Boosting weak classifier performance

1. Application of Adaboost

This section shows how Adaboost was used to produce Figure 14.11 in Davies (2017) Chapter 14. In fact, a complete Matlab implementation of Adaboost (written by Dirk-Jan Kroon, 2010) appears on his website:

https://github.com/timmy00274672/Adaboost.

The relevant example appears as the pair of Matlab scripts:

- adaboost.m
- example.m

Note that redistribution of the software comes with strict copyright conditions, which mean that it is not permissible to reproduce the full listings here.

The first of the Matlab scripts is the Adaboost optimisation routine, and the second is the example calling routine: it uses 50 weak classifiers with horizontal and vertical lines and many possible position offsets.

For the work described in Davies Chapter 14, it proved adequate to use Kroon's adaboost.m script and to radically adapt the example.m calling script in several ways: these can be interpreted as revamping the graphics presentation, with the aim of highlighting the dataset point weighting that Adaboost uses during training. It was also desired (1) to show all weak classifiers by doubly infinite straight lines, (2) to show the resulting class 1 region in green, and the class –1 region in yellow, (3) to superimpose the class 1 test dataset points in red and the class –1 dataset points in blue. In Davies (2017) Figure 14.11, 1–30 weak classifiers were used. However, in the code shown below the number was restricted to 5.

For copyright reasons, neither Kroon's original code nor the complete code we used to produce Figure 14.11 can be shown here, as they contain a substantial proportion of Kroon's original code. However, the key items indicated above are well illustrated by the sections of code reproduced below. It is left as an exercise for the student to patch these sections back into Kroon's original code and so to regenerate our figures (which include the effects of 1–30 weak classifiers).

Details of our new calling routine are:

1. A new data initialisation section.
2. A slightly revised training results section, in which the following numbers are changed: 80 → 160, 81 → 161, 160 → 320, 161 → 321 [defining lim = 80 and working from there, all these numbers are systematically derived].
3. An additional output is obtained from the 'train' mode of Adaboost, which is simply the existing Dweights parameter D in adaboost.m.
4. The remaining sections of example.m (from 'subplot(2,2,2)' onwards) are eliminated and replaced with our totally redesigned output section.

The overall purpose of the new calling routine is to obtain complete clarity in the presentation of the Adaboost data, including particularly the part played by sample reweighting. Figure 1 summarises the situation.
% Adaboost calling routine (Roy Davies, (c) 2018)
% script adapted by Roy Davies from an original by Dirk-Jan Kroon (2010)

K=5; % no. of weak classifiers - typically 1-50

% generate 2D data from normal distributions
% use same no. of points for class 1 (red) and class -1 (blue)
rng('default'); % use the default random no. initialisation
angle = rand(500,1)*2*pi;
r = mvnrnd(0,0.2,500)+1.7;
blue = [sin(angle).*r cos(angle).*r].*2;
angle = rand(500,1)*2*pi;
r = mvnrnd(0,0.1,500)+0.5;
red = [sin(angle).*r cos(angle).*r].*2;

% training data
datafeatures=[blue;red];
dataclass(1:500)=-1;
dataclass(500+1:500*2)=1;
origblue=blue;
origred=red;

% train Adaboost
[classestimate,model,Dweights]= ... 
adaboost('train',datafeatures,dataclass,K);
% in adaboost itself, Dweights translates to parameter D

% training results
... % (c) Kroon (2010)
% as per Kroon, except for change in the lim parameter from 80 to 160

% generate a high resolution grid
resolution = 1000;
a = 6.7;
range = linspace(-a,a,resolution);
[X,Y] = meshgrid(range,range);
xx = X(:); yy = Y(:);
n = size(X,1);
testdata = [ones(1,resolution*resolution)'];
testdata = [xx'; yy']';

% classify the testdata with the trained model
testclass=adaboost('apply',testdata,model);
xx=reshape(xx,n,n);
yy=reshape(yy,n,n);
zz=reshape(testclass,n,n);

% show the classified regions as a green and yellow colourmap
figure;
pcolor(xx,yy,zz);
shading interp;
ymap=[1,1,0.8 % yellow
0.6,1,0.6]; % green
colormap(ymap);
hold on; axis square;
set(gca,'YTick',[],'XTick',[]); % eliminates labels along axes
hold on;
... % '...' denotes continuation
% draw classifier boundary lines
for i=1:length(model)
    x1=-a; y1=-a; x2=a; y2=a;
    if(model(i).dimension==1)
        x1=model(i).threshold; x2=model(i).threshold;
    end
    if(model(i).dimension==2)
        y1=model(i).threshold; y2=model(i).threshold;
    end
    line([x1 x2],[y1 y2],'color','k','linewidth',0.2); hold on
end

% prepare to display weighted points with appropriate sizes
Dweightsb(1:500)=Dweights(1:500);
Dweightsr(1:500)=Dweights(500+1:500*2);
msizeb=int16(Dweightsb*500*16);
msizer=int16(Dweightsr*500*16);
msizeb=msizeb';
msizer=msizer';
for i=1:500
    plot(origblue(i,1),origblue(i,2),'.b','MarkerSize',msizeb(i));
    plot(origred(i,1),origred(i,2),'.r','MarkerSize',msizer(i));
end

% save output figure with 300dpi resolution
print(gcf,'-dtiff','-r300','adaboost-K5');

Figure 1. Output obtained using Adaboost with 5 weak classifiers. (a) shows the full data space; (b) shows the central portion with 2× magnification to demonstrate clearly the effect of weight change on the dataset.
2. Application of Logitboost

This section shows how Logitboost was used to produce Figure 14.15 in Davies (2017) Chapter 14. In fact, a complete Matlab implementation of Logitboost (written by Stefan Stavrev, 2013) appears on the website of Prince's book *Computer Vision* (2012), http://www.computervisionmodels.com. The relevant example appears as the pair of Matlab scripts:

- fit_logitiboost.m
- fit_logitiboost_ex2.m

The first of these is the Logitboost optimisation routine, and the second is the calling routine for an example of the type given in Prince, Chapter 9, Figure 9.16(f): it uses 10 weak classifiers with 20 angles and 40 position offsets.

