The Curse of Dimensionality

Functional Programming and Intelligent Algorithms

Prof Hans Georg Schaathun

Høgskolen i Ålesund

12th February 2015

Outline

Sources of Bad Training

Very often we get a high error rate when we have trained a neural network.

- Why?

1. Too few nodes
2. Too few training iterations
3. Too many training iterations
4. Too little training data

Sources of Bad Training

Very often we get a high error rate when we have trained a neural network.

- Why?

1. Too few nodes
2. Too few training iterations
3. Too many training iterations
4. Too little training data

Underfitting and overfitting

Performance on ...	Underfitting	Overfitting
Training set	Bad	Good
Testing set	Bad	Bad

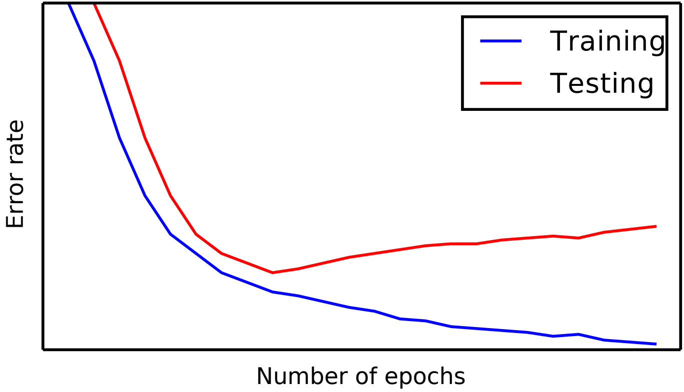
Underfitting	Overfitting
Too few epochs	Too many epochs
Too few nodes	Too little training data

Underfitting and overfitting

Performance on ...	Underfitting	Overfitting
Training set	Bad	Good
Testing set	Bad	Bad

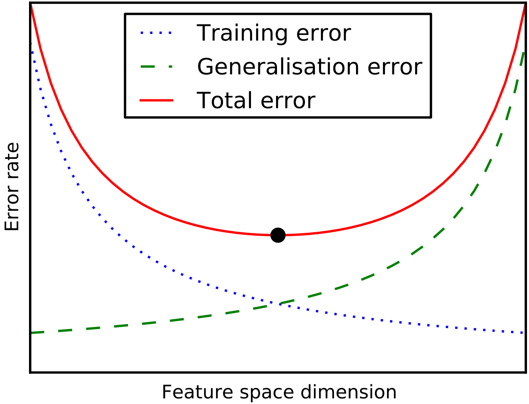
Underfitting	Overfitting
Too few epochs	Too many epochs
Too few nodes	Too little training data

Number of epochs



Degrees of Freedom

Figure 13.1 from Schaathun *Machine Learning in Image Steganalysis*



Curse of Dimensionality

- Dimension = Number of features
- High dimension
  - $\Rightarrow$ large volume
  - $\Rightarrow$ sparse data
- Dimension = Number of weights (less one)
  - $\Rightarrow$ degrees of freedom
  - $\Rightarrow$ flexible model
  - $\Rightarrow$ fits *training data* too well

HØGSKOLEN I ÅLESUND
Aalesund University College

Outline

Dimensionality Reduction

1. Feature selection
   - look for correlation between features
   - remove redundant features
2. Feature extraction
   - transform from feature space to lower dimension
   - generalisation of feature selection
   - individual features no longer recognisable

Dimensionality Reduction

1. Feature selection
   - look for correlation between features
   - remove redundant features
2. Feature extraction
   - transform from feature space to lower dimension
   - generalisation of feature selection
   - individual features no longer recognisable

Dimensionality Reduction

1. Feature selection
   - look for correlation between features
   - remove redundant features
2. Feature extraction
   - transform from feature space to lower dimension
   - generalisation of feature selection
   - individual features no longer recognisable

Feature Selection

The simplest case

- Trial and error
  1. Train and test the classifier
  2. Remove some feature
  3. Train and test again
  4. If performance is improved,
     - discard the removed feature
     - else, reinstate it
  5. Repeat from Step 2
- Alternatively, bottom-up
  - Start with no features
  - Repeatedly add the most useful feature

Feature Selection

The simplest case

- Trial and error
  1. Train and test the classifier
  2. Remove some feature
  3. Train and test again
  4. If performance is improved,
     - discard the removed feature
     - else, reinstate it
  5. Repeat from Step 2
- Alternatively, bottom-up
  - Start with no features
  - Repeatedly add the most useful feature

HØGSKOLEN I ÅLESUND
Aalesund University College

Outline

Separation

What properties make separation simple? What makes it harder?

