Neural Networks: III. Loss (Part 2)

Jan Bauer

jan.bauer@dhbw-mannheim.de

14.05.19

Loss

Aim:

- Predicted scores should be consistent with training data

Intuition:

- Loss is high when doing a poor job
- Loss is low when doing a good job

$\rightsquigarrow$ interested in distance between prediction and what is in fact true

Loss

Sample average over the data loss using a loss function:

$$L(W) \equiv L \equiv \hat{\mathrm{E}} L_d \equiv \frac{1}{D} \sum_{d=1}^{D} L_d(y_{[d]}, X, W)$$

Loss Functions

Distance? Just use a norm!

- $\mathcal{L}_1$ norm

$$L_d = \left|\left|y_{[d]} - y^*_{[d]}\right|\right|_1 \equiv \sum_k \left|y^{(k)}_{[d]} - y^{*(k)}_{[d]}\right|$$

- $\mathcal{L}_2$ norm

$$L_d = \left|\left|y_{[d]} - y^*_{[d]}\right|\right|_2^2 \equiv \sum_k \left(y^{(k)}_{[d]} - y^{*(k)}_{[d]}\right)^2$$

Foreshadowing

- Why is squaring the norm possible?
- Why do we even square?

Foreshadowing

- Why is squaring the norm possible? $\rightarrow$ Squaring is a monotone operation
- Why do we even square? $\rightarrow$ Gradient becomes much simpler

Loss Functions

- Multiclass Support Vector Machine loss (**SVM loss**)

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

  - we want the model to perform better than by a margin of Δ
  - in pratice: $\Delta = 1$ (will be clear in Part 4 - Regularization)
  - squared loss might perform better, i.e. $\sum \max(0, \cdot)^2$

Exercise Time

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

- Example: $y = (12, 8, 13, 5)'$, $\Delta = 2$, $k^* = 1$
- Example: $y = (12, 9, 10, 2)'$, $\Delta = 4$, $k^* = 1$
- Example: $y = (12, 8, 13, 5)'$, $\Delta = 2$, $k^* = 1$ with squared loss
- Example: $y = (12, 9, 10, 2)'$, $\Delta = 4$, $k^* = 1$ with squared loss

Loss Functions

Cross-Entropy loss

$$L_d \equiv -\log \frac{\exp \phi(y_{[d]}^{(k^*)})}{\sum_k \exp y_{[d]}^{(k)}} = -\phi(y_{[d]}^{(k^*)}) + \log \sum_k \exp y_{[d]}^{(k)}$$

- natural counterpart to the softmax activation function

Cross-Entropy Intuition

- measures the difference between the "true" probability distribution p and its estimated counterpart q
- discrete case: $H(p, q) = -\sum_x p(x) \log q(x)$

Cross-Entropy Intuition

Loss: Set $q(y_{[d]}^{(\hat{k})}) = \frac{\exp \phi(y_{[d]}^{(\hat{k})})}{\sum_k \exp y_{[d]}^{(k)}}$ and $p(y_{[d]}^{(k)})$ s.t. $p(y_{[d]}^{(k^*)}) = 1$ and $p(y_{[d]}^{(k)}) = 0$ for $k \neq k^*$. I.e. consider $p(x)$ to be a distribution with all its mass on the correct class. Then

$$
\begin{aligned}
H(p,q) &= -\sum_k p(y_{[d]}^{(k)}) \log q(y_{[d]}^{(k)}) \\
&= -\sum_{k \neq k^*} \underbrace{p(y_{[d]}^{(k)})}_{=0} \log q(y_{[d]}^{(k)}) - p(y_{[d]}^{(k^*)}) \log q(y_{[d]}^{(k^*)}) \\
&\equiv L_d
\end{aligned}
$$