# 基于深度神经网络的语义角色标注模型

## 1、 背景介绍

语义角色标注是实现浅层语义分析的一种方式。在一个句子中，谓词是对主语的陈述或说明，指出“做什么”、“是什么”或“怎么样，代表了一个事件的核心，跟谓词搭配的名词称为论元。语义角色是指论元在动词所指事件中担任的角色。主要有：施事者（Agent）、受事者（Patient）、客体（Theme）、经验者（Experiencer）、受益者（Beneficiary）、工具（Instrument）、处所（Location）、目标（Goal）和来源（Source）等。

语义角色标注（Semantic Role Labeling，SRL）以句子的谓词为中心，不对句子所包含的语义信息进行深入分析，只分析句子中各成分与谓词之间的关系，即句子的谓词（Predicate）- 论元（Argument）结构，并用语义角色来描述这些结构关系，是许多自然语言理解任务（如信息抽取，篇章分析，深度问答等）的一个重要中间步骤。在研究中一般都假定谓词是给定的，所要做的就是找出给定谓词的各个论元和它们的语义角色。

近年来，深度学习在机器学习领域有了较大的进展，已经被广泛的应用于自然语言处理的很多领域上。这种方法的使用将人们从特征工程的工作中解放出来。同时，如 LSTM、GRU 等深度循环神经网络也能够更加科学的在句子上进行计算，有效地缓解句子中长距离依赖的问题。在很多自然语言处理的问题中，深度学习已经表现出其强大的优势。

在之前学者对 SRL 的研究中，需要进行大量特征工程工作，这也导致了模型的泛化能力不足。在最近的研究中，人们发现使用深度学习方法和循环神经网络模型，可以较好地表示上下文和句法特征，能够有效地减少特征工程工作，同时模型也能够得到更好的泛化效果[1-4]。本文将 LSTM 循环神经网络模型应用于 SRL，以对上下文和句法路径特征建模，在模型中自动学习解决问题的有效特征，从而达到减少特征工程，提高模型泛化能力的效果。

## 2、 问题描述

基于语块的 SRL 方法将 SRL 作为一个序列标注问题来解决。如图 1，本文序列标注任务采用 BIO 表示方式来定义序列标注的标签集。在 BIO 表示法中，B 代表语块的开始，I 代表语块的中间，O 代表语块结束。通过 B、I、O 三种标记将不同的语块赋予不同的标签，例如：对于一个角色为 A

的论元，将它所包含的第一个语块赋予标签 B-A，将它所包含的其它语块赋予标签 I-A，不属于任何论元的语块赋予标签 O。

| 输入序列 | 小明 | 昨天 | 晚上 | 在 | 公园 | 遇到 | 了 | 小红 | 。 |
|---|---|---|---|---|---|---|---|---|---|
| 语块 | B-NP | B-NP | I-NP | B-PP | B-NP | B-VP | | B-NP | |
| 标注序列 | B-Agent | B-Time | I-Time | O | B-Location | B-Predicate | O | B-Patient | O |
| 角色 | Agent | Time | Time | | Location | Predicate | O | Patient | |

图 1 BIO 标注方法示例

因此，本文的任务可以描述为，给定句子-论元对$(w, v)$, 得到预测序列 $y$。对于$y_i \ in \ y$, $y_i \epsilon \{O, B_r, I_r\}$。$r$ 代表语义角色。该任务即找出得分最高的标注序列：

$$\tilde{y} = argmax(f(w, y))$$

从根据序列标注结果可以直接得到论元的语义角色标注结果，是一个相对简单的过程。这种简单性体现在：（1）依赖浅层句法分析，降低了句法分析的要求和难度；（2）没有了候选论元剪除这一步骤；（3）论元的识别和论元标注是同时实现的。这种一体化处理论元识别和论元标注的方法，简化了流程，降低了错误累积的风险，往往能够取得更好的结果。

## 3、 解决方案

**本文选用的深度学习框架为 tensorflow 1.4.**

循环神经网络（Recurrent Neural Network）是一种对序列建模的重要模型，在自然语言处理任务中有着广泛地应用。不同于前馈神经网络（Feed-forward Neural Network），RNN 能够处理输入之间前后关联的问题。LSTM 是 RNN 的一种重要变种，常用来学习长序列中蕴含的长程依赖关系。

