ELMo

原创 GG 熊熊nlp白板报 2018-11-12

本文的目的是介绍ELMo语言模型，即Embeddings from Language Models (Clark et al. 2018.Deep contextualized word representations).

语言模型的目标即：compute the probability of a sentence or a sequence of words:

$$P(w_1 w_2 ... w_N) = \prod_{i=1}^{N} P(w_i | w_1 ... w_{i-1})$$

语言模型的子任务即：compute the probability of an upcoming word:

$$P(w_i | w_1 ... w_{i-1})$$

假设已经有了一个语言模型，其可以计算任何句子的概率，那么如何评价这个模型的优劣呢？于是有了评价语言模型的指标：

Perplexity is the inverse probability of the test set, normalized by the number of words：

$$Perplexity = \sqrt[N]{\frac{1}{P(w_1 w_2 ... w_N)}} = \sqrt[N]{\frac{1}{\prod_{i=1}^{N} P(w_i | w_1 ... w_{i-1})}}$$

如何计算上述公式中的条件概率呢？

传统方法主要包括以下两类模型：Count-based N-gram和Neural-based N-gram LM。

1 Count-based N-gram LM

Count-based N-gram LM是基于马尔可夫假设，即当前状态仅与此前的若干个状态有关，所以条件概率的计算转化为：

$$P(w_i | w_1 ... w_{i-1}) \simeq P(w_i | w_{i-k} ... w_{i-1})$$

Count-based N-gram LM计算条件概率时是基于极大似然估计：

$$P(w_i | count(w_{i-k} ... w_{i-1})) \simeq \frac{count(w_{i-k} ... w_i)}{count(w_{i-k} ... w_{i-1})} \qquad (k > 1)$$

当k很大的时候，容易出现数据稀疏，所以通常k=[2, 5]，这也意味着模型包含的上文信息比较有限。

2 Neural N-gram LM

Neural N-gram LM就是通过神经网络来计算上述的条件概率，传统上有以下两种结构：

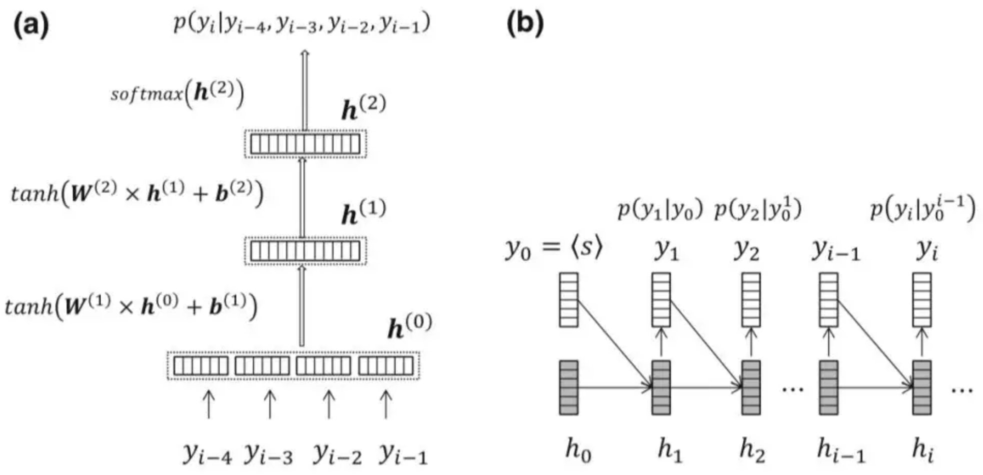
(a)
$p(y_i|y_{i-4}, y_{i-3}, y_{i-2}, y_{i-1})$
$softmax(\boldsymbol{h}^{(2)})$
$\boldsymbol{h}^{(2)}$
$tanh(\boldsymbol{W}^{(2)} \times \boldsymbol{h}^{(1)} + \boldsymbol{b}^{(2)})$
$\boldsymbol{h}^{(1)}$
$tanh(\boldsymbol{W}^{(1)} \times \boldsymbol{h}^{(0)} + \boldsymbol{b}^{(1)})$
$\boldsymbol{h}^{(0)}$
y_{i-4} y_{i-3} y_{i-2} y_{i-1}
(b)
$p(y_1|y_0)$ $p(y_2|y_0^1)$ $p(y_i|y_0^{i-1})$
$y_0 = \langle s \rangle$ y_1 y_2 y_{i-1} y_i
...
h_0 h_1 h_2 h_{i-1} h_i

