法律声明

- 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

- 课程详情请咨询
  - 微信公众号：小象
  - 新浪微博：ChinaHadoop

神经序列模型 I

主讲人： 史兴

07/05/2017

提纲

- 机器学习回顾
- 神经网络基础
- 小试牛刀
- NNLM
- Word2Vec

机器学习回顾

- 模型
  - $\hat{y}_i = f(x_i, \Theta)$
    - linear model 线性模型
      - $\hat{y}_i = \sum_{j=1}^{d} w_j x_{ij} \quad x_i \in R^d$
  - 参数 parameter
    - $\Theta = \{w_j | j = 1, \ldots, d\}$
  - 超参数 hyperparameter
    - d

机器学习回顾

- 数据
  - Training Data 训练集
    - $T = \{(x_i, y_i) | i = 1, \ldots, N\}$
  - Development/Validation Data 验证集
    - 防止过拟合 (overfitting)
  - Test Data 测试集
  - Train Data <> Dev Data <> Test Data 互不重合
- 正确流程(以线性模型为例)
  1. 选择超参数d
  2. 在训练集上训练，得到参数Θ
  3. 在验证集上进行预测，回到步骤1
  4. 选出最佳超参数，重新训练，得到最优模型
  5. 用最优模型在测试集上进行测试

机器学习回顾

- 正确流程(以考试为例为例)
  - Train/Dev/Test = 五年模拟/三年高考/当年高考
  - 超参数：做题的顺序
  - 正确流程
    1. 选择一种做题的顺序d
    2. 做一遍“五年模拟”训练一下自己
    3. 在“三年高考”上测试一下自己，回到步骤1
       1. 如果没有这一步？
       2. 如果使用“五年模拟”来测试？
    4. 用最佳顺序在五年模拟上重新训练一遍自己。
    5. 参加当年高考
       1. 如果提前搞到了高考试题？

机器学习回顾

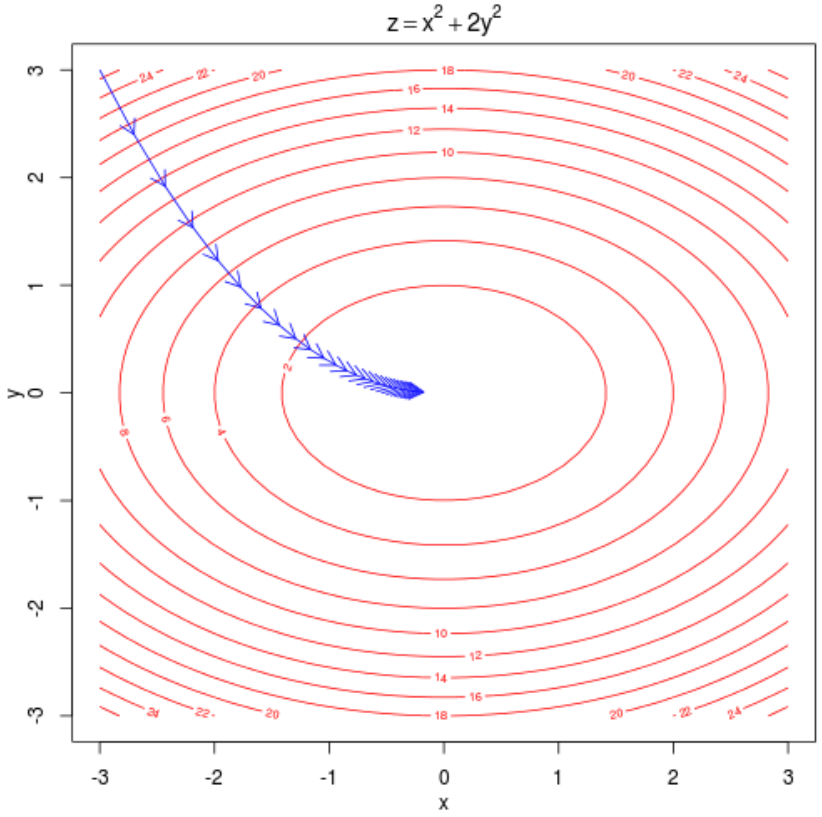
- 目标函数
  - $Obj(\Theta, T) = L(\Theta, T) + \Omega(\Theta)$
  - $L(\Theta, T)$: Loss function
    - 衡量模型对训练集的拟合度
    - Square Loss: $L(\Theta, T) = \sum_{i=1}^{N}(y_i - \hat{y_i})^2$
  - $\Omega(\Theta)$: Regularization
    - 衡量模型的复杂度(死记硬背vs解题规律)
    - L2 Norm: $\Omega(\Theta) = \sum_{j=1}^{d} w_j^2$

机器学习回顾

- 优化目标函数
  - $Obj(\Theta, T) = L(\Theta, T) + \Omega(\Theta)$
  - Batch Gradient Descent 梯度下降法
    1. 初始化参数
       - $\Theta = \text{uniform(d)}$
    2. 计算偏导数
       - $\nabla\Theta = \frac{\partial \text{Obj}(\Theta,\text{T})}{\partial\Theta} = \frac{\partial}{\partial\Theta}\sum_{i=1}^{N} Obj(\Theta, x_i, y_i)$
    3. 更新参数
       - $\Theta = \Theta - \eta\,\nabla\Theta$
    4. 更新learning rate η，返回2，直到收敛

机器学习回顾

- Supervised Learning基本框架



机器学习回顾

- 优化目标函数
  - Batch Gradient Descent
    - 必须遍历所有的训练数据才更新一次
  - Stochastic Gradient Descent 随机梯度下降法
    1. 初始化参数
       - $\Theta = \text{uniform(d)}$
    2. 随机抽取一个数据点(x_i, y_i),计算偏导数
       - $\nabla\Theta = \frac{\partial \text{Obj}(\Theta, x_i, y_i)}{\partial \Theta} = \frac{\partial}{\partial \Theta} Obj(\Theta, x_i, y_i)$
    3. 更新参数
       - $\Theta = \Theta - \eta \nabla\Theta$
    4. 适当的条件更新learning rate η，返回2，直到收敛

机器学习回顾

- 优化目标函数
  - Batch Gradient Descent
    - 必须遍历所有的训练数据才更新一次
  - Stochastic Gradient Descent
    - 每看见一个数据点就更新，非常不稳定
  - Mini-batch Gradient Descent Mini-batch梯度下降法

