基于 RNN 的中文二分结构句法分析

谷波[1]　王瑞波[2]　李济洪[2]　李国臣[3]

（1.山西大学 计算机与信息技术学院, 山西 太原, 030006;

2.山西大学 软件学院, 山西 太原, 030006;

3. 太原工业学院, 山西 太原,030008)

摘 要： 为了构建一个简单易扩展的中文句法分析器，我们依据朱德熙和陆俭明先生的中文二分结构的层次分析句法理论，手工构建了一个三万句的二分结构的中文句法树库，并使用哈夫曼编码方式来简化表示完全二叉树的层次结构。本文将中文句法分析转换为迭代二分的序列标注问题，并根据该任务的特点，提出了在词的间隔上进行标记的序列标注模型（RNN-Interval，简称 RNN-INT），与常用的循环神经网络（RNN,LSTM）模型和条件随机场（CRF）进行了对比实验，使用 mx2 交叉验证序贯 t-检验来比较模型。实验结果表明， RNN-INT 模型在窗口为 1 的词特征就可达到了最好的性能，并好于其他窗口大小和其他序列标注模型（RNN，LSTM，CRF)。最后，在测试集上，在人工分词下，RNN-INT 在短语级别的 F1 值(块 F1)达到 71.25%，在句子级别的准确率达到了 43%。

关键词： 层次句法分析；循环神经网络(RNN)；mx2 CV 序贯 t-检验

中图分类号： TP391　　　　**文献标识码：** A

Chinese Syntactic Parsing with Binary Tree Structure Based on RNN Model

Gu Bo[1]　Wang RuiBo[2]　Li JiHong[2]　Li GuoChen[3]

（1 School of Computer and Information Technology, Shanxi University, TaiYuan,ShanXi,030006,China;

2 School of Software, Shanxi University, TaiYuan, ShanXi,030006,China;

3Taiyuan Institute of Technology, TaiYuan, ShanXi,030008,China）

Abstract: We built a 30,000 sentences binary Chinese Treebank which is base on Chinese syntactic theory proposed by Zhu DeXi and Lu JianMin, and we think that binary structure is more suitable for Chinese syntactic analysis and can make parsing simpler and extensible. A binary tree of each sentence in our corpus is a full binary tree and represented by Huffman coding for simplicity. The parsing can be accomplished by recursively splitting a input sequence into two subsequence, so sequential labeler can be utilized. According to the characteristics of our parsing, we proposed a sequential labeling model(RNN-Interval, abbr RNN-INT) based on RNN(recurrent neural network) tagging the intervals between words. In experiments on our corpus, we compared our model RNN-INT with primary RNN, LSTM and CRF models, and employed the mx2 cross-validated sequential t-test for comparison of the different models. The experiment results show that RNN-INT achieves the best performance when setting the RNN size of windows as 1 under two performance measures with constituency F1 and sentence accuracy. Some outperformance are very significant under mx2 cross-validated sequential t-test. Finally, the RNN-INT model achieved 71.25% for constituent F value and 43% sentence accuracy on our given testing dataset.

Keyword: hierarchical syntactic parsing; RNN (recurrent neural network) ; mx2 cross-validated sequential t-test

基金项目：国家社科基金资助项目[16BTJ034]

0 引言

句法分析是自然语言处理中核心技术之一，许多自然语言处理任务中都依赖于它。通常来说，句法分析的任务是将词的线性序列表示的一个句子，分析成一棵树结构的形式。

句法分析的理论大致上可以分为两类，一类是基于乔姆斯基提出的短语结构文法[1][2][3]的上下文无关文法（CFG），以及之后扩展的概率上下文无关文法（PCFG）[4][5]，一般将这种分析方法称之为成分（constituency）句法分析，分析的结果是一棵短语结构树，叶子节点是词，树的所有的中间节点都是短语（成分）；另一类是基于 L. Tesniere 提出的依存语法[2]理论的，这种分析一般称之为依存（dependency）分析，依存分析的结果是一棵依存树，依存树描述的是句子中词语之间非对称的二元关系。本文所指的层次句法分析是前一种，即成分句法分析。

早期的成分句法分析大多基于给定的文法（也称为产生式或规则）进行的。这种方法需要事先人工构建句法规则，容易实施。但缺点是不能解决句法多义性问题，且人工构建的句法规则通常不能覆盖现全部的句子或短语的形式。为此，Briscoe[6]提出了一个基于概率的自底向上的 LR 分析器，而 Collins[7]则是在自顶向下的句法分析过程中使用概率来进行剪枝。近年来，主流的一些句法分析方法则转向于基于无规则的自底向上的移进归约的分析[8][9][10]，这类方法使用分类器根据当前分析栈和输入队列的特征，来对移进和归约的动作进行概率判断，选择当前特征下最有可能的动作执行，从而自底向上地构建出句法树。Liu[11]则在移进归约分析中通过 BiLSTM 抽取 Lookahead 特征来提高分析器的性能，该系统在宾州英文树库上的 F1 值达到了 91.7%，在宾州中文树库 CTB5.1 上的 F1 值达到了 85.5 %。Cross[12]也利用了 LSTM 基于句子成分的跨度（span），提出了一个移进归约的系统，在英文宾州树库上的实验结果F1值为91.3%，法语上F1值为83.31%。Socher[13]则直接使用一个结构递归的神经网络（Recursive Neural Network, 简称 RecNN）学习句法树的递归结构，进行了英文的句法分析。另外一方面，也有研究者融合多个句法分析器进行句法分析，通过重排候选树集合提高了句法分析的性能[14]，当使用人工正确的分词和词性标注时，该分析器在中文宾州树库 CTB2 上的 F1 值为 89.8%，在中文宾州树库 CTB5 上的 F1 值为 86.8%。朱慕华[15]则是使用向上学习策略，提高了系统的速度，最终的系统在中文宾州树库 CTB5.1 的 F1 值达到了 82.4%。目前这些方法中都用到词性标注，而且词性标注正确率对于系统的最终性能影响非常大。