使用神经网络模型解决问题的思路通常是：前层网络学习输入的特征表示，网络的最后一层在特征基础上完成最终的任务。在 SRL 任务中，深层 LSTM 网络学习输入的特征表示，softmax 在特征的基础上完成序列标注，处于整个网络的末端。图 2 展示了端到端的语义角色标注网络，最终可以用$\sum_{t=1}^{n} log\ (y_t|w)$刻画每个可能的标注序列的得分。为了保证 BIO 结构一致性，定义如下包含惩罚因子的打分函数：

$$f(w, y) = \sum_{t=1}^{n} log\ (y_t|w) - \sum_{c \epsilon C} c(w, y_{1:t})$$

在解码时采用维特比算法过滤掉结构不一致的不合理序列。

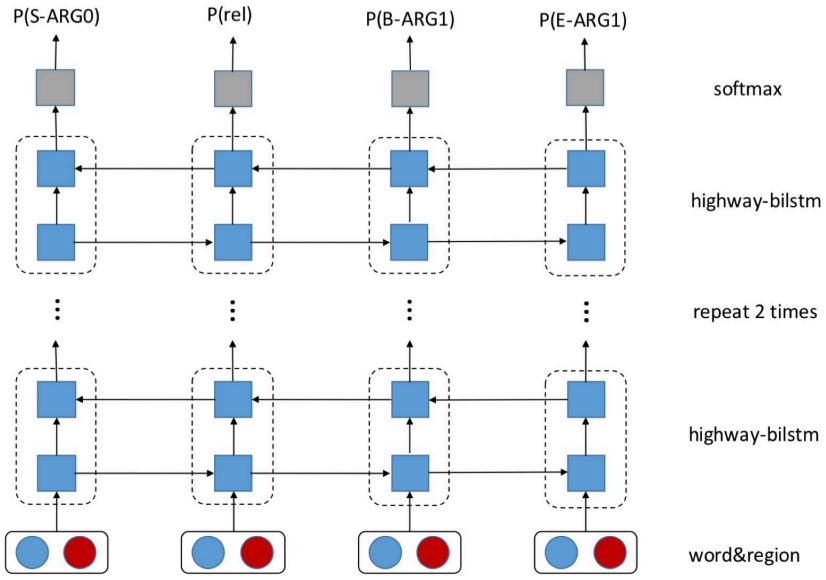


图 2 端到端的语义角色标注网络

下面分别阐述该网络结构的每一层。

3.1 词向量层(word_embedding layer)

受文献[1]和文献[2]的启发，不同于传统的 word_embedding 层，该层选取两个特征：word_id 和 region_mark_id。其中，region 被定义为，以标记是 rel 的谓语动词为窗口中心，分别向左右各拓展 context_length 步得到的包含(2*context_length+2)个词的上下文。图 2 中 context_length 设置为 0.

例如：我们 希望 台湾 当局 顺应 历史 发展 潮流 ， 把握/rel 时机 ， 就 两 岸 政治 谈判 作出 积极 回应 和 明智 选择 。

region

region_mark_id 定义如下：

$$region\_mark\_id = \begin{cases} 1 & if\ word\ position\ in\ the\ region \\ 0 & otherwise \end{cases}$$

该层 $t$ 位置的输出，也即神经网络第一层 $t$ 时间步输入, 可以表示为,

$$x_{1,t} = [W_{emb}(one - hot(word\_id)), W_{mask}(one - hot(region\_mark\_id)]$$

3.2 深度双向 LSTM

LSTM 结构如图 3 所示。第 $l$ 层 LSTM 单元在时刻 t 都有一个记忆单元$c_{l,t}$。LSTM 分为三个门控单元。

a) “忘记门”$f_{l,t}$ 决定会在多大程度上忘记旧记忆。

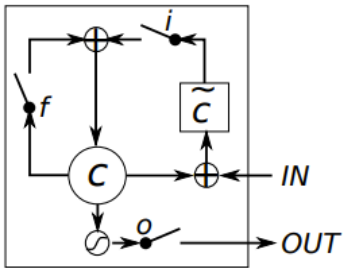


图 3 LSTM 结构

$$f_{l,t} = \sigma\left(W_{l,f}x_{l,t} + U_{l,f}h_{l,t-1} + V_{l,f}c_{l,t-1}\right)$$

