Slide 0

Lecture 2.1

Validation

Jonathan Shapiro

Department of Computer Science, University of Manchester

February 2, 2003

Slide 1

Evaluating Performance

How do we measure the performance of the network? Depends on the application. Some possibilities:

1. Root-mean-square error — the square root of the mean-squared error. A measure of the deviation from actual.
2. Misclassification error (for a classifier).
3. Financial loss, etc.

Call that performance measure, the “error”.

Slide 2

Terminology

Apparent error: (training error) the error on the training data. What the learning algorithm tries to optimize.

True error: the error that will be obtained in use. *What we want to optimize*. Unknown.

The apparent error is *not* a good estimate of the true error. It is optimistic.

Test error: (out-of-sample error) an estimate of true error obtained by testing the network on independent data. The larger the test set, the more accurate the estimate.

$$\text{accuracy of error estimate} = \frac{\text{std dev}}{\sqrt{\text{number of examples used to estimate it}}}$$

Slide 3

Testing procedures

Train-and-test: (Hold out method) Partition the data into two sets. One for training, one for testing. (e.g. $2/3$ of the data for training, $1/3$ for testing.)

Resampling: do train-and-test multiple times over independent partitions.

Slide 4

Why resample?

What if you have a very small dataset.

- Problem — you need to use as much data as possible for training.
- You need data for testing.
- Uses all data for training *and* testing.
- Greatly increased computation cost.

Slide 5

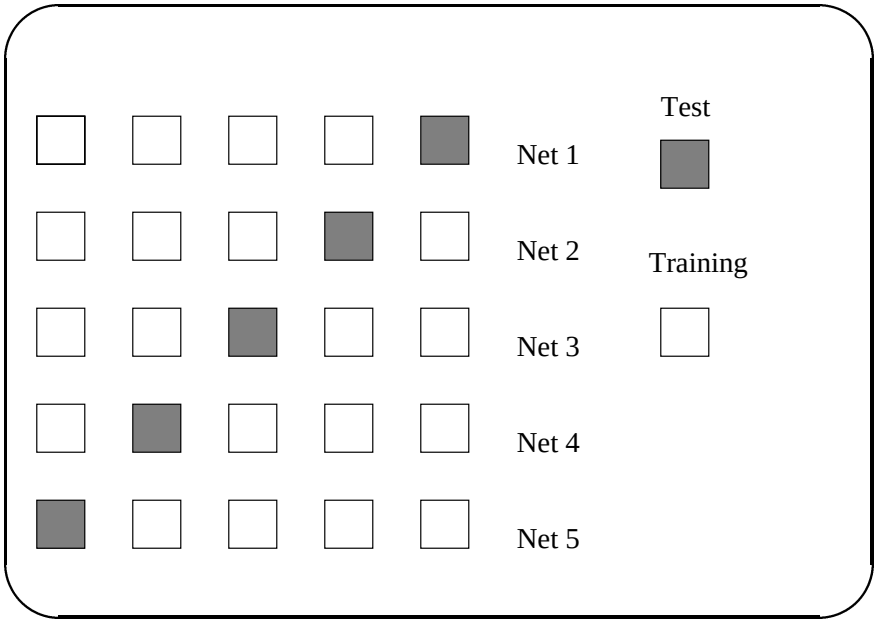
Cross-validation:

- Divide the data into S sets.
- Make S networks, each one trained on $S-1$ of the sets and tested on the (different) remaining one.
- The test error averaged over all S networks is an unbiased estimate of the true error on any randomly chosen network.

Called *leave-one-out* if the test set size is 1 for each partition.

Note: cannot take the best network, must take a random one. Or, use all of them and average the output (called a *committee machine*).

Slide 6



Slide 7

Cross-validation pseudo-code I

Input: Training data of size N. The number of partitions S which evenly divides N.

Returns: A network *net* and its estimated true error $estimate$.

