Deep contextualized word representations

- 动机

  相比于以往的语言模型elmo具有以下两大提升

  - 能够学习到词汇用法的复杂性，例如语法，句法
  - 能够学习到不同上下文的语义多样性
- 模型结构

  ELMo是从大量的文本和双向语言模型学习得到，从中得到内部状态

  - 双向语言模型
    - 前向语言模型是指得到序列$t_1, t_2 ..... t_N$的概率，公式如下所示：

      $p(t_1, t_2 ... t_N) = \prod_{k=1}^{N} p(t_k | t_1, t_2 ..... t_{k-1})$

      最近，如《Exploring the limits of language modeling》、《On the state of the art of evaluation in neural language models》和《Regularizing and optimizing lstm language models》等论文中，首先使用character-level的RNN或CNN，计算得到“上下文无关”（context-independent）词向量表示X_k^{LM}，然后将这个词向量表示放入L层LSTM中，对于位置k的词向量来说，每层LSTM输出一个上下文相关的$\vec{h}_{k,j}^{LM}$，其中$j = 1, 2....L$，最顶层的LSTM输出为$\vec{h}_{k,L}^{LM}$，然后加上softmax来预测t_{k+1}。
    - 后向语言模型即通过后面的词语来预测前面的词语

      $p(t_1, t_2 ... t_N) = \prod_{k=1}^{N} p(t_k | t_{k+1}, t_{k+2} ..... t_N)$
    - 双向语言模型（biLM）将前后向语言模型结合起来，最大化前向、后向模型的联合似然函数即可，如下式所示：

      $\sum_{k=1}^{N} (logp(t_k | t_1, t_2, ..., t_{k-1}; \Theta_x, \vec{\Theta}_{LSTM}, \Theta_s) + logp(t_k | t_{k+1}, t_{k+2} ..... t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s))$

      其中Θ_x和Θ_s是context-independent词向量训练时和softmax时的参数，$\vec{\Theta}_{LSTM}$和$\overleftarrow{\Theta}_{LSTM}$是双向语言模型的参数。
    - ELMo

      ELMo是中间双向LSTM层的结合，对于一个词语t_k，一个L层的双向语言模型可以计算得到t_k的2L+1个状态

      $R_k = X_k^{LM}, \vec{h}_{k,j}^{LM}, \overleftarrow{h}_{k,j}^{LM} | j = 1, ..., L$

      $R_k = h_{k,j}^{LM} | j = 0, ..., L$

      其中，$h_{k,j}^{LM} = [\vec{h}_{k,j}^{LM}, \overleftarrow{h}_{k,j}^{LM}]$。最终得到线性组合的单元集合

      $$\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}$$

      其中，s_j^{task}是softmax-normalized权重，γ^{task}是根据具体的任务去放缩ELMO的参数
  - 利用双向语言模型来完成有监督的语言任务

        - 在RNN任务中，将原先的词向量x_k替换为$[x_k; ELMo_k^{task}]$
        - 可以h_k替换为$[h_k; ELMo_k^{task}]$
        - 增加dropout层和L2 loss的方法来提升效果，其中L2的系数λ越大，越代表了取各层平均的意思，越小，越代表了各层发挥不同的效果。
    - 预训练双向模型结构
        - 这里采用的还是语言模型去训练网络. 并且这篇的模型是有借鉴先行研究的. Exploring the limits of language modeling. Character-Aware Neural Language Models 这两个先行研究都是基于字符卷积网络来做的, 但是这其中有一个问题是字符卷积网络的低效问题, 详细见论文， 通过这篇论文我们也知道, 能够实现最大效率利用字符共现来表达词义的网络就是 biLSTMs 网络, 这也是为什么这个模型采用这个网络的原因. 但是不同于这两个论文的是, 这个论文增加了
            - 联合双向训练
            - LSTM层之间增加了残差连接
    - 结果分析
        - 各层加权方案

          上面说过, L2 loss的系数λ给模型带来的影响, 就是, 其值越大, 最后的$ELMO_t^{task}$越趋近于各层之间的平均值. 下面是对四个不同的构造进行的比较,

            - 第一列是baseline模型(使用普通的词向量, 例如 CoVe)
            - 第二列是只使用了 BiLSTMs 最后一层输出的结果.
            - 第三列是使用了各层次状态值平均的结果(倾向于平均, 并非完全平均, $\lambda = 1$)
            - 第四列是使用了各层次状态值加权和的结果($\lambda = 0.001$)

Task	Baseline	Last Only	All layers	
			λ=1	λ=0.001
SQuAD	80.8	84.7	85.0	**85.2**
SNLI	88.1	89.1	89.3	**89.5**
SRL	81.6	84.1	84.6	**84.8**

          可以看出第四列效果最好，说明将各层发挥自己的特点是最好的

        - 将ELMo加入到哪里

          前面说过可以在output层也加入ELMo模型，实验结果如下所示：

Task	Input Only	Input & Output	Output Only
SQuAD	85.1	**85.6**	84.8
SNLI	88.9	**89.5**	88.7
SRL	**84.7**	84.3	80.9

