命名实体识别

Author：王文惠

2017.11.15

目录
CONTENTS

基本概念

1 命名实体识别（NER）任务

- 识别出非结构化文本中七类（人名、地名、组织机构名、时间、日期、货币量和百分比）命名实体

例：[[中国]乒乓球男队主教练][刘国梁]出席了会议，[他]指出了当前经济工作的重点。
实体概念“刘国梁”的指称项有三个：“中国乒乓球男队主教练”是名词性指称，“刘国梁”命名性指称，“他”是代词性指称，我们关注的是命名性指称。

- 两个子任务：实体边界识别（因为实体通常由几个词组成）和确定实体类别
- NER是自然语言处理的基本任务，是机器翻译、问答系统、主题模型、信息抽取等多种自然语言处理技术必不可少的组成部分。

2 命名实体识别的特点

- 时间、日期、货币量和百分比的构成有明显规律，识别起来相对容易；
- 人名、地名、组织机构名的用字灵活，内部结构复杂、形势复杂多变，识别难度大，如：
  - 人名：杜甫、杜子美、子美、杜工部
  - 机构名：北京大学附属小学、中国奥委会、北师大二附
- 上下文密切相关
  - 不同语境下，可能具有不同的实体类型；或者在某些条件下是实体，在另外的条件下就不是实体
    如：彩霞、河南、新世界

3 命名实体识别的方法

- 无论何种方法，都在试图充分发现和利用所在的上下文特征和实体的内部特征
- 命名实体识别，是序列标注问题。方法包括：

• 基于规则和词典的方法：构造规则模板和字典，选用特征包括统计信息、标点符号、关键字、指示词和方向词、中心词等方法，以模式和字符串相匹配为主要手段。

缺点：制作规则和大字典费时费力，不能判断unseen的实体，特别容易产生错误，系统移植性不好。

• 统计机器学习方法：最大熵方法、HMM、CRF（more recent method，小数据集上效果非常好）、SVM等，需要从文本中选择对该任务有影响的各种特征，有效特征可以分为具体的单词特征、上下文特征、词典及词性特征、停用词特征、核心词特征以及语义特征等。

缺点：对特征选取的要求特别高。

• 结合统计机器学习方法与规则：目前，没有只使用统计模型而不使用规则知识的命名识别系统。

• 基于深度学习的方法：LSTM或CNN提取文本特征，softmax或统计机器学习方法进行标注。

特征包括字向量、字组成构建向量、字位向量（在词中的位置，如“百\B度\E”）、词向量、词性向量等。

4 命名实体识别的评估

- 系统性能表现主要通过准确率（P）、召回率（R）和F-测度值（F1）3个指标来衡量，计算公式如下：

$$\text{准确率} = \frac{\text{正确识别的实体数}}{\text{总的识别实体数}} \times 100\,\%$$

$$\text{召回率} = \frac{\text{正确识别的实体数}}{\text{总的实体数}} \times 100\,\%$$

$$\text{F1} = \frac{2 \times \text{准确率} \times \text{召回率}}{\text{准确率} + \text{召回率}} \times 100\,\%$$

问题描述

问题描述

关注互联网创业

问题描述

项目内容：

- **命名实体识别：**从非结构化文本中自动化抽取出创业公司名称及投资机构名称。如：

太平洋\B网络\I有限公司\E还\O将给\O快乐\B妈咪\E注入\O资源\O，\O并\O根据\O市场\O发展\O，\O追加\O投资\O。\O

事先将文本用分词工具进行分词，每个词的标签有五种：O、S、B、I、E。根据公司名称中包含的词语个数分为两种情况，一是S表示一个词代表一个公司名，二是公司名由多个词语组成，B是公司名的第一个词、I是公司名的中间词语、E表示公司名的最后一个词。O表示该词不是公司名称中的某一部分。

