Your Name 你的名字
CLASS NAME 课程名/课程号
2023 年 5 月 14 日

Homework-1
深度学习部分基础理论公式推导与分析

Question 1. MLP for Classification:

In a multi-class classification problem, e.g. a problem with M classes, we use a model f_θ to produce the output $\hat{y}$. $\hat{y}$ is a vector of dimension M with sum 1, whose components indicate the probability of x belong to each class. In this section, you will use a simple model – the 3-layer MLP – to tackle this problem. Assume that we have K labeled data (x_k, y_k) with $k = 1, 2, \ldots, K$ where $x_k \in \mathbb{R}^N$ and $y_k \in \mathbb{R}^M$ is a M-dimensional one-hot vector.

a) Define the cross entropy loss for a multi-class classification problem.

b) A 3-layer MLP consists of an input layer, a hidden layer and an output layer with two learned parameter matrices W^1, W^2 between successive layers. Please note that there is generally a Softmax layer after the output layer to scale the output, which we omitted in this sub-question for simplicity. When given an input x, this model performs the following forward computations sequentially:

$$\begin{aligned} a_1 &= W^1 x, \\ h &= \sigma(a_1), \\ a_2 &= W^2 h, \\ \hat{y} &= \sigma(a_2). \end{aligned}$$

where $W^1 \in \mathbb{R}^{D\times N}, W^2 \in \mathbb{R}^{M\times D}$ are parameter matrices and $\sigma(z) = \frac{1}{1+e^{-z}}$ is the element-wise Sigmoid function. Use the loss function you defined in sub-question a), apply an SGD update to the parameters W^1 and W^2 with learning rate η via back-propagation. You must identify the closed-form gradient of each parameter, which is $\frac{\partial \mathcal{L}}{\partial W^2(m,d)}$ and $\frac{\partial \mathcal{L}}{\partial W^1(d,n)}$ in your derivation.

HINT: 1. For simplicity, you can assume that the SGD program uses only one training data per batch. 2. Use formula $\sigma'(z) = \sigma(z)(1 - \sigma(z))$ for a scalar z.

a) 由题意，对于多类别分类问题，某一样本的真实标签为 $y \in \mathbb{R}^M$，预测标签为 $\hat{y} \in \mathbb{R}^M$，则交叉熵损失函数定义为：

$$\mathcal{L}(y, \hat{y}) = -\sum_{i=1}^{M} y_i \log \hat{y}_i$$

b) 对于一个输入 $x \in \mathbb{R}^N$，标签 $y \in \mathbb{R}^M$ 的训练数据，模型的前向计算如下：

$$\begin{aligned} a_1 &= W^1 x, \\ h &= \sigma(a_1), \\ a_2 &= W^2 h, \\ \hat{y} &= \sigma(a_2). \end{aligned}$$

其中 $W^1 \in \mathbb{R}^{D\times N}, W^2 \in \mathbb{R}^{M\times D}$ 是待训练的权重矩阵，$\sigma(z) = \frac{1}{1+e^{-z}}$ 是 Sigmoid 函数。

使用交叉熵损失函数，可以将模型在 K 个训练数据上的损失表示为：

$$\mathcal{L} = \frac{1}{K}\sum_{k=1}^{K}\mathcal{L}\left(y_k, \hat{y}_k\right) = -\frac{1}{K}\sum_{k=1}^{K}\sum_{i=1}^{M} y_{k,i}\log \hat{y}_{k,i}$$

对于模型参数 W^1 和 W^2，我们需要计算它们对损失函数的梯度，从而进行梯度下降更新。根据链式法则，可以将梯度计算分为两个部分，即输出层到隐藏层的梯度和隐藏层到输入层的梯度。

未待完续……

Question 2. Dropout and Regularization

Dropout is a well-known way to prevent neural networks from overfitting. In this section, you will show this regularization explicitly for linear regression. Recall that linear regression optimizes $w \in \mathbb{R}^d$ to minimize the following MSE objective:

$$\mathcal{L}(w) = \|y - Xw\|^2$$

where $y \in \mathbb{R}^n$ is the response to design matrix $X \in \mathbb{R}^{n \times d}$. One way of using dropout during training on the d-dimensional input features x_i involves keeping each feature at random with probability p (and zero out it if not kept).

a) Show that when we apply such dropout, the learning objective becomes

$$\mathcal{L}(w) = \mathbb{E}_{M \sim \text{Bernoulli}(p)} \|y - (M \odot X)w\|^2$$

where $\odot$ denotes the element-wise product and $M \in \{0, 1\}^{n \times d}$ is a random mask matrix whose element $m_{i,j}$ have *i.i.d.* Bernoulli distribution with success probability p.

b) Show that we can manipulate the dropout learning objective to a explicit regularized objective:

$$\mathcal{L}(w) = \|y - pXw\|^2 + p(1-p)\|\Gamma w\|^2$$

and define a suitable matrix Γ.

第二题的回答……

Question 3. Figure 1 shows two cipher wheels. The left one is from Jeffrey Hoffstein, et al. [1]. Write a Python 3 program that uses it to encrypt: `FOUR SCORE AND SEVEN YEARS AGO`.

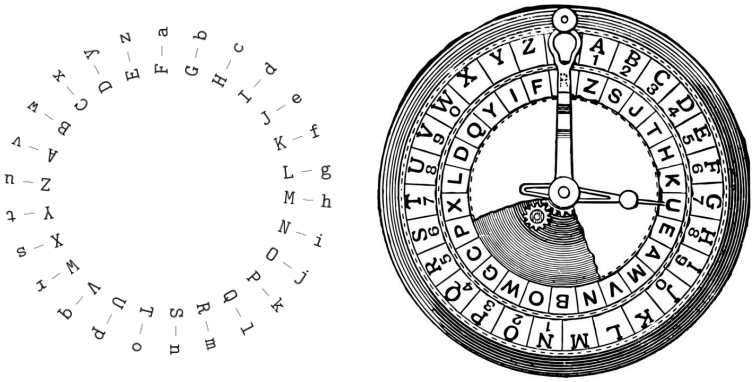

图 1. Cipher wheels.

The Python program is given in listing 1 and the encryption is given in table 1.

```
1 def encrypt(plain):
2     cipher = ''
3     for c in plain:
4         cipher = cipher+c if c==' ' else cipher+chr(((ord(c)-60) % 26)+65)
5     return cipher
6 print(encrypt("FOUR SCORE AND SEVEN YEARS AGO"))
```

Listing 1. Python 3 implementing figure 1 left wheel.

Plain Text	FOUR	SCORE	AND	SEVEN	YEARS	AGO
Cipher Text	KTZW	XHTWJ	FSI	XJAJS	DJFWX	FLT

表 1. Caesar cipher

References

[1] Jeffrey Hoffstein, Jill Pipher, Joseph H Silverman, and Joseph H Silverman. *An introduction to mathematical cryptography*, volume 1. Springer, 2008.