        - 双向语言模型中捕获了哪些信息
            - 语义消歧

任务这里使用了 SemCor3.0 这个预料库, 这是一个标注了多义的预料库, 库中的每个词汇都对应着 wordnet 的一个位置. 基于该预料库进行计算的方法是, 先利用 BiLMs 来计算出了语料库中所有的词汇向量表示. 然后将位于wordnet相同位置的词汇的向量取了平均. 测试的时候, 对于一个给出的target word in a target sentence. 利用 BiLM 得出结果后, 利用训练时获得的每个wordnet位置中的词汇的初始向量, 再利用 1近邻法求这个词汇可能的位置. 这个模型本身可以说是非常简单了, 只用到了 1-近邻. 但是结果还是显示, 有很高的F1值. 如下图:

Model	F_1
WordNet 1st Sense Baseline	65.9
Raganato et al. (2017a)	69.9
Iacobacci et al. (2016)	**70.1**
CoVe, First Layer	59.4
CoVe, Second Layer	*64.7*
biLM, First layer	67.4
biLM, Second layer	*69.0*

- **POS tagging**

  上面的结果测试了 ELMo 表达语义的效果, 接下来利用 POS tagging 检测其对语法的表达效果. 这个是用向量作为输入进一个分类器, 去分辨词性, 也可以说是非常简单了. 其结果如下:

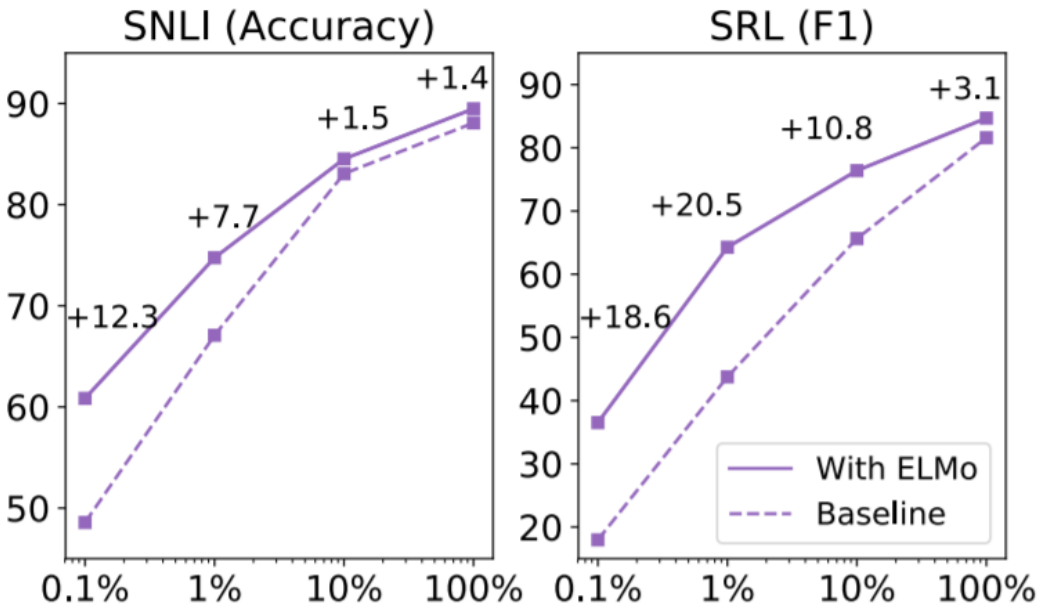
Model	Acc.
Collobert et al. (2011)	97.3
Ma and Hovy (2016)	97.6
Ling et al. (2015)	**97.8**
CoVe, First Layer	*93.3*
CoVe, Second Layer	92.8
biLM, First Layer	*97.3*
biLM, Second Layer	96.8

- **Implications for supervised tasks**

  biLM中不同层的LSTM代表了不同的信息，对于下游任务来说，要将所有层的信息都结合进来

- **Implications for supervised tasks**

  在不加 ELMo 的情况下去训练 SRL model , 达到最佳 F1 值要在 486 个epoch之后, 但是!!!!!在加了 ELMo 之后, 只需要十个回合便超越了原来的 F1 值， 降低了98%的时间消耗。只需要更少的样品就能取得同样的结果，实验如下图所示：



- 结论

  作者提出了一种简单而有效的模型来替代了以往的语言模型，其实也为transformer和bert模型的提出埋下了伏笔。深度学习模型一般来说，越深越宽的模型肯定比浅层模型效果要好。network embedding的一些方法是基于word2vec的方法来完成的，在阅读这篇论文时，我就在思考一个问题，能够将这些模型套入到network embedding的方法里面呢，现在看来是有难度的，在语言模型中，一句话中的词地位应该是差不多的，但是在网络或者图中，一阶邻居，二阶邻居，乃至三阶邻居对于一个结点的重要性是不一样的。