Assignment 1. MLPs, CNNs and Backpropagation

Maurice Frank
11650656
maurice.frank@posteo.de
Code: github

1 MLP backprop

1.1

1.1.a

$$
\begin{aligned}
\frac{\partial \boldsymbol{L}}{\partial x_i^{(N)}} &= -\frac{\partial}{\partial x_i^{(N)}} \sum_i t_i \log x_i^{(N)} && (\frac{\partial \boldsymbol{x}^{(N)}}{\partial \boldsymbol{L}}) \\
&= -t_i \cdot \frac{1}{x_i^{(N)}} \\
&\Leftrightarrow \\
\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{x}^{(N)}} &= -[\dots \frac{t_i}{x_i^{(N)}} \dots] \\
&\in \mathbb{R}^{d_N}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial x_i^{(N)}}{\partial \tilde{x_j}^{(N)}} &= \frac{\partial}{\partial \tilde{x}_j^{(N)}} \frac{\exp \tilde{x}_i^{(N)}}{\sum_k \exp \tilde{\boldsymbol{x}}_k^{(N)}} && (\frac{\partial \boldsymbol{x}^{(N)}}{\partial \tilde{\boldsymbol{x}}^{(N)}}) \\
&= \frac{(\frac{\partial}{\partial \tilde{x}_j^{(N)}} \exp \tilde{x}_i^{(N)}) \cdot \sum_k \exp \tilde{x}_k^{(N)} - \exp \tilde{x}_i^{(N)} \cdot \frac{\partial}{\partial \tilde{x}_j^{(N)}} \sum_k \exp \tilde{x}_k^{(N)}}{(\sum_k \exp \tilde{x}_k^{(N)})^2} \\
&= \frac{\delta_{ij} \exp \tilde{x}_j^{(N)}}{\sum_k \exp \tilde{x}_k^{(N)}} - \frac{\exp \tilde{x}_i^{(N)} \cdot \exp \tilde{x}_j^{(N)}}{(\sum_k \exp \tilde{x}_k^{(N)})^2} \\
&= \text{softmax}(\tilde{x}_j^{(N)}) \cdot (\delta_{ij} - \text{softmax}(\tilde{x}_i^{(N)})) \\
&\Rightarrow \\
\frac{\partial \boldsymbol{x}^{(N)}}{\partial \tilde{\boldsymbol{x}}^{(N)}} &= \begin{bmatrix} & \vdots & \\ \dots & \text{softmax}(\tilde{x}_j^{(N)}) \cdot (\delta_{ij} - \text{softmax}(\tilde{x}_i^{(N)})) & \dots \\ & \vdots & \end{bmatrix} \\
&\in \mathbb{R}^{d_N \times d_N}
\end{aligned}
$$

$$\frac{\partial \boldsymbol{x}^{(l<N)}}{\partial \tilde{\boldsymbol{x}}^{(l<N)}} = \frac{\partial}{\partial \tilde{\boldsymbol{x}}^{(l<N)}} \max(0, \tilde{\boldsymbol{x}}^{(l<N)}) \qquad (\frac{\partial \boldsymbol{x}^{(l<N)}}{\partial \tilde{\boldsymbol{x}}^{(l<N)}})$$
$$= \boldsymbol{x}^{(l<N)} \oslash \tilde{\boldsymbol{x}}^{(l<N)}$$
$$\in \mathbb{R}^{d_l}$$

$$\frac{\partial \tilde{\boldsymbol{x}}^{(l)}}{\partial \boldsymbol{x}^{(l-1)}} = \frac{\partial}{\partial \boldsymbol{x}^{(l-1)}} \boldsymbol{W}^{(l)} \boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)} \qquad (\frac{\partial \tilde{\boldsymbol{x}}^{(l)}}{\partial \boldsymbol{x}^{(l-1)}})$$
$$= \boldsymbol{W}^{(l)}$$
$$\in \mathbb{R}^{d_l \times d_{l-1}}$$

