Task 3-1 & 2

Devosmita Chatterjee

910812-7748

Figure 1 shows the classification error of the training set and of the validation set as a function of epoch for the four networks. C_Train1 & C_Valid1, C_Train2 & C_Valid2, C_Train3 & C_Valid3 and C_Train4 & C_Valid4 denote the classification errors of the training set and the validation set for the four networks, respectively.

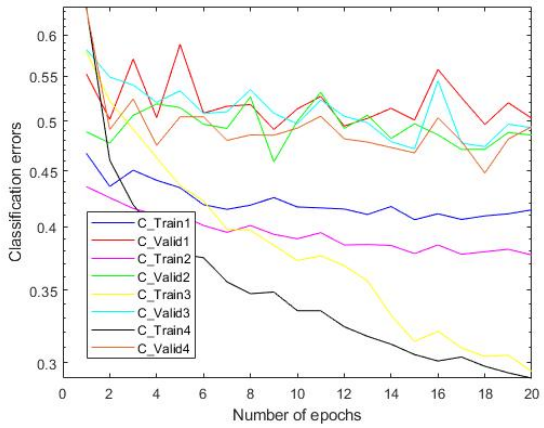

Figure 1: This figure shows the classification errors of the training set and the validation set for the four networks for 20 epochs.

Table 1: The table presents the lowest classification errors on the validation set (C_Valid) and the classification errors on the training set (C_Train) and the test set (C_Test) obtained for Networks 1-4 at the epoch (Epoch_No) where the lowest classification error on the validation set was obtained.

Network	C_Train	C_Valid	C_Test	No_Epochs
Network 1	0.4254	0.4912	0.4910	9
Network 2	0.3936	0.4588	0.4577	9
Network 3	0.3139	0.4715	0.4711	15
Network 4	0.2978	0.4481	0.4520	18

1 Results and Discussion

The results and discussion are the following:

1. From table 1, we find that the classification errors on the training sets are low and the classification errors on the validation set and the test set are higher for all networks.
2. The classification errors on all datasets for network 2 and network 3 are lower than that for network 1. The reason behind this is the inclusion of one hidden layer.
3. The classification errors on all datasets for network 4 are the least because of the inclusion of two hidden layers.

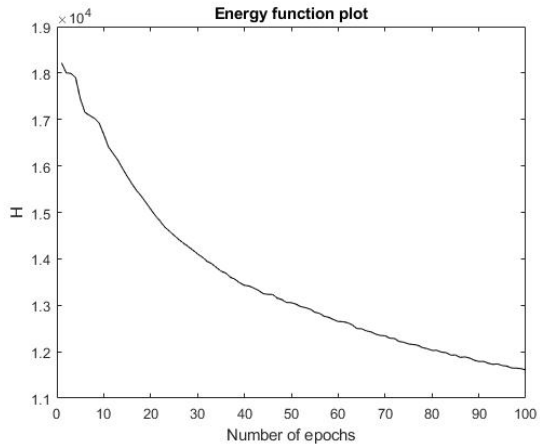

Figure 1: The plot shows H evaluated for all training data as a function of epoch where H is the energy function.

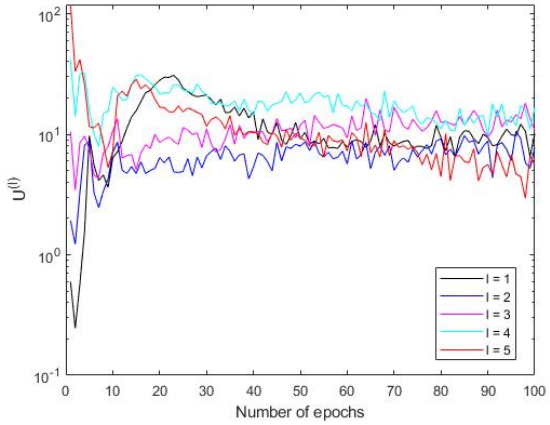

Figure 2: The plot shows $U^{(l)}$ as a function of epoch for l=1 to l=5 where $U^{(l)}$ is the learning speed of layer l.