1. 初始化参数
   - $\Theta = \text{uniform(d)}$
2. 随机抽取m个数据点$\mathrm{T_m} = \{(x_i, y_i) | i = k_1, \ldots, k_m\}$,计算偏导数
   - $\nabla\Theta = \frac{\partial \mathrm{Obj}(\Theta, T_m)}{\partial \Theta} = \frac{\partial}{\partial \Theta} \sum_{i=k_1}^{k_m} Obj(\Theta, x_i, y_i)$
3. 更新参数
   - $\Theta = \Theta - \eta \nabla\Theta$
4. 适当的条件更新learning rate η，返回2，直到收敛

机器学习回顾

优化目标函数

优化方法	特点	类比
Batch GD	稳定，更新慢	看24史
Stochastic GD	不稳定，更新快	看“万历十五年”
Mini-batch GD	稳定，更新快	看三国演义

机器学习回顾

- 超参数选择
  - Grid Search
    - 在高维空间中对一定区域进行遍历
  - Random Search
    - 随机在高维空间中选择若干超参数

机器学习回顾

- 超参数选择
  - Grid Search
    - 在高维空间中对一定区域进行遍历
  - Random Search
    - 随机在高维空间中选择若干超参数

机器学习回顾

- 判别式/生成式模型
  - 判别式模型 discriminative model
    - P(y|x)
    - 关注于从x到y的映射
      - 情感分析 p(+1/0/-1|句子)
    - 只能predict，不能生成数据
  - 生成式模型generative model
    - P(y,x)
    - 关于于对x和y的联合分布
      - 情感分析 p(+1/0/-1, 句子) = p(-1/0/1) * p(句子| -1/0/1)
    - 可以用来生成数据
    - 每个生成式模型都有一个生成故事 Generative Story

机器学习回顾

总结

- 模型：$\hat{y}_i = f(x_i, \Theta)$
- 数据：Train/Dev/Test 正确流程
- 目标函数：$Obj(\Theta, T) = L(\Theta, T) + \Omega(\Theta)$
- 优化目标函数: batch/stochastic/minibatch GD
- 超参数选择: Grid Search/Random Search
- 判别式模型/生成式模型

神经网络基础

- 神经网络 Neural Network
  - 模型：$\hat{y}_i = f(x_i, \Theta)$
    - 线性运算符：$y = Wx + b$
    - 非线性运算符/激活函数/activation function
      - sigmoid, tanh, ReLU, softmax, embedding lookup …
  - 数据：Train/Dev/Test 正确流程
  - 目标函数：$Obj(\Theta, T) = L(\Theta, T) + \Omega(\Theta)$
    - loss function: cross-entropy
    - Regulation: Dropout
    - 计算目标函数：Forward Propagation
    - 计算$\frac{\partial \mathrm{Obj}(\Theta,\mathrm{T})}{\partial \Theta}$： Backward Propagation
  - 优化目标函数: batch/stochastic/minibatch GD
  - 超参数选择: Grid Search/Random Search
  - 判别式模型/生成式模型

神经网络基础

- Forward/Backward Propagation 前向/反向传播算法
  - Forward Propagation
    - $\hat{y_i} = f(x_i, \Theta)$
      $= f_n(f_{n-1} \cdots f_1(x_i) \dots)$
      $= f_{\theta_n} \circ f_{\theta_{n-1}} \circ \cdots \circ f_{\theta_1}(x_i)$
    - $J(\Theta, D) = Loss(\Theta, D) + \Omega(\Theta)$
      $= \sum L(\Theta, y_i, \hat{y_i}) + \Omega(\Theta)$
      $= \sum l_{y_i} \circ f_{\theta_n} \circ f_{\theta_{n-1}} \circ \cdots \circ f_{\theta_1}(x_i)$
  - Backward Propagation 反向传播算法
    - $\frac{\partial J(\Theta, D)}{\partial \Theta} = \{\frac{\partial J}{\partial \theta_j} | j = 1, \dots, n\}$

神经网络基础

- Backward Propagation 反向传播算法
  - 导数计算的链式法则
    - $J = g(f(x))$
    - $\frac{\partial J}{\partial x} = \frac{\partial J}{\partial f}\frac{\partial f}{\partial x} = g(f(x))' f(x)'$
    - 例子： 求导： $\log(x^2)$

