APM 598: Homework 2 (03/03)

1 Two-layers neural networks

Ex 1.

We consider two-layers neural networks of the form (see fig. 1):

$$f(\mathbf{x}) = \mathbf{b}_2 + W_2\Big(\sigma(\mathbf{b}_1 + W_1 \cdot \mathbf{x})\Big), \tag{1}$$

where $\mathbf{x}$, $\mathbf{b}_1$, $\mathbf{b}_2 \in \mathbb{R}^2$ and $W_1, W_2 \in \mathcal{M}_{2\times 2}(\mathbb{R})$ are matrices (2×2). The activation function σ is the ReLu function (i.e. $\sigma(x) = \max(x, 0)$). We denote by $\mathbf{s} = f(\mathbf{x})$ the *score* predicted by the model with $\mathbf{s} = (s_1, s_2)$ where s_1 is the score for class 1 and s_2 the score for class 2.

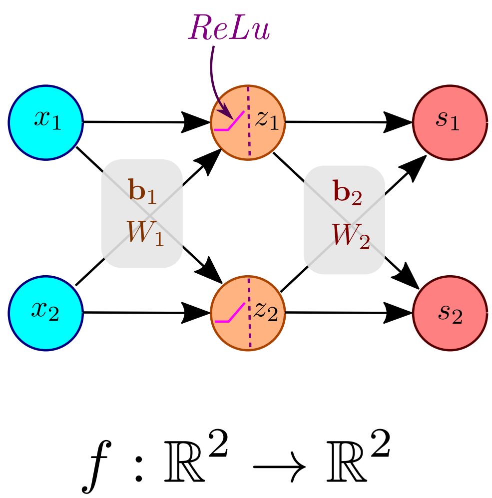

Figure 1: Illustration of a two-layer neural network using ReLu activation function.

a) Consider the points given in figure 2-left where each color correspond to a different class:

$$\begin{aligned} &\text{class 1:} \qquad \mathbf{x}_1 = (1,0) \text{ and } \mathbf{x}_2 = (-1,0), \\ &\text{class 2:} \qquad \mathbf{x}_3 = (0,1) \text{ and } \mathbf{x}_4 = (0,-1). \end{aligned}$$

Find (numerically or analytically) some parameters $\mathbf{b}_1$, $\mathbf{b}_2$, W_1 and W_2 such that the scores $\mathbf{s}$ satisfy:

$$s_1 > s_2 \text{ for } \mathbf{x}_1 \text{ and } \mathbf{x}_2 \;, \quad s_1 < s_2 \text{ for } \mathbf{x}_3 \text{ and } \mathbf{x}_4.$$

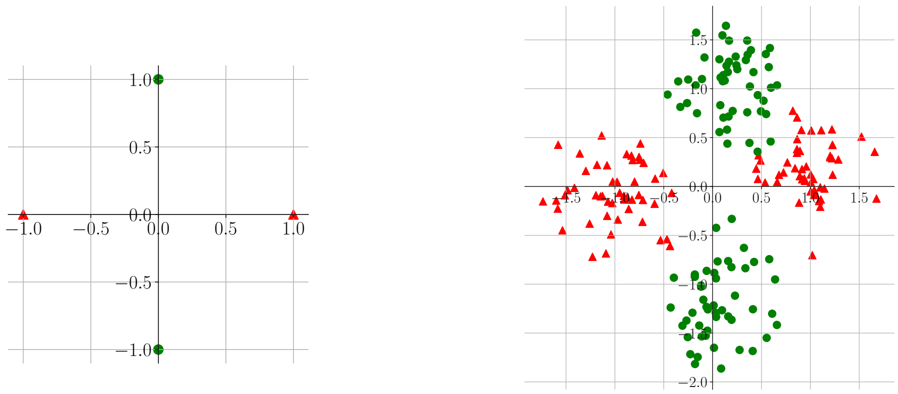

Figure 2: Data points to classify.

b) Consider now the data-set given in figure 2-right (see code below to load the data). Train a two-layer neural network of the form (1) to classify the points. Provide the accuracy of the model (percentage of correctly predicted labels).

```
##################################
##          Exercise 1          ##
##################################
import numpy as np
import pandas as pd
df = pd.read_csv('data_HW2_ex1.csv')
X = np.column_stack((df['x1'].values,df['x2'].values))
y = df['class'].values
```

Ex 2.

The goal of this exercise is to show that two-layers neural networks with ReLu activation can approximate any continuous functions. To simplify, we restrict our attention to the one-dimensional case and fix a continuous function:

$$g : [0,1] \longrightarrow \mathbb{R}.$$

We claim that for any $\varepsilon > 0$, there exists f_θ two-layers neural network such that:

$$\max_{x \in [0,1]} |g(x) - f_\theta(x)| < \varepsilon. \tag{2}$$

The key idea is to show that f_θ can interpolate (exactly) g at as many points as needed (see figure 3-right).