b) “输入门”$i_{l,t}$决定在多大程度上将新产生的记忆内容$\tilde{c}_{l,t}$添加到 $t$ 时刻的记忆单元$c_{l,t}$中。

$$i_{l,t} = \sigma\left(W_{l,i}x_{l,t} + U_{l,i}h_{l,t-1} + V_{l,i}c_{l,t-1}\right)$$

$$\tilde{c}_{l,t} = tanh(W_{l,c}x_{l,t} + U_{l,c}h_{l,t-1})$$

依据输入门和遗忘门更新记忆：

$$c_{l,t} = f_{l,t}c_{l,t-1} + i_{l,t}\tilde{c}_{l,t}$$

c) 引入“输出门”$o_{l,t}$，对总体的记忆状态进行再一次过滤。

$$o_{l,t} = \sigma\left(W_{l,o}x_{l,t} + U_{l,o}h_{l,t-1} + V_{l,o}c_{l,t-1}\right)$$

$$h_{l,t} = o_t tanh(c_t)$$

一个双向 LSTM 由前向 LSTM 和后向 LSTM 组成。前向 LSTM 正序读序列输入，计算出前向隐藏状态的序列$(\overrightarrow{h}_1, \dots, \overrightarrow{h}_T)$; 后向 LSTM 反序读序列输出，计算出后向隐藏状态的序列$(\overleftarrow{h}_1, \dots, \overleftarrow{h}_T)$。综合以上，为了让隐藏状态同时保留前面词以及后面词的记忆，将时刻 $t$ 的隐藏状态记为$h_t = [\overrightarrow{h}_t^T; \overleftarrow{h}_t^T]^T$。对于深度多层模型来说，$l$ 层 $t$ 时刻的输入表示为$x_{l,t} = h_{l-1,t}$。

- **Highway Connection**

为了解决深层神经网络训练时梯度消失问题，层与层之间的连接选用 highway connection[6]。

这一操作启发自 Highway network。一个典型的神经网络是一个仿射变换加一个非线性函数，即

$y = H(x, W_H)$，在文献中，为了训练更深的神经网络，受 lstm 门机制的启发，为输出添加 transfer gate 和 carry gate，形成 Highway network：

$$y = H(x, W_H) \circ T(x, W_T) + x \circ C(x, W_C)$$

为简化计算，可以取 *C*=1-*T*。本质是在输出和网络层之间加了一个连接，直接让输入 X 的信息直接通过，不需要通过神经网络层，跟高速公路一样，因此命名为 highway network。

本文训练 4 层的双向 LSTM，层与层之间选用 highway connection，添加一个 transfer gate $r_{l,t}$ 控制输出，如图 5 所示。

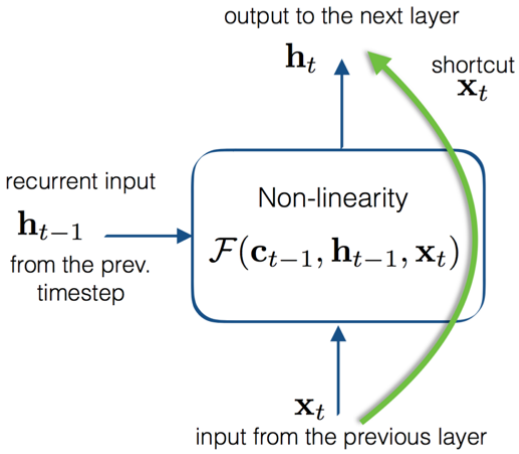


图 4 highway connection

$$r_{l,t} = \sigma(W_r^l[h_{l,t-1}, x_t] + b_r^l)$$

$$h'_{l,t} = o_{l,t} \circ \tanh(c_{l,t})$$

$$h_{l,t} = r_{l,t} \circ h'_{l,t} + (1 - r_{l,t}) \circ W_h^l x_{l,t}$$

- **dropout 机制**

为了避免过拟合，选用时间步共享的 dropout 机制[5]，如图 5 所示。

$$\tilde{h}_{l,t} = r_{l,t} \circ h'_{l,t} + (1 - r_{l,t}) \circ W_h^l x_{l,t}$$

$$h_{l,t} = z_l \circ \tilde{h}_{l,t}$$

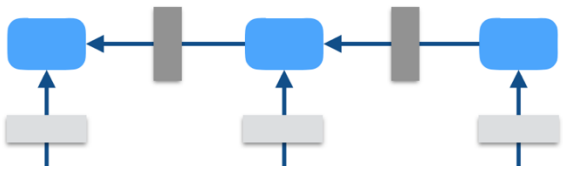

图 5 dropout 机制

3.3 维特比解码层(Viterbi decoder layer)

