

Word Window Classification

交叉熵损失函数

Background: What is “cross entropy” loss/error?

- Concept of “cross entropy” is from information theory
- Let the true probability distribution be p
- Let our computed model probability be q
- The cross entropy is:

$$H(p, q) = - \sum_{c=1}^C p(c) \log q(c)$$

- Assuming a ground truth (or true or gold or target) probability distribution that is 1 at the right class and 0 everywhere else: $p = [0, \dots, 0, 1, 0, \dots, 0]$ then:
- **Because of one-hot p , the only term left is the negative log probability of the true class**

11

上图是交叉熵的公式，在逻辑回归中， $p(c)$ 就是 y 和 $1 - y$, $q(c)$ 就是 $h(x)$ 和 $1 - h(x)$ 。

Classification over a full dataset

- Cross entropy loss function over full dataset $\{x_i, y_i\}_{i=1}^N$

$$J(\theta) = \frac{1}{N} \sum_{i=1}^N -\log \left(\frac{e^{f_{y_i}}}{\sum_{c=1}^C e^{f_c}} \right)$$

- Instead of

$$f_y = f_y(x) = W_y \cdot x = \sum_{j=1}^d W_{yj} x_j$$

We will write f in matrix notation:

$$f = Wx$$



上图是真实training时的损失函数，实际情况中 $p(c)$ 为1。

Traditional ML optimization

- For general machine learning θ usually only consists of columns of W :

$$\theta = \begin{bmatrix} W_{\cdot 1} \\ \vdots \\ W_{\cdot d} \end{bmatrix} = W(:) \in \mathbb{R}^{Cd}$$

- So we only update the decision boundary via

$$\nabla_{\theta} J(\theta) = \begin{bmatrix} \nabla_{W_{\cdot 1}} \\ \vdots \\ \nabla_{W_{\cdot d}} \end{bmatrix} \in \mathbb{R}^{Cd}$$



Visualizations with ConvNetJS by Karpathy

上图是隐藏层的参数矩阵 $W \in R^{Cd}$, 即输入 $X \in R^{md}$, 乘以 W , 得到 $O \in R^{mC}$ 。

Classification difference with word vectors

Classification difference with word vectors

- Commonly in NLP deep learning:
 - We learn **both** W and word vectors x
 - We learn **both** conventional parameters **and** representations
 - The word vectors re-represent one-hot vectors—move them around in an intermediate layer vector space—for easy classification with a (linear) softmax classifier via layer $x = L\theta$

$$\nabla_{\theta} J(\theta) = \begin{bmatrix} \nabla_{W_{11}} \\ \vdots \\ \nabla_{W_{d1}} \\ \nabla_{x_{aardvark}} \\ \vdots \\ \nabla_{x_{zebra}} \end{bmatrix} \in \mathbb{R}^{Cd+Vd}$$

Very large number of parameters!

16



上图是用NN分类的区别: 不仅要学习 W , 还要同时学习词向量 x 。

NN的简介

略

Named Entity Recognition (NER)

4. Named Entity Recognition (NER)

- The task: **find** and **classify** names in text, for example:

The European Commission [ORG] said on Thursday it disagreed with German [MISC] advice.

Only France [LOC] and Britain [LOC] backed Fischler [PER]'s proposal .

"What we have to be extremely careful of is how other countries are going to take Germany 's lead", Welsh National Farmers ' Union [ORG] (NFU [ORG]) chairman John Lloyd Jones [PER] said on BBC [ORG] radio .

- Possible purposes:

- Tracking mentions of particular entities in documents
- For question answering, answers are usually named entities
- A lot of wanted information is really associations between named entities
- The same techniques can be extended to other slot-filling classifications
- Often followed by Named Entity Linking/Canonicalization into Knowledge Base

25

Binary word window classification

一般不会对single word分类，而是放在context中进行分类。

Window classification

- Idea:** classify a word in its context window of neighboring words.
- For example, **Named Entity Classification** of a word in context:
 - Person, Location, Organization, None
- A simple way to classify a word in context might be to **average** the word vectors in a window and to classify the average vector
 - Problem: that would **lose position information**

如果直接将一个window取平均，会损失位置信息。

一种改进方法是把window的wordvec都concat起来：

Window classification: Softmax

- Train softmax classifier to classify a center word by taking concatenation of word vectors surrounding it in a window
 - Example: Classify “Paris” in the context of this sentence with window length 2:

... museums in Paris are amazing

$$\mathbf{x}_{\text{window}} = [x_{\text{museums}} \quad x_{\text{in}} \quad x_{\text{Paris}} \quad x_{\text{are}} \quad x_{\text{amazing}}]^T$$

- Resulting vector $x_{\text{window}} = \boxed{x \in \mathbb{R}^{5d}}$, a column vector!

