Evaluation and Validation. Design for Generalization

**From Last Lecture**

- Multilayer Perceptrons are universal approximators.
- Given a large enough number of hidden units, a network with sigmoid neurons can approximate any function with an arbitrarily small error.
Today

- How to evaluate performance?
- How many units should we use in the hidden layer?
- When to stop training?

The Training Set

- Among all possible patterns, the system learns from a limited training set.
- The training set should follow, as well as possible, the distribution of the other samples in the universe.
- This can be done by selecting random samples in sufficient number among the true population.

Universe

- Training Samples
- Other possible samples
Evaluating Learning Quality

- Up to now we have considered, as measure of fit, the mean squared error in the training set:
  \[ E_{\text{train}} = \frac{1}{P} \sum_{p=1}^{P} e^2(x^p) \]

- However, what we really would like to know is the expected error for all possible samples.
  \[ E_{\text{true}} = E\{e^2(X)\} = \int e^2(X)p(X)dX \]

- \( X \) is a random variable with density function \( p(X) \) from which samples \( x \) are drawn.

- \( E_{\text{true}} \) can not be evaluated explicitly. We must rely on estimates of its value.

- \( E_{\text{train}} \) is not a good estimate ...

Generalization

- The way the system generalizes its knowledge to samples outside the training set depends on many factors.

- Overfitting: when the system approximates very well the training samples but generalizes badly to previously unseen patterns.

- Generalization Ability = Prediction Accuracy
To evaluate the generalization ability of the system after training is complete, we draw different samples (test set) from the population and compute the Testing Error.

\[ E_{test} = \frac{1}{T} \sum_{t=1}^{T} e^2(x^t) \]

- \( T \) – number of test samples
- \( x^t \) – test samples.

In which of the sets (training or test sets) the error is smaller?

How to train for generalization?

- The generalization ability of a system depends on many factors, including:
  - Complexity (e.g. number of hidden units in a NN).
  - Amount of training.

- Usually the generalization ability decreases (error in a test set increases) when the number of hidden units increase and/or the amount of training exceeds a reasonable value.
Validation Set

- To make decisions in the training process to improve the generalization capability (number of hidden units, amount of training) we choose another independent set of samples from the population.
- This is the so-called Validation Set.
- The validation error:
  \[ E_{\text{valid}} = \frac{1}{T} \sum_{v=1}^{V} e^2(x^v) \]
- Goal of training: Minimize the validation error.

When to stop training?

- The training error usually decreases monotonously with the amount of training.
- Instead, the validation error starts increasing after a certain amount of training.
- Training should stop when the error in the validation set starts increasing.
- Because error in the validation set tends to oscillate, we should let training continue for a while and then choose the set of weights with the best performance.
Model Selection by Validation

- How to choose the number of hidden units in the network?

- Try several alternatives and choose the one with the least validation error.

Model Selection by Regularization

- When the amount of data is small, we may not be able to have independent training and validation sets.

- We can use Regularization to improve generalization:

  \[ E_{\text{total}} = E + \lambda E_{\text{reg}} \]

  - \( E \) - the (usual) data fit term.
  - \( E_{\text{reg}} \) - the regularization term.
  - \( E_{\text{total}} \) - the cost to minimize.
  - \( \lambda \) - How much regularization to use
  - Role of \( E_{\text{reg}} \) is to promote smooth output functions.

- Term to penalize high weights:

  \[ E_{\text{reg}} = \sum_{i,j} w_{ij}^2 \]

- The total gradient is:

  \[ \frac{\partial E_{\text{total}}}{\partial w_{ij}} = \frac{\partial E}{\partial w_{ij}} + 2\lambda w_{ij} \]

  - If derivative of \( E \) is zero, the weight will decay exponentially to zero.

  \[ \frac{\partial E}{\partial w_{ij}} = 0 \quad \Rightarrow \quad w_{ij}^{t+1} = (1 - 2\eta\lambda)w_{ij}^t \]

  - Is stability guaranteed?
Pruning

- Big networks (more weights than training patterns) usually generalize poorly.

- A technique to improve generalization consists in removing connections whose weights approach zero during training, effectively reducing its size.
  - skeletonization method of Mozer and Smolensky (1989),
  - optimal brain damage (Le Cun et al 1990b)
  - optimal brain surgeon (Hassibi et al 1993)

- This type of pruning is time consuming because after each pruning the networks must be retrained.

Pruning by Regularization

- Use the following regularization term (Williams 1994):
  \[ E_{reg} = \sum_{i,j} |w_{ij}| \]

- The gradient becomes:
  \[ \frac{\partial E_{total}}{\partial w_{ij}} = \frac{\partial E}{\partial w_{ij}} + \lambda \text{sgn}(w_{ij}) \]
  - sgn(.) is the sign function.