然而宾州树库的词性标注体系与国内的主流的中文词性标注体系还是有较大差别的。在国内主流的词性标注体系中，中文中的词无论在句子中主语还是谓语位置，其词性基本上是确定的，也就是说，动词可以做主语、宾语是常态。在中文中连续多个词标注为动词也是常见之事，因此，按照英文的以词性为基础的句法分析方法似乎不太合适。

对中文句法分析的研究，文献中大多是借鉴英语的句法分析方法，少有从中文的句法分析理论出发来研发相应的技术。许多语言学家认为，中文是意合语言，形式标记不明显，相比较英文而言，中文句法分析较难。特别是完全中文句法分析（既有树结构又有树节点成分的短语类型）技术更难。朱德熙[16]和陆俭明[17]先生的现代汉语语法理论认为中文的短语和句子是属于同一个层面的范畴，中文的句子结构与短语结构是同构的。中文句子并不像英文那样词构成短语，短语构成句子，句子主要是 SVO 结构。中文的词构成短语，词也可直接构成句子，短语与句子之间是实现关系，不是构成关系。句子与短语的结构都可以二分结构来分析，中文的层次分析法就是二分结构的句法分析。依照这个理论体系，自动句法分析只需要每次对各成分一分为二。本文探索基于循环神经网络（RNN）机器学习模型，来实现二分结构的句法分析。相较于传统的成分句法分析，本文方法主要有一下几个特点：

第一，对句子只做二分的结构分析，不做短语类型识别。依据层次分析理论，中文句子的结构是层次二分结构，结构类型与短语结构类型相同。在中文中，相比较而言句子的层次结构是理解句意的关键，因此我们只做层次结构分析。这样可以使得中文的短语以及整个句子的分析统一在一个二分结构的框架内，使得句法分析的问题从形式上得以简化，也使得算法实现起来比较简洁、高效。

第二，基于哈夫曼编码来表示句法树。由于采用了二分的结构，我们语料库中标注的句子都是一个满二叉树。把中间节点按照哈夫曼编码的方式进行了标记，使得语料库中的句子标注形式非常简单（没有像传统的树库那样使用多重括号对的形式），方便阅读、存储以及后期的分析处理。

第三，使用 RNN 模型，只用词特征。该方法没有使用基于转移的移进归约的方法，也不是基于 chart 的自顶向下的方法，而是通过递归地使用序列标注器进行句法分析。相对于基于 CFG 的句法分析方法，这种方法的优点是不需要词性的信息，也不需要中心词信息或者分析栈和输入队列等的特征。本文的方法直接以词序列为输入，通过同时学习词表示向量（word-embedding）和模型参数，训练得到一个序列标注器，递归的使用序列标注器来进行

句法分析。

第四，使用 hinge-loss 的词间隔标注策略。我们把句法分析看成是一个将序列逐层二分的问题，其任务就是每次寻找一个可以将序列一分为二的最优的间隔标记位置，即寻找使得分割序列的间隔最大得分位置，并以 hinge-loss 作为损失函数。这样在使用 RNN 时可以有效地利用了间隔左右两边所有词的信息来共同确定词间隔的标注。

第五，用 m×2 交叉验证序贯检验比较模型的性能。传统模型比较方法是在验证集上直接对比大小，从统计意义上来说是不可靠的。本文采用了 m×2 交叉验证（以下简写为 m×2 CV）序贯 t-检验。m×2 CV 是将语料进行 m 次两等份切分，从而得到指标估计的方差估计，逐步增加 m 构建相应的序贯 t-检验，使得对算法模型的性能的对比更加准确、可靠。

除了本文提出的在 hinge-loss 损失下，对词间隔进行标注的 RNN 模型（简称 RNN-INT）外，还对比了 RNN[18][19]（Recurrent Neural Network, 简称 RNN）和 LSTM[20][21]，以及条件随机场(CRF)[22]，各自在不同窗口下的性能。结果表明本文提出的 RNN-INT 模型在窗口为 1 时取得最好的结果，并且在 m×2 交叉验证的序贯 t-检验中是显著优于其它模型的。在测试集上，以 PARSE-VAL[23]的评测体系，短语的块 F1 值达到了 71.25%，整句的正确率达到了 43%。

论文内容安排如下，第一节介绍中文二分结构树库的标注方案，第二节介绍把二分结构句法分析转换为序列分割问题以及采用的标记方案，第三节介绍 RNN-INT 模型，第四节是实验设置方案以及 m×2 CV 序贯 t-检验方法，第五节是结果与分析，第六节是总结以及下一步计划进行的工作。

1 中文二分结构句法树库与编码方案

1.1 中文二分结构句法树库

根据中文层次分析法的理论，中文的每个句法成分都可以看成是由左右两个子成分构成，给定的中文句子均可分析成一棵满二叉树（full binary tree）。从朱德熙先生的观点来看，除了少数外来词，中文的绝大多数多字词内部也是二分结构的，词的内部结构关系、短语的内部结构以及句子的内部结构关系是一致的。 因此，如果将字作为最终叶节点的话，这种二分结构甚至可以将中文的分词一并纳入到一个句子的二分句法结构中。

中文的二分句法结构中的句法成分有 6 种内部结构[16][17]，包括偏正结构，述宾结构，述补结构，主谓结构，联合结构和连谓结构。除了联合和连谓结构可能出现三个或三个以上的并列的成分，其它 4 种都是自然的二分结构。对于三个或三个以上的成分构成的联合或连谓结构，采用从左到右的顺序将其依次进行二分处理。

依据以上，我们标注了一个中文树库，每棵标注的句法树是满二叉树，即树中任何一个非叶子节点的句法成分，都有左右两个子树（子成分）组成。目前树库中有 30034 个简单句（无标点）都是从北大新闻语料（有人工分词和词性标注，但我们没有使用词性）中选取的，没有标注成分的内部结构。