- 关系抽取：从非结构化文本中自动化抽取出创业公司、投资机构等之间的关系，如收购、并购、融资等。
- 创业公司分类。

模型介绍

模型介绍

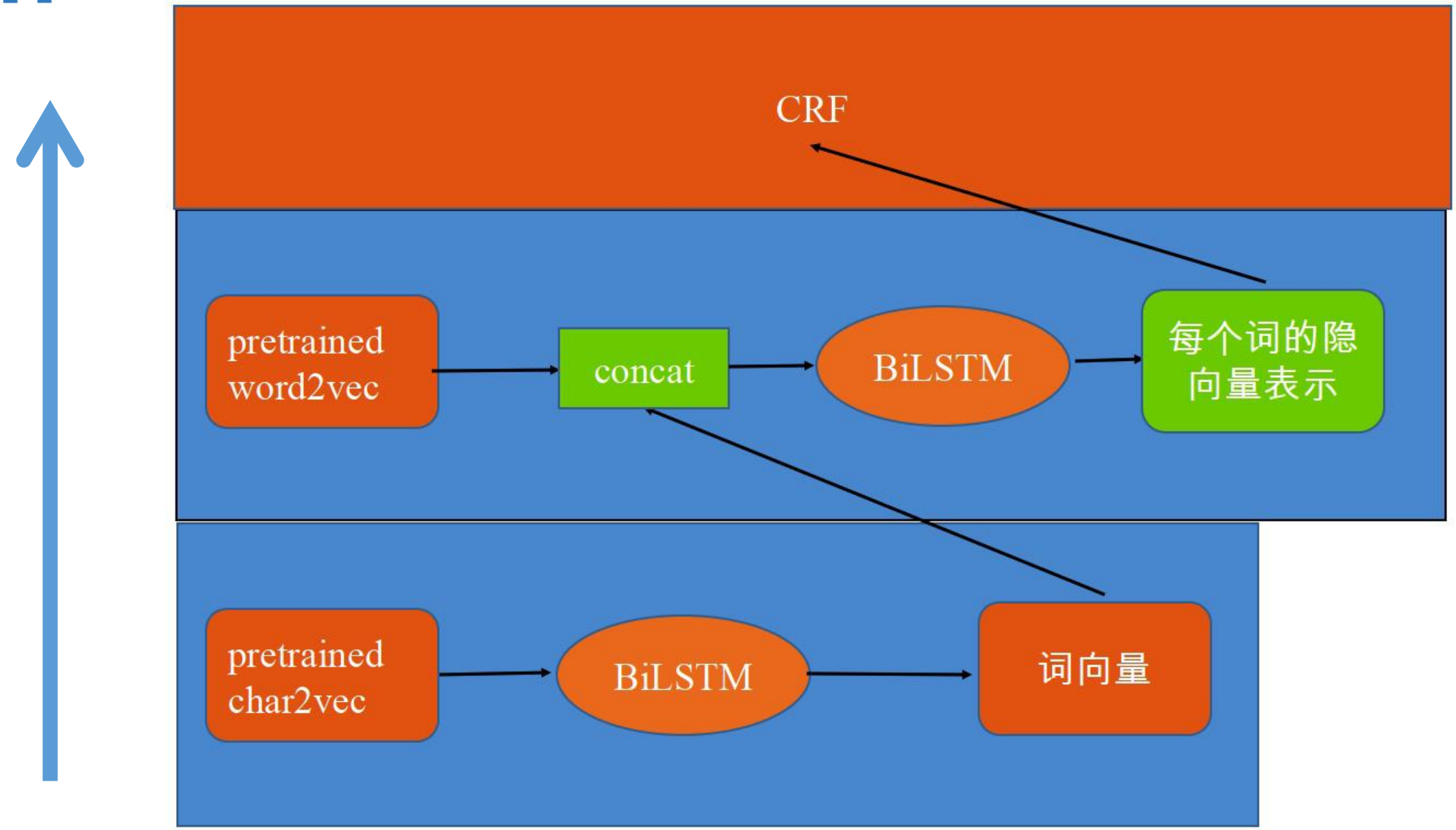


Figure1 模型流程示意图

整体思路

- 用分词工具jieba对文本进行分词；
- 首先利用双向LSTM训练出词语的字级别向量，并从大规模无标注的背景语料中通过word2vec获得具有语义特征信息的词向量，然后将二者组合作为输入，并且可以尝试组合词语的词性向量；
- 构建适合公司名称实体识别的双向LSTM，提取文本的上下文特征表示向量；
- 并在上述特征向量中，组合表示该词语是否出现在人工字典中的向量；
- 将上述特征向量作为序列标注算法CRF的输入，进行实体标注。

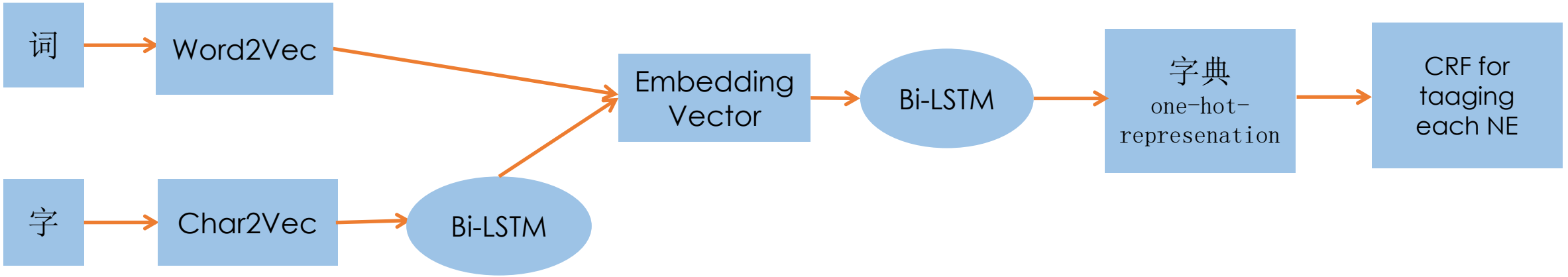


Figure2 模型流程示意图

LSTM输入-输出序列的三种形式：

many to one

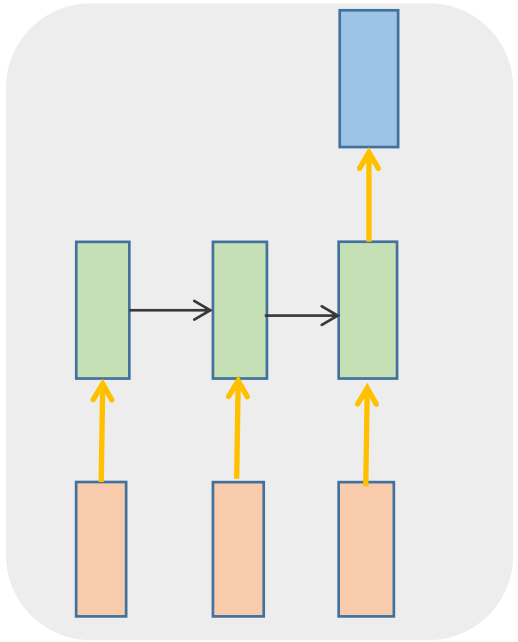


Figure3 多个字向量通过LSTM生成词的向量表示，如：
“百”、“度” LSTM “百度”

