Basics on deep learning for vision

Charles Vin, Aymeric Delefosse

S1-2023

1 Introduction to neural networks

1.1 Theorical Foundation

1.1.1 Supervised dataset

1. What are the train, val and test sets used for ? The train dataset is used to train the model. The test dataset is used to test the model on data it has nether seen before. Finaly the validation set is a separate portion of the dataset used to fine-tune and optimize the model's hyperparameters.

2. What is the influence of the number of exemples N ? A large the number of example can help the model to generalize more and be more robust to noise or outlier. A small number of example can prone to overfitting. Increasing N can also increase the computational complexity of training the model.

1.1.2 Network architecture

3. Why is it important to add activation functions between linear transformations? Otherwise we just sum linear functions so it stays linear. So activation functions introduce non-linearity to the network which permit the model to capture and learn more complex patern than linear.

4. What are the sizes n_x, n_h, n_y in the figure 1? In practice, how are these sizes chosen?

- $n_x = 2$ is the size of the input, the dimension of our data.
- $n_h = 4$ is the size of the hidden layer. It chosen proportionaly to the conplexity of the feature we want to develop in the hiden layer. A large size can lead to overfitting
- $n_y = 2$ is the size of the output, it's choosen in function of the number of class of y

5. What do the vectors $\hat{y}$ and y represent? What is the difference between these two quantities? $y \in \{0, 1\}$ is the ground truth while $\hat{y} \in [0, 1]$ is like a probabilty for each class. $\hat{y}$ express the model's confidence in each class prediction.

6. Why use a $SoftMax$ function as the output activation function? $\tilde{y} \in \mathbb{R}$ so we have to transform it into a probability distribution. There is many way to do that but the $SoftMax$ is commonly used in multi-class classification problems.

7. Write the mathematical equations allowing to perform the *forward* pass of the neural network, i.e. allowing to successively produce $\hat{h}$, h, $\tilde{y}$, $\hat{y}$, starting at x. Let note W_i, b_i the parameter for the i layer, $f_i(x) = W_i x + b_i$ and $g_i(x)$ the activation function of the layer i.

$$
\begin{aligned}
\tilde{h} &= f_0(x) \\
h &= g_0(\tilde{h}) \\
\tilde{y} &= f_1(h) \\
\hat{y} &= g_1(\tilde{y})
\end{aligned}
$$

1.2 Loss function

During training, we try to minimize the loss function. For cross entropy and squared error, how must the $\hat{y}_i$ vary to decrease the global loss function $\mathcal{L}$? $?$

How are these functions better suited to classification or regression tasks?