1 Results and Discussion

The results and discussion are the following:

1. The learning speed shows a lognormal distribution with time, and it is different in the different layers.
2. In the initial phase of training that is, phase I, we observe the learning slowdown due to severity of the vanishing gradient problem.

Appendix

Fully connected networks 2019

Fullyconnectednetworks2019.m

```
clc; clear;
%%
% Name: Devosmita Chatterjee
% Assignment 3.1
[xTrain, tTrain, xValid, tValid, xTest, tTest] = LoadCIFAR(1);
xTrain = xTrain-mean(xTrain,2);
xValid = xValid-mean(xTrain,2);
xTest = xTest-mean(xTrain,2);

[C_Train1,C_Valid1,C_Test1,lastepoch1] =
network1(xTrain,tTrain,xValid,tValid,xTest,tTest);
[C_Train2,C_Valid2,C_Test2,lastepoch2] =
network2(xTrain,tTrain,xValid,tValid,xTest,tTest);
[C_Train3,C_Valid3,C_Test3,lastepoch3] =
network3(xTrain,tTrain,xValid,tValid,xTest,tTest);
[C_Train4,C_Valid4,C_Test4,lastepoch4] =
network4(xTrain,tTrain,xValid,tValid,xTest,tTest);
%%
disp(lastepoch1)
disp(C_Train1(lastepoch1))
disp(C_Valid1(lastepoch1))
disp(C_Test1)
%9
%0.4254
%0.4912
%0.4910
%%
disp(lastepoch2)
disp(C_Train2(lastepoch2))
disp(C_Valid2(lastepoch2))
disp(C_Test2)
%9
%0.3936
%0.4588
%0.4577
%%
disp(lastepoch3)
disp(C_Train3(lastepoch3))
disp(C_Valid3(lastepoch3))
disp(C_Test3)
%15
%0.3139
%0.4715
%0.4711
%%
disp(lastepoch4)
disp(C_Train4(lastepoch4))
disp(C_Valid4(lastepoch4))
disp(C_Test4)
%18
%0.2978
%0.4481
```

```
%0.4520
%%
x = 1:20;
y1 = C_Train1;
y2 = C_Valid1;
y3 = C_Train2;
y4 = C_Valid2;
y5 = C_Train3;
y6 = C_Valid3;
y7 = C_Train4;
y8 = C_Valid4;

plot(x,y1,'b')
hold on
plot(x,y2,'r')
hold on
plot(x,y3,'m')
hold on
plot(x,y4,'g')
hold on
plot(x,y5,'y')
hold on
plot(x,y6,'c')
hold on
plot(x,y7,'k')
hold on
plot(x,y8,'color',[0.9100    0.4100    0.1700])

set(gca, 'YScale', 'log')
xlabel('Number of epochs')
ylabel('Classification errors')

legend('C\_Train1','C\_Valid1','C\_Train2','C\_Valid2','C\_Train3','C
\_Valid3','C\_Train4','C\_Valid4','Location', 'Best')
```