图1 （来源：Li Deng， Yang Liu. 2018. Deep Learning in Natural Language Processing)

上图中，(a) 表示基于固定窗口上文（a fix-sized window context）的前馈NN，(b)表示基于所有上文(all context before the word)的RNN。

在训练的时候，上述两种结构的目标是，在整个语料库上最大化（maximize）下列公式：

$$log(P(w_1 w_2 ... w_N)) \qquad (N\text{为语料中词的个数})$$

为了便于学习，上述优化目标也等同于，最小化（minimize）整个语料库的复杂度：

$$log(Perplexity) = \frac{-1}{N}\sum_{i=1}^{N} log(P(w_i|w_{i-k}...w_{i-1})) \qquad (N\text{为语料中词的个数})$$

在训练的时候，通常采用交叉熵计算损失：

$$L_{CE} = -\sum_{i=1}^{N}\sum_{k=1}^{C} t_{ik} log(y_{ik})$$

$$(i\text{为样本序号，}k\text{为预测类别的序号，}t\text{为样本的值，}y\text{为模型预测的值，}N\text{为样本数})$$

在LM中，样本值 t 是词表的one-hot表示，模型预测值 y 是上文的条件概率，因此：

$$L_{CE} = -\sum_{i=1}^{N} 1 * log(y_{ik}) = -\sum_{i=1}^{N} log(P(w_i|w_{i-k}...w_{i-1})) \qquad (t_{ik} = 1)$$

因此，训练结束后，从损失就可以直接得到模型在当前训练语料上的复杂度，进而可以评价模型优劣。

3 biLMs

上文中的语言模型都是基于上文的forward-only LMs, Peters et al.(2017)用了基于上文和下文的LM来做序列标注任务，这个模型即Bidirectional LM（biLM），包括forward LM和backward LM：

$$P(w_1 w_2 \ldots w_N) = \prod_{i=1}^{N} P(w_i | w_1 \ldots w_{i-1})$$
$$P(w_1 w_2 \ldots w_N) = \prod_{i=1}^{N} P(w_i | w_{i+1} \ldots w_N)$$

因此对位置为k的词的LM embedding为:

$$h_k^{LM} = [\overrightarrow{h}_k^{LM} \quad ; \quad \overleftarrow{h}_k^{LM}]$$

