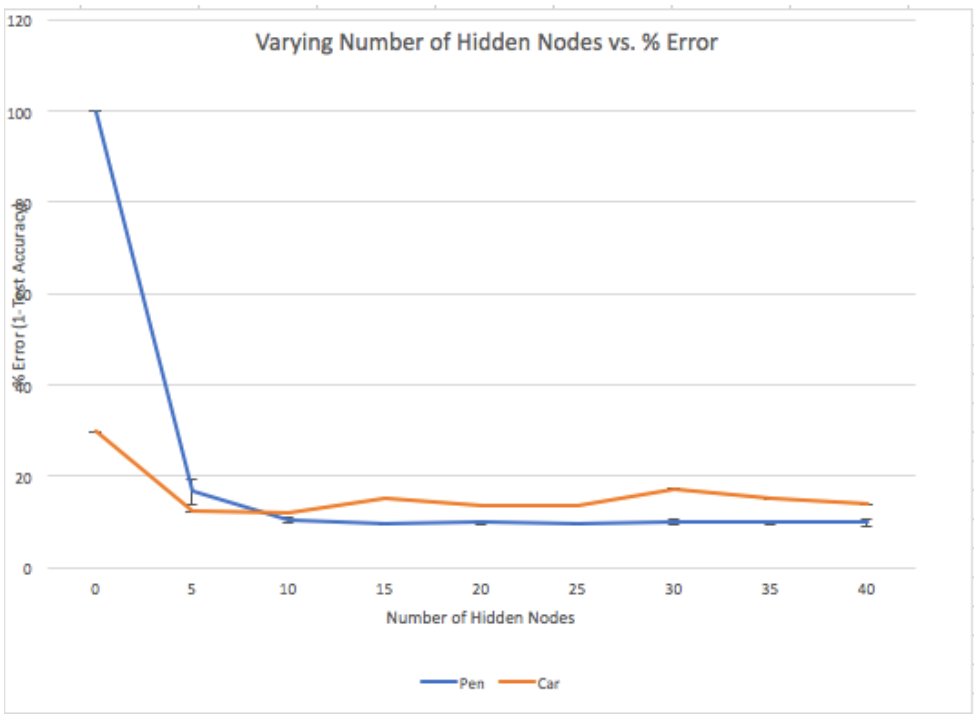
Car							
		Trial 1	Trail 2	Trial 3	Trial 4	Trial 5	Avg.	Std. Dev
	default	0.893979	0.907068	0.894634	0.893325	0.894634	0.896728	0.005193
	0	0.701571	0.70157068	0.70157068	0.701571	0.701571	0.701571	0
	5	0.887435	0.882199	0.896597	0.849476	0.86911	0.876963	0.016369
	10	0.866492	0.865183	0.886126	0.890707	0.885471	0.878796	0.010741
	15	0.855366	0.859293	0.854712	0.842932	0.835733	0.849607	0.008833
	20	0.848168	0.870419	0.879581	0.864529	0.854058	0.863351	0.01124
	25	0.860602	0.865838	0.867147	0.860602	0.861257	0.863089	0.002819
	30	0.824607	0.831806	0.83377	0.820681	0.829843	0.828141	0.00482
	35	0.839005	0.856021	0.85733	0.850785	0.844895	0.849607	0.006881
	40	0.861911	0.859293	0.850785	0.871073	0.863874	0.861387	0.006589

Pen								
		Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Average	Std. Dev.
	default	0.904517	0.909091	0.904803	0.904803	0.90566	0.905775	0.001702
	0	0	0	0	0	0	0	0
	5	0.846198	0.851058	0.843911	0.845626	0.777301	0.832819	2.78600342
	10	0.897942	0.9008	0.906518	0.891652	0.888222	0.897027	0.65049572
	15	0.901944	0.901658	0.907376	0.90566	0.901372	0.903602	0.2448555
	20	0.901658	0.891081	0.906804	0.906232	0.904517	0.902058	0.5773313
	25	0.903659	0.903087	0.907089	0.904231	0.909377	0.905489	0.238223
	30	0.889937	0.901372	0.899657	0.908519	0.9008	0.900057	0.59407548
	35	0.903087	0.892224	0.904517	0.905089	0.903087	0.901601	0.475417
	40	0.902516	0.905946	0.90709	0.888222	0.904231	0.901601	0.68667786

The first row of the tables addresses question 5. I included the default values at the top

By increasing the number of hidden layer perceptrons, we increase the accuracy of the Neural Net. With zero hidden nodes it becomes a linear classifier, so you cannot solve nonlinear problems like xor. As you increase the number of nodes, you increase the complexity of the problems the network can solve. The pens data set is much larger and as a result the percent error is lower and percent accuracy is higher compared to the other data set.

Note: However sometimes training a very large and complex network on a small data set can lead to overfitting. K-fold cross validation and withholding some sample data for validation can prevent overfitting.



Extra Credit:
As mentioned previously, with zero hidden nodes a Neural Network is like a linear classifier with input nodes and weights to output nodes. For xor, the four classic examples are:
(0,0) -> 0
(0,1) -> 1
(1,0) -> 1
(0,0) -> 0

A Neural Network with zero hidden nodes adjusts its weights between its input nodes and output node essentially drawing a line between the data separating positive and negative values. It is not possible to draw a line to separate the points of xor in two dimensions. The network will be unable to classify and it will usually only get 2 out of the four correct.