Exercise Sheet #8

Due: 19.12.22 11h00

Roman Remme

General Regulations.

- Please hand in your solutions in groups of three people. A mix of attendees from Monday and Tuesday tutorials is fine.
- Your solutions to theoretical exercises can be either handwritten notes (scanned), or typeset using LaTeX.
- For the practical exercises, the data and a skeleton for your jupyter notebook are available at `https://github.com/hci-unihd/mlph_sheet08`. Always provide the (commented) code as well as the output, and don't forget to explain/interpret the latter. Please hand in both the notebook (`.ipynb`), as well as an exported pdf.
- Submit all your files in the Übungsgruppenverwaltung, only once for your group of three.

1 Projection Trick

The data in `data2d.npy`, `labels.npy` is a 2D classification problem.

(a) Can this problem be solved with a linear decision boundary in 2D? Apply logistic regression (as implemented in scikit-learn, `sklearn.linear_model.LogisticRegression`), and visualize the decision boundary of the classifier in a scatter plot of the data. Which accuracy do you get? (2 pts)

(b) Come up with a nonlinear transformation to enhance the feature space, such that the classes can be linearly separated. Visualize the points in the new, 3D feature space[1]. Demonstrate that the problem can be solved in the enhanced space using logistic regression. (3 pts)

(c) Describe (in words) how this problem could be solved by an MLP with a single hidden layer. (2 pts)

2 Training of an MLP

In this exercise, we use `http://playground.tensorflow.org` to gain some intuition on what happens throughout the training of a neural network and how the different pieces interact and work together. For each part submit a screenshot and a short discussion of what you observed.

(a) Fitting a Neural Net. Consider the spiral data set and the first two features (X_1, X_2). Come up with an architecture that can learn to classify the pattern well. You are free to use any number layers/neurons/activation functions/regularization/... (2 pts)

(b) Exploring Regularization. Pick the largest network size (i.e. 6 layers of 8 neurons each) and one of the data sets. Train it first without regularization, observing the behavior. Retrain it with L1 and L2 regularization and observe how the weight structure changes. What kind of behavior do you expect and does it fit with what you observe? (2 pts)

(c) Breaking Things. As discussed in the lecture, a net with enough parameters can fit any kind of pattern, even if there is none. Try to replicate this observation. Have your net learn a pattern "perfectly" (i.e. very low training error), but without having predictive power (i.e. the test error stays larger than

[1] You could for example use `https://matplotlib.org/stable/gallery/mplot3d/scatter3d.html`.

random (which would be 0.5)). *Hint: If you consider the spiral data set with the minimal amount of training data and the maximum amount of noise, you get a collection of points with most structure removed.* (2 pts)

3 Reverse Mode Automatic Differentiation

As discussed in the lecture we can describe the flow of information through a "standard" neural network as falling into two phases. First we propagate information forward through the network from the input layer to the output (sometimes referred to as *forward propagation*). The second phase then consists of backpropagating the error we received from the scalar loss/error/cost function that we try to optimize. The function we consider in this exercise is

$$y(\mathbf{x}) = \left(\sin\frac{x_1}{x_2} + \frac{x_1}{x_2} - \exp(x_2)\right) \cdot \left(\frac{x_1}{x_2} - \exp(x_2)\right),$$

evaluated at $\mathbf{x} = (1.5, 0.5)$.

(a) Give the computational graph of the function. (2 pts)

(b) Give the forward trace at the given $\mathbf{x}$. (2 pts)

(c) Give the backward/reverse trace. (3 pts)

4 Bonus: Number of linear regions in a deep network

Show that the maximum number of linear regions in a ReLU network scales (at least) exponentially with the depth, by explicitly constructing a sequence of ever deeper networks that demonstrate this behavior. Hint: First do it using the absolute value as the activation function, then modify the resulting networks to use ReLU while not changing the functions they describe. (5 pts)