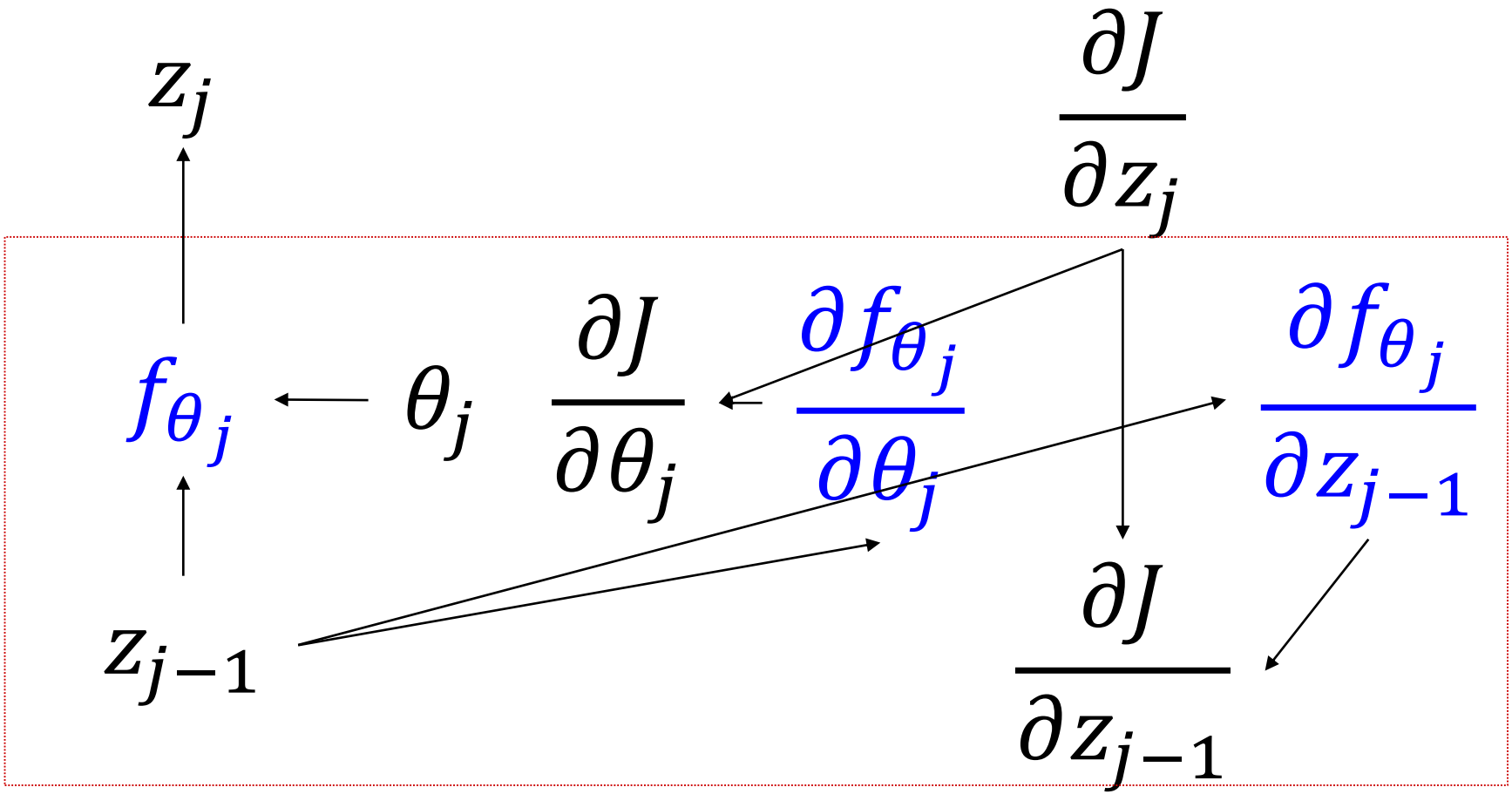
神经网络基础

- Backward Propagation 反向传播算法
  - $\frac{\partial J}{\partial \theta_j} = \sum_i \frac{\partial}{\partial \theta_j} l_{y_i} \circ f_{\theta_n} \circ \cdots \circ f_{\theta_j} \circ \cdots \circ f_{\theta_1}(x_i)$
  - 设 $z_j = f_{\theta_j} \circ \cdots \circ f_{\theta_1}(x_i)$
  - 所以，$\frac{\partial J}{\partial \theta_j} = \sum_i \frac{\partial J}{\partial f_{\theta_j}} \frac{\partial f_{\theta_j}}{\partial \theta_j} = \sum_i \frac{\partial J}{\partial z_j} \frac{\partial f_{\theta_j}}{\partial \theta_j}$
  - $\frac{\partial f_{\theta_j}}{\partial \theta_j}$ 只跟f_{θ_j}的形式有关
  - $\frac{\partial J}{\partial z_j} = \frac{\partial J}{\partial z_n} \frac{\partial z_n}{\partial z_{n-1}} \cdots \frac{\partial z_{j+1}}{\partial z_j}$，其中$\frac{\partial z_k}{\partial z_{k-1}}$只跟$f_{\theta_k}$的形式有关

神经网络基础

- $J(\Theta, D) = \sum l_{y_i} \circ f_{\theta_n} \circ f_{\theta_{n-1}} \circ \cdots \circ f_{\theta_1}(x_i)$
- $z_n \quad z_{n-1} \quad z_1$
- $\frac{\partial J}{\partial z_n} \quad \frac{\partial z_n}{\partial z_{n-1}} \quad \cdots \quad \frac{\partial z_{j+1}}{\partial z_j} \quad \cdots \quad \frac{\partial z_2}{\partial z_1}$
- $\frac{\partial f_{\theta_n}}{\partial \theta_n} \quad \frac{\partial f_{\theta_{n-1}}}{\partial \theta_{n-1}} \quad \cdots \quad \frac{\partial f_{\theta_j}}{\partial \theta_j} \quad \cdots \quad \frac{\partial f_{\theta_1}}{\partial \theta_1}$
- $\frac{\partial J}{\partial \theta_n} \quad \frac{\partial J}{\partial \theta_{n-1}} \quad \cdots \quad \frac{\partial J}{\partial \theta_j} \quad \cdots \frac{\partial J}{\partial \theta_1}$

神经网络基础



黑色的为数值(numpy array)，蓝色的为函数(function)

神经网络基础

```
operator f {

matrix input;
matrix d_input;

matrix *output;
matrix *d_output;

matrix theta;
matrix d_theta;
// cpu version
function forward(){}
function compute_d_input(){}
function compute_d_theta(){}
// gpu version
function forward_gpu(){}
function compute_d_input_gpu(){}
function compute_d_theta_gpu(){}

}
```

神经网络基础

矩阵求导的定义

- d标量/d向量

$$\frac{\partial y}{\partial \mathbf{x}} = \left[\frac{\partial y}{\partial x_1} \quad \frac{\partial y}{\partial x_2} \quad \cdots \quad \frac{\partial y}{\partial x_n} \right].$$

- d标量/d矩阵

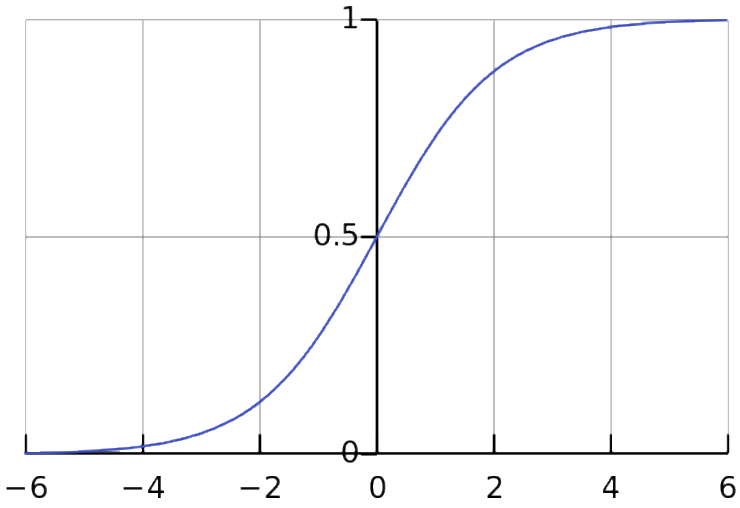
$$\frac{\partial y}{\partial \mathbf{X}} = \begin{bmatrix} \frac{\partial y}{\partial x_{11}} & \frac{\partial y}{\partial x_{21}} & \cdots & \frac{\partial y}{\partial x_{p1}} \\ \frac{\partial y}{\partial x_{12}} & \frac{\partial y}{\partial x_{22}} & \cdots & \frac{\partial y}{\partial x_{p2}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y}{\partial x_{1q}} & \frac{\partial y}{\partial x_{2q}} & \cdots & \frac{\partial y}{\partial x_{pq}} \end{bmatrix}.$$

神经网络基础

- 线性变换
  - 为了统一，假设x,y都是行向量
  - $y = xW + b$
  - $\frac{\partial J}{\partial W} = \frac{\partial J}{\partial y}\frac{\partial y}{\partial W} = x^T \frac{\partial J}{\partial y}$
  - $\frac{\partial J}{\partial b} = \frac{\partial J}{\partial y}$
  - $\frac{\partial J}{\partial x} = \frac{\partial J}{\partial y} W^T$
  - 保证形状是关键

