Chapter 7 Neural Networks and Neural Language Models

A modern neural network: a network of small computing units,each of which takes a vector of input values and produces a single output value. In this chapter we use them to classification.

feedforward network: computation proceeds iteratively from one layer of units to the next

deep learning: the use of modern neural nets

compared with logistic regression:

more powerful classifier, minimal neural network

logistic regression: apply the regression classifier to many tasks by developing many rich kinds of feature templates based on domain knowledge

neural networks: build neural networks that take raw words as inputs and learn to induc features as part of the process of learning to classify.

Units: the building block

- **bias term**: taking a weighted sum of its inputs, with one additional term in the sum, $z = b + \Sigma_i w_i x_i$
- **activation**: z non-linear function f to z

three popular non-linear function f() below

- **sigmoid function**: $y = \sigma(z) = \frac{1}{1+e^{-z}}$

  map the output in the range [0,1], differentiable

  the output of a neural unit: $y = \sigma(w \cdot x + b) = \frac{1}{1+e^{-w \cdot x + b}}$
- **tanh function**: $y = \frac{e^z - e^{-z}}{e^z + e^{-z}}$

  map the output in range [-1,+1]
- **ReLU**: $y = max(x, 0)$

In the sigmoid or tanh function,very high values of z result in values of y that are **saturated**.By contrast, the tanh function has the nice properties of being smoothly differentiable and mapping outlier values toward the mean.

The XOR problem

- **perceptron**: a very simple neural unit that has a binary output and does **not** have a non-linear activation function.

  The output is 0 or 1

  $$y = \begin{cases} 0, & if\ w \cdot x + b \leq 0 \\ 1, & if\ w \cdot x + b > 0 \end{cases}$$

  A perceptron is a linear classifier

The XOR problem is not a **linearly separable** function

Feed-Forward Neural Networks

A feedforward network is a multilayer network in which the units are connected with no cycles; the outputs from units in each layer are passed to units in the next higher layer, and no outputs are passed back to lower layers.

In the standard architecture, each layer is **fully-connected**, meaning that each unit in each layer takes as input the outputs from all the units in the previous layer, and there is a link between every pair of units from two adjacent layers

The weight matrix is multiplied by its input vector h to produce the intermediate output z: $z = Uh$

Training Neural Nets

The goal of training procedure: learn parameters $W^{[i]}$ and $b^{[i]}$ for each layer i

Loss function

- First, we'll need a **loss function** that models the distance between the system output and the gold output, and it's common to use the loss function used for logistic regression, the **cross-entropy loss**.
- Second, to fifind the parameters that minimize this loss function, we'll use the **gradient descent** optimization algorithm introduced in Chapter 5.
- Third, gradient descent requires knowing the gradient of the loss function, the vector that contains the partial derivative of the loss function with respect to each of the parameters.

Computing the Gradient

- **error backpropagation**: the solution to computing tthe gradient
- **backward differentiation**: the same as a more general procedure, depends on the notion of **computation graphs**

Computation Graphs

computation graph: is a representation of the process of computing a mathematical expression, in which the computation is broken down into separate operations, each of which is modeled as a node in a graph.

forward pass to compute the result

Backward differentiation on computation graphs

The importance of the computation graph comes from the **backward pass**, which

is used to compute the derivatives that we'll need for the weight update.

Backwards differentiation makes use of the **chain rule** in calculus.

Neural Language Models

- **language modeling**: predicting upcoming words from prior word context.

Embeddings

- **pretraining**: The method of using another algorithm to learn the embedding representations we use for input words

- **one-hot vector** is a vector that has one element equal to 1—in the dimension corresponding to that word's index in the vocabulary— while all the other elements are set to zero

Training the neural language model