network1.m

```
function [C_Train,C_Valid,C_Test,lastepoch1] =
network1(xTrain,tTrain,xValid,tValid,xTest,tTest)

epochs = 20;
eta = 0.1;
mB = 100;
p = size(xTrain,2);
batches = p/mB;

w = normrnd(0,1/sqrt(3072),size(tTrain,1),size(xTrain,1));
theta = zeros(size(tTrain,1),1);

w1=zeros(size(tTrain,1),size(xTrain,1),epochs);
theta1=zeros(size(tTrain,1),1,epochs);

C_Train = zeros(1,epochs);
C_Valid = zeros(1,epochs);

for t = 1:epochs
    t

    p = 40000;
    rng(55)
    tmp = randperm(length(xTrain));
    xTrain =  xTrain(:,tmp);
    tTrain =  tTrain(:,tmp);

    for nbr = 1:batches
        nbr

        tempw=zeros(size(tTrain,1),size(xTrain,1));
        tempt=zeros(size(tTrain,1),1);
        del = 0;
        err = 0;
        for mu=(nbr-1)*mB+1:nbr*mB
            batch_xTrain = xTrain(:,mu);
            batch_tTrain = tTrain(:,mu);
            V0 = batch_xTrain;
            b = w*V0-theta;
            V1 =1./(1+exp(-b));
            error=(batch_tTrain-V1).*V1.*(1-V1);
            err = error+err;
            delta = error*V0';
            del = delta +del;
        end

        tempw=eta*del;
        tempt=-eta*err;
        w=w+tempw;
        theta=theta+tempt;
    end

    %training set
    C_Terr = 0;
    for mu = 1:size(xTrain,2)
        V0_Train = xTrain(:,mu);
```

```
        b_Train = w*V0_Train-theta;
        V1_Train = 1./(1+exp(-b_Train));
        out_Train = zeros(size(tTrain,1),1);
        index_Train = find(V1_Train == max(V1_Train));
        out_Train(index_Train) = 1;
        C_Terr = (1/(2*size(xTrain,2)))*norm(tTrain(:,mu)-out_Train)
+ C_Terr;
    end

    %Validation set
    C_Verr = 0;
    for mu = 1:size(xValid,2)
        V0_Valid = xValid(:,mu);
        b_Valid = w*V0_Valid-theta;
        V1_Valid = (1+exp(-b_Valid)).^(-1);
        out_Valid = zeros(size(tValid,1),1);
        index_Valid = find(V1_Valid == max(V1_Valid));
        out_Valid(index_Valid) = 1;
        C_Verr = (1/(2*size(xValid,2)))*norm(tValid(:,mu)-out_Valid)
+ C_Verr;

    end
    %calculate classification errors
    C_Train(t) = C_Terr;
    C_Valid(t) = C_Verr;
    w1(:,:,t) = w;
    theta1(:,:,t) = theta;
end

index_min = find(C_Valid == min(C_Valid));
weight = w1(:,:,index_min(end));
threshold = theta1(:,:,index_min(end));
lastepoch1 = index_min(end);

%Test set
C_err = 0;
for mu = 1:size(xTest,2)
    V0_Test = xTest(:,mu);
    b_Test = weight*V0_Test-threshold;
    V1_Test = 1./(1+exp(-b_Test));
    out_Test = zeros(size(tTest,1),1);
    index_Test = find(V1_Test == max(V1_Test));
    out_Test(index_Test) = 1;
    C_err = (1/(2*size(xTest,2)))*norm(tTest(:,mu)-out_Test) + C_err;
end
%calculate classification errors
C_Test = C_err;

end
```

