Machine Learning 2

4. Neural Networks

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute for Computer Science
University of Hildesheim, Germany

Syllabus

		A. Advanced Supervised Learning
Tue. 5.4.	(1)	A.1 Generalized Linear Models
Tue. 12.4.	(2)	A.2 Gaussian Processes
Tue. 19.4.	(3)	A.2b Gaussian Processes (ctd.)
Tue. 26.4.	(4)	A.3 Advanced Support Vector Machines
Tue. 3.5.	(5)	A.4 Neural Networks
Tue. 10.5.	(6)	A.5 Ensembles
Tue. 17.5.	(7)	A.5b Ensembles (ctd.)
Tue. 24.5.	—	— Pentecoste Break —
Tue. 31.5.	(8)	A.6 Sparse Linear Models — L1 regularization
Tue. 7.6.	(9)	A.6b Sparse Linear Models — L1 regularization (ctd.)
Tue. 14.6.	(10)	A.7. Sparse Linear Models — Further Methods
		B. Complex Predictors
Tue. 21.6.	(11)	B.1 Latent Dirichlet Allocation (LDA)
Tue. 28.6.	(12)	B.2 Deep Learning
Tue. 5.7.	(13)	Questions and Answers

Outline

1. Network Topologies

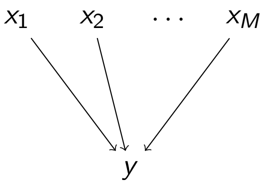
2. Stochastic Gradient Descent (Backpropagation)

3. Regularization

Outline

1. Network Topologies
2. Stochastic Gradient Descent (Backpropagation)
3. Regularization

Logistic Regression (0 hidden layers)



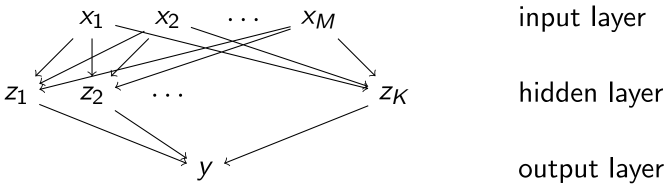
input layer

output layer

logistic regression:

$$\hat{y}(x) := \hat{p}(y = 1 \mid x) = \text{logistic}(\beta^T x), \quad x \in \mathbb{R}^M$$

Feedforward Neural Network (1 hidden layer)

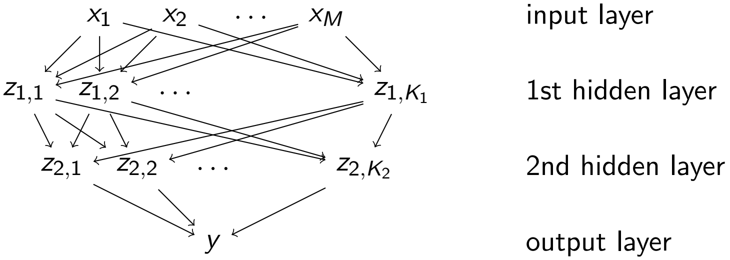


feedforward neural network (1 hidden layer):

$$\hat{y}(x) := \hat{p}(y = 1 \mid x) = \text{logistic}(\beta_2^T z(x))$$

$$z_k(x) := \text{logistic}(\beta_{1,k}^T x), \quad k = 1, \ldots, K, x \in \mathbb{R}^M$$

Feedforward Neural Network (2 hidden layers)



feedforward neural network (2 hidden layers):

$$\hat{y}(x) := \hat{p}(y = 1 \mid x) = \text{logistic}(\beta_3^T z_2(x))$$
$$z_{2,k}(x) := \text{logistic}(\beta_{2,k}^T z_1), \quad k = 1, \dots, K_2$$
$$z_{1,k}(x) := \text{logistic}(\beta_{1,k}^T x), \quad k = 1, \dots, K_1, x \in \mathbb{R}^M$$

Different Targets y

Binary classification:

$$\hat{y}(x) := \hat{p}(y = 1 \mid x) = \text{logistic}(\beta_{L+1}^T z_L(x))$$

Regression:

$$\hat{y}(x) := \beta_{L+1}^T z_L(x)$$

Regression with multiple outputs:

$$\hat{y}(x) := \beta_{L+1} z_L(x), \quad \beta \in \mathbb{R}^{K_{\text{out}} \times K_L}$$

Multi-class classification:

$$\hat{y}(x) := \hat{p}(y \mid x) = \text{softmax}(\beta_{L+1} z_L(x))$$

Notes:

- L hidden layers
- at hidden nodes always are logistic/sigmoid functions (**activation function**, **transfer function**).