- If the derivative of E is zero, the weight will decay linearly to zero.
  \[ \frac{\partial E}{\partial w_{ij}} = 0 \quad \Rightarrow \quad w_{ij}^{t+1} = w_{ij}^t - \lambda \text{sgn}(w_{ij}^t) \]

- This acts simultaneously as a regularizer, tending to keep the weights small, and as a pruner, since it automatically sets the least important weights to zero.
Cross Validation

- Another technique that uses **all available data in training** and reduce variance in the generalization measure, is called **cross-validation**.

- The training data is **split into multiple disjoint sets**.

- In each round, **different sets for training and validation** are employed.

- The **final validation measure** is obtained by computing the statistics of the individual validation rounds (average and variance).

---

Cross-Validation Techniques

- **K-fold cross-validation:**
  - Partition original data in K subsets.
  - Sequentially choose one of the sets and use it for validation, training the system with all the other (K-1) sets.
  - Average the validation results.

- **Leave-one-out cross-validation (LOOCV)**
  - Use a single observation as the validation data and use all other to train the system.
  - Repeat for all data samples.

- **Repeated random sub-sampling validation.**
  - Randomly split the data set in training and validation data.
  - Repeat with different random splits.
Other uses of cross-validation

- To choose learning method.
  - Neural Networks
  - Polynomial Regression
  - Nearest Neighbor
  - ...

- To choose the training algorithm.
  - Gradient descent
  - Newton’s method
  - ...

- Feature selection - to choose which features from the input data (entries of the input vectors) are truly informative.

Validation vs Testing

- When validation data is used during the training process (decide when to stop training, select the model, e.g. number of hidden units, number of layers), an independent test set must still be used to evaluate the final quality of the network.

- The reason is that training will be biased towards good results in the validation set.

- During the training process we may tend to overtrain the system for having good results in the validation set, but again, what is the performance in an independent set?

- Bottom line: Test sets must be used only once - to perform the final evaluation of a learning system.
Evaluation Criteria

- The mean squared error (MSE) is not a good evaluation measure. It depends on the magnitude of the samples.

- It allows to compare different solutions but do not let us know if the solutions are good or bad.

- If we have a data set in millimeters, the MSE will be much larger then for a data set in meters (why?). Does this mean that learning is better in meters?

- The following measures can used to assess quality independently of scale:
  - For fitting problems:
    - Pearson Correlation Coefficient between outputs (predictions) and desired values.
  - For classification problems:
    - Percentage of misclassifications
    - Confusion matrix
    - Receiver Operating Characteristic

Evaluating Fitting Problems

- The Pearson Correlation coefficient is a measure of correlation (linear dependence between two variables).

\[ r \equiv \frac{1}{n-1} \sum_{i=1}^{n} \frac{X_i - \bar{X}}{s_x} \frac{Y_i - \bar{Y}}{s_y} \]

\[ \bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i \]
\[ s_x = \frac{1}{n-1} \sum_{i=1}^{n} (X_i - \bar{X})^2 \]

- \( r \) is a value between 0 and 1:
  - \( r = 1 \), perfect correlation.
  - \( r = 0 \), no correlation.
Evaluating Classification Problems

- **Binary problem:**
  - 1 – positive sample/classification
  - 0 – negative sample/classification

- **Classification Types:**
  - True Positive – the output of the classifier is 1 for a input of 1.
  - True Negative – the output of the classifier is 0 for a input of 0.
  - False Positive – when the output of the classifier is 1 for a input of 0.
  - False negative – when the output of the classifier is 1 for a input of 0.

- **Types of Errors:**
  - Type I – False positive/False alarm / Misdetection:
    - Poor specificity
    - High sensitivity
  - Type II – False negative /Detection Failure:
    - High specificity
    - Poor sensitivity

Confusion Matrix

Number of patterns of class i classified by the network in elements of class j, and corresponding percentages with respect to the whole data set.

Alexandre Bernardino, alex@isr.ist.utl.pt Machine Learning, 2009/2010
Receiver Operating Characteristic

- In a network of sigmoidal units, the decision to classify a pattern in the positive or negative class depends on a threshold.
- Usually the decision threshold is half the range of the sigmoidal unit.
- This value determines the balance between true positives and false positives.
- A too low threshold will tend to accept many false positives.
- A too high threshold will tend to reject many true positives.
- The ROC curve displays this effect, as the threshold is changed.

Examples

- Matlab Neural Network Toolbox:
  - Fitting: nftool
  - Classification: nprtool