network2.m

```
function [C_Train,C_Valid,C_Test,lastepoch2] =
network2(xTrain,tTrain,xValid,tValid,xTest,tTest)

epochs = 20;
eta = 0.1;
mB = 100;
p = size(xTrain,2);
batches = p/mB;

w1 = normrnd(0,1/sqrt(3072),10,size(xTrain,1));
theta1 = zeros(10,1);
w2 = normrnd(0,1/sqrt(10),size(tTrain,1),10);
theta2 = zeros(size(tTrain,1),1);

save_w1=zeros(10,size(xTrain,1),epochs);
save_theta1=zeros(10,1,epochs);
save_w2=zeros(size(tTrain,1),10,epochs);
save_theta2=zeros(size(tTrain,1),1,epochs);

C_Train = zeros(1,epochs);
C_Valid = zeros(1,epochs);

%%

for t = 1:epochs
    t

    p = 40000;
    rng(55)
    tmp = randperm(length(xTrain));
    xTrain =  xTrain(:,tmp);
    tTrain =  tTrain(:,tmp);


    for nbr = 1:batches
        nbr

        tempw1=zeros(10,size(xTrain,1));
        tempt1=zeros(10,1);

        tempw2=zeros(size(tTrain,1),10);
        tempt2=zeros(size(tTrain,1),1);

        del1 = 0;
        err1 = 0;
        del2 = 0;
        err2 = 0;
        for mu=(nbr-1)*mB+1:nbr*mB
            batch_xTrain = xTrain(:,mu);
            batch_tTrain = tTrain(:,mu);
            V0 = batch_xTrain;
            b1 = w1*V0-theta1;
            V1 =1./(1+exp(-b1));
            b2 = w2*V1-theta2;
            V2 =1./(1+exp(-b2));
```

```
            error2=(batch_tTrain-V2).*V2.*(1-V2);
            error1=w2'*error2.*V1.*(1-V1);

            err1 = error1+err1;
            err2 = error2+err2;

            delta2 = error2*V1';
            del2 = delta2 +del2;
            delta1 = error1*V0';
            del1 = delta1 +del1;

        end

        tempw1=eta*del1;
        tempt1=-eta*err1;
        w1=w1+tempw1;
        theta1=theta1+tempt1;

        tempw2=eta*del2;
        tempt2=-eta*err2;
        w2=w2+tempw2;
        theta2=theta2+tempt2;
    end




    %training set

    C_Terr = 0;
    for mu = 1:size(xTrain,2)
        V0_Train = xTrain(:,mu);
        b1_Train = w1*V0_Train-theta1;
        V1_Train =1./(1+exp(-b1_Train));
        b2_Train = w2*V1_Train-theta2;
        V2_Train =1./(1+exp(-b2_Train));
        out_Train = zeros(size(tTrain,1),1);
        index_Train = find(V2_Train == max(V2_Train));
        out_Train(index_Train) = 1;
        C_Terr = (1/(2*size(xTrain,2)))*norm(tTrain(:,mu)-out_Train)
+ C_Terr;
    end

    %Validation set
    C_Verr = 0;
    for mu = 1:size(xValid,2)
        V0_Valid = xValid(:,mu);
        b1_Valid = w1*V0_Valid-theta1;
        V1_Valid =1./(1+exp(-b1_Valid));
        b2_Valid = w2*V1_Valid-theta2;
        V2_Valid =1./(1+exp(-b2_Valid));
        out_Valid = zeros(size(tValid,1),1);
        index_Valid = find(V2_Valid == max(V2_Valid));
        out_Valid(index_Valid) = 1;
```

```
        C_Verr = (1/(2*size(xValid,2)))*norm(tValid(:,mu)-out_Valid)
+ C_Verr;

    end
    %calculate classification errors
    C_Train(t) = C_Terr;
    C_Valid(t) = C_Verr;
    save_w1(:,:,t) = w1;
    save_theta1(:,:,t) = theta1;
    save_w2(:,:,t) = w2;
    save_theta2(:,:,t) = theta2;
end

index_min = find(C_Valid == min(C_Valid));
weight1 = save_w1(:,:,index_min(end));
threshold1 = save_theta1(:,:,index_min(end));
weight2 = save_w2(:,:,index_min(end));
threshold2 = save_theta2(:,:,index_min(end));
lastepoch2 = index_min(end);
%%
%Test set
C_err = 0;
for mu = 1:size(xTest,2)
    V0_Test = xTest(:,mu);
    b1_Test = weight1*V0_Test-threshold1;
    V1_Test =1./(1+exp(-b1_Test));
    b2_Test = weight2*V1_Test-threshold2;
    V2_Test =1./(1+exp(-b2_Test));
    out_Test = zeros(size(tTest,1),1);
    index_Test = find(V2_Test == max(V2_Test));
    out_Test(index_Test) = 1;
    C_err = (1/(2*size(xTest,2)))*norm(tTest(:,mu)-out_Test) + C_err;

end
%calculate classification errors
C_Test = C_err;

end
```