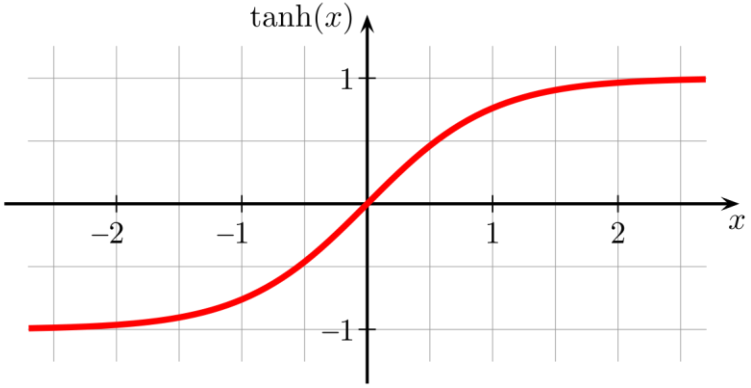
神经网络基础

- 非线性变换：sigmoid
  - $y = \sigma(x) = \frac{1}{1+e^{-x}}$
  - $\frac{\partial y}{\partial x} = y(1-y)$
  - 值域[0,1]
    - 做概率
    - 做开关
    - 做“挤压”
  - 导数的绝对值
    - 只在0附近比较大
  - elementwise



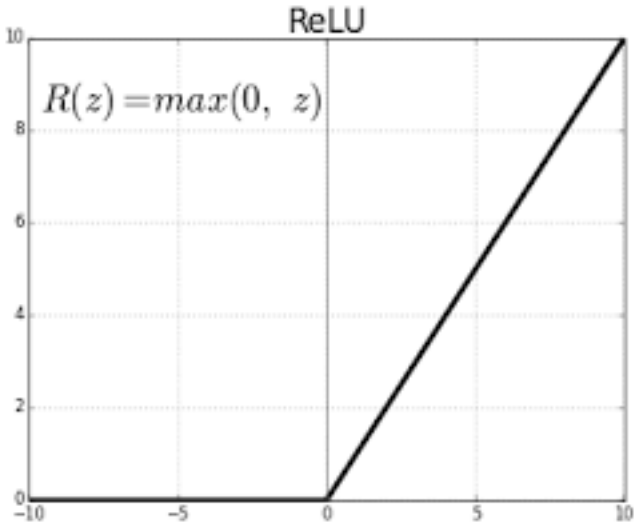
神经网络基础

- 非线性变换：tanh
  - $y = tanh(x) = \frac{1-e^{-2x}}{1+e^{-2x}}$
  - $\frac{\partial y}{\partial x} = 1 - y^2$
  - 值域[-1,1]
    - 做“挤压”
  - 导数的绝对值
    - 只在0附近比较大
    - google: batch normalization
  - elementwise



神经网络基础

- 非线性变换：ReLU
  - $y = ReLU(x) = \max(0, x)$
  - $\frac{\partial y}{\partial x} = 1\ if\ x > 0; 0\ if\ x < 0$
  - 值域[0,+∞]
    - 仅仅是线性变换
  - 导数的绝对值
    - 0 或 1
  - elementwise



神经网络基础

- 非线性变换：softmax
  - $y_j = softmax\,(x)_j = \frac{e^{x_j}}{\sum_{i=1}^{d} e^{x_i}}$
  - $\frac{\partial y_i}{\partial x_i} = y_i(1 - y_i)$; $\frac{\partial y_i}{\partial x_j} = -y_i y_j$
  - 值域[0,1]
    - 做概率
  - 导数的绝对值
    - 0 或 1
  - not elementwise

x	4	-4	3
y	0.7308	0.0003	0.2689

神经网络基础

- 非线性变换：embedding lookup
  - $y = \mathrm{embl}(x) = W[x,:] \quad w \in R^{|v|*d}$
  - $\frac{\partial y}{\partial W[x,:]} = \frac{\partial J}{\partial y}$
  - 作用：int -> embedding

|V| = 5; d=3

0.4	1	-1
0.2	0.4	0.3
-0.3	2	-3
0.4	2	3
0.8	3	-0.5

W

x = 2

Forward

0.2	0.4	0.3

y

神经网络基础

- 非线性变换：embedding lookup
  - $y = \mathrm{eb}l\,(x) = W[x,:] \quad w \in R^{|v|*d}$
  - $\frac{\partial y}{\partial W[x,:]} = \frac{\partial J}{\partial y}$
  - 作用：int -> embedding

0	0	0
0.3	-0.2	0.4
0	0	0
0	0	0
0	0	0

$\frac{\partial y}{\partial W}$

|V| = 5; d=3

x = 2

0.3	-0.2	0.4

Backward

$\frac{\partial J}{\partial y}$

神经网络基础

- Loss function：cross-entropy
  - $CE(y,\hat{y}) = -\sum_{j=1}^{d} y_j \log \hat{y}_j$
  - 作业题1：拉格朗日法求CE的最小值
  - 多分类问题，y一般为one-hot vector.
    - $y = [0,0,1]$ $\hat{y} = [0.2,0.2,0.6]$ CE = ?
    - $CE(y,\hat{y}) = y_j \log \hat{y}_j$ ； $y_j = 1$
  - $\frac{\partial CE}{\partial \hat{y}_j} = -\frac{y_j}{\hat{y}_j} = -\frac{1}{\hat{y}_j}$;

神经网络基础

- Loss function：cross-entropy
  - $\frac{\partial CE}{\partial \widehat{y_j}} = -\frac{y_j}{\widehat{y_j}} = -\frac{1}{\widehat{y_j}}$;
  - 结合softmax
    - $y_j = softmax\,(x)_j = \frac{e^{x_j}}{\sum_{i=1}^{d} e^{x_i}}$
    - $\frac{\partial y_i}{\partial x_i} = y_i(1 - y_i)$; $\frac{\partial y_i}{\partial x_j} = -y_i y_j$
  - $\frac{\partial CE}{\partial x_i} = \frac{\partial CE}{\partial \widehat{y_j}} \frac{\partial \widehat{y_j}}{\partial x_i}$
    - $-\frac{1}{\widehat{y_j}} \widehat{y_j}(1-\widehat{y_j}) = \widehat{y_j} - 1;\ \ if\ i = j$
    - $-\frac{1}{\widehat{y_j}} * -\widehat{y_i}\widehat{y_j} = \widehat{y_i};\ \ if\ i \neq j$