We consider neural networks f_θ of the form:

$$f_\theta(x) = \mathbf{W}^{(2)}\Big(\sigma(\mathbf{W}^{(1)}x + \mathbf{b}^{(1)})\Big) + b^{(2)} = \sum_{k=1}^{m} w_k^{(2)}\sigma(w_k^{(1)}x + b_k^{(1)}) \;+\; b^{(2)} \tag{3}$$

where m is the size of hidden layer, the unknown parameters θ are the two weight matrices $\mathbf{W}^{(1)} = \{w_k^{(1)}\}_{k=1:m}$, $\mathbf{W}^{(2)} = \{w_k^{(2)}\}_{k=1:m}$ and the two bias $\mathbf{b}^{(1)} = \{b_k^{(1)}\}_{k=1:m}$ and $b^{(2)}$. The activation function σ is taken as the ReLu function. The hidden layer is intended to have a large dimension, i.e. the intermediate value $\mathbf{z} \in \mathbb{R}^m$ with $m \gg 1$ (see figure 3-left).

a) Consider three points: (x_0, y_0), (x_1, y_1), (x_2, y_2) with $x_0 = 0$, $x_1 = \frac{1}{2}$, $x_2 = 1$ (the values y_i are arbitrary). Find f_θ such that $f_\theta(x_i) = y_i$ for $i = 0, 1, 2$.
Hint. Use $m = 2$ with the functions $\sigma(x - x_0)$ and $\sigma(x - x_1)$

b) Generalize: write a program that take as inputs $\{(x_i, y_i)\}_{0 \le i \le N}$ with $x_i < x_{i+1}$ and return a two layers n.n. such that $f_\theta(x_i) = y_i$ for all $i = 0 \dots N$.
Hint. Use $m = N$ and the functions $\sigma(x - x_i)$.

Extra) Prove (2).
Hint: use that g is uniformly continuous on $[0, 1]$.

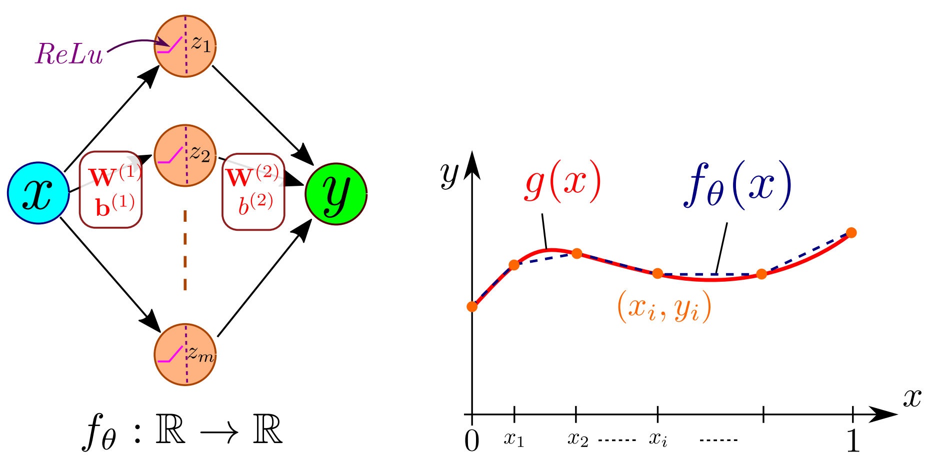

Figure 3: **Left**: two layers neural network used to approximate continuous function. The *hidden* layer (i.e. $\mathbf{z} = (z_1, \dots, z_m)$) is in general quite large. **Right:** to approximate the continuous function g, we interpolate some of its values (x_i, y_i) by a piece-wise linear function using the functions $\sigma(x - x_i)$.

2 Convolutional Neural Networks

Ex 3.

Using convolutional layers, max pooling and ReLu activation functions, build a classifier for the Fashion-MNIST database (see a sketch example in figure 4) with a neural network with at most 500 parameters.

Provide the evolution of the loss for the training and test sets. Give the accuracy on both sets after the training.

The three groups with the highest accuracy get additional points.

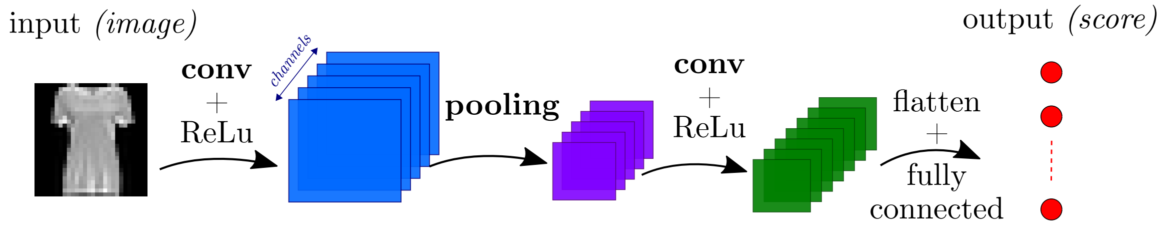

Figure 4: Schematic representation of a neural network for image classification.