objectsb1 = CS6640_objects(fname);
[r, c, dtb1] = CS6640_veh_track(objectsb1);
Xb1 = CS6640_features(objectsb1, dtb1);

↓ 2
↓ ↓3

Training Data	num-samples	
objectsb1, dtb1, Xb1	138	⎤ 353
objectsb2, dtb2, Xb2	215	⎦
Xc1	120	⎤ 289
Xc2	169	⎦
Xt1	142	⎤ 308
Xt2	166	⎦

Testing Data

	num-samples
objectsb3, dtb3, Xb3	284
objectsc3, dtc3, Xc3	86
objectst3, dtt3, Xt3	149

net = feedforwardnet([10,10]);
net.divideParam.valRatio = 0;
net.divideParam.testRatio = 0;
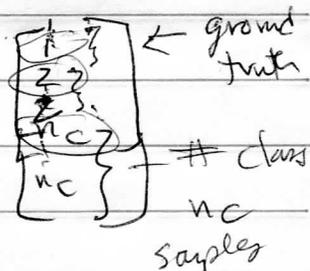net.divideParam.trainRatio = 1;
net.trainParam.epochs = 10000;

$\begin{array}{c} n \text{ samples} \\ x_1 \; x_2 \; \dots \; x_n \end{array}$

$m \Big[\quad \Big]$

m-tuple features

[net, tr, Y, E] = train(net, P, T);
class = round(net(X));

← ground truth

class

nc

nc samples

NN's take features produced by user

CNN's take images directly & find features

convolution + Bias + Activation   subsample e.g., max   regular net

image   Feature Maps   Pooled Feature maps   output
                                                          Fully connected Network