30

这是一篇经典论文，ICML2018的test of time奖。

Matrix calculus review

多元求导

分母标量，分子向量

Gradients

- Given a function with 1 output and n inputs

$$f(\mathbf{x}) = f(x_1, x_2, \dots, x_n)$$

- It's gradient is a vector of partial derivatives with respect to each input

$$\frac{\partial f}{\partial \mathbf{x}} = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n} \right]$$

42

分母m维向量，分子n维向量 (m个输出， n个输入)

Jacobian Matrix: Generalization of the Gradient

- Given a function with **m outputs** and n inputs

$$\mathbf{f}(\mathbf{x}) = [f_1(x_1, x_2, \dots, x_n), \dots, f_m(x_1, x_2, \dots, x_n)]$$

- It's Jacobian is an **$m \times n$ matrix** of partial derivatives

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

$$\boxed{\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)_{ij} = \frac{\partial f_i}{\partial x_j}}$$

43

Derivative with respect to Matrix

- What does $\frac{\partial s}{\partial \mathbf{W}}$ look like? $\mathbf{W} \in \mathbb{R}^{n \times m}$
- 1 output, nm inputs: 1 by nm Jacobian?
 - Inconvenient to do $\theta^{new} = \theta^{old} - \alpha \nabla_{\theta} J(\theta)$
- Instead follow convention: shape of the gradient is shape of parameters
 - So $\frac{\partial s}{\partial \mathbf{W}}$ is n by m :
$$\begin{bmatrix} \frac{\partial s}{\partial W_{11}} & \cdots & \frac{\partial s}{\partial W_{1m}} \\ \vdots & \ddots & \vdots \\ \frac{\partial s}{\partial W_{n1}} & \cdots & \frac{\partial s}{\partial W_{nm}} \end{bmatrix}$$

70

Derivative with respect to Matrix

- Remember $\frac{\partial s}{\partial \mathbf{W}} = \boldsymbol{\delta} \frac{\partial z}{\partial \mathbf{W}}$
 - $\boldsymbol{\delta}$ is going to be in our answer
 - The other term should be \mathbf{x} because $\mathbf{z} = \mathbf{W}\mathbf{x} + \mathbf{b}$
- It turns out $\frac{\partial s}{\partial \mathbf{W}} = \boldsymbol{\delta}^T \mathbf{x}^T$

$\boldsymbol{\delta}$ is local error signal at \mathbf{z}
 \mathbf{x} is local input signal

71

Other Jacobians

$$\frac{\partial}{\partial \mathbf{x}}(\mathbf{Wx} + \mathbf{b}) = \mathbf{W}$$

$$\frac{\partial}{\partial \mathbf{b}}(\mathbf{Wx} + \mathbf{b}) = \mathbf{I} \text{ (Identity matrix)}$$

$$\frac{\partial}{\partial \mathbf{u}}(\mathbf{u}^T \mathbf{h}) = \mathbf{h}^T$$

- Compute these at home for practice!
 - Check your answers with the lecture notes

一个例子

Example Jacobian: Elementwise activation Function

$\mathbf{h} = f(\mathbf{z})$, what is $\frac{\partial \mathbf{h}}{\partial \mathbf{z}}$? $\mathbf{h}, \mathbf{z} \in \mathbb{R}^n$
 $h_i = f(z_i)$

$$\begin{aligned}\left(\frac{\partial \mathbf{h}}{\partial \mathbf{z}}\right)_{ij} &= \frac{\partial h_i}{\partial z_j} = \frac{\partial}{\partial z_j} f(z_i) && \text{definition of Jacobian} \\ &= \begin{cases} f'(z_i) & \text{if } i = j \\ 0 & \text{if otherwise} \end{cases} && \text{regular 1-variable derivative}\end{aligned}$$

$$\frac{\partial \mathbf{h}}{\partial \mathbf{z}} = \begin{pmatrix} f'(z_1) & & 0 \\ & \ddots & \\ 0 & & f'(z_n) \end{pmatrix} = \text{diag}(f'(\mathbf{z}))$$

上图是激活函数，所以 $h_i = f(z_i)$ ，不用考虑其他 z_j ，也因此最后结果是对角矩阵。

重复计算部分可以保留

Re-using Computation

- Suppose we now want to compute $\frac{\partial s}{\partial \mathbf{W}}$
 - Using the chain rule again:

$$\frac{\partial s}{\partial \mathbf{W}} = \frac{\partial s}{\partial \mathbf{h}} \frac{\partial \mathbf{h}}{\partial \mathbf{z}} \frac{\partial \mathbf{z}}{\partial \mathbf{W}}$$
$$\frac{\partial s}{\partial \mathbf{b}} = \frac{\partial s}{\partial \mathbf{h}} \frac{\partial \mathbf{h}}{\partial \mathbf{z}} \frac{\partial \mathbf{z}}{\partial \mathbf{b}}$$

The same! Let's avoid duplicated computation...

67

Shape convention

What shape should derivatives be?

- $\frac{\partial s}{\partial \mathbf{b}} = \mathbf{h}^T \circ f'(\mathbf{z})$ is a row vector
 - But convention says our gradient should be a column vector because \mathbf{b} is a column vector...
- Disagreement between Jacobian form (which makes the chain rule easy) and the shape convention (which makes implementing SGD easy)
 - We expect answers to follow the **shape convention**
 - But Jacobian form is useful for computing the answers

74