Slide 8

Algorithm:

```
Divide the data into S sets: D(1), D(2), ... , D(S).

estimate=0
for i = 1 to S
   remove D(i) from the training set
   create new network net(i)
   train net(i) on training data
   E(i) = test error on net(i) measured on D(i)
   estimate=estimate+E(i)/S
end loop

i=random number between 1 and S
return(net(i),estimate)
```

Slide 9

Comparing networks

Since,

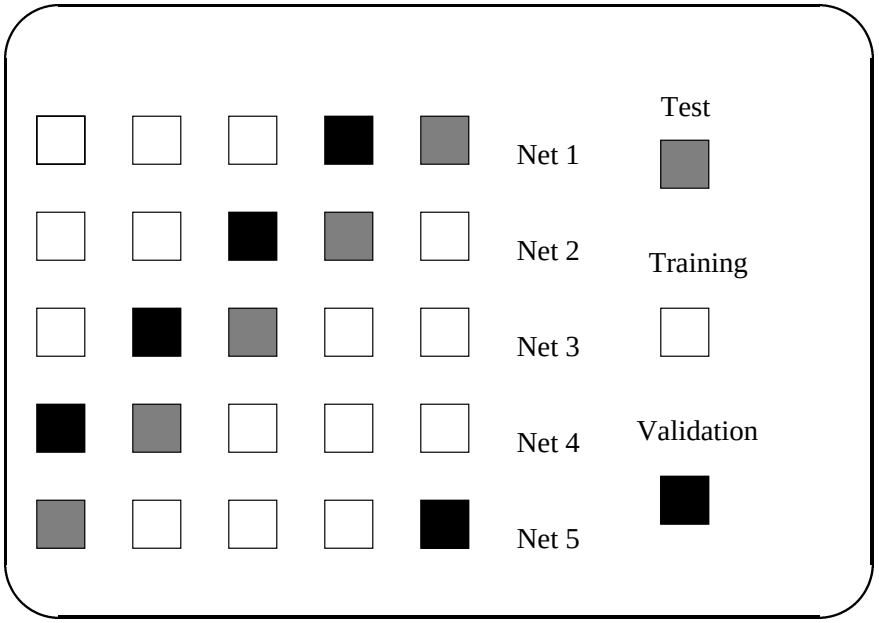
1. we want to optimize true error, not apparent error, and
2. apparent error is not a fair estimate of true error

It is not correct to pick networks with lowest training error.

Better to introduce a *validation* set to compare networks. Network which performs better on validation set (independent of the data used to train either) is chosen.

Test set still needed to give a fair estimate of true performance.

Slide 10



Slide 11

Model Complexity

Complex models: models with many adjustable weights and biases will

1. Be more likely to be able to solve your task,
2. be more likely to store the training data without solving the task.

Simple models: The simpler the model that can store the training data, the more likely that it will generalize.

This is the fundamental trade-off:

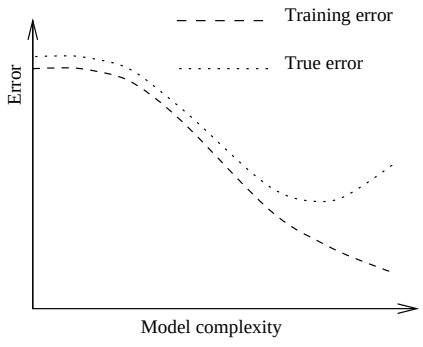
- Too simple — cannot do the task
- too complex — cannot learn the task from small datasets.

Slide 12

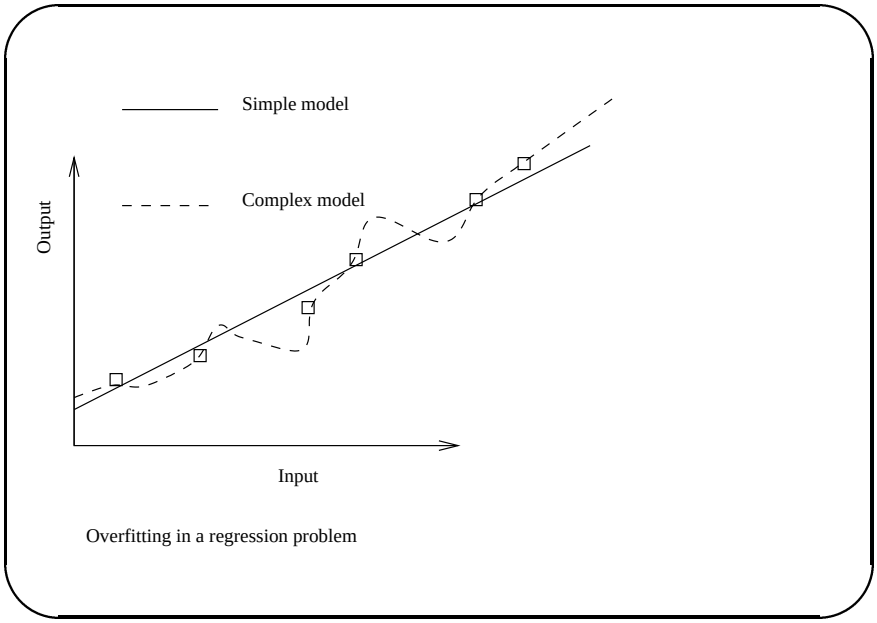
- Selecting appropriate complexity $\longrightarrow$ "model selection"
- Controlling complexity $\longrightarrow$ "regularization"

