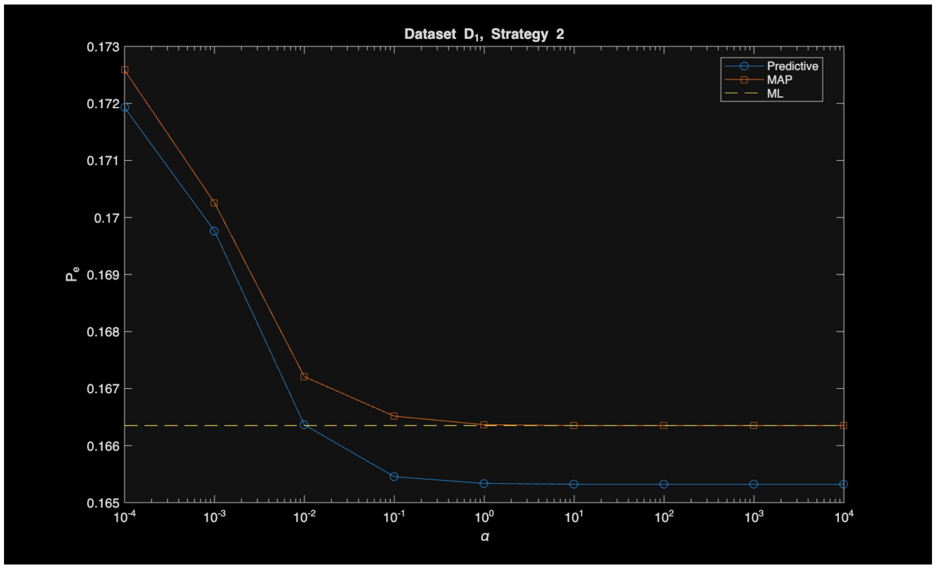

Figure 2: Histogram of DCT coefficients for background (cheetah).

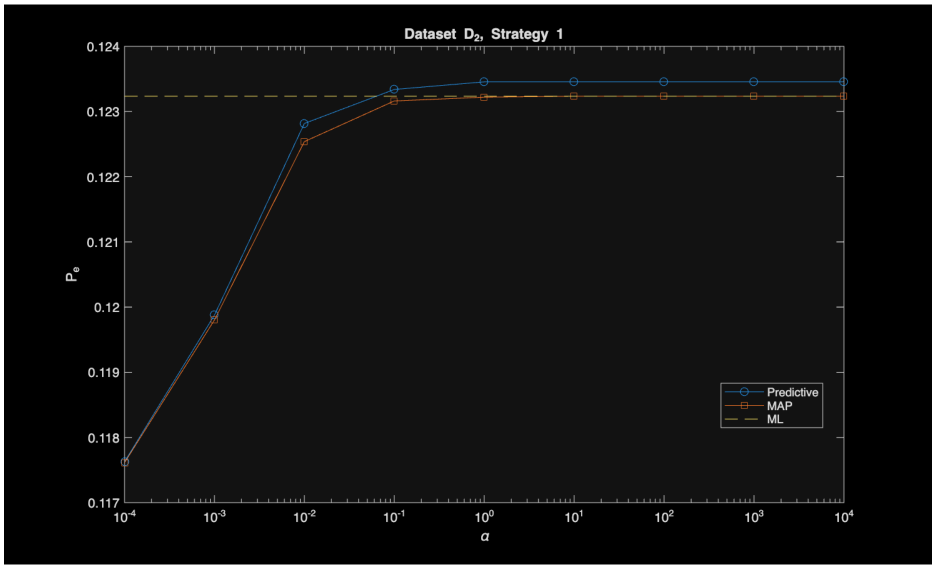

Figure 3: Histogram of DCT coefficients for background (cheetah).

