This is continuation of the last homework. Just like before, your homework submission must contain the following to receive full credit:

- A single script called "hmwk4.m" or "hmwk4.py" that requires no arguments and prints the things I ask for in bold below. Each output should be labeled.
- A pdf called hmwk1_results.pdf with your solutions to all theory problems and also the text output from the hmwk1 script. See the posted homework example.
- As many other code files as you wish - but only things you wrote or solutions to previous homeworks.

1. (a) Create a function to implement the gradient of the neural net objective function you created in your last homework assignment. Your function should have prototype

```
function dW = net_grad(W,D,L)
```

and the following specifications:

- The function should return a cell-array or list of matrices containing the gradients with respect to the weights. The dimensions of these gradients should match those of the weight matrices handed in as argument `W`.
- You can never evaluate the exponential of a positive number.
- Your function may contain only 2 `for` loops (to loop over layers of the net during the forward and backward pass). You may NOT loop over feature vectors.

(b) Load the mnist dataset, and create a random neural network for this problem with 2 hidden layers of 50 neurons each. Build a function handle that accepts an array of size 50×50, sets the second weight matrix equal to this argument, and then return the cross-entropy objective. Build a second function handle that accepts a 50×50 array of weights, calls `net_grad`, and then returns the gradient with respect to the second weight matrix. These functions should use the entire MNIST training dataset to evaluate the objective/gradient. **Verify your gradient by calling `check_gradient` using these two function handles, and capture the output in the console.**

Congratulations! You're now a gradient expert! Have cold beer.