Shirui Ye

PS5 Programming Report

Source code:

Part A

%read in data and preprocess

source=imread('Bayesnoise\_textbook.png');

%extract

extract\_s=source(:,:,1);

extract\_s=int8(extract\_s);

%get greyscale for source

[r,c]=size(extract\_s);

for i=1:r

for j=1:c

if extract\_s(i,j)<119

extract\_s(i,j)=-1;

else

extract\_s(i,j)=1;

end

end

end

gc=extract\_s;

or=gc;

s=size(gc);

n=7;

yd=s(1);

xd=s(2);

h=-0.01;

fp=1;

c=0;

b=5;

while (fp)

c=c+1;

fp=0;

for i=2:xd-1

for j=2:yd-1

fpe=(-gc(j,i))\*(h-(b\*(gc(j,i+1)+gc(j,i-1)+gc(j+1,i)+gc(j-1,i)))-(n\*gc(j,i)));

nfpe=gc(j,i)\*(h-(b\*(gc(j,i+1)+gc(j,i-1)+gc(j+1,i)+gc(j-1,i)))-(n\*gc(j,i)));

if nfpe>fpe

gc(j,i)=-gc(j,i);

fp=1;

end

end

end

end

%correction read in and process

correction=imread('Bayes\_textbook.png');

corr\_coe=int8(correction(:,:,1));

%get greyscale for correction

[r,c]=size(corr\_coe);

for i=1:r

for j=1:c

if corr\_coe(i,j)<119

corr\_coe(i,j)=-1;

else

corr\_coe(i,j)=1;

end

end

end

corr\_b=corr\_coe;

[r,c]=size(corr\_b);

sum=r\*c;

comparison=0;

for i=1:r

for j=1:c

if corr\_b(i,j)==gc(i,j)

comparison=comparison+1;

end

end

end

%report recovery rate

recovery=(comparison/sum)\*100;

fprintf('The recovery is %.4f \n', recovery)

%get image

imshow(uint8(gc)\*255);

figure();

imshow(uint8(or)\*255);

Part B

%read in data and preprocess

source=imread('Lenanoise.png');

source=int16(source);

src=source;

s=size(source);

yd=s(1);

xd=s(2);

form=@(x,N)(mod(x-1,N)+1);

d\_lam=1;

lam\_s=1;

check=true;

procedure=1;

while(check)

check=false;

for i=1:xd

for j=1:yd

%1st case

minu=(-d\_lam\*abs(max(0,source(j,i)-procedure)-src(j,i)))-(lam\_s\*(abs(max(0,source(j,i)-procedure)-source(form(j-1,yd),i))+abs(max(0,source(j,i)-procedure)-source(j,form(i+1,xd)))+abs(max(0,source(j,i)-procedure)-source(j,form(i-1,xd)))+abs(max(0,source(j,i)-procedure)-source(form(j+1,yd),i))));

%2nd case

plus=(-d\_lam\*abs(min(255,source(j,i)+procedure)-src(j,i)))-(lam\_s\*(abs(min(255,source(j,i)+procedure)-source(form(j-1,yd),i))+abs(min(255,source(j,i)+procedure)-source(j,form(i+1,xd)))+abs(min(255,source(j,i)+procedure)-source(form(j+1,yd),i))+abs(min(255,source(j,i)+procedure)-source(j,form(i-1,xd)))));

%3rd case

same=(-d\_lam\*abs(source(j,i)-src(j,i)))-(lam\_s\*(abs(source(j,i)-source(form(j-1,yd),i))+abs(source(j,i)-source(j,form(i+1,xd)))+abs(source(j,i)-source(form(j+1,yd),i))+abs(source(j,i)-source(j,form(i-1,xd)))));

%variable and compare

xi=source(j,i);

if plus>same

source(j,i)=min(255,xi+procedure);

check=true;

end

if same<minu

source(j,i)=max(0,xi-procedure);

check=true;

end

end

end

end

recover=imread('Lena.png');

%get image

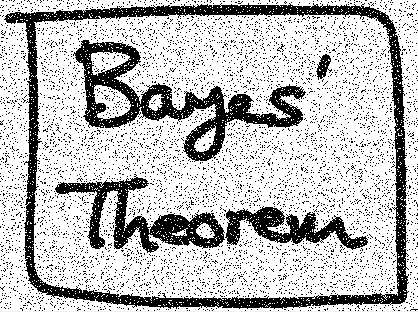
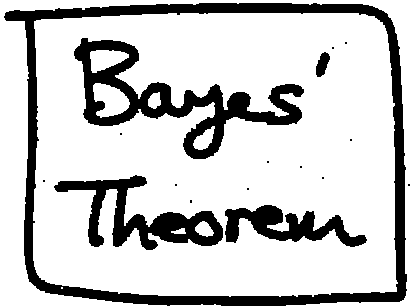
imshow(uint8(source));

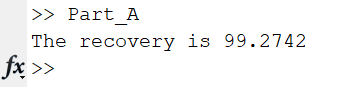
figure();

imshow(uint8(src));

Report:

Part A - the optimum values I have for h, β, η are 0.01, 5 and 7. The accuracy I get with these values is 99.2742%.



Please run my part A to see the exact image outcomes and accuracy above.

The clean image is gotten from the noisy image. The image cannot be recovered exactly, but we have a pretty good result. Markov Random Frields are used.

Noise yi is in {-1,1} orginal xi is in {-1,1}

I write the Energy function:

E(x,y)=h

and correction in my Part A so that they can be used directly in the file. Then I implement Coordinate-descent algorithm.

{xi} (xi=yi)

For xi if -x 🡪E(x,y) decreases x=-x

I started with values 0.03, 15, 8. The accuracy started from around 94, then I adjust these values step by step and finally get to 99.2742% accuracy.

Part B – We still cannot recover Lena exactly. This part is harder than recovering the image in Part A.

Graph Model:

P(X|Y, lamda(d), lamda(s))=

X=output clean, Y=input noise, 2 lambdas are weights

L1 norm:P(z)=|z|, L2norm P(z)=|z|2

Max-sum alg🡪MAP🡪X

Argmaxlamda(d), lamda(s)p(X|Y,lamda(d),lamda(s))

Please see images below:



You can run my code to see the results above. The model is the extension of what we talked about.

p(X|Y, ,λd,λd)=

X is to restore, Y is noisy Lena. Max sum is used to get MAP solution. I started from small values from 32 to 256. Then get multiple restore results. Then I chose the best case to report. Run my code to see the result.