many to many

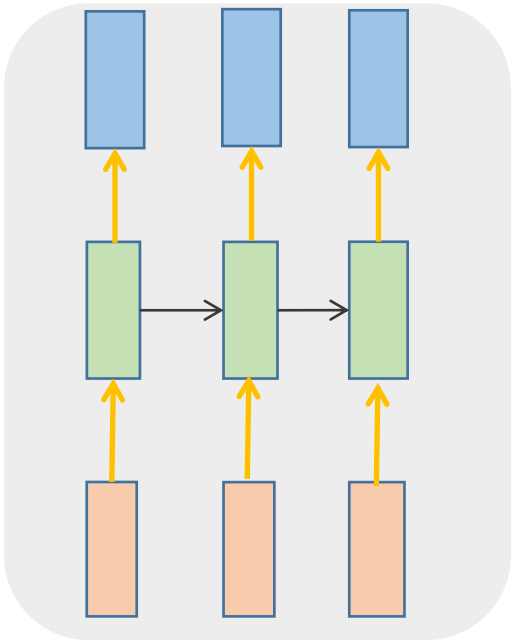


Figure4 词向量通过LSTM捕捉上下文信息，获得每个词的隐状态向量表示。

many to many

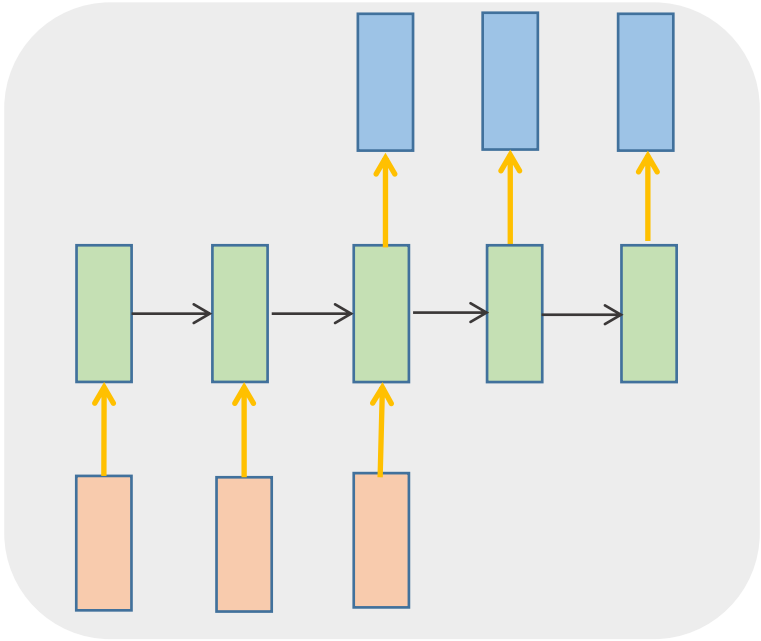


Figure5 该形式通常用于神经网络翻译中，先对源语言编码，再解码成目标语言。

模型介绍

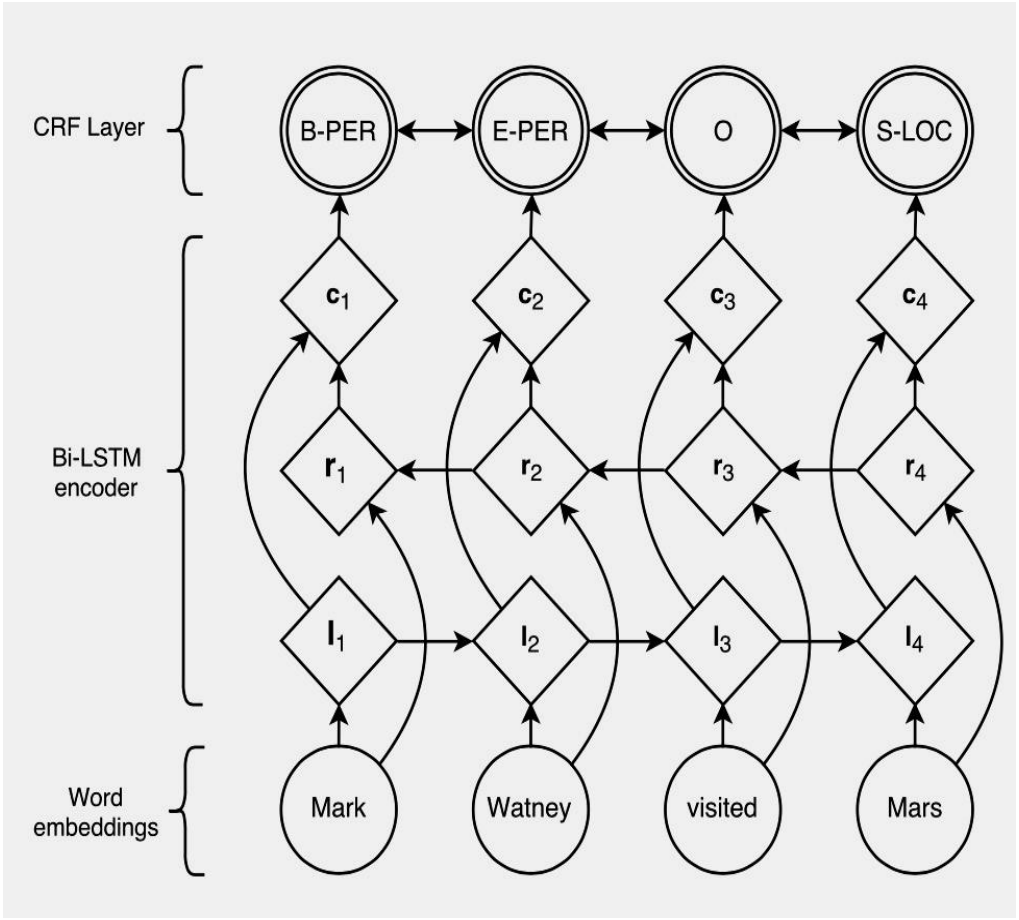