（Peters et al.2017)采用了下图的结构来获取其LM embedding：

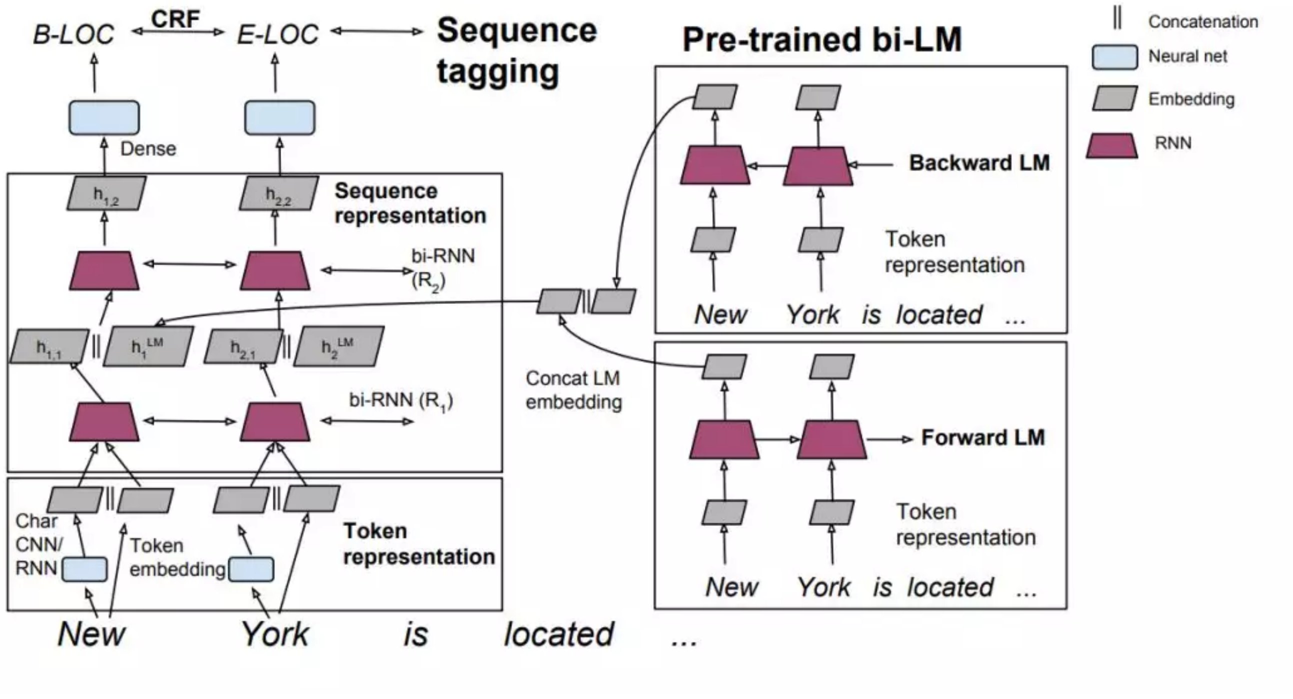

图2 （Peters et al.2017)的TagML

在（Peters et al. 2017)之前，在LM结构上与其相似的是（Kim et al. 2015）以及（Jozefowicz et al. 2016），如下两个图所示：

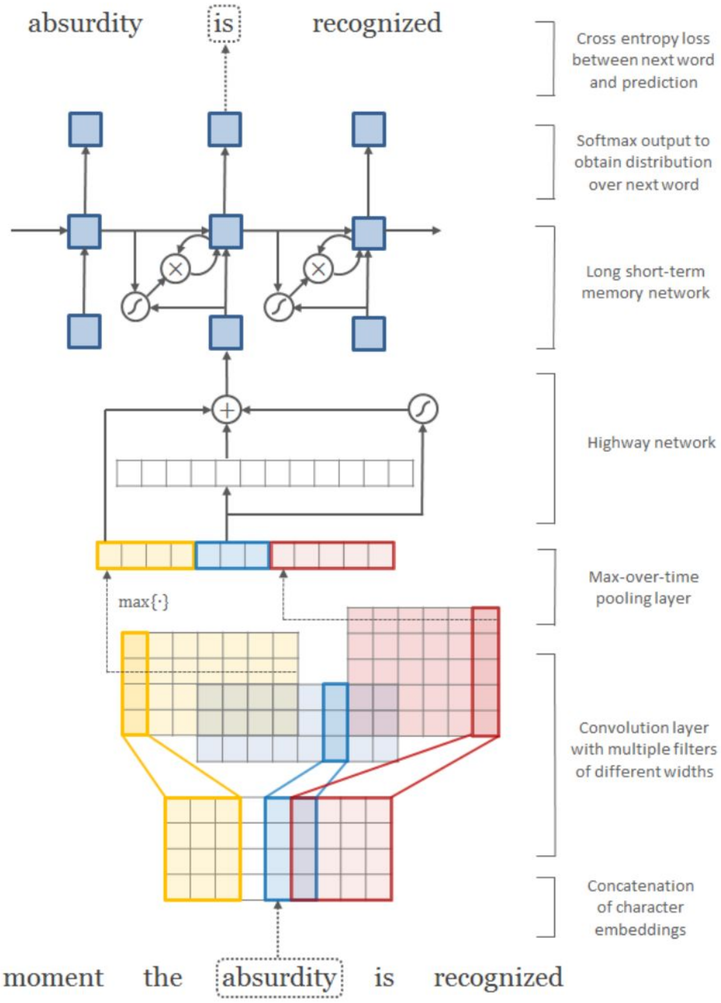
absurdity
is
recognized
Cross entropy loss between next word and prediction
Softmax output to obtain distribution over next word
Long short-term memory network
Highway network
Max-over-time pooling layer
max{·}
Convolution layer with multiple filters of different widths
Concatenation of character embeddings
moment
the
absurdity
is
recognized