network3.m

```
function [C_Train,C_Valid,C_Test,lastepoch3] =
network3(xTrain,tTrain,xValid,tValid,xTest,tTest)

epochs = 20;
eta = 0.1;
mB = 100;
p = size(xTrain,2);
batches = p/mB;

w1 = normrnd(0,1/sqrt(3072),50,size(xTrain,1));
theta1 = zeros(50,1);
w2 = normrnd(0,1/sqrt(50),size(tTrain,1),50);
theta2 = zeros(size(tTrain,1),1);

save_w1=zeros(50,size(xTrain,1),epochs);
save_theta1=zeros(50,1,epochs);
save_w2=zeros(size(tTrain,1),50,epochs);
save_theta2=zeros(size(tTrain,1),1,epochs);

C_Train = zeros(1,epochs);
C_Valid = zeros(1,epochs);

%%

for t = 1:epochs
    t

    p = 40000;
    rng(55)
    tmp = randperm(length(xTrain));
    xTrain =  xTrain(:,tmp);
    tTrain =  tTrain(:,tmp);


    for nbr = 1:batches
        nbr

        tempw1=zeros(50,size(xTrain,1));
        tempt1=zeros(50,1);

        tempw2=zeros(size(tTrain,1),50);
        tempt2=zeros(size(tTrain,1),1);

        del1 = 0;
        err1 = 0;
        del2 = 0;
        err2 = 0;
        for mu=(nbr-1)*mB+1:nbr*mB
            batch_xTrain = xTrain(:,mu);
            batch_tTrain = tTrain(:,mu);
            V0 = batch_xTrain;
            b1 = w1*V0-theta1;
            V1 =1./(1+exp(-b1));
            b2 = w2*V1-theta2;
            V2 =1./(1+exp(-b2));
```

```
            error2=(batch_tTrain-V2).*V2.*(1-V2);
            error1=w2'*error2.*V1.*(1-V1);

            err1 = error1+err1;
            err2 = error2+err2;

            delta2 = error2*V1';
            del2 = delta2 +del2;
            delta1 = error1*V0';
            del1 = delta1 +del1;

        end

        tempw1=eta*del1;
        tempt1=-eta*err1;
        w1=w1+tempw1;
        theta1=theta1+tempt1;

        tempw2=eta*del2;
        tempt2=-eta*err2;
        w2=w2+tempw2;
        theta2=theta2+tempt2;
    end




    %training set

    C_Terr = 0;
    for mu = 1:size(xTrain,2)
        V0_Train = xTrain(:,mu);
        b1_Train = w1*V0_Train-theta1;
        V1_Train =1./(1+exp(-b1_Train));
        b2_Train = w2*V1_Train-theta2;
        V2_Train =1./(1+exp(-b2_Train));
        out_Train = zeros(size(tTrain,1),1);
        index_Train = find(V2_Train == max(V2_Train));
        out_Train(index_Train) = 1;
        C_Terr = (1/(2*size(xTrain,2)))*norm(tTrain(:,mu)-out_Train)
+ C_Terr;
    end

    %Validation set
    C_Verr = 0;
    for mu = 1:size(xValid,2)
        V0_Valid = xValid(:,mu);
        b1_Valid = w1*V0_Valid-theta1;
        V1_Valid =1./(1+exp(-b1_Valid));
        b2_Valid = w2*V1_Valid-theta2;
        V2_Valid =1./(1+exp(-b2_Valid));
        out_Valid = zeros(size(tValid,1),1);
        index_Valid = find(V2_Valid == max(V2_Valid));
        out_Valid(index_Valid) = 1;
```

```
        C_Verr = (1/(2*size(xValid,2)))*norm(tValid(:,mu)-out_Valid)
+ C_Verr;

    end
    %calculate classification errors
    C_Train(t) = C_Terr;
    C_Valid(t) = C_Verr;
    save_w1(:,:,t) = w1;
    save_theta1(:,:,t) = theta1;
    save_w2(:,:,t) = w2;
    save_theta2(:,:,t) = theta2;
end

index_min = find(C_Valid == min(C_Valid));
weight1 = save_w1(:,:,index_min(end));
threshold1 = save_theta1(:,:,index_min(end));
weight2 = save_w2(:,:,index_min(end));
threshold2 = save_theta2(:,:,index_min(end));
lastepoch3 = index_min(end);
%%
%Test set
C_err = 0;
for mu = 1:size(xTest,2)
    V0_Test = xTest(:,mu);
    b1_Test = weight1*V0_Test-threshold1;
    V1_Test =1./(1+exp(-b1_Test));
    b2_Test = weight2*V1_Test-threshold2;
    V2_Test =1./(1+exp(-b2_Test));
    out_Test = zeros(size(tTest,1),1);
    index_Test = find(V2_Test == max(V2_Test));
    out_Test(index_Test) = 1;
    C_err = (1/(2*size(xTest,2)))*norm(tTest(:,mu)-out_Test) + C_err;

end
%calculate classification errors
C_Test = C_err;

end
```