1.2 中文二分结构句法树的哈夫曼编码表示

一个句子的二分结构句法形式是一个满二叉树，可以采用哈夫曼编码进行标记，这种表示方式可以以一种非递归的线性的形式描述出递归的二分的层次结构的句法树的全部信息。这样要比括号对的表示形式更加简洁直观，方便标注人员阅读和标记，也便于存储。因此我们在语料库中对词的间隔使用了哈夫曼编码进行标记（huff-labels）。给定一个有 n 个词的句子，对其 n-1 个间隔（或称为句法成分的切分位置）进行标记，将一个二叉的句法树的结构，表示成 n-1 个数字。具体的标记方法如下。

(1) 句法树的根节点的编码标记为 1。

(2) 对于句法树中的非根中间节点，假定其父节点的编码为 h，按照如下方式确定其对应的哈夫曼的编码：

A. 如果该节点是其父节点的左子节点，则其编码为父节点的哈夫曼编码乘以 2，即 h*2。

B. 如果该节点是其父节点的右子节点，则其编码为父节点的哈夫曼编码乘以 2，然后再加 1，即 h*2+1。

例如，“我们 班 有 许多 外国 学生”，对应的句法树形式如下图 1 所示。

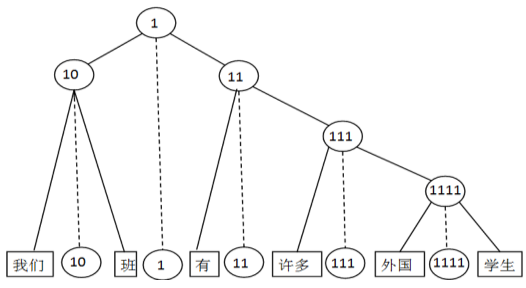

图 1 二分结构句法树的哈夫曼编码标记

图 1 中，矩形表示句法树的叶子节点（词），圆形表示句法树中非叶子节点（短语），圆形中间的数字表示该间隔位置对应的哈夫曼编码（图中显示的是其二进制形式）。通过实线连接的叶子和非叶子节点构成了一棵二分结构的成分句法树（满二叉树）。图中用虚线链接的两个圆形节点，是为了清楚的显

示非叶子节点和它对应的词间隔的位置而增加的。我们以文本格式保存二分结构句法树，并且使用分号‘;’将句子和它的对应的哈夫曼编码十进制序列分开，并列放在一行中，编码之间也用空格分隔。这样就把一个句子和它的二分结构的句法树的编码描述放在一行中，表示成一个线性形式。最终的表示示例如下：

我们 班 有 许多 外国 学生；2 1 3 7 15。

2 将二分结构句法分析转换为递归的序列分割问题

本文采用自顶向下的分析方法，将句法分析看作把一个序列逐层向下分割为两个子序列的递归过程。每次都把一个长序列分割成两个短的子序列，然后再分别对两个短的序列进行递归分割，直到每个词。这样句法分析就转换为一个逐层的序列标注问题，从而可以把机器学习中序列标注技术应用到二分结构的句法分析上。

如果对句子的所有可能句法树都计算其得分，然后选择得分最大句法树为最终分析结果，这样计算量是非常巨大的。Socher[13]使用 RecNN 对英文进行句法分析时使用了贪心算法，他在实验中发现，在每步保存 k 个最好的子树，最终的句法分析结果并不比仅保存一个最好的子树的结果好。本文采用了贪心算法，每一次二分的时候，只保留当前序列划分为左右两个子序列的最大得分的结果，即只考虑当前最优，而不考虑最终得到的树的全局最优。

2.1 二分结构的间隔标记策略

序列标记策略有多种[24][25]，不同的序列标记方案在不同的机器学习模型下的性能是有明显差异的。我们针对二分结构句法分析任务提出了一种对词间隔进行标记的方案，下面以例句“我们 班 有 许多 外国 学生”进行说明。对于切分后的左右两个成分，都按照从左到右深度优先的顺序进行描述。

1)BI 标记：每次切分都是划分成两整块，没有组块分析任务中的块外词，所以把传统的 BIO 标记中的 O 去掉，B 表示当前词是一个成分的开始，I 表示当前词是一个成分的内部；若成分由一个词构成，把这个词标为 B。这种标记关注的是第二个成分的 B 标记（右边成分的第一个词）的识别。见表 1。

2)LR 标记：每次切分出的左右两个子成分，可以通过左右两种标记分别标出对应的词，把左边成分的词都标为 L，把右边的词都标为 R。从 L 到 R 的转换的位置，决定了切分的位置。见表 2。

3)BEO 标记：把紧邻分割位置的左右两个词作为重点考虑，分割位置左边的词标为 B，分割位置右边的词标为 E，其他的词标为 O。这种标记方式重点关注紧邻切分位置的左右两个词。见表 3。

4)INT 标记（间隔标记）：这种标记方式不是标记词，而是标记词与词之间的间隔。n 个词的句子序列，有 n-1 个间隔，把分割位置的间隔标记为 1，其它间隔标记为 0。这种方式直接对间隔进行标记，从而避免了前面其他几种标记方案中，把对词标记转成对间隔的划分时所产生的不一致现象。见表 4。

表 1 BI 标记示例

	我们	班	有	许多	外国	学生
1	B	I	B	I	I	I
2	B	B				
3			B	B	I	I
4				B	B	I
5					B	B

表 2 LR 标记示例

	我们	班	有	许多	外国	学生
1	L	L	R	R	R	R
2	L	R				
3			L	R	R	R
4				L	R	R
5					L	R

表 3 BEO 标记示例

	我们	班	有	许多	外国	学生
1	O	B	E	O	O	O
2	B	E				
3			B	E	O	O
4				B	E	O
5					B	E

表 4 间隔标记(INT)示例

	我们	班	有	许多	外国	学生
1	0	1	0	0	0	
2	1					
3			1	0	0	
4				1	0	
5					1	

2.2 序列的递归分割处理算法

假设给定了一个训练好的序列标注器 M，当给定一个由词序列“$w_1\ w_2\ \cdots w_i \cdots\ w_n$”组成句子 S 作为输入，序列的二分结构的递归分割处理算法如下：