在多层 LSTM 后接 **softmax/CRF** 进行标签预测，为了减少标签种类，将 BIOES 边界标签转化为 BIO 标签；同时为了防止 BIO 混乱问题，引入维特比解码算法，限制不合理的状态转移。例如，拒绝产生 I-ARG1 、 B-ARG0 这样的输出序列。

## 4、 实验

4.1 实验参数

- LSTM 权重初始化方式采用正交初始化
- 优化选取 AdamOptimizer
- 实验环境：**tensorflow 1.4**

表 1 参数设置

| 参数 | 设定值 |
|---|---|
| 词向量维度 | 128 |
| 上下文标记维度 | 128 |
| LSTM 隐层向量维度 | 200 |
| batch_size | 60 |
| LSTM 层数 | 4 |
| 学习率 | 0.001 |
| dropout_rate | 0.1 |
| context_length | 2 |

4.2 实验结果

表 2 模型在验证集上的实验结果

| | Development | | |
|---|---|---|---|
| Method | P | R | F1 |
| **Bilstm+softmax** | **0.7357** | **0.7360** | **0.7358** |
| **Bilstm+CRF** | **0.7727** | **0.7431** | **0.7576** |

此外，在验证集上测试了该模型在论元预测上的准确率，得到了非常惊艳的结果，1115 句话中仅仅有一句话的论元没有被正确预测，模型没有将这句话中的任何一个词预测为论元，可以说，准确率达到了 100%，召回率达到了 99.91%，amazing!

**由于 bilstm+CRF 的实验是最后做的，来不及更新实验报告，因此下面的实验分析是基于 bilstm+softmax 的结果做的！**

1) 错误分析

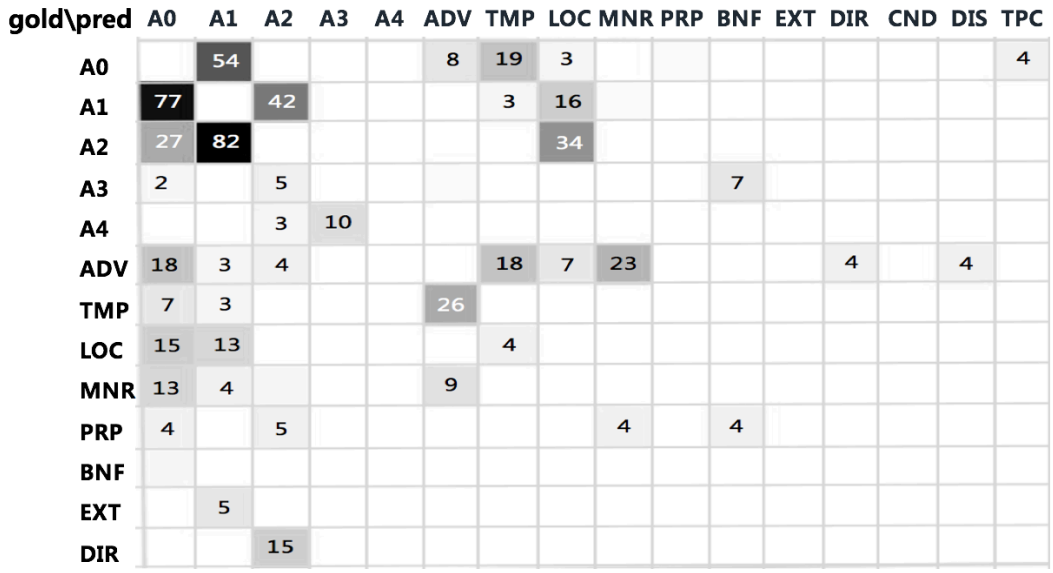


图 6 标注错误 Confusion matrix

由图 6 可以看出，在状语成分中，LOC、TMP、MNR 较容易发生标注错误。在文献[1]中，作者将其解释为动词词条定义的模糊性导致状语成分和 ARG0-ARG2 的混淆。例如：ARG2 在 move.01 词条被定义为 Arg2-GOL: destination，语义界定的模糊性使得模型在决策时也摇摆不定。此外，在 ARG0-ARG4 中，主谓的混淆颠倒也较为严重。