network4.m

```
function [C_Train,C_Valid,C_Test,lastepoch4] =
network4(xTrain,tTrain,xValid,tValid,xTest,tTest)

epochs = 20;
eta = 0.1;
mB = 100;
p = size(xTrain,2);
batches = p/mB;

w1 = normrnd(0,1/sqrt(3072),50,size(xTrain,1));
theta1 = zeros(50,1);
w2 = normrnd(0,1/sqrt(50),50,50);
theta2 = zeros(50,1);
w3 = normrnd(0,1/sqrt(50),50,size(tTrain,1));
theta3 = zeros(size(tTrain,1),1);

save_w1=zeros(50,size(xTrain,1),epochs);
save_theta1=zeros(50,1,epochs);
save_w2=zeros(50,50,epochs);
save_theta2=zeros(50,1,epochs);
save_w3=zeros(50,size(tTrain,1),epochs);
save_theta3=zeros(size(tTrain,1),1,epochs);

C_Train = zeros(1,epochs);
C_Valid = zeros(1,epochs);

%%

for t = 1:epochs
    t

    p = 40000;
    rng(55)
    tmp = randperm(length(xTrain));
    xTrain =  xTrain(:,tmp);
    tTrain =  tTrain(:,tmp);


    for nbr = 1:batches
        nbr

        tempw1=zeros(50,size(xTrain,1));
        tempt1=zeros(50,1);

        tempw2=zeros(50,50);
        tempt2=zeros(50,1);

        tempw3=zeros(50,size(tTrain,1));
        tempt3=zeros(size(tTrain,1),1);

        del1 = 0;
        err1 = 0;
        del2 = 0;
        err2 = 0;
        del3 = 0;
```

```
    err3 = 0;
    for mu=(nbr-1)*mB+1:nbr*mB
        batch_xTrain = xTrain(:,mu);
        batch_tTrain = tTrain(:,mu);
        V0 = batch_xTrain;
        b1 = w1*V0-theta1;
        V1 =1./(1+exp(-b1));
        b2 = w2*V1-theta2;
        V2 =1./(1+exp(-b2));
        b3 = w3'*V2-theta3;
        V3 =1./(1+exp(-b3));

        error3=(batch_tTrain-V3).*V3.*(1-V3);
        error2=w3*error3.*V2.*(1-V2);
        error1=w2'*error2.*V1.*(1-V1);

        err1 = error1+err1;
        err2 = error2+err2;
        err3 = error3+err3;

        delta3 = error3*V2';
        del3 = delta3 +del3;
        delta2 = error2*V1';
        del2 = delta2 +del2;
        delta1 = error1*V0';
        del1 = delta1 +del1;

    end

    tempw1=eta*del1;
    tempt1=-eta*err1;
    w1=w1+tempw1;
    theta1=theta1+tempt1;

    tempw2=eta*del2;
    tempt2=-eta*err2;
    w2=w2+tempw2;
    theta2=theta2+tempt2;

    tempw3=eta*del3';
    tempt3=-eta*err3;
    w3=w3+tempw3;
    theta3=theta3+tempt3;
end




%training set

C_Terr = 0;
for mu = 1:size(xTrain,2)