- High variance within a class: hard
- Big difference between classes: easier

Separation

What properties make separation simple? What makes it harder?

- High variance within a class: hard
- Big difference between classes: easier

Between-class variability

1. Let $\boldsymbol{\mu}_i$ be the mean feature vector of Class i
2. How can we measure the difference between the two Classes?
   - $\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2$
3. Vectors: we measure variability per axis

A classification heuristic

- Classifier: hyperplane $\mathbf{w} \cdot \mathbf{x} - w_0 = 0$
- Heuristic: $h(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} - w_0$ (distance to hyperplane)
  - Class 1: $h(\mathbf{x}) < 0$
  - Class 2: $h(\mathbf{x}) > 0$
  - The larger $|h(\mathbf{x})|$, the better confidence
- Within-class variation (Class i)
  - $\mathrm{Var}(h(\mathbf{x})) = \mathbf{w}^{\mathrm{T}}\Sigma_i^2\mathbf{w}$
  - Where Σ_i^2 is the covariance matrix
  - $\Sigma_i^2 = E\big((\mathbf{x} - \boldsymbol{\mu}_i)(\mathbf{x} - \boldsymbol{\mu}_i)^{\mathrm{T}}\big)$

Fisher Linear Discriminant (FLD)

- Separation:
  - $L = \frac{\sigma^2_{\text{between}}}{\sigma^2_{\text{within}}} = \frac{(\mathbf{w}\boldsymbol{\mu}_1 - \mathbf{w}\boldsymbol{\mu}_2)^2}{\mathbf{w}^{\mathrm{T}}\Sigma_1^2\mathbf{w} + \mathbf{w}^{\mathrm{T}}\Sigma_2^2\mathbf{w}}$
- Minimise separation by
  - $\mathbf{w} = (\Sigma_1^2 + \Sigma_2^2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$

Outline

HØGSKOLEN I ÅLESUND
Aalesund University College

Principal Component Analysis

Figure 6.6 from Marsland

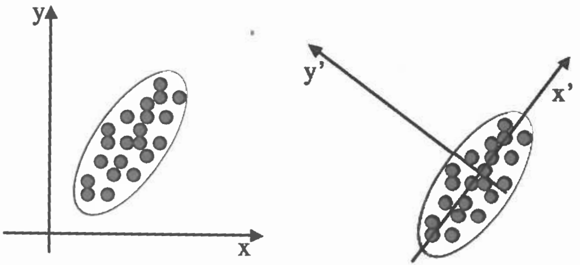


FIGURE 6.6 Two different sets of coordinate axes. The second consists of a rotation and translation of the first and was found using Principal Components Analysis.

Principal Component Analysis

1. PCA finds a new basis
2. First axis – the principal component
   - ... explains most of the variation
3. Next axis chosen perpendicular to previous axes
   - ... to explain most of the remaining variation

PCA Algorithm

1. Write N data points as rows of a matrix X (size $N \times M$)
2. For each column, subtract its mean to get B
3. Compute covariance $C = \frac{1}{N} B^{\mathrm{T}} B$
4. Compute eigenvectors and eigenvalues of C
   - $V^{-1}CV = D$
   - D: diagonal matrix with eigenvalues
   - V: matrix of eigenvectors
5. Sort the columns of D in deacrising order of eigenvalues
   - apply same order to V
6. Discard columns with eigenvalue less than η
7. Transform data by multiplication with V

HØGSKOLEN I ÅLESUND
Aalesund University College

Conclusion

Performance on ...	Underfitting	Overfitting
Training set	Bad	Good
Testing set	Bad	Bad

	Underfitting	Overfitting
Training time	Too few epochs	Too many epochs
Curse of dimensionality	Too few nodes	Too little training data