Figure6 主体模块. ◯表示词向量, ◇l◇、◇r◇表示LSTM单元, ◇c◇ 表示将每个词的上下文隐状态向量组合, ◎ 表示每个单词的标签。

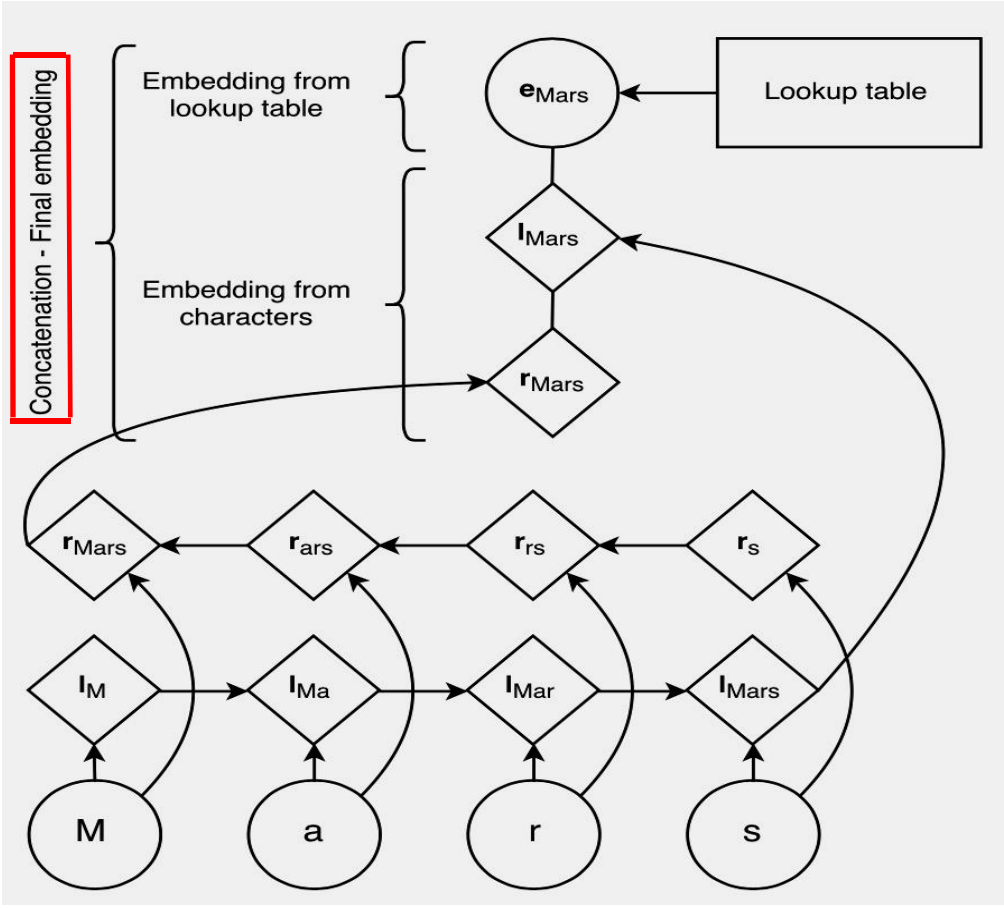


Figure7 字向量表示 —Bi-LSTM→ 词向量表示

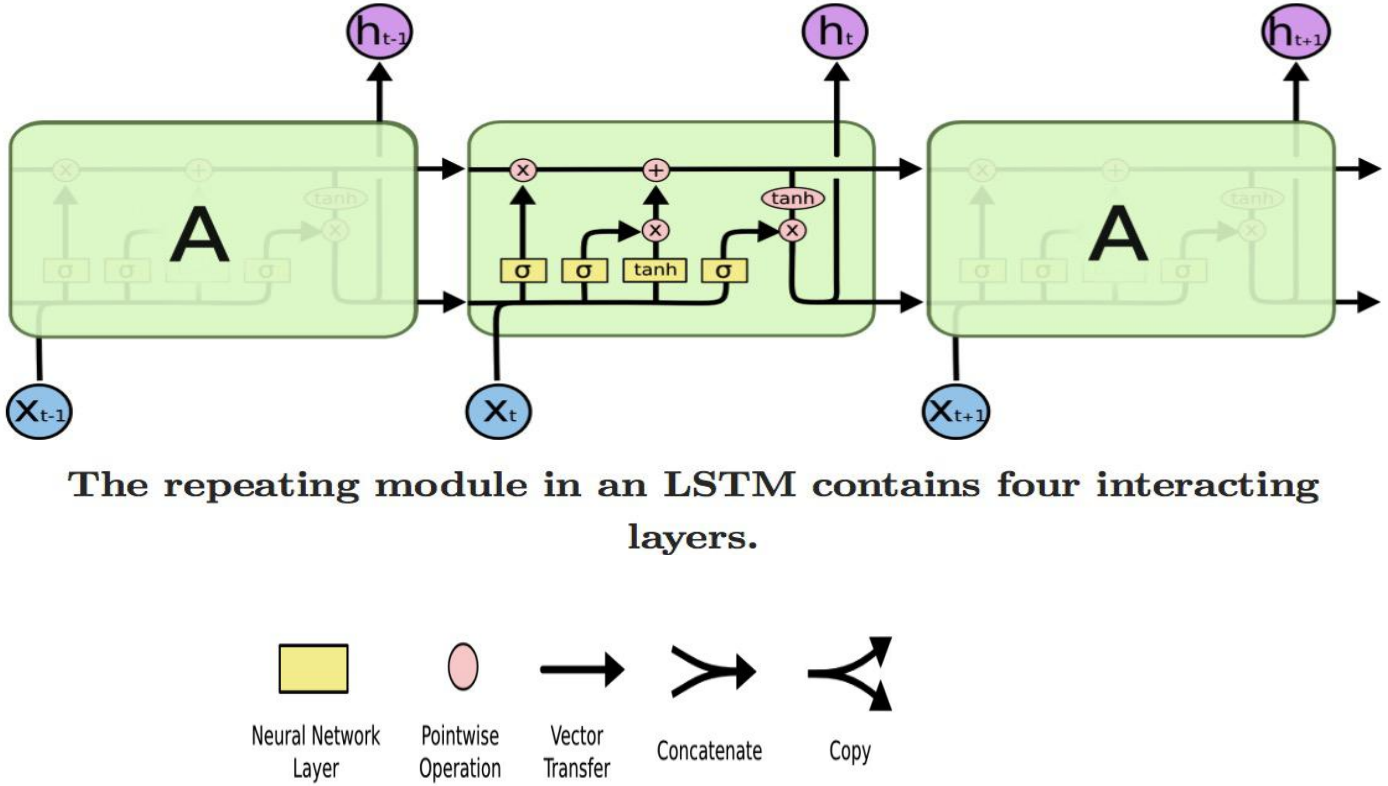
Paper: Named Entity Recognition, 2016, ACL, GLample(CMU)