Network Topologies

- **feedforward neural network** (aka **multiplayer perceptron**, MLP)
  - often just a single hidden layer is used
    - NN with single hidden layer is already a **universal approximator**
  - **skip arcs** can be used to connect layers skipping a hidden layer
  - sometimes layers are not connected completely, but have **sparse connections**.
  - nodes & connections always form a DAG

- **recurrent neural network**
  - neural networks with backward connections / not a DAG.
  - used in language modeling
  - no simple probabilistic interpretation

- **Hopfield networks** / **associative memory**:
  - symmetric connections between hidden units
  - probabilistic counterpart: Boltzmann machine.

Outline

1. Network Topologies
2. Stochastic Gradient Descent (Backpropagation)
3. Regularization

SGD / Loss

feedforward neural network, L hidden layers with $K_1, \ldots, K_L$ nodes each:

$$\begin{aligned} z_{\ell,k}(x) &:= s(\beta_{\ell,k}^T z_{\ell-1}), & \beta_{\ell,k} &\in \mathbb{R}^{K_{\ell-1}}, k = 1, \ldots, K_\ell \\ z_{0,k} &:= x_k, & k &= 1, \ldots, K_0 := M \\ \hat{y}_k(x) &:= z_{L+1,k}(x), & k &= 1, \ldots, K_{L+1} := \dim \mathcal{Y}, \text{ usually } = 1 \end{aligned}$$

$$f(\beta) := \frac{1}{N} \sum_{n=1}^{N} \ell(y_n, \hat{y}(x)) + \frac{\lambda}{2} ||\beta||^2$$

gradient for single sample x_n, y_n:

$$\frac{\partial f}{\partial \beta_{\ell,k}} = \ell'(y_n, \hat{y}(x_n)) \sum_{k'=1}^{K_{L+1}} \frac{\partial \hat{y}_{k'}(x_n)}{\partial \beta_{\ell,k}} + \lambda \beta_{\ell,k} = \quad \ell'(y_n, \hat{y}_n) \sum_{k'=1}^{K_{L+1}} \frac{\partial z_{L+1,k'}}{\partial \beta_{\ell,k}} + \lambda \beta_{\ell,}$$

SGD / Layers

feedforward neural network, L hidden layers with $K_1, \ldots, K_L$ nodes each:

$$z_{\ell,k}(x) := s(\beta_{\ell,k}^T z_{\ell-1}), \qquad \beta_{\ell,k} \in \mathbb{R}^{K_{\ell-1}}, k = 1, \ldots, K_\ell$$

gradient for single sample x_n, y_n:

$$\frac{\partial z_{\ell',k'}}{\partial \beta_{\ell,k}} = \sum_{\tilde{k}=1}^{K_{\ell-1}} \frac{\partial z_{\ell',k'}}{\partial z_{\ell'-1,\tilde{k}}} \frac{\partial z_{\ell'-1,\tilde{k}}}{\partial \beta_{\ell,k}}$$

$$\frac{\partial z_{\ell',k'}}{\partial z_{\ell,k}} = s'(\beta_{\ell',k'}^T z_{\ell'-1}) \sum_{\tilde{k}=1}^{K_{\ell-1}} \beta_{\ell',k',\tilde{k}} \frac{\partial z_{\ell'-1,\tilde{k}}}{\partial z_{\ell,k}}$$

$$\frac{\partial z_{\ell',k'}}{\partial z_{\ell'-1,k}} = s'(\beta_{\ell',k'}^T z_{\ell'-1}) \beta_{\ell',k',k}$$

$$\frac{\partial z_{\ell',k'}}{\partial \beta_{\ell'-1,k}} = s'(\beta_{\ell',k'}^T z_{\ell'-1}) z_{\ell'-1,k}$$

SGD / Arrange Computations

1. Feedforward:

$$z_{0,k} := x_k, \qquad k = 1, \ldots, K_0 := M$$

$$z_{\ell,k} := s(\beta_{\ell,k}^T z_{\ell-1}), \qquad \ell = 1, \ldots, L+1, k = 1, \ldots, K_\ell$$

