Introduction to Deep Learning (I2DL)

Exercise 5: Neural Networks and CIFAR10 Classification
Today's Outline

• Neural Networks
  – Mathematical Motivation
  – Modularization

• Exercise 5
  – Implementation Loop
  – CIFAR10 Classification

• Feedback
Our Goal

$f \rightarrow \$ \text{?}$

I2DL: Prof. Niessner, Prof. Leal-Taixé
Universal Approximation Theorem

Theorem (1989, colloquial)
For any continuous function $f$ on a compact set $K$, there exists a one layer neural network, having only a single hidden layer + sigmoid, which uniformly approximates $f$ to within an arbitrary $\varepsilon > 0$ on $K$. 

[Diagram of a neural network with labels $w = 100$, $b = -40$, and output from top hidden neuron showing $-b/w = 0.40$.]
Universal Approximation Theorem
(Optional)

Readable proof:
https://mcneela.github.io/machine_learning/2017/03/21/Universal-Approximation-Theorem.html
(Background: Functional Analysis, Math Major 3rd semester)

Visual proof:
A word of warning…

Shallow VS Deep

• Shallow
  (1 hidden layer)

• Deep
  (>1 hidden layer)
Obvious Questions

• Q: Do we even need deep networks?
  A: Yes. Multiple layers allow for more representation power given a fixed computational budget in comparison to a single layer.

• Q: So we just build 100 layer deep networks?
  A: Not trivially ;-) Constraints: Memory, vanishing gradients, …
class Classifier(Networke):
    
    Classifier of the form $y = \text{sigmoid}(X \ast W)$
    
    def __init__(self, num_features=2):
        super(Classifier, self).__init__("classifier")
        self.num_features = num_features
        self.W = None

    def initialize_weights(self, weights=None):
        
        Initialize the weight matrix $W$

        :param weights: optional weights for initialization
        :type weights: numpy.ndarray

        if weights is not None:
            assert weights.shape == (self.num_features + 1, 1), 
            "weights for initialization are not in the correct shape"
            self.W = weights
        else:
            self.W = 0.001 * np.random.randn(self.num_features + 1, 1)

    def forward(self, X):
        
        Performs the forward pass of the model.

        :param X: N x D array of training data. Each row is a D-dimensional point.
        :returns: Predicted labels for the data in X, shape N x 1
                  1-dimensional array of length N with classification scores.
        
        assert self.W is not None, "weight matrix W is not initialized"
        # add a column of 1s to the data for the bias term
        batch_size, _ = X.shape
        X = np.concatenate((X, np.ones((batch_size, 1))), axis=1)
        # save the samples for the backward pass
        self.cache = X
        # output variable
        y = None

        y = X.dot(self.W)
        y = self.sigmoid(y)

        # END OF YOUR CODE
Modularization

Chain Rule:
\[
\frac{\partial f}{\partial y} = \frac{\partial f}{\partial d} \cdot \frac{\partial d}{\partial y}
\]

```python
class Sigmoid:
    def __init__(self):
        pass

    def forward(self, x):
        
        :param x: Inputs, of any shape

        :return out: Output, of the same shape as x
        :return cache: Cache, for backward computation, of the same shape as x

    def backward(self, dout, cache):
        
        :return: dx: the gradient w.r.t. input X, of the same shape as X
```

I2DL: Prof. Niessner, Prof. Leal-Taixé
Exercise 3: Dataset

- Data
  - Dataset
  - Dataloader
- Model
  - Network
  - Loss/Objective
- Solver
  - Optimizer
  - Training Loop
  - Validation
Exercise 4: Binary Classification

Data
- Dataset
- Dataloader

Model
- Network
- Loss/Objective

Solver
- Optimizer
- Training Loop
- Validation
Exercise 5: Neural Networks and CIFAR10 Classification
Exercise 6: Neural Networks and Hyperparameter Tuning

Data
- Dataset
- Dataloader

Model
- Network
- Loss/Objective

Solver
- Optimizer
- Training Loop
- Validation
Feedback: Google Forms

- Exercise 3: 
  https://docs.google.com/forms/d/e/1FAIpQLScqEBS-w_UoULQWlY3sYqAPF7vna3o0RvFq6eWIKlwseDpAXg/viewform

- Exercise 4: 
  https://docs.google.com/forms/d/e/1FAIpQLSdQ1MGokyD-aaALcvUBLPYFrWbQL7akP-ZoOv7awDnciqbiOw/viewform

- Exercise 5: 
  https://docs.google.com/forms/d/e/1FAIpQLSf7Vjw_a0s-Z1BQvdEAkDtNANc3GfxwoTsJi2\WiQissYPDchw/viewform
See you next week 😊