图3（Kim et al.2015）的lstm-character-cnn

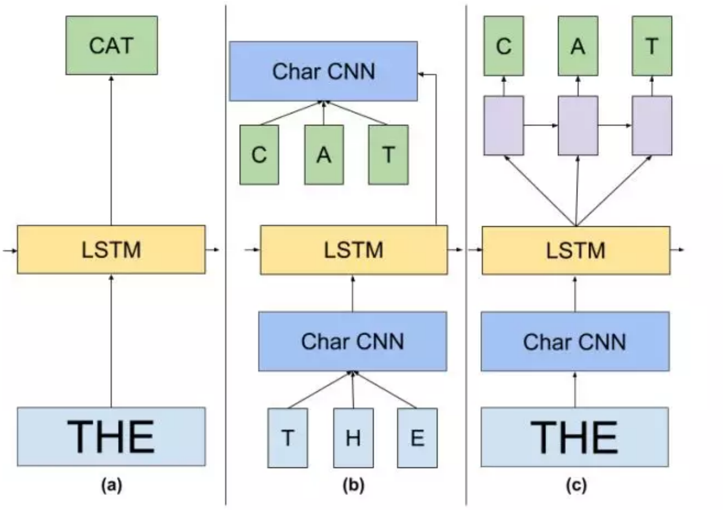

图4 （Jozefowicz et al. 2016）的CNN-BIG-LSTM

上述两者都加入了Char-CNN，好处在于捕捉了更多的subword information(e.g. morphemes)，因此模型能够学习到：

> ***eventful***, ***eventfully***, ***uneventful***, and ***uneventfully*** should have structurally related embeddings in the vector space

（Kim et al.2015）的lstm-character-cnn的结构中还包括了一个Highway Layer（最后实现表示L=2效果最好），Highway的计算方式为：

$$z = t \cdot g(W_H * y + b_H) + (1 - t) \cdot y$$
$$t = \sigma(W_T * y + b_T)$$
$$(t : transform \quad gate, \quad 1 - t : carry \quad gate)$$

对于hightway的作用为，作者说到：

> Before the highway layers the representations seem to solely rely on surface forms—for example the nearest neighbors of ***you*** are ***your***, ***young***, ***four***, ***youth***, which are close to you in terms of edit distance. The highway layers however, seem to enable encoding of semantic features that are not discernable from orthography alone. After highway layers the nearest neighbor of ***you*** is ***we***, which is orthographically distinct from you. Another example is ***while*** and ***though*** these words are far apart edit distance-wise yet the composition model is able to place them near each other.

（Peters et al. 2017）对于（Jozefowicz et al. 2016）的CNN-BIG-LSTM说到：

> (Jozefowicz et al., 2016) pass a token represention (either from a CNN over characters or as token embeddings) through multiple layers of LSTMs to embed the history(t1,...,tk) into a fixed dimensional vector, this is the forward LM embedding of the token at positon k and is the output of the top LSTM layer in the LM.