神经网络基础

- Cross-entropy + Softmax

x	4	-4	3
$\hat{y}$	0.7308	0.0003	0.2689
y	0	0	1
CE	-log(0.2689)		
$\frac{\partial CE}{\partial x_i}$	0.7308	0.0003	0.2689-1

神经网络基础

Cross-entropy + Softmax

Θ	x	4	-4	3
	$\hat{y}$	0.7308	0.0003	0.2689
	y	0	0	1
	CE	-log(0.2689)		
$\nabla\Theta$	$\frac{\partial CE}{\partial x_i}$	0.7308	0.0003	0.2689-1
$\Theta - \eta \nabla\Theta$	x_{new}	4-0.7308	-4 – 0.0003	3-0.2689+1

按比例“推所有，拉一个”

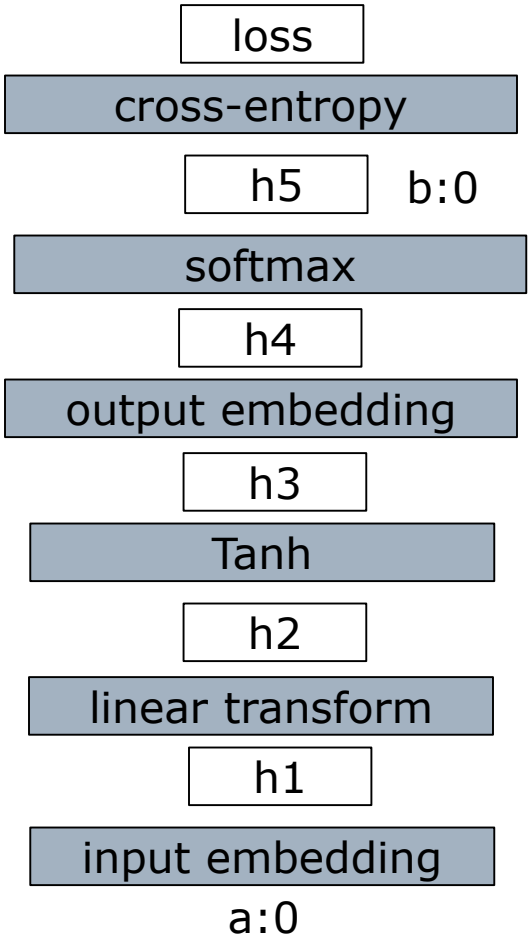
神经网络基础

- 总结一下
  - 线性运算符：
    - $y = Wx + b$
  - 非线性运算符/激活函数/activation function
    - sigmoid
    - tanh
    - ReLU
    - softmax
    - embedding lookup
  - Loss
    - cross-entropy
      - + softmax: “按比例推所有，拉一个”

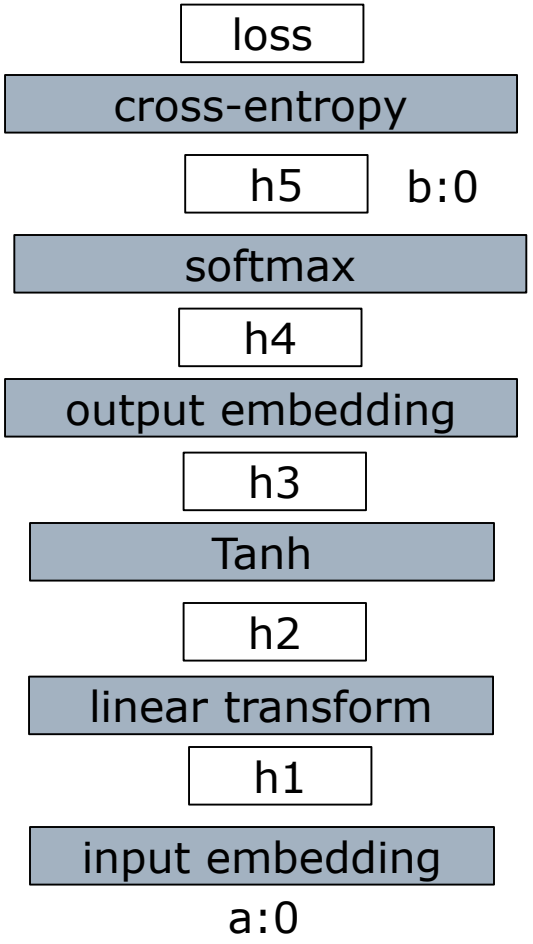
小试牛刀

- bigram neural network
  - $p(w_n|w_{n-1})$
  - 词汇集 V = {a,b,c}
  - 训练数据
    - {abc, acb,bca}
    - ab, bc, ac, cb, bc, ca
  - 字符串转化成数字
    - a:0, b:1, c:2

小试牛刀


loss
cross-entropy
h5
b:0
softmax
h4
output embedding
h3
Tanh
h2
linear transform
h1
input embedding
a:0

小试牛刀



h3
[0.07982977 0.7530659]
h2
[0.08 0.98]

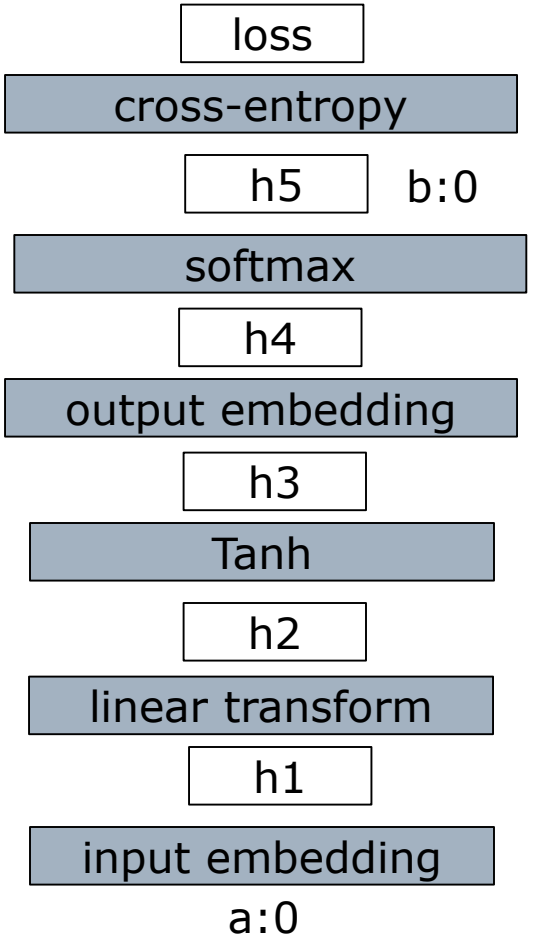
linear_w
[[1.2 0.2]
 [-0.4 0.4]]

linear_b
[0. 0.5]

h1
[0.4 1.]

input_embed
[[0.4 1.]
 [0.2 0.4]
 [-0.3 2.]]

