mat_drop.m

```
%%Dropout

%Setup the parameters you will use for this code
input_layer_size  = 400;  % 20x20 Input Images of Digits
hidden_layer_size = 25;   % 25 hidden units
num_labels = 10;          % 10 labels, from 1 to 10
                          % (note that we have mapped "0" to label 10)

%% ========== Loading and Visualizing Data =============


% Load Training Data
fprintf('Loading and Visualizing Data ...\n')

load('ex4data1.mat');
m = size(X, 1);

% Randomly select 100 data points to display
sel = randperm(size(X, 1));
sel = sel(1:100);

%displayData(X(sel, :));

fprintf('Program paused. Press enter to continue.\n');
%pause;




%% ================Initializing Pameters ================


fprintf('\nInitializing Neural Network Parameters ...\n')

initial_Theta1 = randInitializeWeights(input_layer_size, hidden_layer_size);
initial_Theta2 = randInitializeWeights(hidden_layer_size, num_labels);

% Unroll parameters
initial_nn_params = [initial_Theta1(:) ; initial_Theta2(:)];



%%%%%i am adding %%%%%%%%
for iteration = 1:1
increment = uint16(100);  %used for deciding size of mini-batch

increment
```

```
for xyz = 1:increment:m   %this for-loop iterates over all
fprintf('\n');
xyz
if xyz+increment>m
        X_ran = X(xyz:m, :);
        y_ran = y(xyz:m);
else
        X_ran = X(xyz:xyz+increment, :);
        y_ran = y(xyz:xyz+increment);
end

%%Randomly ommiting hidden units with probability 0.5
c = 0;
percent = 0.5;
ran_mat = rand(hidden_layer_size, 1);
for i=1:hidden_layer_size
   if ran_mat(i)>=percent
        c++;
   end
end



ini_theta1_ran = zeros(c, input_layer_size+1);
ini_theta2_ran = zeros(num_labels, c+1);

k=0;
for i=1:hidden_layer_size
   if ran_mat(i)>=percent
        k++;
        for j=1:input_layer_size+1
                ini_theta1_ran(k,j) = initial_Theta1(i, j);
        end

   end
end



for i=1:num_labels
        ini_theta2_ran(i, 1) = initial_Theta2(i, 1);
end

k=1;
for i=1:hidden_layer_size
   if ran_mat(i)>=percent
        k++;
        for j=1:num_labels
                ini_theta2_ran(j, k) = initial_Theta2(j, (i+1));
        end
   end
end
```

```
initial_nn_params_ran = [ini_theta1_ran(:) ; ini_theta2_ran(:)];


c
%%%%%%%% i have added %%%%



%% ====================  Training NN ====================

fprintf('\nTraining Neural Network... \n')

options = optimset('MaxIter', 50);


lambda = 1;

% Create "short hand" for the cost function to be minimized
costFunction = @(p) nnCostFunction(p, ...
                                   input_layer_size, ...
                                   c, ...
                                   num_labels, X, y, lambda);

% Now, costFunction is a function that takes in only one argument (the
% neural network parameters)
[nn_params_ran, cost] = fmincg(costFunction, initial_nn_params_ran, options);

% Obtain Theta1 and Theta2 back from nn_params
ini_theta1_ran = reshape(nn_params_ran(1:c * (input_layer_size + 1)), ...
                 c, (input_layer_size + 1));

ini_theta2_ran = reshape(nn_params_ran((1 + (c * (input_layer_size + 1))):end), ...
                 num_labels, (c + 1));




%%%%%
%putting back these in the original theta
%%%

k=0;
for i=1:hidden_layer_size
   if ran_mat(i)>=percent
```

```
        k++;
        for j=1:input_layer_size+1
                initial_Theta1(i, j) = ini_theta1_ran(k,j);
        end

   end
end


k=1;
for i=1:num_labels
         initial_Theta2(i, 1) = ini_theta2_ran(i, 1);
end

for i=1:hidden_layer_size
   if ran_mat(i)>=percent
        k++;
        for j=1:num_labels
                initial_Theta2(j, (i+1)) = ini_theta2_ran(j, k);
        end
   end
end



end %ending loop of xyz

end %end of iterations loop


Theta1 = initial_Theta1;
Theta2 = initial_Theta2;




%% ================= Implement Predict =================


for i=1:size(Theta2, 1)
  for j=1:size(Theta2, 2)
        Theta2(i, j) = Theta2(i, j)/2;  %outgoing weights halved
   end
end


pred = predict(Theta1, Theta2, X);

fprintf('\nTraining Set Accuracy: %f\n', mean(double(pred == y)) * 100);
```

%max iter 50 .. and increment 100 -> 94%
%max iter 20 .. and increment 100 -> 89%
%max iter 15 .. and increment 100 -> 91-93%
%max iter 10 .. and increment 100 -> 89 %