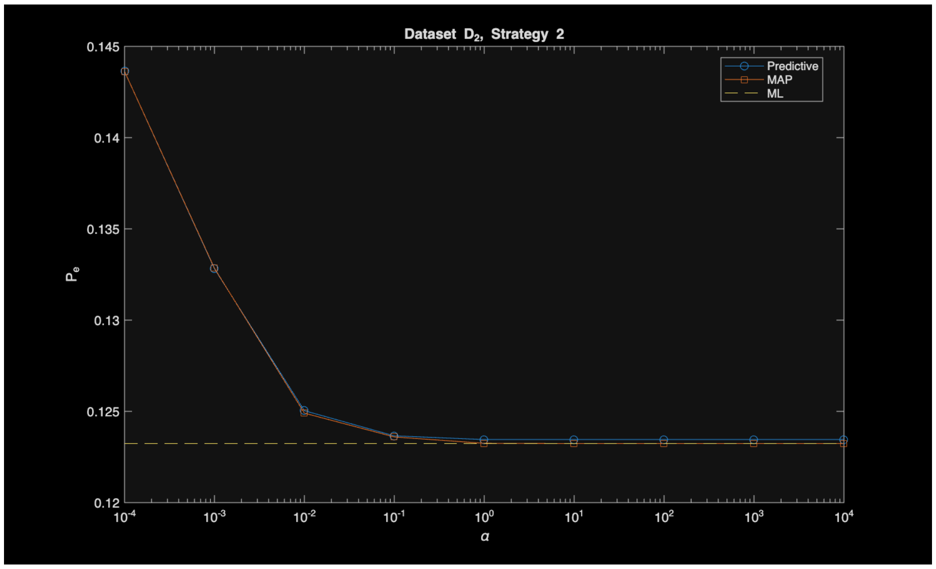

Figure 4: Histogram of DCT coefficients for background (cheetah).

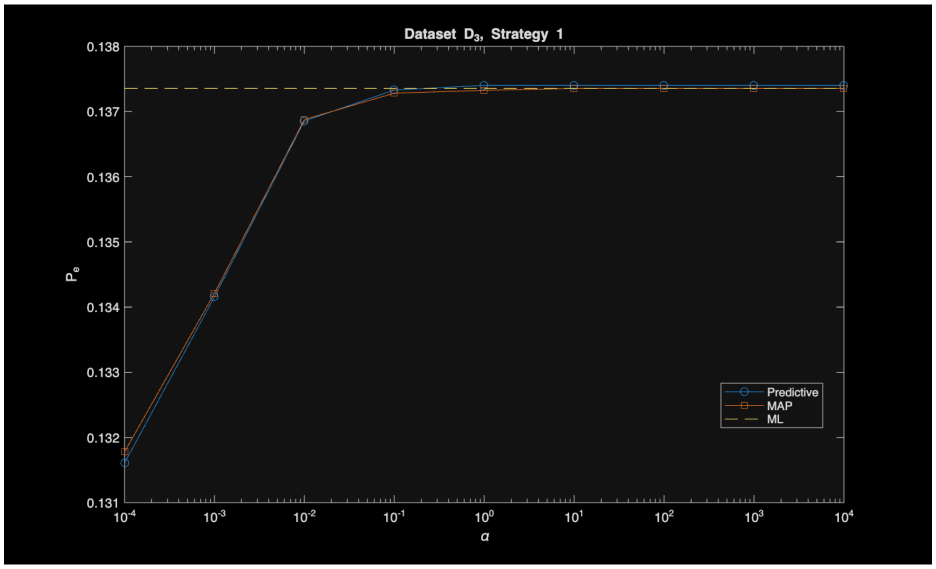

Figure 5: Histogram of DCT coefficients for background (cheetah).

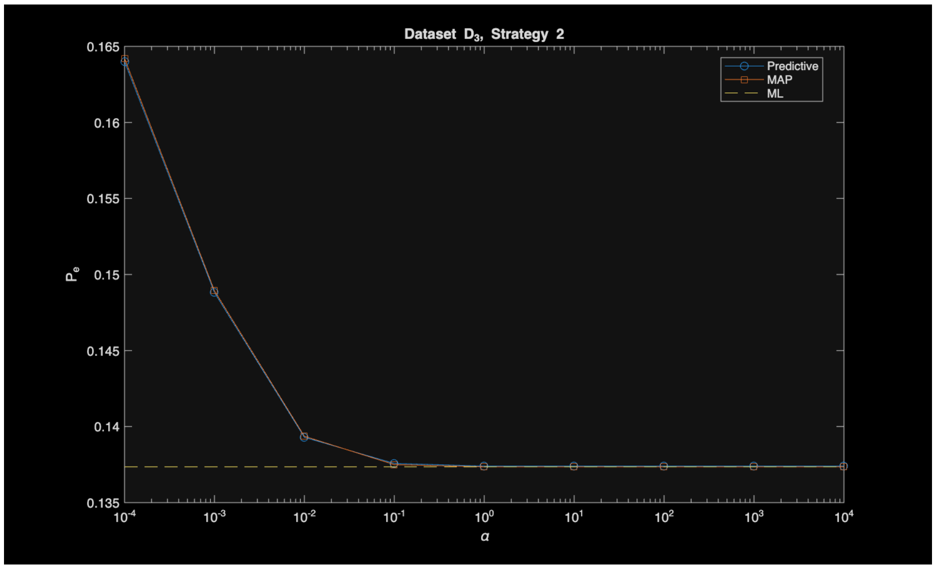

Figure 6: Histogram of DCT coefficients for background (cheetah).