Pretrained

- **需求**：得到好的embeddings，通常需要大量的训练数据。大量研究证明，好的初始化对训练好模型至关重要，并且能够加快收敛速度；
- **实现方法**：为了充分利用未标注数据，用Word2Vec模型预训练每个字/词的embbeddings（并在训练命名识别模型时，对char/word embeddings进行细微调整）。Word2Vec的思想是出现在相似上下文环境中的词是相似的，因此通常它们会集聚在空间的某一区域。Word2Vec有两种模型：CBOW和Skip-gram，大量实验表明，Skip-gram的效果更好；
- **优点**：比起随机初始化embeddings，pretrained embeddings达到收敛所需要的数据量少，且结果更好。

Bi-LSTM

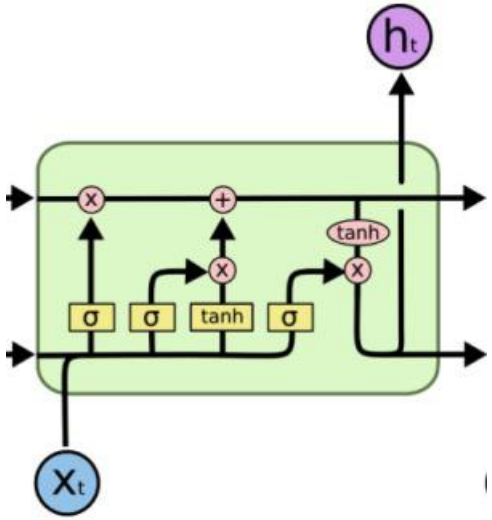
长短期记忆网络（LSTM），是一种特殊的循环神经网络，克服了传统RNN模型由于序列过长而产生的梯度弥散问题，是自然语言处理任务中使用最多的模型之一。LSTM模型通过门结构使得模型可以有选择地保存上下文信息，因此LSTM适合命名实体识别任务。LSTM网络的主要结构为：



The repeating module in an LSTM contains four interacting layers.

LSTM介绍：http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Bi-LSTM



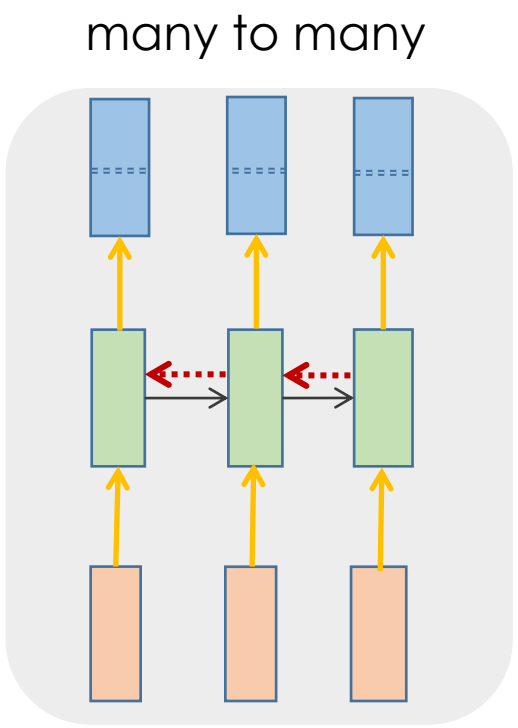
$$i_t = \sigma(x_t \cdot w_{xh}^i + h_{t-1} \cdot w_{hh'}^i + b_h^i)$$
$$f_t = \sigma(x_t \cdot w_{xh}^f + h_{t-1} \cdot w_{hh'}^f + b_h^f)$$
$$o_t = \sigma(x_t \cdot w_{xh}^o + h_{t-1} \cdot w_{hh'}^o + b_h^o)$$
$$\tilde{c}_t = \tanh(x_t \cdot w_{xh}^c + h_{t-1} \cdot w_{hh'}^c + b_h^c)$$
$$c_t = i_t \otimes \tilde{c}_t + f_t \otimes c_{t-1}$$
$$h_t = o_t \otimes \tanh(c_t)$$

σ是sigmoid函数，取值范围[0,1]。LSTM通过控制输入门及遗忘门的大小来控制memory cell对长短期信息的记忆，从而解决传统RNN梯度消失的问题。梯度爆炸的问题通过gradient clipping解决,

$$\frac{\partial L}{\partial \theta} = \begin{cases} LB \ \ldots\ if \ (\frac{\partial L}{\partial \theta})_0 < LB \\ (\frac{\partial L}{\partial \theta})_0 \ \ldots\ else \\ UB \ \ldots\ if \ (\frac{\partial L}{\partial \theta})_0 > UB \end{cases}$$

Bi-LSTM

由于LSTM中，当前输出仅依赖于当前输入与上文的信息。为了能够有效利用上下文信息，引入双向LSTM（Bi-LSTM）结构，双向LSTM对每个句子分别采用顺序（从第一个词开始，从左往右递归）和逆序（从最后一个词开始，从右往左递归）计算得到两套不同的隐层表示，然后通过向量拼接得到最终的隐层表示。举例说明：



线性CRF

CRF能够考虑到相邻标签的关系获得一个全局最优的标记序列，例如：百度\B网络\I有限公司\E，采用的标签是IOBES，考虑到“有限公司”的前一个词的标注是I，“有限公司”的可能标签是I或E。

将Bi-LSTM的输出作为CRF的输入，获得全局最优的标注序列。对于一个句子S={W1,W2,.....,Wn}输入网络进行训练，定义矩阵P是Bi-LSTM的输出结果，其中P的大小N*K，N是单词个数，K是标签的种类。定义P_{ij}表示句子中第i个单词的第j个标签的概率。对于一个预测序列$y=\{y_1,y_2,...,y_n\}$，它的概率可以表示为

$$S(X,y)=\sum_{i=0}^{n}A_{y_i,y_{i+1}}+\sum_{i=0}^{n}P_{i,y_i} \qquad (1)$$

矩阵A是转移矩阵，例如A_{ij}表示由标签i转移到j的概率，y_0、y_n是句子起始和结束的标记，因此A是一个大小为K+2的方阵。

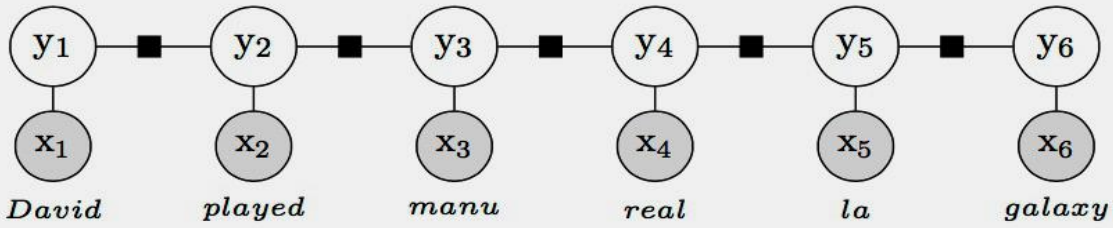


Figure 1: Linear-chain model (CRF).

