Neural Networks: III. Regularization (Part 4)

Jan Bauer

jan.bauer@dhbw-mannheim.de

21.05.19

Example (Motivation): Underfitting

Let $x = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$, $y^* = 1$, $w_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$, $w_2 = \begin{pmatrix} \frac{1}{4} \\ \frac{1}{4} \\ \frac{1}{4} \\ \frac{1}{4} \end{pmatrix}$

$\Rightarrow w_1' x = w_2' x = y^*$

Which weight is better?

Example (Motivation): Underfitting

$\rightsquigarrow$ w_2, since it is more general

Example (Motivation): Underfitting

- How to solve underfitting?

- One solution can be: Use more training data

$\rightsquigarrow$ underfitting is easy to fix in this case

Example (Motivation): Overfitting

Example (Motivation): Overfitting

How to solve overfitting?

$\rightarrow$ Regularization

Regularization

- good performance training data $\nRightarrow$ good performance (unseen) test data
- gap between training and test data is large when
  - models are complex
  - training data is small
- increasing number of training instances improves generalization power ($\rightsquigarrow$ popularity of NN in the present)
- model complexity $\nearrow$ $\Rightarrow$ generalization $\searrow$

Most common regularization: $\mathcal{L}_2$ norm

- i.e. "elementwise" $\mathcal{L}_2$

$$R(W) \equiv \frac{1}{2} ||W||_2^2 \equiv \frac{1}{2} \sum_i \sum_j \sum_{N_{i-1}} \sum_{N_i} \left(W_{i,j}^{(N_{i-1}, N_i)} \right)^2$$

(don't learn this by heart!)

- rather above's expression than $R(W) \equiv ||W||_2$

- $\mathcal{L}_1$ is trivial

- $\mathcal{L}_1$ is less 'hard'

To punish non-generalization, regularization is implemented in the loss:

$$L(W) \equiv L \equiv \frac{1}{D}\sum_{d=1}^{D} L_d(y_{[d]}, X, W) + \lambda \frac{1}{2}\,||W||_2^2$$

where we used the $\mathcal{L}_2$ norm as an Example. In short:

$$L \equiv \underbrace{\hat{\mathrm{E}} L_d}_{\text{Data loss}} + \underbrace{\lambda R(W)}_{\text{Reg. loss}}$$

Example: $||w_1||_2^2 > ||w_2||_2^2$

Question Time:

$$L \equiv \hat{\mathrm{E}} L_d + \lambda R(W), \quad R(W) \equiv \frac{1}{2} ||W||_2^2$$

- What does $\lambda = 0$ mean?
- Why $||\cdot||_2^2$?
- Why $\frac{1}{2}$?
- Why is $\frac{1}{2}$ and $||\cdot||_2^2$ "allowed"?
- Is $L = 0$ still a thing? (Is it possible and/or meaningful?)

SVM Loss:

$$L_d \equiv \sum_{k \neq k^*} \max\left(0, y_{[d]}^{(k)} - y_{[d]}^{(k^*)} + \Delta\right)$$

Setting $\Delta = 1$ is fine, because it is connected to (and therefore can be controled by) λ

- the score values are connected to the weights (higher weights $\rightarrow$ higher scores)
- if we want the correct score to be way larger than the wrong ones, we could easily increase the weights by some $\mu > 0$ to get $\mu \cdot W$ as weights.
  - Example: Whiteboard
- but higher values for the weights get punished by $R(W)$. And the larger λ, the more punishes R(W) the loss

Regularization in the bigger picture

- **Aim:** Reduce test error
  - increasing training error is fine
- **How:** Improve generalization
- **How:** Punish input dimensions with large influence on the score
- **How:** Extend the loss by a regularization penalty $R(W)$