2) 句子长度对预测结果的影响

表 3 不同句子长度下验证集实验结果

| | | Development | | |
|---|---|---|---|---|
| 句子长度 | 句子数 | P | R | F1 |
| [6, 25] | 310 | 0.7433 | 0.7884 | 0.7653 |
| [26, 35] | 284 | 0.7538 | 0.7356 | 0.7446 |
| [36, 55] | 291 | 0.7373 | 0.7226 | 0.7299 |
| [56, 145] | 230 | 0.6972 | 0.6830 | 0.6901 |

由表可以看到，随着句子长度的增加，准确率、召回率、F1 均呈下降趋势，这也充分说明模型在长距离语义依赖中的表现略差，长距离语义建模成为限制模型性能的一大因素。

4.3 模型分析 ablation study

**在得到最好的结果之前，我们做了大量的实验，进行了各种尝试，下面是我们的一些结论。**

**注：下面的 ours 指的是 bilstm+softmax 模型。**

1) 词向量 embedding-layer 分析

采用 2017-08-20 的中文维基百科语料预训练词向量，得到约 38w 个词的词向量，可以覆盖 79.2%的训练集词汇,覆盖 87%的验证集词汇。不能找到的词被映射为<unk>, 采用[-0.05, 0.05]的均匀分布随机化。

使用大规模语料预训练词向量后，和随机初始化相比，F1 从 68.4%提升至 73.58%。另外，如果在 word_embedding 层加上词性特征，F1 值会由 73.58%降至 69.61%，由 76 可以看到，加入 POS 特征，模型在 30 轮次左右即过拟合，然而不加 POS 特征的模型会持续上升至 80 轮次左右。推断原因可能是(1) 词性特征对于语义角色标注任务的重要性并不显著；(2) 词性标注的错误也会影响模型性能，增加模型错误标注的概率。

表 4 不同 embedding-layer 在验证集上的实验结果

| | Development | | |
|---|---|---|---|
| Method | P | R | F1 |
| **Ours(pre-trained)** | **0.7357** | **0.7360** | **0.7358** |
| Random initialized | 0.6917 | 0.6767 | 0.6841 |
| pre-trained + pos feature | 0.6965 | 0.6957 | 0.6961 |

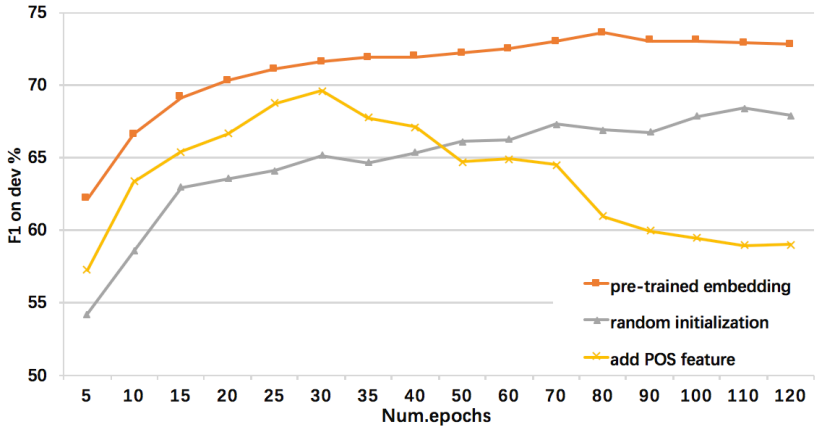


图 7 探究不同 embedding-layer 对验证集上 F1 值的影响

2) 层间连接分析

选取了 Random initialized 词向量(简写为 ran)的模型进行实验，对比有无 highway connection

对模型性能的影响。

训练深层神经网络时，为了防止反向传播时梯度消失，层与层之间的 highway 连接方式是值得借鉴的，F1 从 66.94%提升至 68.41%。由图 8 可知，highway connection 的性能在每一轮次均领先于 no highway_connection。

**注：Ran 指的是 word_embedding 采用随机化初始方式的。**

表 5 有无 highway connection 在验证集上的实验结果

| Method | Development | | |
|---|---|---|---|
| | P | R | F1 |
| **Ran+highway_connection** | **0.6917** | **0.6767** | **0.6841** |
| Ran+ no highway_connection | 0.6965 | 0.6957 | 0.6694 |