For the work described in Davies Chapter 14, it proved adequate to use Stavrev's fit_logitiboost.m script and to suitably adapt the fit_logitiboost_ex2.m calling script in several ways: these can be interpreted as revamping the graphics presentation. In particular, it was desired (1) to show all weak classifiers by doubly infinite straight lines, (2) to show the resulting class 1 region in green, and the class –1 region in yellow, (3) to superimpose the class 1 test dataset points in red and the class –1 dataset points in blue. 10 weak classifiers were used together with 40 angles and 40 position offsets.

For copyright reasons, neither Stavrev's original code nor the complete code we used to produce Figure 14.15 can be shown here, as they contain a substantial proportion of Stavrev's original code. However, items 1–3 above are well illustrated by the sections of code reproduced below. It is left as an exercise for the student to patch these sections back into Stavrev's original code and so to regenerate our figures (which include the effects of 1, 2, 3, 4 and 10 weak classifiers).

It has been noticed that small differences appear in the output of this code when used under Matlab R2015a and R2018a: these are presumed to be due to variations in the default initialisation options for these versions of Matlab. Therefore, to maintain compatibility with the figures and other data in Davies 2017, it is advised to use R2015a where possible. In addition, more seriously, the observed variations include not only differences in output but also certain failure modes being initiated, e.g., with *fminunc* and other routines. In particular, the basic offset was needed to eliminate some *fminunc* problems.
% Logitboost calling routine (Roy Davies, (c) 2018)
% script adapted by Roy Davies from an original by Stefan Stavrev (2013)

K=6; % no. of weak classifiers

% generate a high resolution grid
resolution = 1000;
a = 6.7;
range = linspace(-a,a,resolution);
[X,Y] = meshgrid(range,range);
x = X(:); y = Y(:);
n = size(X,1);

% generate 2D data from normal distributions
% use same no. of points for class 1 (red) and class -1 (blue)
rng('default');
angle = rand(500, 1)*2*pi;
r = mvnrnd(0,0.2,500)+1.7;
blue = [sin(angle).*r cos(angle).*r].*2;
angle = rand(500, 1)*2*pi;
r = mvnrnd(0,0.1,500)+0.5;
red = [sin(angle).*r cos(angle).*r].*2;
X_data = [blue; red];

% prepare the training input
% ... % (c) Stavrev (2010)

% construct the weak classifier parameters
offsetdiv=20; anglediv=40; noffsets=40; nangles=40; basicoffset=0.1;
% (original Stavrev values: 40, 20, 40, 20, 0)
% ... % (c) Stavrev (2010)

% fit logitboost
[predictions,cc] = fit_logitboost(X_train, w, X_test, Alpha, K);
% note that 'K' is now being provided by the calling function
% note that additional output 'cc' is being obtained from logitboost

angle_delta = 2*pi/anglediv;
column = 1;
offset_delta = 1/offsetdiv;
angle = 0;
optoffset=zeros(K,1);
angl=zeros(K,1);
mm=1;
for i = 1:nangles
    x = cos(angle); y = sin(angle);
    offset = basicoffset;
    for j = 1:noffsets
        offset = offset + offset_delta;
        for k=1:K
            if mm==cc(k) % obtain optimum values
                optoffset(k)=offset;
                optangle(k)=angle;
            end
        end
    end
    mm=mm+1;
end
angle = angle + angle_delta;
```matlab
% plot the results
figure;
Z = reshape(predictions,n,n);
pcolor(X,Y,Z);
shading interp;
ymap=[1,1,0.8]% yellow
0.6,1,0.6];% green
colormap(ymap);
set(gca,'YTick',[],'XTick',[]);
hold on;

% draw doubly infinite lines for weak classifiers
for kk=1:K
    if abs(sin(optangle(kk)))> abs(cos(optangle(kk)))
        x = -a:(a+1);
        y=-(optoffset(kk) + x*cos(optangle(kk)))/sin(optangle(kk));
    else
        y = -a:(a+1);
        x=-(optoffset(kk) + y*sin(optangle(kk)))/cos(optangle(kk));
    end
    plot(x,y,'k'); hold on;
end;

% superimpose data points
hold on;
selector = 1:500;
scatter(X_data(selector,1), X_data(selector,2), 10, 'fill',...
'MarkerFaceColor','b','LineWidth',1);
hold on; axis square;
selector = 500+1:2*500;
scatter(X_data(selector,1), X_data(selector,2), 10, 'fill',...
'MarkerFaceColor','r','LineWidth',1);
hold off;
saveas(gcf,'logitboost-K6.tif');
```

Figure 2. Output obtained using Logitboost with 6 weak classifiers. An identical dataset is used with Adaboost (Figure 1) and Logitboost.