1. 把句子 S: $w_1\ w_2 \ldots w_i \ldots w_n$，看成词序列作为输入，每个词的索引固定。
2. 调用序列分割程序，S 为其参数
   1)如果 S 长度为 1，返回。
   2)如果 S 长度为 2，将两个词分成两个左右子序列，并把第一个词的索引，记为此次分割的位置，返回。
   3)如果 S 长度大于 2，调用 M 对序列 $w_1\ w_2 \ldots w_i \ldots w_n$ 进行分割。假设得到的两个子序列分别为 $w_1 \ldots w_i$ 和 $w_{i+1} \ldots w_n$，将左边序列最后一个词的索引 i 记为此次分割的位置。然后分别把 $w_1 \ldots w_i$ 和 $w_{i+1} \ldots w_n$ 分别作为新的参数 S，递归调用序列分割程序（转 2）。
3. 根据序列分割程序每次保存的分割的索引信息，生成一棵满二叉树作为分析结果。

3 间隔标注的 RNN 模型(RNN-INT)

上节将二分结构句法分析转换为递归的序列分割（标注）问题，对序列标注任务，目前性能比较好是条件随机场和神经网络(RNN,LSTM)。在最初的实验中（数据分为训练集，验证集，测试集，没有使用交叉验证)，对 RNN 和 CRF 中都使用了除间隔标记（INT）之外的多种标记策略进行了实验，发现 RNN 使用 BI 标记要好很多，而 CRF 中使用 BEO 和 LR 的结果也优于其他标记。因此后续的交叉验证实验中，对 CRF 选择了 LR 和 BEO 两种标记，对 RNN 和 LSTM 则只选择了 BI。针对词间隔标记策略，我们基于 RNN 结构提出了 RNN-INT 模型，采用 hinge-loss 为损失函数，在每个词间隔上计算得分并进行标记。

3.1 RNN 模型

本文的 RNN-INT 模型，基于 Elman[19]提出的 RNN 结构，所以我们先简单介绍 RNN 的结构。三层的 RNN 网络，分为输入层，隐层和输出层。

输入层：可以采用 Collobert[26]的方法，将词库中的每个词都表示成一个向量(word-embedding)，并对当前词采用开窗口的方式。假设每个词对应的 word-embedding 维数大小为 d，窗口（对称）大小为 s，拼接后可以得到一个大小为 d*s 的输入向量x。

隐层：对于句子中的词按照从左向右的顺序依次传递给网络，假设当前时刻 t 的输入向量为x^{t}，输入层连接到隐层的权重矩阵为$\mathrm{W_{hx}}$，隐层连接到隐层的权重矩阵是$\mathrm{W_{hh}}$，隐层的偏置向量为$\mathrm{b_h}$，隐层的激活函数为σ，t 时刻隐层的计算公式如下(1)式：

$$h^t = \sigma(\mathrm{W_{hx}}\boldsymbol{x}^{\boldsymbol{t}} + W_{\mathrm{hh}}h^{t-1} + \boldsymbol{b}_{\mathrm{h}}) \qquad (1)$$

输出层：是一个 softmax 层，有几个类别标记，最终的输出层就有几个输出单元，假定当前时刻 t，隐层连接到输出层的权重矩阵是$\mathrm{W_{ho}}$，输出层的偏置向量是$\mathrm{b_o}$ 输出层的公式如下(2)式所示。

$$\mathrm{y^t} = \mathrm{softmax}(W_{\mathrm{ho}}\mathrm{h^t} + \boldsymbol{b}_{\boldsymbol{o}}) \qquad (2)$$

3.2 使用 hinge-loss 来度量词间隔标注的损失

Socher[13]在 用 RecNN 来做句法分析，也用了 hinge-loss 损失函数，通过计算相邻成分的得分，递归的对相邻成分进行合并。本文与其不同之处是，首先我们是自顶向下的分析，从整句出发，递归对句子进行二分，直到不能分割为止；Socher 则使用自底向上的方案，递归的对两个成分进行融合直至形成一个节点。其次是 Socher 对两个相邻节点计算融合得分的时候，只使用这两个节点的信息，而我们则是采用了 RNN 的方式，考虑了间隔左右两边的整个成分的信息。第三，Socher 的方法为了递归应用 RecNN，要保证每个节点的向量等长，需要对两个子节点连接后得到的两倍长的向量进行线性变换，从而使父节点的向量和子节点等长的向量，而我们在序列二分后，对两个子序列独立在分，所以不涉及节点的融合。

从中文的特点知道，句法中短语的识别不仅由构成短语的词决定，还需要由短语前后的所有词共同决定的，而传统的组块分析方法主要依赖相邻的词。考虑到这种二分结构句法的形式，采用直接对词之间的间隔进行标记的方法，并且把能分割的位置标记为 1，不能分割的标记为 0。采用贪心策略进行逐层二分结构的句法分析，每一次分析都是把一个成分切分成左右两个子成分，即只能有一个词间的间隔位置被标记为 1，其它的间隔位置都是 0。

3.3 RNN-INT 模型的结构

RNN-INT 模型可以看作在输入层和隐层有两个独立的子网络，分别对应正向词序列（来自于间隔左边的词）和反向词序列（来自于间隔右边的词）。正向隐层和反向隐层的两个输出向量，在同一个输出层进行了合并。我们需要使得真实切分的间隔的得分，大于其它间隔的得分，所以输出层没有进行 softmax，而是对正反两个隐层的输出，和对应的权重向量进行内积运算，然后再把这两个值求和作为该间隔上的得分。网络结构如下图 2 所示。