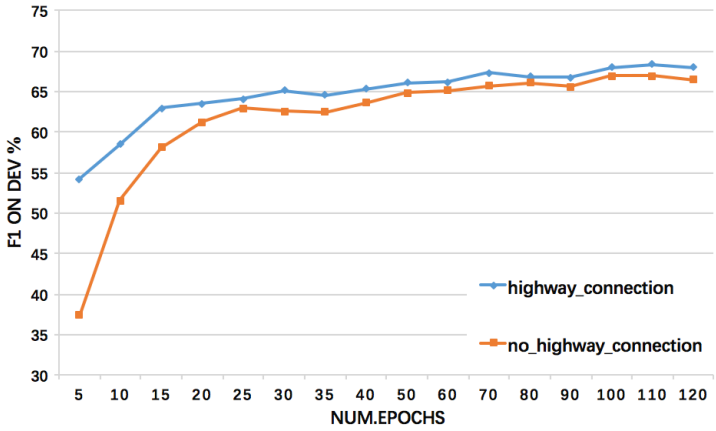


图 8 探究不同层间连接方式对验证集上 F1 值的影响

3) dropout 机制分析

选取了 Random initialized 词向量的模型进行实验，探究 dropout 机制对模型性能的影响。

没有 dropout 机制，模型在 80 轮次达到 67.84%之后就开始过拟合，加入 dropout 机制后模型在 80 轮次后在验证集上的 F1 值仍有提升，最终至 68.41%，dropout 机制可以提高模型特征学习的能力，从而增强模型的鲁棒性，使其避免过早过拟合，提高性能。

**注：Ran 指的是 word_embedding 采用随机化初始方式的。**

表 6 有无 dropout 在验证集上的实验结果

| Method | Development | | |
|---|---|---|---|
| | P | R | F1 |
| Ran + dropout | 0.6917 | 0.6767 | 0.6841 |
| Ran + No dropout | 0.6888 | 0.6684 | 0.6784 |

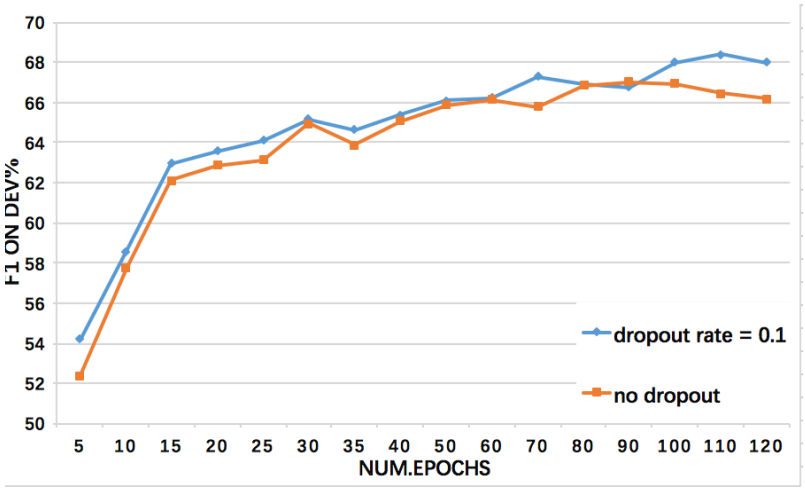


图 9 探究 dropout 机制对验证集上 F1 值的影响

4) 比较层数对模型性能的影响

表 7 不同网络深度在验证集上的实验结果

| | Development | | |
|---|---|---|---|
| Method | P | R | F1 |
| **Ours(num_layers = 4)** | **0.7357** | **0.7360** | **0.7358** |
| Num_layers = 2 | 0.7211 | 0.7012 | 0.7110 |