小试牛刀



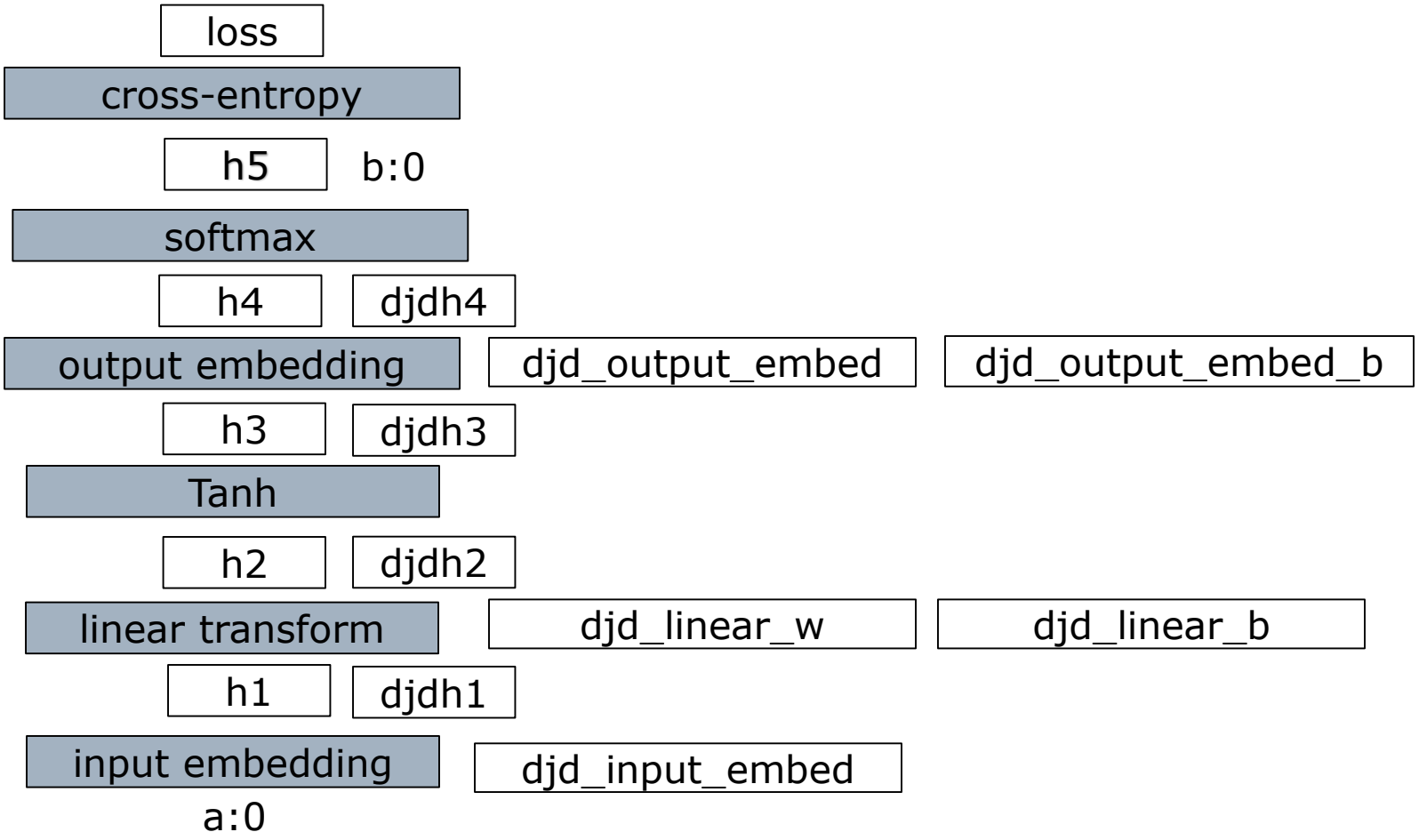
ce
0.810028205586
h5
[0.351601 0.444846 0.203553]

h4
[0.673236 0.908465 0.12666425]

output_embed
[[-1. 1.]
[0.4 0.5]
[-0.3 0.2]]

output_embed_b
[0. 0.5 0.]

小试牛刀


loss
cross-entropy
h5
b:0
softmax
h4
djdh4
output embedding
djd_output_embed
djd_output_embed_b
h3
djdh3
Tanh
h2
djdh2
linear transform
djd_linear_w
djd_linear_b
h1
djdh1
input embedding
djd_input_embed
a:0

小试牛刀

- 程序展示
  - 地址：
  - https://github.com/shixing/xing_nlp/tree/master/LM/NNLM/toy.py
  - 运行环境
    - 需要安装numpy (如果之前安装了anaconda, 那么anaconda中是包含numpy的，不需要再次安装)

小试牛刀

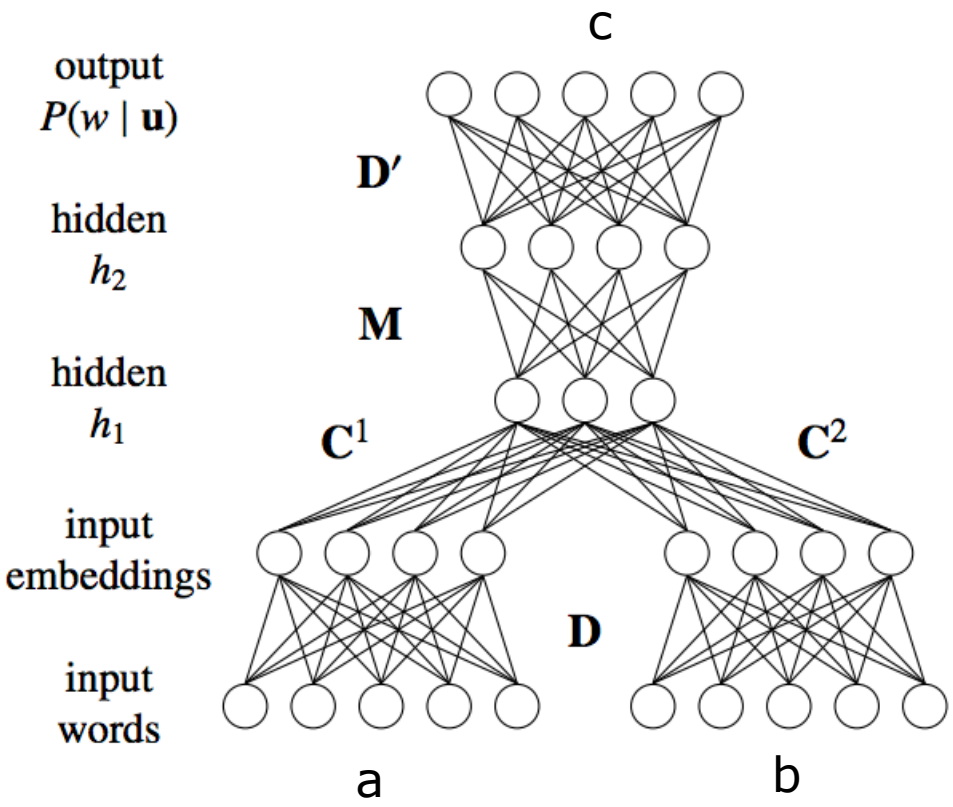
- 程序展示
  - 挑战：
    - toy.py：包含了五处错误，找到这五处错误。
    - 10ite.correct.txt 包含了正确的结果的输出。
    - 理解神经网络调试的困难之处以及常见bug。
    - Debug的一般方法：
      - 看代码
      - 在小样例上面手算
      - 做gradient_check