线性CRF

所以，在原语句S的条件下产生标记序列y的概率为

$$P(y \mid S) = \frac{e^{s(X,y)}}{\sum_{\tilde{y} \in Y_X} e^{s(X,\tilde{y})}} \qquad (2)$$

在训练的过程中，标记序列的似然函数为 $\log(p(y \mid S)) = s(X,y) - \log(\sum_{\tilde{y} \in Y_X} e^{s(X,\tilde{y})})$，其中，$Y_X$ 表示所有可能的标记集合，包含不符合IOBES标记规则的标记序列。通过公式（2）得到有效合理的输出序列。预测时，由（3）输出整体概率最大的一组序列

$$y^* = \arg\max_{\tilde{y} \in Y_X} s(X,\tilde{y}) \qquad (3)$$

(paper:Natural Language Processing (almost) from Scratch,2011,Collobert)

Dropout

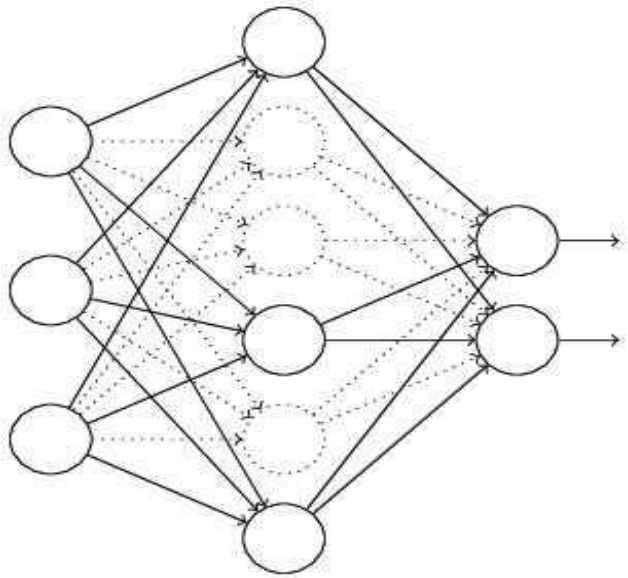
在主体模块中，在Bi-LSTM的输入部分——word embeddings增加Dropout，可以减轻模型过拟合的问题，dropout值选为0.5。使用dropout后，训练过程变为：

首先，随机删掉网络中一半（p=0.5）的隐藏神经元，输入输出神经元保持不变（虚线表示部分临时被删除的神经元）；

然后把输入x通过修改后的网络向前传播，把得到的损失结果通过修改的网络反向传播，一小批训练样本执行完这个过程之后就按照随机梯度下降法更新没有被删除的神经元相应的参数（w，b）；

恢复被删除的神经元。此时，被删除的神经元保持原样，而没有被删除的神经元已经有所更新。

重复上述过程。



Dropout

在训练每一小批数据时，随机地临时删掉一部分神经元，反向传播时只更新未删除的神经元对应的权值，删除的神经元的权值保持不变。因此，dropout掉不同的神经元等价于训练不同的网络。

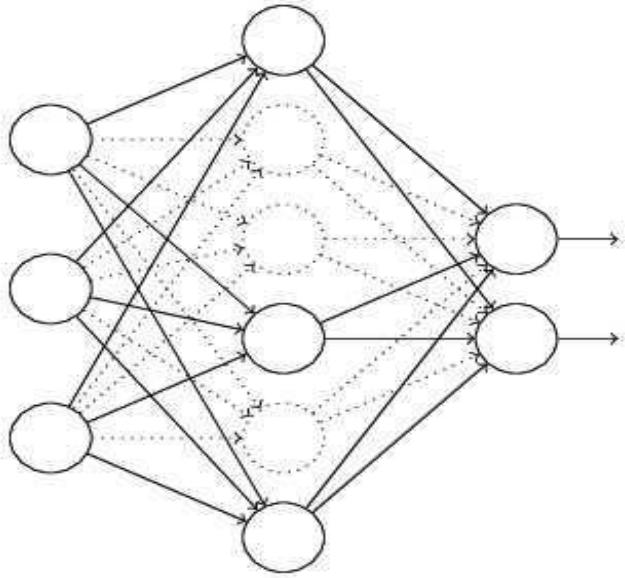
为什么dropout可以防止过拟合？

“综合取平均”策略

- 整个dropout过程相当于对多个不同的神经网络取平均。这种“综合取平均”的策略通常可以有效地防止过拟合问题。因为不同的网络可能产生不同的过拟合，取平均有可能让一些“相反”的拟合相互抵消掉，从而在整体上减少过拟合。

减少神经元间的共适应关系

- dropout使得两个神经元不一定每次都出现在dropout后的网络中，阻止了某些特征仅在其他特定特征下才有效果的情况，迫使网络去学习更加鲁棒的特征。



优化器

- 梯度下降法、随机梯度下降法、批量梯度下降法
- 动量法
- Adagrad
- Adadelta
- Rmsprop
- Adam

优化器

优化目标函数：

$$J(\theta_0,\theta_1)=\frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})^2$$

其中，$x^{(i)}$表示向量x中的第i个元素，$y^{(i)}$表示向量y中的第i个元素，$h_\theta(x^{(i)})$表示已知的假设函数，m为训练集的数量。