    V0_Train = xTrain(:,mu);
    b1_Train = w1*V0_Train-theta1;
    V1_Train =1./(1+exp(-b1_Train));
    b2_Train = w2*V1_Train-theta2;
    V2_Train =1./(1+exp(-b2_Train));
    b3_Train = w3'*V2_Train-theta3;
```

```
        V3_Train =1./(1+exp(-b3_Train));
        out_Train = zeros(size(tTrain,1),1);
        index_Train = find(V3_Train == max(V3_Train));
        out_Train(index_Train) = 1;
        C_Terr = (1/(2*size(xTrain,2)))*norm(tTrain(:,mu)-out_Train)
+ C_Terr;
    end

    %Validation set
    C_Verr = 0;
    for mu = 1:size(xValid,2)
        V0_Valid = xValid(:,mu);
        b1_Valid = w1*V0_Valid-theta1;
        V1_Valid =1./(1+exp(-b1_Valid));
        b2_Valid = w2*V1_Valid-theta2;
        V2_Valid =1./(1+exp(-b2_Valid));
        b3_Valid = w3'*V2_Valid-theta3;
        V3_Valid =1./(1+exp(-b3_Valid));
        out_Valid = zeros(size(tValid,1),1);
        index_Valid = find(V3_Valid == max(V3_Valid));
        out_Valid(index_Valid) = 1;
        C_Verr = (1/(2*size(xValid,2)))*norm(tValid(:,mu)-out_Valid)
+ C_Verr;

    end
    %calculate classification errors
    C_Train(t) = C_Terr;
    C_Valid(t) = C_Verr;
    save_w1(:,:,t) = w1;
    save_theta1(:,:,t) = theta1;
    save_w2(:,:,t) = w2;
    save_theta2(:,:,t) = theta2;
    save_w3(:,:,t) = w3;
    save_theta3(:,:,t) = theta3;
end

index_min = find(C_Valid == min(C_Valid));
weight1 = save_w1(:,:,index_min(end));
threshold1 = save_theta1(:,:,index_min(end));
weight2 = save_w2(:,:,index_min(end));
threshold2 = save_theta2(:,:,index_min(end));
weight3 = save_w3(:,:,index_min(end));
threshold3 = save_theta3(:,:,index_min(end));
lastepoch4 = index_min(end);
%%
%Test set
C_err = 0;
for mu = 1:size(xTest,2)
    V0_Test = xTest(:,mu);
    b1_Test = weight1*V0_Test-threshold1;
    V1_Test =1./(1+exp(-b1_Test));
    b2_Test = weight2*V1_Test-threshold2;
    V2_Test =1./(1+exp(-b2_Test));
    b3_Test = weight3'*V2_Test-threshold3;
    V3_Test =1./(1+exp(-b3_Test));
    out_Test = zeros(size(tTest,1),1);
    index_Test = find(V3_Test == max(V3_Test));
    out_Test(index_Test) = 1;
    C_err = (1/(2*size(xTest,2)))*norm(tTest(:,mu)-out_Test) + C_err;
```

```
end
%calculate classification errors
C_Test = C_err;

end
```

