```
M = [1,2,4,8];
learningRate = 0.01;
probability = 1/4;
trials = 500;
numVisibleNeurons = 3;
miniBatchSize = 20;
kCD = 250;
nOut = 3000;
nIn = 2000;

givenPatterns = [1,  1, -1;
                 1, -1,  1;
                -1,  1,  1;
                -1, -1, -1;
                 1,  1,  1;
                 1, -1, -1;
                -1,  1, -1;
                -1, -1,  1];
Pd = [0.25, 0.25, 0.25, 0.25, 0, 0, 0, 0];
pB = zeros(size(givenPatterns,1),size(M,2));
Dkl = zeros(8,size(M,2));

runTimeWeights = zeros(trials,M(3),numVisibleNeurons);
runTimeHidden = zeros(trials,1,M(3));
runTimeVisible = zeros(trials,1,numVisibleNeurons);
%runtimeDeltaWeights = zeros(trials, M(mCounter),numVisibleNeurons);

for mCounter = 3:3%size(M,2)
    weights = normrnd(0,1,M(mCounter),numVisibleNeurons); % !Set diagonal to 0
    weights(1:1+size(weights,1):end) = 0;
    thetaHidden = zeros(1,M(mCounter));
    thetaVisible = zeros(1,numVisibleNeurons);
    for iTrial = 1:trials
        deltaWeight = zeros(M(mCounter),numVisibleNeurons);
        deltaThresholdVisible = zeros(1,numVisibleNeurons);
        deltaThresholdHidden = zeros(1,M(mCounter));
        for mB = 1:miniBatchSize
            h = zeros(M(mCounter),kCD);
            mu = randi(4,1,1);
            vInput = givenPatterns(mu,:);
            bHidden0 = weights*(vInput') - thetaHidden';
            for i = 1:M(mCounter)
                r = rand;
                if r < 1/(1+exp(-2*bHidden0(i)))
                    h(i,1) = 1;
                else
                    h(i,1) = -1;
                end
            end
            bVisible = zeros(numVisibleNeurons,kCD);
            v = zeros(numVisibleNeurons,kCD);
            bHidden = zeros(M(mCounter),kCD);
            for t = 2:kCD
                bVisible(:,t-1) = h(:,t-1)'*weights - thetaVisible;
                for j = 1:numVisibleNeurons
                    r = rand;
                    if r < 1/(1+exp(-2*bVisible(j,t-1)))
                        v(j,t) = 1;
                    else
                        v(j,t) = -1;
                    end
                end
                bHidden(:,t) = weights*v(:,t)-thetaHidden';
                for i = 1:M(mCounter)
                    r = rand;
                    if r < 1/(1+exp(-2*bHidden(i,t)))
                        h(i,t) = 1;
                    else
                        h(i,t) = -1;
                    end
                end
            end
            deltaWeight = deltaWeight + (tanh(bHidden0)*vInput - tanh(bHidden(:,kCD))*v↙
(:,kCD)'); %Behöver troligen kollas på row/colums
            deltaThresholdVisible = deltaThresholdVisible + (vInput-v(:,kCD)');
            deltaThresholdHidden = deltaThresholdHidden + (tanh(bHidden0) - tanh↙
(bHidden(:,kCD)))';


        end
        weights = weights + learningRate*deltaWeight; %Learning rate above according to↙
Ludvig
        thetaVisible = thetaVisible - learningRate*deltaThresholdVisible;
        thetaHidden = thetaHidden - learningRate*deltaThresholdHidden;

%         runTimeWeights(iTrial,:,:) = deltaWeight;
%         runTimeVisible(iTrial,:,:) = deltaThresholdVisible;
%         runTimeHidden(iTrial,:,:) = deltaThresholdHidden;

        runTimeWeights(iTrial,:,:) = weights;
        runTimeVisible(iTrial,:,:) = thetaVisible;
        runTimeHidden(iTrial,:,:) = thetaHidden;
        %runtimeDeltaWeights(iTrial,:,:) = deltaWeight;
    end
    %%

    tic
    for i = 1:nOut
        mu = randi([1,8],1,1);
        vInput = givenPatterns(mu,:)';
        bHidden = weights*vInput-thetaHidden';
        h = zeros(1,M(mCounter));
        v = zeros(numVisibleNeurons,1);
        for m = 1:M(mCounter)
            r = rand;
            if r < 1/(1+exp(-2*bHidden(m)))
                h(m) = 1;
            else
                h(m) = -1;
            end
        end
        for j = 1:nIn
            bVisible = h*weights - thetaVisible;
            for n = 1:numVisibleNeurons
                r = rand;
                if r < 1/(1+exp(-2*bVisible(n)))
                    v(n) = 1;
                else
                    v(n) = -1;
                end
            end
            bHidden = weights*v-thetaHidden'; %%Back an dforth
            %h = zeros(M(mCounter));
            for k = 1:M(mCounter)
                r = rand;
                if r < 1/(1+exp(-2*bHidden(m)))
                    h(m) = 1;
                else
                    h(m) = -1;
                end
            end
            for pa = 1:8
                if v == givenPatterns(pa,:)'
                    pB(pa,mCounter) = pB(pa,mCounter) + 1/(nIn*nOut);
                end
            end
        end
    end

    for mu = 1:8
        if pB(mu,mCounter) == 0
            Dkl(mu,mCounter) = Dkl(mu,mCounter) + Pd(mu).*(log(Pd(mu)));
        else
            Dkl(mu,mCounter) = Dkl(mu,mCounter) + Pd(mu).*(log(Pd(mu)-log(pB(mu,↙
mCounter))));
        end
    end
    %Sum dkl and add to another vector
    toc
end
```

```
%% PLot DKL
clf
x = 0:0.01:10;
KL = numVisibleNeurons - (log2(x+1))-(x+1)./(2.^(log2(x+1))); %Paranthesis take integer
figure(1)
plot(M,sum(Dkl,1),'o')
hold on
plot(x,KL)
xlim([0,8])
ylim([0,2])

%% Testkod
Dkl = zeros(8,size(M,2));
for mCounter = 1:4
    for mu = 1:8
        if pB(mu,mCounter) == 0
            Dkl(mu,mCounter) = Dkl(mu,mCounter) + Pd(mu).*(log(Pd(mu)));
        else
            Dkl(mu,mCounter) = Dkl(mu,mCounter) + Pd(mu).*(log(Pd(mu)-log(pB(mu,↙
mCounter))));
        end
    end
end
```