This is why a validation set is used.

Slide 13

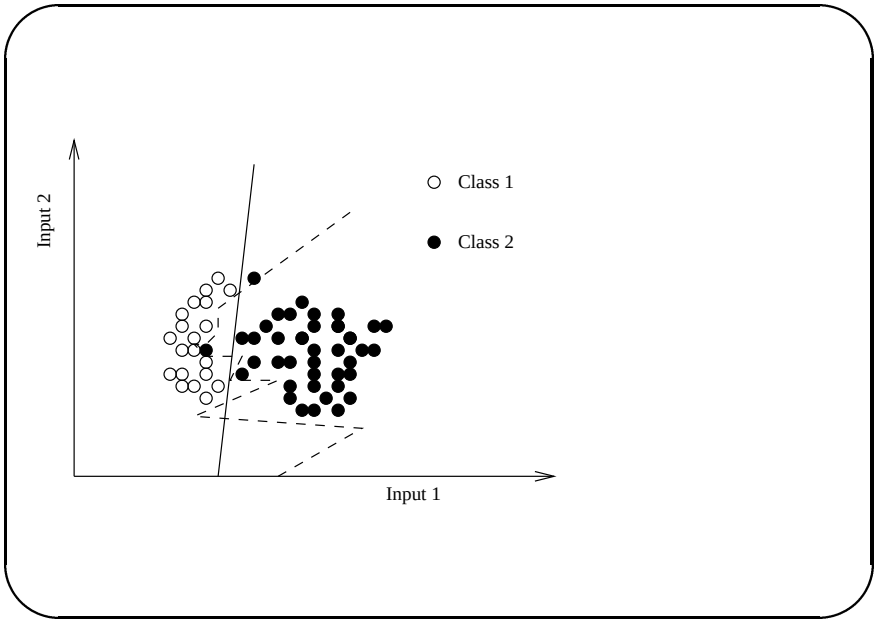


Slide 14



Overfitting in a regression problem

Slide 15



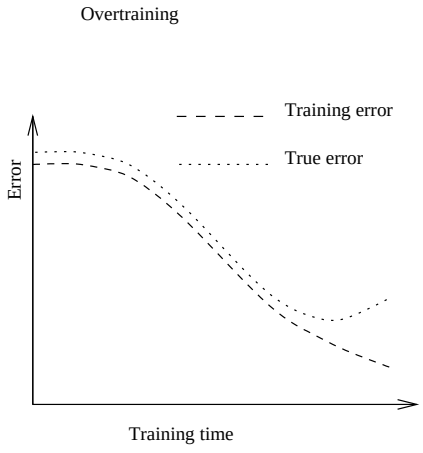
Slide 16

Overtraining

- For neural networks, complexity can increase during training. Weights become more tuned to the training data.
- Overtraining — when true error starts to decrease during training. Network becomes tuned to particular noise or spurious properties of the training set.
- Regularization method — early stopping. Stop training when validation set error starts increasing.

In general, it is called “overfitting” when learning model becomes tuned to features specific to the training set.

Slide 17



Slide 18

Validation and Model Selection

- To selection a model of appropriate complexity, test models of different complexity. Use best one.
- Train several networks of different complexity on training set. Find which has the best performance on validation set. Choose that one. Test it on test set.
- Example — Early stopping. Stop training the network when the error on the validation set starts to increase.

Slide 19

Conclusions

1. Lower apparent error does not necessarily lead to the best generalization.
2. Basic trade-off: Network should be sufficiently complex to be able to do the task, but not so complex that it can overfit the data, or memorize the data without generalizing. Model selection or regularization may be required to balance the trade-off.
3. Testing must always be done on data which had nothing to do with the creation or selection of the network to give an unbiased estimate of the true error.