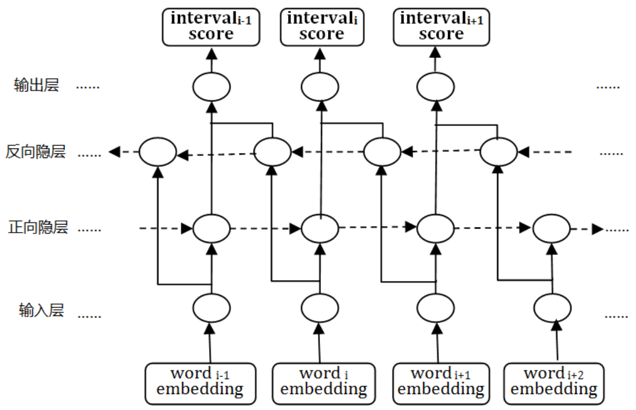

图 2 RNN-INT 网络结构

输入的中文句子 sen=” w_1 w_2 ... w_n”， w_i 对应句子中的从左向右看的第 i 个词。

输入层：前面所述相似，对每个间隔，有两个输入向量（分别由间隔左右两边的词得到）。所以对

于每个间隔，可以看作有两个输入层，一个正向输入层从左向右依次接收词序列（$w_1\ w_2\cdots w_i$），另一个反向输入层从右向左依次接收词序列（$w_n\ w_{n-1}\cdots w_{i+1}$）。

隐层：和输入层对应有两个隐层。接收间隔左边词为输入，和正向输入层对应的隐层为正向隐层记作$\mathrm{h}_\mathrm{f}^\mathrm{t}$；接收间隔右边的词为输入，和反向输入层对应的隐层为反向隐层记作$\mathrm{h}_\mathrm{b}^\mathrm{t}$。分别在原来的权重矩阵和偏置向量的下标最后加上 f 和 b，来表示正向和正向隐层对应的参数。

$$h_f^t = \sigma\left(\mathrm{W}_{\mathrm{hxf}}\boldsymbol{x}^{\boldsymbol{t}} + W_{\mathrm{hhf}}h_f^{t-1} + \boldsymbol{b}_{\mathrm{hf}}\right) \quad (3)$$

$$h_b^t = \sigma(\mathrm{W}_{\mathrm{hxb}}\boldsymbol{x}^{\boldsymbol{t}} + W_{\mathrm{hhb}}h_b^{t+1} + \boldsymbol{b}_{\mathrm{hb}}) \quad (4)$$

输出层：两个隐层的输出在同一个输出层进行合并。因为输出层最终的输出是一个得分 score 所以输出层只有一个输出单元，也不需要 softmax 运算。$\mathrm{W_{of}}$和$\mathrm{W_{ob}}$分别表示正向隐层，反向隐层与输出层之间的权重矩阵；$\mathrm{b_o}$表示输出层的偏置向量。在 t 时刻输出层输出的$\mathrm{score^t}$计算公式如下。

$$\mathrm{score^t} = h_f^t\mathrm{W_{of}} + h_b^t\mathrm{W_{ob}} + \boldsymbol{b}_{\mathrm{o}} \quad (5)$$

损失函数：对序列进行一次切分时，只有一个词间隔可以是实际的切分点，因此需要使得这个间隔点的得分，大于其他词的间隔点的得分。假设有 n 个间隔位置，每个间隔位置的真实切分标记为$\mathrm{y_i}$（需要把前面的 0 转成-1 表示不在此处切分，1 表示在此处切分），则对序列的一次分割的 hinge-loss 损失函数为（6）式。

$$\sum_{\mathrm{i=1}}^{\mathrm{n}} \max(0, 1 - \mathrm{score(s_i)y_i}) \quad (6)$$

训练采用随机梯度下降算法，损失进行反向传播，更新词表示向量和神经网络的权重。测试时，对一个词序列，把得分最大的间隔作为切分的位置。

4 实验设置

4.1 实验语料库的切分

目前我们的树库共有 30034 个简体中文单句，句子只使用分词信息，没有标点也没有使用词性。先从整个语料库中随机抽出 6000 句子作为测试集，剩下的 24034 句作为建模的数据(以下记为 D)，然后将 D 使用 m×2 交叉验证（简记为 m×2CV），Wang[27][28]从理论和模拟实验上证明，m×2CV 方法是较好的模型比较方法，并且更适用于文本数据。

除了将句子包括在训练集中外，同时将句子的二叉树的每个成分都抽取出来，也独立看作一个句子，并加入训练集中作为训练样本（只含两个词的成分除外，在之前实验中发现，如果在训练集中加入这些两个词的成分，会对序列标注器产生干扰，训练序列标注器去掉这些会轻微提高句法分析的性能，所以在最终的训练集合中把两个词的成分都删除了）。例如，上面图 1 中的句子，“我们 班 有 许多 外国 学生”，经过抽取成分，可抽取出三个样例：我们 班 | 有 许多 外国 学生，有 | 许多 外国 学生，许多 | 外国 学生。

4.2 实验中的模型与标记方案

在 CRF[1]模型的实验中，对于 LR，BEO 标记方案，分别记为 CRF-LR 和 CRF-BEO；RNN 和 LSTM 模型基于 Theano[29]进行了实现，这两个模型下都用 BI 标记，所以模型后面不再加 BI 后缀。并且在这两个模型下对是否采用双向(bi)，以及是否在最后一层使用了 viterbi 算法(vtb)的方式（即 CRF 方式）都进行了实验。本文提出的在词的间隔上进行标记的方案，称为 RNN-INT。在每种模型下，都使用了对称窗口，窗口大小从 0（只取当前词自己）到 7。因为语料中的句子都是简单句，而且平均长度在 10 个词左右，所以没有再取更大的窗口。在 RNN 和 LSTM 的实验中，均采用 sigmoid 激活函数。