$$\frac{\partial \tilde{\boldsymbol{x}}^{(l)}}{\partial \boldsymbol{W}^{(l)}} = \frac{\partial}{\partial \boldsymbol{W}^{(l)}} \boldsymbol{W}^{(l)} \boldsymbol{x}^{(l-1)} \qquad (\frac{\partial \tilde{\boldsymbol{x}}^{(l)}}{\partial \boldsymbol{W}^{(l)}})$$
$$= \begin{bmatrix} \vdots \\ \frac{\partial \tilde{\boldsymbol{x}}_i^{(l)}}{\partial \boldsymbol{W}^{(l)}} \\ \vdots \end{bmatrix}$$
$$\in \mathbb{R}^{d_l \times (d_l \times d_{l-1})}$$
with
$$\frac{\partial \tilde{\boldsymbol{x}}_i^{(l)}}{\partial \boldsymbol{W}^{(l)}} = \begin{bmatrix} \vdots \\ \boldsymbol{x}^{(l-1)^T} \\ \vdots \end{bmatrix}$$
$$\in \mathbb{R}^{d_l \times d_{l-1}}$$

$$\frac{\partial \tilde{\boldsymbol{x}}^{(l)}}{\partial \boldsymbol{b}^{(l)}} = \frac{\partial}{\partial \boldsymbol{b}^{(l)}} \boldsymbol{b}^{(l)} \qquad (\frac{\partial \tilde{\boldsymbol{x}}^{(l)}}{\partial \boldsymbol{b}^{(l)}})$$
$$= \boldsymbol{b}^{(l)} \otimes \boldsymbol{b}^{(l)}$$
$$\in \mathbb{R}^{?????}$$

Note the use of $\oslash$ for element-wise division and the use of δ for the Kronecker-Delta.

1.1.b

$$\frac{\partial \boldsymbol{L}}{\partial \tilde{\boldsymbol{x}}^{(N)}} = \frac{\partial \boldsymbol{L}}{\partial \boldsymbol{x}^{(N)}} \frac{\partial \boldsymbol{x}^{(N)}}{\partial \tilde{\boldsymbol{x}}^{(N)}} \qquad (\frac{\partial \boldsymbol{L}}{\partial \tilde{\boldsymbol{x}}^{(N)}})$$
$$= \frac{\partial \boldsymbol{L}}{\partial \boldsymbol{x}^{(N)}} \cdot \begin{bmatrix} & & \vdots & \\ \dots & \text{softmax}(\tilde{x}_j^{(N)}) \cdot (\delta_{ij} - \text{softmax}(\tilde{x}_i^{(N)})) & \dots \\ & & \vdots & \end{bmatrix}$$

$$\frac{\partial \boldsymbol{L}}{\partial \tilde{\boldsymbol{x}}^{(l<N)}} = \frac{\partial \boldsymbol{L}}{\partial \tilde{\boldsymbol{x}}^{(l)}} \frac{\partial \boldsymbol{x}^{(l)}}{\partial \tilde{\boldsymbol{x}}^{(l)}} \qquad (\frac{\partial \boldsymbol{L}}{\partial \tilde{\boldsymbol{x}}^{(l<N)}})$$
$$= \frac{\partial \boldsymbol{L}}{\partial \tilde{\boldsymbol{x}}^{(l)}} \cdot \boldsymbol{x}^{(l)} \oslash \tilde{\boldsymbol{x}}^{(l)}$$

Layers	LR	Batch size	Test Accuracy	Test Loss
100	2e-3	200	46.86	1.51
10	2e-3	200	39.38	1.68
200	1e-3	200	**50.37**	**1.42**
80,50	1e-5	200	11.17	2.30

Table 1: Results for the NumPy MLP under different parameter settings. Layers shows the sorted list of number of neurons for the hidden layers. LR is learning rate. Reported are the highest accuracy and lowest loss on the test set during training.