目标函数的作用是，让模型分类的错误率尽可能小。为了达到这一目的，模型的训练通常需要给参数一个随机初始值，然后用最优化方法寻找使得分类错误率更小的参数设置。

目前，比较常用的神经网络参数初始化方法——Xavier initialization，特点是根据某一层网络的输入、输出节点数量自动调整最合适的分布。一般采用的分布有两种：

高斯分布：$W \sim N(0,\frac{2}{n_{in}+n_{out}})$

均匀分布：$W\sim U(-\sqrt{\frac{6}{n_{in}+n_{out}}},\sqrt{\frac{6}{n_{in}+n_{out}}})$

优化器

批量梯度下降法

在整个数据集上，对每个参数θ求目标函数J(θ)的梯度。

Repeat {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

(simultaneously update θ_j for $j = 0, \ldots, n$)

}

批量梯度下降的代码实现：

```
for i in range(epoches):
    params_grad = grad(loss,data,parms)
    params = params - lr * params_grad
```

缺点：

1.每次更新都需要在整个数据集上计算出目标函数的偏导数，计算速度慢；

2.对于数据量大、内存无法容纳的数据集，该方法无法使用；

3.批量梯度下降法不能“在线”更新模型。

随机梯度下降法

在每轮迭代中，随机选择一个样本$x^{(i)}$对目标函数求偏导数，更新参数：

$$\theta = \theta - \eta \nabla_\theta J(\theta; x^{(i)}, y^{(i)})$$

随机梯度下降法代码：

```
for i in range(epoches):
    np.random.shuffle(data)
    for example in data:
        params_grad = grad(loss,data,parms)
        params = params - lr * params_grad
```

优点：

运行速度快，支持“在线”学习；

缺点：

随机梯度下降法更新值的方差大，在频繁的更新下，目标函数存在剧烈波动。

Mini-batch梯度下降法

Mini-batch梯度下降法结合了前两种方法的优点。在每次更新中，对n个样本构成的一批数据计算目标函数，并对相应的参数求偏导数：

$$\theta = \theta - \eta \nabla_\theta J(\theta; x^{(i:i+n)}, y^{(i:i+n)})$$

Mini-batch梯度下降法代码：

```
for i in range(epoches):
    np.random.shuffle(data)
    for batch in get_batech(data,bs=50)
        params_grad = grad(loss,batch,parms)
        params = params - lr * params_grad
```

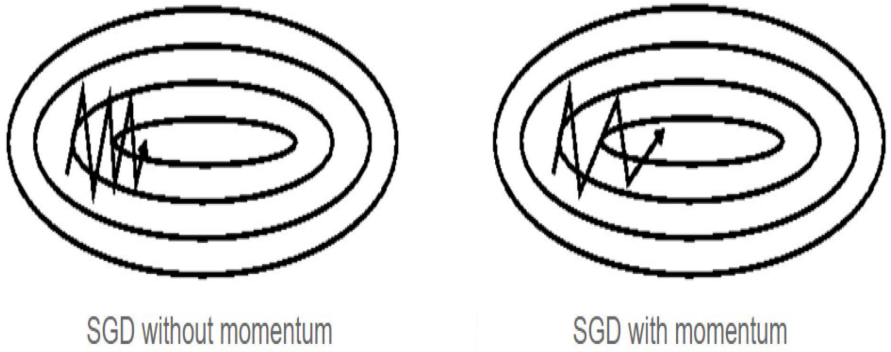
优点：

1.降低了更新参数的方差，使得收敛更为稳定；

2.能够利用深度学习库中高度优化的矩阵计算模块，高效求出每一小批量数据的梯度。

优化器

动量法



$$v_0 = 0$$

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta J(\theta)$$

$$\theta = \theta - v_t$$

动量法，模仿物理中动量的概念。在梯度指向相同的维度增加动量，在梯度指向相反的方向减少动量。
动量法，在相关方向上加速SGD，能够抑制动荡，从而加快收敛。

Adagrad

上述优化方法所有参数的学习率都是相同的，而Adagrad为不同的参数指定不同的学习率，对出现频次低的参数进行较大的更新，对出现频次高的参数进行较小更新。该方法适合处理稀疏数据。

$$g_{t,i} = \nabla_\theta J(\theta_i)$$

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \varepsilon}} \cdot g_{t,i}$$

G_t 矩阵是一个对角阵，对角线上的第i个元素等于从一开始到t时刻参数 θ_i 的梯度平方和，ε是平滑项，以避免分母为0.
缺点：
当梯度平方和累积到一定程度，学习率最终会趋向于一个无限小的值，此时已经不能从训练数据中学习到其他的信息。

优化器

Adadelta

Adadelta为解决Adagrad的缺点诞生，该方法仅计算一个时间区间内的梯度平方和。但该方法不存储之间t-1步的梯度平方和，而是按照递归的方法将梯度值累积，当前的梯度平方平均值等于之前梯度平方平均值与当前梯度平方的加权和。而且，该方法解决了动量法以及Adagrad方法的量纲不一致问题。

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\lambda) g_t^2, \quad RMS[g]_t = \sqrt{E[g^2]_t + \varepsilon}$$

$$E[\Delta\theta^2]_t = \gamma E[\Delta\theta^2]_{t-1} + (1-\lambda)\Delta\theta_t^2, \quad RMS[\Delta\theta]_t = \sqrt{E[\Delta\theta^2]_t + \varepsilon}$$

最终参数更新的公式为

$$\Delta\theta_t = -\frac{RMS[\Delta\theta]_{t-1}}{RMS[g]_t} g_t$$

$$\theta_{t+1} = \theta_t + \Delta\theta_t$$

RMSprop

该方法同样是用来解决Adagrad问题的，与Adadelta几乎同时发展出来。

$$E[g^2]_t = 0.9 E[g^2]_{t-1} + 0.1 g_t^2$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \varepsilon}} g_t$$

作者Geoff Hinton建议γ取0.9，学习率η取0.001.