4.3 评价指标

完全句法分析中，多数文献中采用了块准确率，或者块召回率，或者块 F1 值作为评价指标[23]。 本文以块 F1 值作为评价指标，由于语料基本都是单句，我们把整句正确率作为考察指标。

4.4 两模型性能对照的 m×2 交叉验证序贯 t-检验

Wang[27][28]分析了常用的模型性能对比的 t-检验存在的问题，提出模型性能比较的正则化 mx2 交叉验证序贯 t-检验方法。理论和模拟验证该检验是相对保守的检验，可以得到更为可信的结论。Wang[27]所给的 3×2 CV t-检验自然可以扩展为相应的两模型性能比较的 m×2 CV 序贯 t-检验，描述如下：

给定两个机器学习模型A、B及模型评价指标$\mathcal{M}^{\mathrm{A}}$、$\mathcal{M}^{\mathrm{B}}$，记$\mathcal{M} = \mathcal{M}^{\mathrm{A}} - \mathcal{M}^{\mathrm{B}}$为两个模型性能之差，不妨假设$\mathcal{M}^{\mathrm{B}}$为 Baseline 模型的性能。

原假设：H0 ：$\mathcal{M} \le 0$　备择假设：H1 ：$\mathcal{M} > 0$

记$\mathcal{M}$在数据集$\mathcal{D}$上的第 i 次切分上的 2 折交叉

[1] https://github.com/taku910/crfpp

验证估计为$\widehat{\mathcal{M}}_i = \frac{\left(\widehat{\mathcal{M}}_1^{(i)}+\widehat{\mathcal{M}}_2^{(i)}\right)}{2}, i = 1,2,\ldots m$；$\mathcal{M}$的 mx2 交叉验证估计为m次 2 折交叉验证估计的平均，记为$\widehat{\mathcal{M}} = \sum_{i=1}^{m} \widehat{\mathcal{M}}_i /m$，其检验统计量如下式 7 所示：

$$T_{m\times 2} = \frac{\widehat{\mathcal{M}}}{\widehat{\sigma}} \backsim C_m \cdot t(2m-1) \qquad (7)$$

这里，$\widehat{\sigma} = \sqrt{\frac{1}{2m}\sum_{i=1}^{m}\sum_{k=1}^{2}(\widehat{\mathcal{M}}_k^{(i)} - \mathcal{M})^2}$，$C_m = \sqrt{\frac{2m+1}{2m-1}}$。可证明$T_{m\times 2}$近似服从自由度为 2m-1 的 t-分布. Wang[28]推荐最小的 m 取 3，之后逐步增加实验次数 m. 若到一个较大的实验次数（比如 m=6），仍然没有停止，可强行停止。

5 结果与分析

5.1 基于 m×2 交叉验证的实验结果

本文 m 先取 3。表 5 是模型的块 F1 和整句正确率在 3X2 交叉验证下的平均值。无论块 F1 值还是整句正确率，RNN-INT 模型取窗口 1 的结果最好，达到 68.93 和 39.88。图 3 和图 4 显示了模型的块 F1 值和整句正确率随着窗口大小的变化趋势，不同模型各自在不同的窗口下取得其最好性能。但趋势都是随着窗口增大，结果变好，随后趋于平稳或者略有下降。窗口增大模型的性能逐渐趋于接近。我们认为可能当窗口增大时对于之前性能低的模型会增加信息，性能也提升；对于性能已经很高的模型，反而增加了噪声使其性能下降。和之前预想不同，RNN 和 LSTM 在该任务下性能相差不大，这可能和句子长度相对较短有关，使得 LSTM 没有发挥其长时记忆的优势，且由于其参数明显比 RNN 多，所以性能不高，甚至当窗口为 0 的时候，由于参数个数较多，但输入数据较少，性能反而是所有模型下最低的。双向模型和使用 viterbi 对性能都有明显的提升，但窗口小于 3 时，双向模型性能提高的多，当窗口大于等于 3 时，viterbi 对性能提升的多，同时使用双向且结合 viterbi 都取得各自最好的结果。