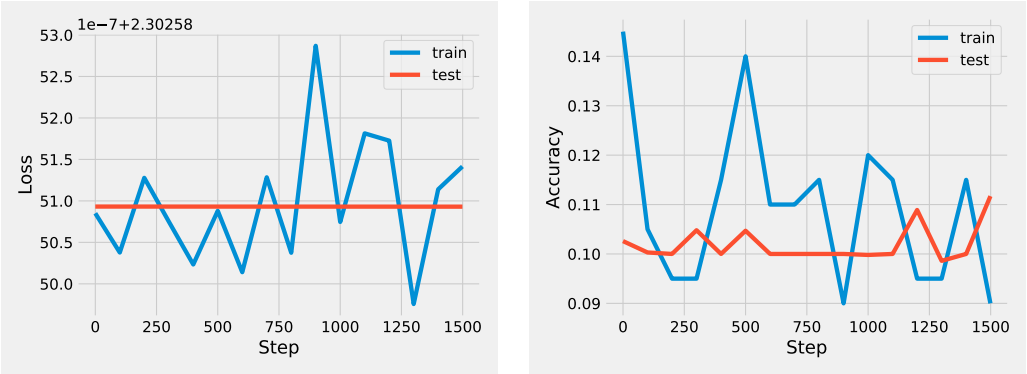

Figure 1: **Left** the loss and **right** the accuracy during training of the NumPy MLP implementation.

$$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{x}^{(l<N)}} = \frac{\partial \boldsymbol{L}}{\partial \tilde{\boldsymbol{x}}^{(l+1)}} \frac{\partial \tilde{\boldsymbol{x}}^{(l+1)}}{\partial \boldsymbol{x}^{(l)}} = \frac{\partial \boldsymbol{L}}{\partial \tilde{\boldsymbol{x}}^{(l+1)}} \cdot \boldsymbol{W}^{(l+1)} \quad (\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{x}^{(l<N)}})$$

$$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{W}^{(l)}} = \frac{\partial \boldsymbol{L}}{\partial \tilde{\boldsymbol{x}}^{(l)}} \frac{\partial \tilde{\boldsymbol{x}}^{(l)}}{\partial \boldsymbol{W}^{(l)}} \quad (\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{W}^{(l)}})$$

$$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{b}^{(l)}} = \frac{\partial \boldsymbol{L}}{\partial \tilde{\boldsymbol{x}}^{(l)}} \frac{\partial \tilde{\boldsymbol{x}}^{(l)}}{\partial \boldsymbol{b}^{(l)}} \quad (\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{b}^{(l)}})$$

1.2 NumPy MLP

The performance of the NumPy implementation of the MLP under differnt hyperparameter settings are show in Tabel 1. The loss and accuracy curves of the best model are shown in Figure 1. Source code is in files `mlp_numpy.py`, `train_mlp_numpy.py` and `modules.py`.

2 PyTorch MLP

The performance of the PyTorch MLP model under diffent settings are shown in Table 2. The loss and accuracy curves during training of the best performing model are shown in Figure 2. Source code is in files `mlp_pytorch.py` and `train_mlp_pytorch.py`.

Layers	LR	Batch size	Test Accuracy	Test Loss
100	2e-3	200	45.58	1.61
10	2e-3	200	39.52	1.70
200	1e-3	200	**48.14**	**1.53**
80,50	1e-5	200	31.08	2.26
300	1e-5	200	33.27	4.25

Table 2: Results of the PyTorch MLP using differnt settings for the hyperparameters.

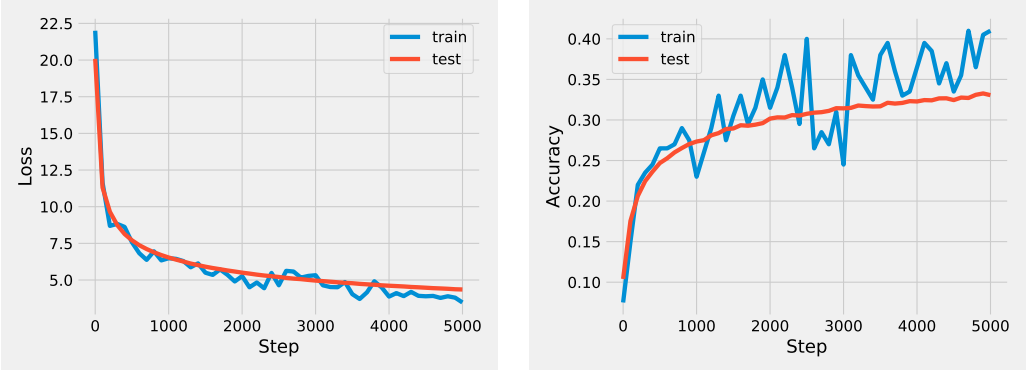

Figure 2: **Left** the loss and **right** the accuracy during training of the PyTorch MLP implementation.