4 ELMo

终于来到了本文的终点和重点ELMo，其简要的网络结构如下图所示：

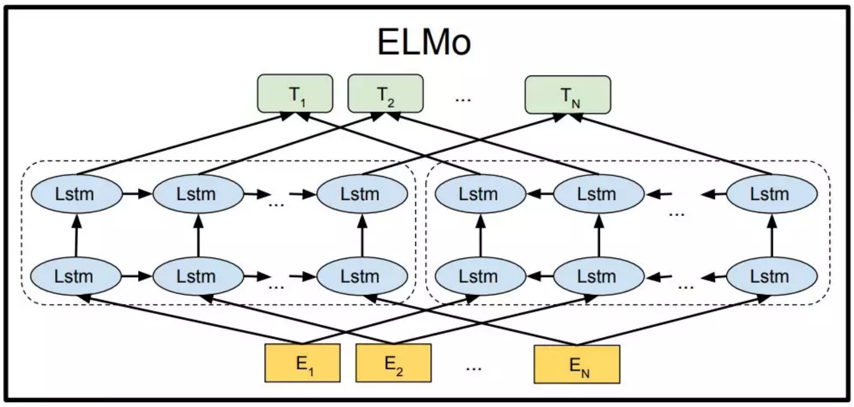

图5 Google BERT的ELMo structure

ELMo从biLM获取所有层的中间表示，根据不同的任务对其进行线性组合，如下所示：

$$
\begin{aligned}
R_k &= [X_k^{LM}; \overrightarrow{h}_{kj}^{LM}; \quad \overleftarrow{h}_{kj}^{LM} \; | j = 1, ..., L] \\
X_k^{LM} &= h_{k,0}^{LM} \\
R_k &= [h_{k,j}^{LM} \quad | j = 0, ...L] \\
ELMo_k^{task} &= E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} h_{k,j}^{LM}
\end{aligned}
$$

$$
(X_k^{LM}\text{是当前词的}TokenEmbedding, k\text{是当前词的位置，层数}j = 0, ..., L)
$$

作者论文中预训练的biLM模型与与（Jozefowicz et al. 2016）的结构相似。从其公开的官方tf项目看到，其biLM的实现包括了此前的所有特点：bidirectional + word embedding (or char cnn embedding) + highway + residual connection between lstm layers，作者自己说到：

> The final model uses L=2 biLSTM layers with 4096 units and 512 dimension projections and a residual connection from the first to second layer.
>
> The pretrained biLMs in the paper are similar to the architectures in（Jozefowicz et al. 2016），but modified to support joint training of both directions and add a residual connection between LSTM layers. We focus on large scale biLMs.

总的来说，ELMo的特点总结如下：

- 捕捉subword information: 利用了 character convolutions；
- 捕捉词的multi-sense：利用了context-dependent embedding，即先对句子建模(functions of input sentence)；
- 捕捉不同层次的语法信息：2-layer LSTM中第一层倾向捕捉词法信息，第二层倾向捕捉语义信息。

对于捕捉不同层次的信息，作者进行实验的结果如下图所示：

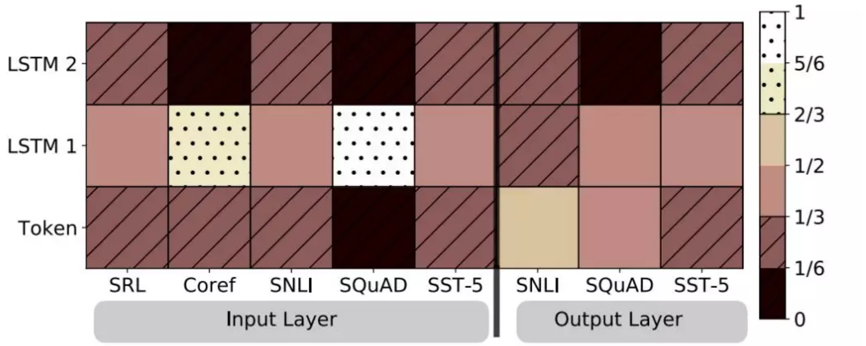

图6 （Clark et al. 2018）的biLM各层权重的可视化

如上图所示，横坐标表示不同的任务，纵坐标表示s_j_task的权重大小，其区间为[0, 1]。emlo embedding用于input layer时，lstm-layer-1的作用更大，尤其对于指代和阅读理解任务（权重取值

区间为[2/3, 5/6]）。

此外作者的实验也包含了WSD词义消解任务和POS词性标注任务，前者更喜top-layer，后者是lstm-layer-1的作用更大。

5 Reference

1. Clark et al. 2018.Deep contextualized word representations.
2. Peters et al. 2017. Semi-supervised sequence tagging with bidirectional language models.
3. Kim et al.2015. Character-Aware Neural Language Models.
4. Jozefowicz et al. 2016. Exploring the Limits of Language Modeling.

6 About

```
@author:  donggui@uzoo.cn

@date:  Nov.  11,  2018
"""

本文只对elmo的历史和原理进行了介绍，还确少中文任务上的实验(//todo)

"""
```

喜欢此内容的人还喜欢

如何评价《唐人街探案 3 》
卢克文工作室

“你男票8套房，能让给我吗？”过年千万不要带男朋友回家！否则...
吐槽星君