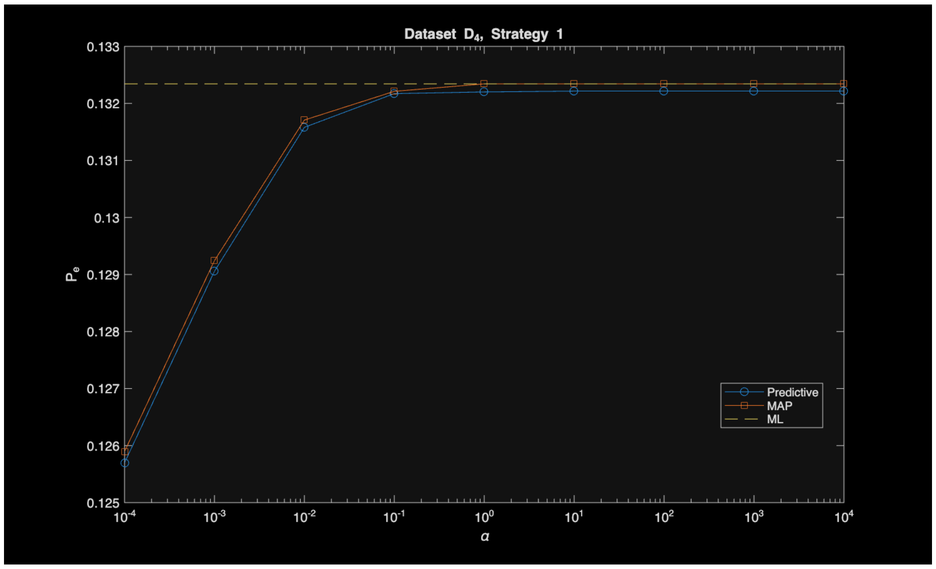

Figure 7: Histogram of DCT coefficients for background (cheetah).

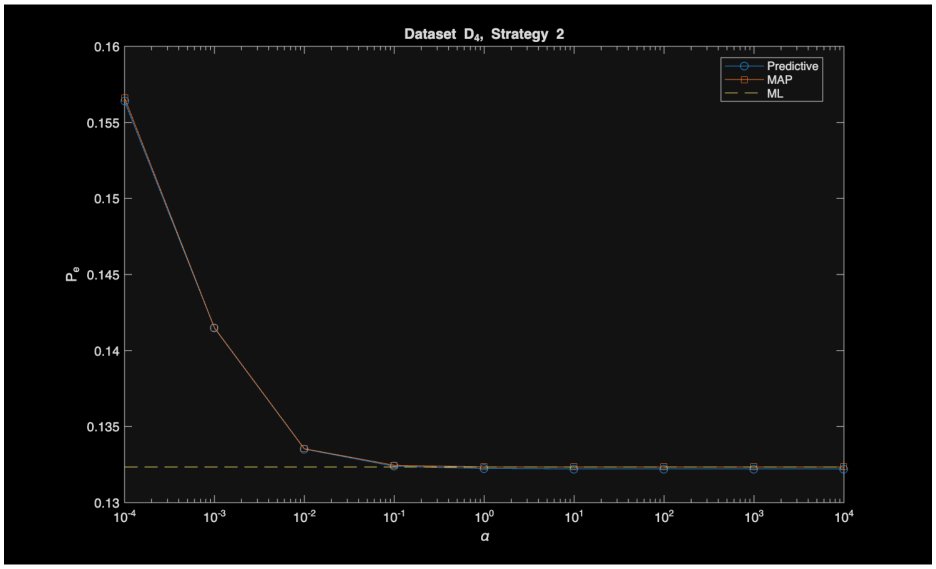

Figure 8: Histogram of DCT coefficients for background (cheetah).