NNLM

- n-gram 潜在的问题:
  - P(梨|水果 包含) P(苹果|水果 包含)
    - 对“词”的理解有限
      - Neural Network Language Model
    - N-gram 上下文的长度有限
      - Recurrent NN Language Model

NNLM

“a b c”



Bengio, Yoshua, et al. "A neural probabilistic language model." *Journal of machine learning research* 3.Feb (2003): 1137-1155.

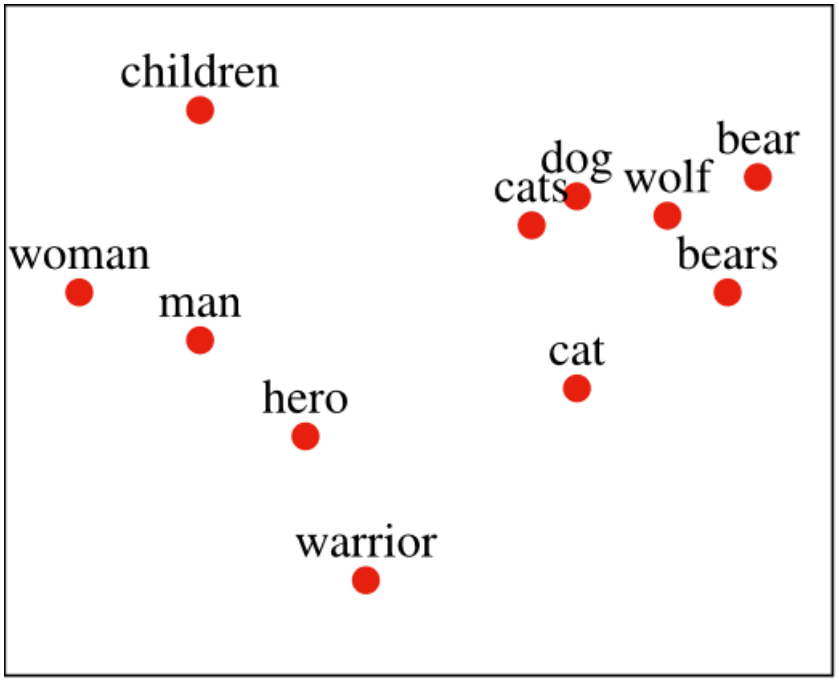
NNLM

Embedding的作用：~~对“词”的理解有限~~

词法的相似性
good, better

语法的相似性
see, saw

语义相似性
dog, cat



NNLM

为什么Embedding会有这样的作用？

Embedding的每一维相当于是机器学习出来的特征

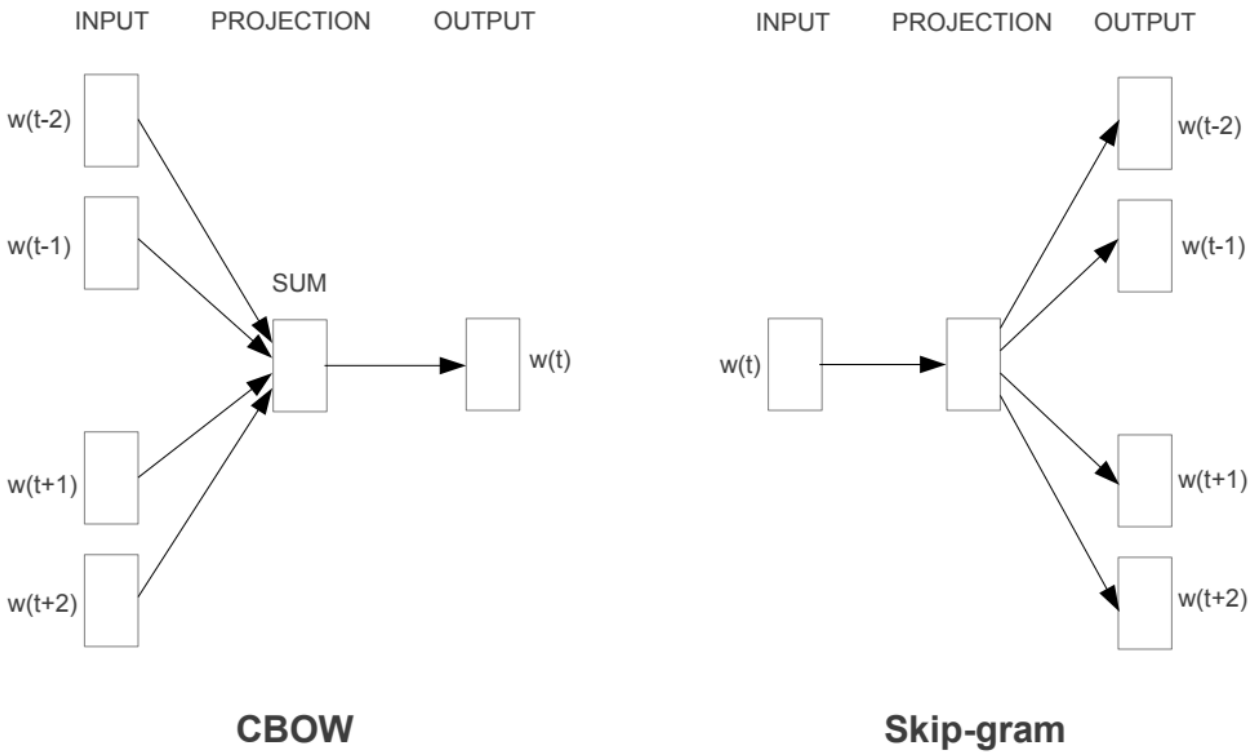
所谓的“特征学习/表示学习”
representation learning

ICLR
International Conference on Learning Representations

Deep learning的核心原因之一

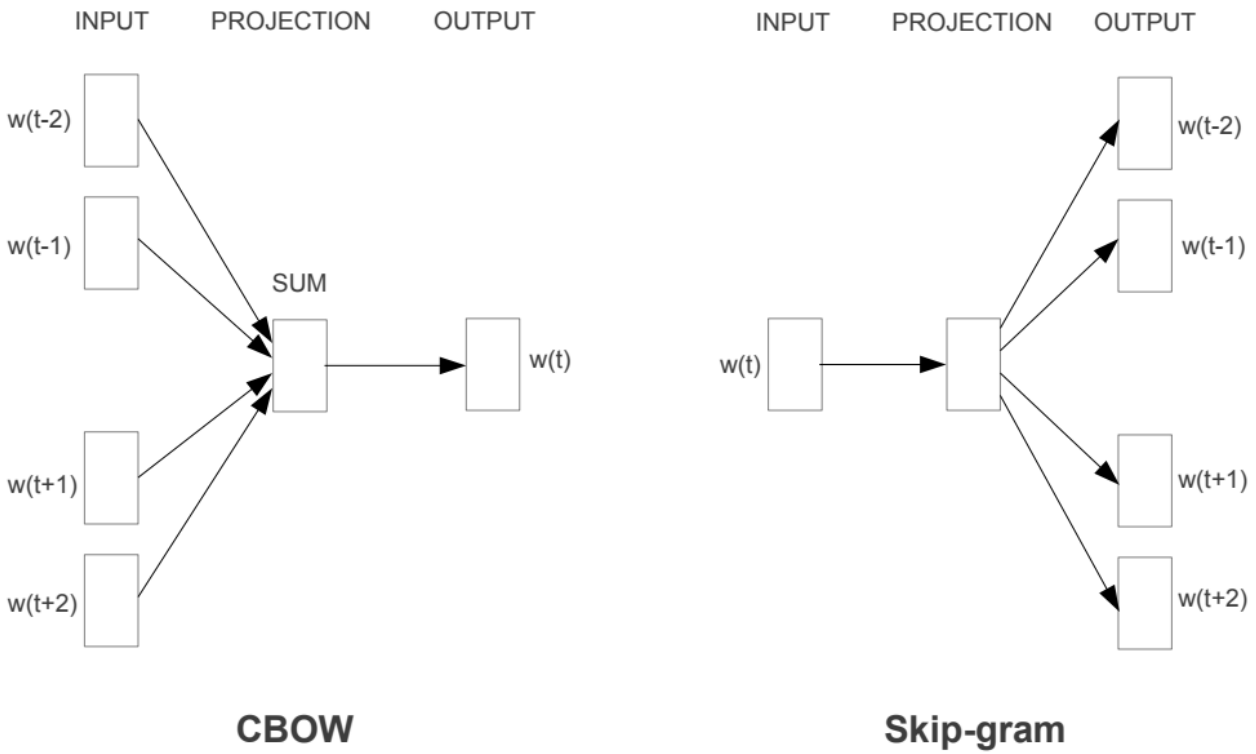
Word2Vec

- 线性模型！快+大数据



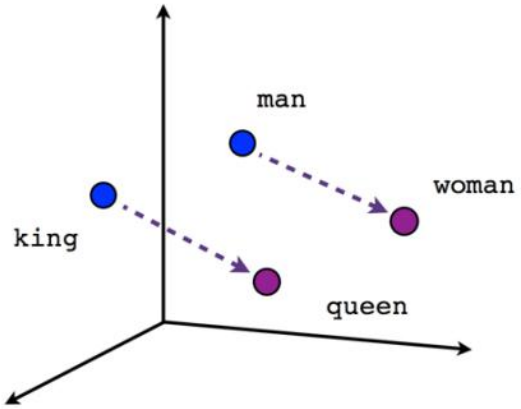
Word2Vec

- 线性模型？！ 快+大数据



Word2Vec

- 著名的类比
  - King – Man + Woman = Queen



Male-Female

Word2Vec

- 著名的类比
  - King – Man + Woman = Queen
  - King – Queen = Man - Woman
  - 进一步的实验：
    - King, Man, Woman, ? (easy)
    - King, Man, ?, ? (hard!)
    - 没有那么神奇
      - 只要把King/Queen，Man/Woman聚类在一起

Word2Vec

- 著名的类比
  - King – Man + Women = Queen
- 评价Word2Vec的一个任务
  - 4词类比：
    - Athens Greece Rome Italy (语义)
    - write writes sit sits (词法)
    - 一共19558对
  - 准确率越高，word2vec越好么？

Word2Vec

- 用途
  - 寻找近义词