2. Backpropagation:

for $\ell := L+1, \ldots, 1, \quad k := 1, \ldots, K_\ell$:

for $k' := 1, \ldots, K_{L+1}$:

$$\frac{\partial z_{L+1,k'}}{\partial z_{\ell,k}} = \sum_{\tilde{k}=1}^{K_{\ell+1}} \frac{\partial z_{L+1,k'}}{\partial z_{\ell+1,\tilde{k}}} \frac{\partial z_{\ell+1,\tilde{k}}}{\partial z_{\ell,k}} = \sum_{\tilde{k}=1}^{K_{\ell+1}} \frac{\partial z_{L+1,k'}}{\partial z_{\ell+1,\tilde{k}}} s'(\beta_{\ell+1,\tilde{k}}^T z_\ell) \beta_{\ell+1,\tilde{k},k}$$

$$\frac{\partial z_{L+1,k'}}{\partial \beta_{\ell,k}} = \frac{\partial z_{L+1,k'}}{\partial z_{\ell,k}} \frac{\partial z_{\ell,k}}{\partial \beta_{\ell,k}} = \frac{\partial z_{L+1,k'}}{\partial z_{\ell,k}} s'(\beta_{\ell,k}^T z_{\ell-1}) z_{\ell-1}$$

$$\frac{\partial f}{\partial \beta_{\ell,k}} = \ell'(y_n, \hat{y}_n) \sum_{k'=1}^{K_{L+1}} \frac{\partial z_{L+1,k'}}{\partial \beta_{\ell,k}} + \lambda \beta_{\ell,k}$$

SGD / Arrange Computations

2. Backpropagation:

for $\ell := L+1, \ldots, 1, \quad k := 1, \ldots, K_\ell$:

for $k' := 1, \ldots, K_{L+1}$:

$$\frac{\partial z_{L+1,k'}}{\partial z_{\ell,k}} = \sum_{\tilde{k}=1}^{K_{\ell+1}} \frac{\partial z_{L+1,k'}}{\partial z_{\ell+1,\tilde{k}}} \frac{\partial z_{\ell+1,\tilde{k}}}{\partial z_{\ell,k}} = \sum_{\tilde{k}=1}^{K_{\ell+1}} \frac{\partial z_{L+1,k'}}{\partial z_{\ell+1,\tilde{k}}} s'(\beta_{\ell+1,\tilde{k}}^T z_\ell)\beta_{\ell+1,\tilde{k},k}$$

$$\frac{\partial z_{L+1,k'}}{\partial \beta_{\ell,k}} = \frac{\partial z_{L+1,k'}}{\partial z_{\ell,k}} \frac{\partial z_{\ell,k}}{\partial \beta_{\ell,k}} = \frac{\partial z_{L+1,k'}}{\partial z_{\ell,k}} s'(\beta_{\ell,k}^T z_{\ell-1}) z_{\ell-1}$$

$$\frac{\partial f}{\partial \beta_{\ell,k}} = \ell'(y_n, \hat{y}_n) \sum_{k'=1}^{K_{L+1}} \frac{\partial z_{L+1,k'}}{\partial \beta_{\ell,k}} + \lambda \beta_{\ell,k}$$

$$\beta_{\ell,k}^{(t)} = \beta_{\ell,k}^{(t)} - \eta^{(t)} \frac{\partial f}{\partial \beta_{\ell,k}}$$

Outline

1. Network Topologies

2. Stochastic Gradient Descent (Backpropagation)

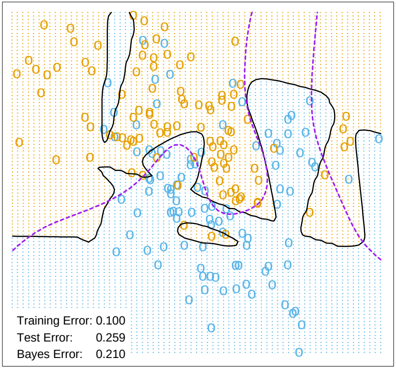
3. Regularization

Regularization of Neural Networks

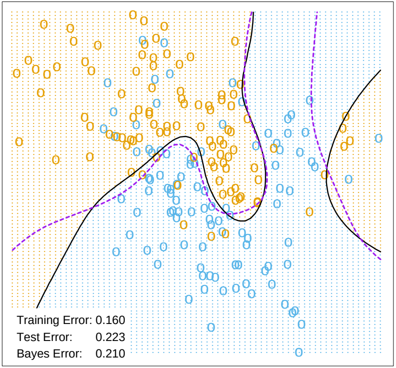
- L2 regularization
  - aka **weight decay**
  - most frequently used method
- L1 regularization
- **early stopping**
- use a random sample of connections
  - **drop connect**

L2 regularization / Example

Neural Network - 10 Units, No Weight Decay



Neural Network - 10 Units, Weight Decay=0.02



[HTFF05, p. 399]

Further Readings

- See [Mur12, chapter 16.5] and [HTFF05, chapter 11].

References

Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin.
The elements of statistical learning: data mining, inference and prediction, volume 27.
Springer, 2005.

Kevin P. Murphy.
Machine learning: a probabilistic perspective.
The MIT Press, 2012.