Vanishing Gradient Problem 2019

Vanishinggradientproblem2019.m

```
clc; clear;
%%
% Name: Devosmita Chatterjee
% Assignment 3.1
[xTrain, tTrain, xValid, tValid, xTest, tTest] = LoadCIFAR(2);
xTrain = xTrain-mean(xTrain,2);

epochs = 100;
eta = 0.01;
mB = 100;
p = size(xTrain,2);
batches = p/mB;

w1 = normrnd(0,1/sqrt(3072),20,size(xTrain,1));
theta1 = zeros(20,1);
w2 = normrnd(0,1/sqrt(20),20,20);
theta2 = zeros(20,1);
w3 = normrnd(0,1/sqrt(20),20,20);
theta3 = zeros(20,1);
w4 = normrnd(0,1/sqrt(20),20,20);
theta4 = zeros(20,1);
w5 = normrnd(0,1/sqrt(20),size(tTrain,1),20);
theta5 = zeros(size(tTrain,1),1);

energy_function = zeros(1,epochs);



epoch_err1 = zeros(1,epochs);
epoch_err2 = zeros(1,epochs);
epoch_err3 = zeros(1,epochs);
epoch_err4 = zeros(1,epochs);
epoch_err5 = zeros(1,epochs);
%%

for t = 1:epochs
    t

    p = 40000;
    rng(55)
    tmp = randperm(length(xTrain));
    xTrain =  xTrain(:,tmp);
    tTrain =  tTrain(:,tmp);

    out = zeros(10,40000);

    err1_f =0;
    err2_f =0;
    err3_f =0;
    err4_f =0;
    err5_f =0;

    for nbr = 1:batches
        nbr
```

```
%tempw1=zeros(20,size(xTrain,1));
%tempt1=zeros(20,1);

%tempw2=zeros(20,20);
%tempt2=zeros(20,1);

%tempw3=zeros(20,20);
%tempt3=zeros(20,1);

%tempw4=zeros(20,20);
%tempt4=zeros(20,1);

%tempw5=zeros(20,size(tTrain,1));
%tempt5=zeros(size(tTrain,1),1);

del1 = 0;
err1 = 0;
del2 = 0;
err2 = 0;
del3 = 0;
err3 = 0;
del4 = 0;
err4 = 0;
del5 = 0;
err5 = 0;



for mu=(nbr-1)*mB+1:nbr*mB

    batch_xTrain = xTrain(:,mu);
    batch_tTrain = tTrain(:,mu);
    V0 = batch_xTrain;
    b1 = w1*V0-theta1;
    V1 =1./(1+exp(-b1));
    b2 = w2*V1-theta2;
    V2 =1./(1+exp(-b2));
    b3 = w3'*V2-theta3;
    V3 =1./(1+exp(-b3));
    b4 = w4*V3-theta4;
    V4 =1./(1+exp(-b4));
    b5 = w5*V4-theta5;
    V5 =1./(1+exp(-b5));
    for i=1:10
        out(i,mu) = V5(i);
    end

    error5=(batch_tTrain-V5).*V5.*(1-V5);
    error4=w5'*error5.*V4.*(1-V4);
    error3=w4'*error4.*V3.*(1-V3);
    error2=w3*error3.*V2.*(1-V2);
    error1=w2'*error2.*V1.*(1-V1);

    err1 = error1+err1;
    err2 = error2+err2;
    err3 = error3+err3;
    err4 = error4+err4;
```

```
        err5 = error5+err5;

        delta5 = error5*V1';
        del5 = delta5 +del5;
        delta4 = error4*V3';
        del4 = delta4 +del4;
        delta3 = error3*V2';
        del3 = delta3 +del3;
        delta2 = error2*V1';
        del2 = delta2 +del2;
        delta1 = error1*V0';
        del1 = delta1 +del1;

    end

    tempw1=eta*del1;
    tempt1=-eta*err1;
    w1=w1+tempw1;
    theta1=theta1+tempt1;

    tempw2=eta*del2;
    tempt2=-eta*err2;
    w2=w2+tempw2;
    theta2=theta2+tempt2;

    tempw3=eta*del3';
    tempt3=-eta*err3;
    w3=w3+tempw3;
    theta3=theta3+tempt3;

    tempw4=eta*del4';
    tempt4=-eta*err4;
    w4=w4+tempw4;
    theta4=theta4+tempt4;

    tempw5=eta*del5;
    tempt5=-eta*err5;
    w5=w5+tempw5;
    theta5=theta5+tempt5;

    err1_f =err1+err1_f;
    err2_f =err2+err2_f;
    err3_f =err3+err3_f;
    err4_f =err4+err4_f;
    err5_f =err5+err5_f;

end

stored=0;
for m = 1:40000
    stored = sum(abs(tTrain(:,m)-out(:,m)).^2)+stored;
end
H=stored/2;
energy_function(t) = H;

epoch_err1(t) = norm(err1_f);
epoch_err2(t) = norm(err2_f);
epoch_err3(t) = norm(err3_f);
```

```
    epoch_err4(t) = norm(err4_f);
    epoch_err5(t) = norm(err5_f);

end


%%
x = 1:100;

plot(x,energy_function,'k')

xlabel('Number of epochs')
ylabel('H')
title('Energy function plot')

%%
x = 1:100;

plot(x,epoch_err1,'k')
hold on
plot(x,epoch_err2,'b')
hold on
plot(x,epoch_err3,'m')
hold on
plot(x,epoch_err4,'c')
hold on
plot(x,epoch_err5,'r')

set(gca, 'YScale', 'log')
xlabel('Number of epochs')
ylabel('U^{(l)}')

legend('l = 1','l = 2','l = 3','l = 4','l = 5','Location','Best')
```