  - 用来作为别的自然语言任务的特征值
  - 用来为别的Neural Network做初始化

Word2Vec

- 应用展示：超参数搜索
  - 地址：
    - https://github.com/shixing/xing_nlp/tree/master/word2vec
  - 运行环境
    - Tensorflow 1.2
  - 实验流程
    - 首先进入word2vec文件夹
    - 首先运行 bash prepare.sh (mac机器运行bash prepare.mac.sh)

Word2Vec

- 应用展示：超参数搜索
  - 实验流程
    - 首先进入word2vec文件夹
    - 首先运行 bash prepare.sh (mac机器运行bash prepare.mac.sh)
    - 运行默认设置
      - python word2vec.py --train_data text8.small --eval_data questions-words.txt --save_path temp
    - 可以改变的超参数及其范围(其他参数不能修改)：
      - embedding_size: 10-400
      - learning_rate: 0.0001 – 10
      - batch_size: 4– 512
      - window_size: 1-20
      - python word2vec.py --train_data text8.small --eval_data questions-words.txt --save_path temp --embedding_size 20 --learning_rate 1.0 --batch_size 48 --window_size 10

Word2Vec

- 应用展示：超参数搜索
  - 超参数搜索策略
    - Gird Search
    - Random Search
    - Coordinate Descent
      - 假设有两个超参数x,y
      - 随机初始化x，线性搜索最佳的y, 得到y_best
      - 固定y_best, 线性搜索x, 得到x_best
      - 不断迭代，知道x,y稳定下来
  - 工程技巧(不能让机器闲着)
    - 多机器运行
    - 写bash script连续运行

Word2Vec

- 应用展示：超参数搜索
  - 排行榜
    - 将你的最好的结果写在这里：
      - http://collabedit.com/qvpc6
      - 只写昵称和分数，不要暴露自己的超参数
    - 第五次课前，我们会统计结果
      - 第一名，得到神秘奖励一份。

联系我们

小象学院：互联网新技术在线教育领航者

- 微信公众号：大数据分析挖掘
- 新浪微博：ChinaHadoop