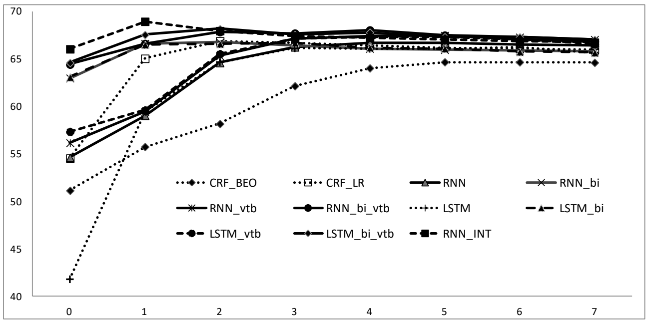

图 3 模型在不同窗口下的模型块 F1 值

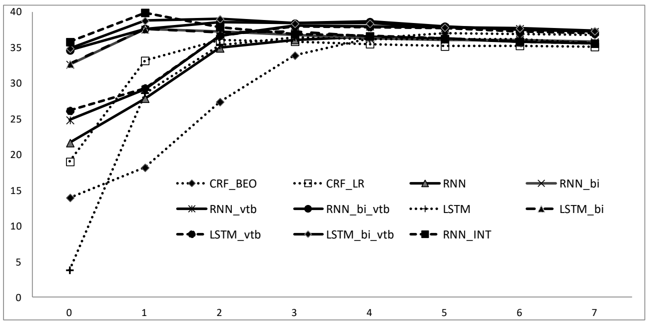

图 4 模型在不同窗口下的模型整句正确率

表 5 模型块 F1 值和整句正确率表

	块 F1								整句正确率							
窗口大小	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
CRF_LR	51.12	55.73	58.15	62.13	64.01	***64.65***	64.65	64.60	13.93	18.15	27.37	33.88	36.25	***37.04***	36.83	36.79
CRF_BEO	54.51	65.03	***66.87***	66.70	66.38	66.15	66.07	65.98	18.98	33.15	***36.03***	35.84	35.49	35.16	35.26	35.10
RNN	54.59	59.02	64.57	66.23	***66.70***	66.66	66.45	66.39	21.62	27.85	34.90	36.08	***36.45***	36.19	35.74	35.56
RNN_bi	62.96	66.60	***66.73***	66.38	66.06	65.95	65.86	65.80	32.61	***37.53***	37.17	36.79	36.22	36.06	36.08	35.77
RNN_vtb	56.13	59.40	65.29	67.10	***67.34***	67.45	67.27	67.02	24.81	29.12	36.57	***38.04***	37.97	37.78	37.63	37.22
RNN_bi_vtb	64.41	66.59	67.84	67.69	***68.02***	67.47	66.98	66.83	34.61	37.63	38.55	38.38	***38.59***	37.95	37.42	37.17
LSTM	41.75	59.44	64.64	66.10	66.04	66.08	***66.21***	65.64	3.72	28.57	35.30	***36.44***	36.20	36.18	36.20	35.59
LSTM_bi	63.07	66.50	66.58	***66.68***	66.17	66.08	65.74	65.65	32.71	***37.53***	37.12	37.28	36.45	36.21	35.79	35.71
LSTM_vtb	57.32	59.58	65.51	67.07	67.30	***67.30***	66.99	66.56	26.12	29.29	36.66	***37.99***	37.87	37.72	37.17	36.96
LSTM_bi_vtb	64.58	67.53	***68.14***	67.53	67.77	67.30	67.27	67.00	34.85	38.74	***38.98***	38.35	38.36	37.80	37.69	37.31
RNN_INT	66.03	***68.93***	67.94	67.40	67.22	67.02	66.82	66.72	35.78	***39.88***	37.87	36.88	36.59	36.21	35.82	35.56

我们在 m×2CV 下对 RNN-INT 和其他模型进行差异的显著性 t-检验。在各模型各个窗口下，块 F1 和整句正确率的相关性非常大，整体相关系数为 0.98，说明这两个指标有高度的正相关。这也符合我们的直观认识，句法块识别对的越多，对应的整句识别正确的也越多。我们取模型的块 F1 值最高的窗口，为该模型的窗口大小配置，都与 RNN-INT 在 1 窗口下进行 t 检验。表 6 是模型比较的 mx2CV t-检验的结果。从表 6 中的结果看，RNN-INT 与其他模型的性能之差都大于零，最少也高出近 1 个百分点，且大部分检验在 m=3 时 t-检验就显著，少部分在 m=6 下才显著。RNN-INT 与 RNN_bi_vtb 和 LSTM_bi_vtb 的整句正确率，即使在 m=6 时也不显著，且差异值及方差的估计在 m=4，5，6 时已经基本稳定，说明再增加 m 已无明显效果，故停止序贯实验。

表 6　RNN-INT 与其他模型对照的 t-检验 p-值表

	块 F1 值		整句正确率	
MODEL	diff±sv	p	diff±sv	p
CRF_BEO	4.27±0.19	3.48E-06**	2.84±0.41	1.02E-03**
CRF_LR	2.06±0.28	8.40E-04**	3.85±0.29	4.94E-05**
RNN	2.22±0.33	1.11E-03**	3.43±0.61	2.46E-03**
RNN_bi	2.20±0.31	9.22E-04**	2.71±0.45	1.96E-03**
RNN_vtb	1.59±0.33	5.01E-03**	1.91±0.28	1.11E-03**
RNN_bi_vtb	0.91±0.40	5.79E-02	1.30±0.57	5.58E-02
LSTM	2.72±0.40	1.16E-03**	3.68±0.37	1.94E-04**
LSTM_bi	2.25±0.67	1.83E-02*	2.60±0.51	3.88E-03**
LSTM_vtb	1.62±0.35	5.78E-03**	2.16±0.28	6.67E-04**
LSTM_bi_vtb	0.79±0.20	1.04E-02*	0.89±0.43	6.91E-02

注：**表示显著水平为 0.01 下显著，　*表示显著水平为 0.05 下显著.

根据以上分析，以数据集 D 进行模型选择的最终结果为，词特征在的 1 窗口下的 RNN-INT 模型，以此模型将全部数据 D（即交叉验证的训练、验证集合并）为最终的训练集，重新训练 RNN-INT 模型。在最终的 RNN-INT 模型上，用 6 组不同的随机数初始化权重和词表示向量，得到在测试集合结果，如表 7 所示。最终在测试集（6000 句）上得到平均块 F1 值为 71.252%，平均整句正确率的结果为 42.998%。

表 7　RNN-INT 模型不同初始值在测试集的结果

	1	2	3	4	5	6	平均
块 F1	70.94	70.91	71.43	71.43	***71.65***	71.15	71.252
整句正确率	42.8	42.7	***43.6***	42.78	43.03	43.08	42.998

5.2　域外词(out of domain)上的实验对比

上面的表 5 显示基于神经网络的方法优于 CRF 方法，一个可能的原因是这类方法在域外词的处理上有优势。不同于 CRF 模型直接使用词本身作为离散特征，在神经网络模型中将词转换成了词的向量表示形式，是一种连续的特征，当遇到域外词时，通过其周围的域内词的向量也可以反映出一部分域外词的信息，从而降低了域外词的影响。

在 3×2 交叉验证实验中，我们以验证集中的句子中是否出现了训练集中没有见过的词为准，把验证集分成含有域外词和不含域外词两个子集，简记为 OOD(out of domain)，IND(in domain)，分别在两个子集上计算指标值。经统计六组交叉验证的验证集中 OOD 的句子占比均大致在一半左右，占比平均为 51.5%。我们对比了 RNN-INT,与 CRF-LR 模型。下面是在 OOD 和 IND 上分开统计的结果。