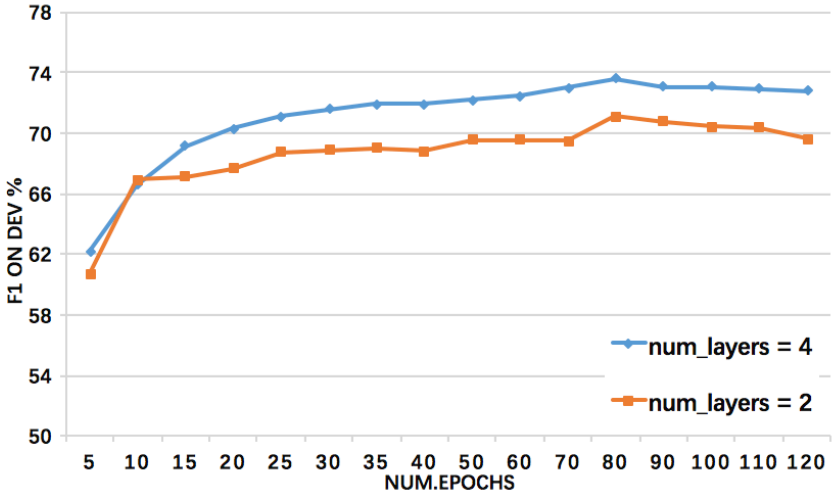


图 10 探究层数对验证集上 F1 值的影响

由图 10 和表 7 看出，2 层的性能不如 4 层。深层神经网络特征提取能力更强，在序列标注任务上优于浅层神经网络。

但是，层数加深时，也存在优化难题。

本文采用了文献中 8 层 LSTM 网络结构（正反向各视为一层）和本文的 4 层双向 LSTM 网络结构，在训练集的损失值上进行了比较。两种网络结构其他部分均相同，并且参数个数也在同一

数量级。如图 11 所示，在起始训练阶段，4 层网络 loss 值下降很快，15 轮次时 loss 将为 0.0822，但是 8 层网络在 50 轮次时才会降到 0.08 左右。4 层网络最终收敛至 0.005，8 层网络收敛至 0.0372，由此可见 8 层网络优化难度高于 4 层，可以尝试通过加大数据量和更精细的 highway connection 来解决这一优化难题。

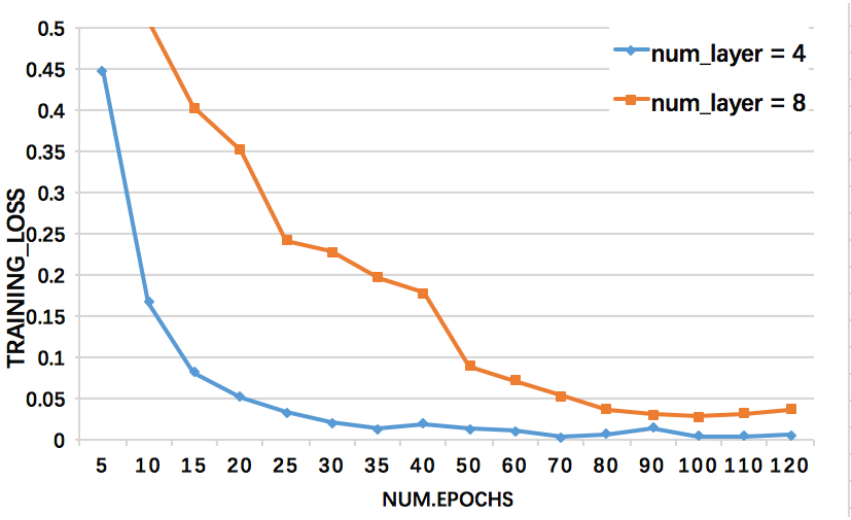


图 11 探究层数对训练集损失的影响

5) 加入层正则化对模型性能的影响

表 8 层正则化在验证集上的实验结果

| Method | Development | | |
|---|---|---|---|
| | P | R | F1 |
| **Ours** | **0.7357** | **0.7360** | **0.7358** |
| + layer_norm | 0.7387 | 0.7340 | 0.7363 |

从表 8 结果看，加入层正则化优化效果并不明显。

## 5、 展望

1) 借鉴文献[4]探讨 self-attention 机制对 SRL 任务的贡献

2) 将句法知识引入到深度神经网络中

参考文献


[1] He, Luheng, et al. "Deep semantic role labeling: What works and what's next." Proceedings of the Annual Meeting of the Association for Computational Linguistics. 2017.

[2] Zhou, Jie, and Wei Xu. "End-to-end learning of semantic role labeling using recurrent neural networks." ACL (1). 2015.

[3] Wang, Zhen, et al. "Chinese Semantic Role Labeling with Bidirectional Recurrent Neural Networks." EMNLP. 2015.

[4] Tan, Zhixing, et al. "Deep Semantic Role Labeling with Self-Attention." arXiv preprint arXiv:1712.01586 (2017).

[5] 123 Gal, Yarin, and Zoubin Ghahramani. "A theoretically grounded application of dropout in recurrent neural networks." Advances in neural information processing systems. 2016.

[6] Srivastava, Rupesh K., Klaus Greff, and Jürgen Schmidhuber. "Training very deep networks." Advances in neural information processing systems. 2015.