优化器

Adam

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$$

由于m_0和v_0初始化为0，在开始的几步，m_t和v_t趋于0，因此加上纠正系数，

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

所以，$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \varepsilon}} \hat{m}_t$

作者认为参数的默认值分别为：$\beta_1 = 0.9, \beta_2 = 0.009, \varepsilon = 10^{-8}$。

优化器

如何选择优化器？

如果输入数据较为稀疏，使用自适应性学习类型的算法会有助于得到好的结果。此外，使用该方法的另一好处是，在不调参、直接使用默认值的情况下，就能得到很好的结果。

总的来说，RMSprop是一种基于Adagrad拓展的算法，从根本上解决了学习率骤缩问题；Adadelta法与RMSprop大致相同。而Adam法，则基于RMSprop法添加了偏差修正项和动量项。RMSprop、Adadelta及Adam法都是非常相似的算法，在相似情况下都能做的很好。

实验分析

Research Analysis

实验环境
Working environment

机器：MacBook Pro　处理器 2.7GHz Intel Core i5　　内存 8GB

操作系统：OS X EI Capitan 10.11.6

编程语言：python

分词工具库：jieba

CRF库：sklearn-crfsuite

深度学习框架：Theano

word2vec模型：Skip-gram

一、缺乏标注语料——人工标注

由于语料中的创业公司名称及投资机构名存在大量重复，而阅读速度明显高于标注速度。因此，在标注的过程中，不是直接标注，而是将公司名称摘录到实体字典中，最后根据模式匹配快速生成标注，从而节约劳动力、提高标注效率，同时为后续模型的训练提供高质量的词典。

基于字典的标注方法：
初始版本：倒排索引。由于jieba分词存在错误，不能保证形成哈希表时的关键字与语料分词一致，导致字典中出现的部分公司名称不能正确标注。

```
#"搜索引擎模式,该方法适合用于搜索引擎构建倒排索引的分词，粒度比较细"
for x in jieba.cut_for_search(line):
    if line.lower() not in hash_words[x.lower()]:
        hash_words[x.lower()] += [line.lower()]
```

解决方法：以字典中每个公司名的每个字为哈希表的关键字，值是公司名，然后对每个key中的value按照长度由大到小排序。

```
#将中文公司名中的字作为索引
for x in ' '.join(line).split():
    if line.lower() not in hash_words[x.lower()]:
        hash_words[x.lower()] += [line.lower()]

for k in hash_words.keys():
    hash_words[k].sort(key = lambda x: len(x), reverse = True)
```

二、训练时间长，调参成本高

考虑到这一点，在实验中使用的优化器都是Adam，可以减少对学习率的调节，同时只需要二十几轮迭代就可以得到最佳效果；并使用early stopping。（平均10 epoches需要6h左右，至少训练30 epoches）

三、过拟合问题

由于在标注数据集时顺便生成了字典，数据集中的实体基本都在字典中，因此，在加上字典特征后，验证集和测试集的F1分别达到97.5%、96.51%。进一步实验，在训练集上使用字典特征训练，而在进行验证集及测试集评估的时候，字典特征向量置零，表示所有的词语都不在字典中，此时验证集和测试集的F1都不到80%，存在严重的过拟合。因此，通过实验选择一定的比例，从字典中随机采样一部分公司名形成小规模的字典加进模型进行训练。

对于过拟合问题，还可以通过增加训练数据量来解决。

四、数据分布

在将数据集随机划分为训练集、验证集及测试集时，尽量保证三个数据集数据同分布。个人觉得第一衡量标准：当样本量大致相当时，文件大小差异不大。

实验数据
Working environment

Train set sents	Dev set sents	Test set sents	unique words	unique chars	words initialized by pretrained word embeddings
6294	1150	1250	19011	3025	86.3%

实验结果
Research Results

method	Dev F1	Test F1
baseline CRF	89.91%	90.5%
1 char+word embeddings，gaze，Bi-LSTM,CRF	94.41%	94%
2 char+word embeddings,no gaze,Bi-LSTM,CRF	93.9%	92.64%
3 word embeddings,gaze,Bi-LSTM,CRF	94.3%	94%
4 word embeddings,no gaze,Bi-LSTM,CRF	93.58%	92.31%
5 radical+char embeddings,gaze,Bi-LSTM,CRF	92.84%	91.83%
6 radical+char embeddings,no gaze,Bi-LSTM,CRF	86.94%	86.37%
7 char embeddings,layer 1,gaze,Bi-LSTM,CRF	88.74%	88.47%
8 char embeddings,layer 1,no gaze,Bi-LSTM,CRF	86.33%	86.67%
9 char embeddings,layer 2,gaze,Bi-LSTM,CRF	89.11%	88.98%
10 char embeddings,layer 2,no gaze,Bi-LSTM,CRF	87.41	86.23%

Table 1 NER results on dev and test data.(除CRF，其他模型框架与前面的介绍类似)

注：训练集：用于训练模型的数据集；验证集：用于模型的选择、参数调试；测试集：用于对学习方法的评估。在学习到的不同复杂度的模型中，选择对验证集有最小预测误差的模型。

Paper: Character-Based LSTM-CRF with Radical-Level Features for Chinese Named Entity Recognition, 2015, NLPCC,（自动化所）

分析

从上述结果可知，在中文公司名命名实体识别任务中：

- 使用词向量的效果明显优于字向量，词向量所包含的信息量远比字向量丰富；
- 在词向量的基础上，加入字向量信息，能够对pretrained word embeddings中未见词汇有所帮助。

在char+word embeddings、gaze、Bi-LSTM、CRF这组实验中，测试集中：

- 不在训练集的实体数约占总数的35%；
- 预测错误的实体数约占总数的13%；
- 预测错误的实体中，超过70%的不在训练集中；
- 其他预测错误的实体主要是由于在训练集中仅出现一两次。

解决方案：由于错误实体数量不大，可以将判错的实体加入到人工字典中。

method	Dev F1	Test F1
1 char+word embeddings,no gaze,Bi-LSTM,CRF	93.9%	92.64%
2 char+word embeddings,gaze,Bi-LSTM,CRF	94.41%	94%
3 char+word embeddings, gaze+wrong,Bi-LSTM,CRF	94.76%	94.14%

谢谢