Softmax with temperature from the paper
" Distilling the Knowledge in a Neural Network "

Neural networks typically produce class probabilities by using a "softmax" output layer that converts the logit, z_i, computed for each class into a probability, q_i, by comparing z_i with the other logits.

$$\boxed{q_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)}} \quad (1)$$

2

where T is a temperature that is normally set to 1. Using a higher value for T produces a softer probability distribution over classes.

* Key Concept

Model learns probability distribution

Loss measures difference between model distribution and data distribution

Optimizer trains model to decrease loss

Details about "Loss"

Training dataset $D = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$

Key assumption : independent and identically distributed (i.i.d.)

$$(x_i, y_i) \perp\!\!\!\perp (x_j, y_j) \qquad (x_i, y_i) \sim p(x, y)$$

$$\Rightarrow p(D) = \prod p(x_i, y_i)$$
$$= \prod p(x_i)\, p(y_i \mid x_i)$$

$$p(D) = \prod p(x_i)\, p_\theta(y_i \mid x_i)$$ model of true $p(y_i \mid x_i)$

choose θ such that $p(D)$ is maximized

$$\log p(D) = \sum \cancel{\log} p(x_i) + \log p_\theta(y_i \mid x_i)$$

$$\theta^* \leftarrow \operatorname*{argmax}_\theta \sum \log p_\theta(y_i \mid x_i)$$ maximum likelihood estimation (MLE)

$$\theta^* \leftarrow \operatorname*{argmin}_\theta - \sum \log p_\theta(y_i \mid x_i)$$ negative log-likelihood (NLL)

our loss function

Information of an event E

$$I(E) = -\log_2 p(E)$$

Entropy of random variable X

$$H(X) = E(I(X))$$
$$= -\sum p(x_i) \log p(x_i)$$

Cross entropy of the distribution q relative to a distribution p

$$H(p, q) = -E_p(\log q)$$
$$= -\sum p(x) \log q(x)$$

KL Divergence (Kullback - Leibler divergence)

relative entropy

statistical distance measuring how probability distribution Q (model) is different from reference probability distribution P (data)

$$D_{KL}(P \| Q) = \sum p(x) \log \frac{p(x)}{q(x)}$$
$$= -\sum p(x) \log \frac{q(x)}{p(x)}$$

JS Divergence (Jensen - Shannon divergence)

$$JSD(P \| Q) = \frac{1}{2} D_{KL}\left(P \| \frac{P+Q}{2}\right) + \frac{1}{2} D_{KL}\left(Q \| \frac{P+Q}{2}\right)$$

Summary

for 1 data point (x_i, y_i)

- cross entropy : $-\sum_y p(y|x_i) \log p_\theta(y|x_i)$
- negative log-likelihood : $-\log p_\theta(y_i|x_i)$
- KL divergence : $-\sum_y p(y|x_i) \log \frac{p_\theta(y|x_i)}{p(y|x_i)}$
- binary cross entropy : $-p(y|x_i) \log p_\theta(y|x_i) - (1-p(y|x_i)) \log(1-p_\theta(y|x_i))$

KL Divergence $\longleftrightarrow$ Cross Entropy

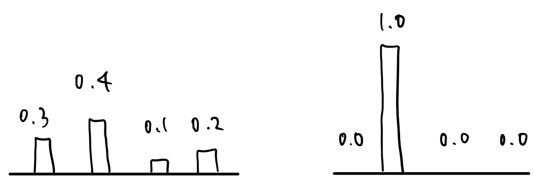
$$-\sum p(x) \log q(x) + \cancel{\sum p(x) \log p(x)} \qquad\qquad -\sum p(x) \log q(x)$$

Cross Entropy $\longleftrightarrow$ NLL

Consider only (x_i, y_i) $\rightarrow$ model output is probability distribution

$$H(p, p_\theta) = -\sum_y p(y|x_i) \log p_\theta(y|x_i) \qquad\qquad -\log p_\theta(y_i|x_i)$$

if $p(y|x_i)$ is one-hot encoded ...



model $\qquad\qquad$ data

$p_\theta(y|x_i) \qquad\qquad p(y|x_i)$

$$H(p, p_\theta) = -\sum_y p(y|x_i) \log p_\theta(y|x_i)$$

$$= -p(y_1|x_i) \log p_\theta(y_1|x_i) - p(y_2|x_i) \log p_\theta(y_2|x_i)$$

$$\cdots \quad - p(y_n|x_i) \log p_\theta(y_n|x_i)$$

$$= -p(y_i|x_i) \log p_\theta(y_k|x_i)$$

$$= -\log p_\theta(y_k|x_i)$$

$$= \text{NLL}$$