表 8 OOD 上的结果

	块 F1	整句正确率
CRF-LR	63.872±0.412	31.112±0.514
RNN-INT	**66.214±0.234**	**35.293±0.639**

表 9 IND 上的结果

	块 F1	整句正确率
CRF-LR	70.087±0.303	41.262±0.404
RNN-INT	**71.830±0.453**	**44.740±0.600**

从表 8 可以看出，在域外词集 OOD 上，RNN-INT 比 CRF-LR 的块 F1 高出 2.3%，整句正确率高出 4.4%；而在域内词集 IND 表 9 上却只高出 1.7%，　整句正确率指标上高出 3.4%。 因此，在处理域外词上，RNN-INT 与 CRF-LR 相比还是有一定的优势的。

5.3 结论

对于我们的实验结果进行分析总结，可以得出以下结论：

（1）RNN-INT 性能最优

从所有的实验中，本文所提出的 RNN-INT 模型性能最好，对比其它模型有显著的性能上的提高。可能主要有以下一些原因。

词的间隔是中文成分之间分割的重要信息。我们认为不论是组块分析还是完全句法分析，成分的

形成本质上是由两个成分两边的所有词共同决定的，不仅仅是两成分间隔相邻的少数词。特别是在二分结构的句法分析中，采用 RNN 对词之间的间隔进行标记，是利用了间隔左右的所有词，抓住了成分整体的信息，从而提高标记的正确率。

在 RNN-INT 中使用 hinge-loss 损失函数，并使用正向、反向 RNN 计算词的间隔的得分，这在一定程度上考虑了一个序列的全局信息，从而使得训练得到的网络更可靠。另外，实验中发现 RNN-INT 模型相对于普通的 RNN 模型训练和预测的速度都快一些，我们认为这是由于该模型下不需要像传统的分类任务那样需要在最后一层进行 softmax，RNN-INT 只需要计算一个得分即可，并且 hinge-loss 损失函数也相对于交叉熵损失计算简单，这使得模型在训练和预测时速度得以变快。

在域外词的处理上 RNN 模型相对 CRF 模型有优势。这可能归功于 RNN 中使用了词的向量表示，这种表示方式本身会将词和它周围的词之间的信息进行融合，并包含在循环的隐层向量中，这使得即使在预测时没有见过的词，也可以通过相邻见过的词的表示向量，间接地获得了该词的一些信息，从而使得 RNN 比 CRF 对域外词处理上有优势。

在 RNN-INT 中采用词特征的 1 窗口有利于 RNN。不同于 CRF 模型中使用的是离散特征，随着窗口增大，特征越来越稀疏，提供给模型的信息会越来越少，模型性能不再提升。而从 RNN-INT 的结果可以看出，在窗口为 1 时要好于其他窗口的情形，这说明在中文中使用当前词语与左右两个词的拼接向量，做双向的循环迭代，构建两个成分的隐含表示（隐层向量）是最好的，并不是拼接的向量越长越好。

（2）mx2 交叉验证进行模型选择和评估，结论更为可靠

采用 mx2 交叉验证，通过对语料数据集进行多次划分，进行多次实验，可以得到重复实验结果，并可以获得估计的方差，以及相应的统计显著性检验，从而对模型的性能评价更加准确。这样可以克服传统的简单地将语料切分为训练、验证、测试集，只能有一个实验的结果，难以从统计的显著性来评价好坏。

6 总结与展望

完全句法分析是中文信息处理里的一个难题。为了构建一个简单的中文句法分析，我们以朱德熙和陆俭明先生的层次分析理论为基础，将中文句子的结构表示为逐层二分结构的形式，简化中文句子的分析过程，使得句法分析任务变得相对简单些。

我们将二分结构的句法分析问题转换为逐层的序列标注问题，在已经构建的大约 3 万句规模的中文语料库上，使用了 RNN、LSTM 和 CRF 模型，在多种标注策略和不同的词特征窗口设置下，进行了模型对比的实验，最终在我们的测试集上的结果表明，本文提出的 RNN-INT 模型，效果是最好的，最终的块 F1 值达到 71.25%，整句的正确率达到了 43%左右。

目前我们的语料库规模还比较小，且都是单句，句子也没有标注成分的内部结构和短语类别。下一步，一方面我们计划继续扩充语料库，研究复杂句的二分结构标注方案，并且不断完善相应的标注规范，特别是针对一些中文特殊结构的句子提出其合理的标注方案（比如把字句，被子句等等）；另一方面我们计划将每个成分的内部的短语结构也进行标注。在算法方面，我们计划探索结合自底向上的方法，即同时利用词以及短语的合并的信息，与词的间隔上的分割的信息进行二分句法分析，以期提高分析器的性能。

参考文献

[1] Chomsky N. Syntactic structures.[J]. International Journal of American Linguistics, 1958, 149(3):174–196.

[2] 宗成庆.统计自然语言处理 [M]. 北京: 清华大学出版社, 2008.

[3] 蒋宗礼, 姜守旭. 形式语言与自动机理论[M]. 清华大学出版社, 2003.

[4] Manning C D, Schutze H. Foundations of statistical natural language processing[M].Vol. 999. Cambridge: MIT Press, 1999.

[5] 吴伟成, 周俊生, 曲维光. 基于统计学习模型的句法分析方法综述[J]. 中文信息学报, 2013, 27(03):9-19.

[6] Briscoe T, Carroll J. Generalized Probabilistic LR Parsing of Natural Language (Corpora) with Unification-Based Grammars[J]. Computational Linguistics, 1993, 19(1):25-59.

[7] Collins M. Three generative, lexicalised models for statistical parsing[C]. Eighth Conference on European Chapter of the Association for

Computational Linguistics. Association for Computational Linguistics, 1997:16-23.
[8] Sagae K, Lavie A. A classifier-based parser with linear run-time complexity[C]. International Workshop on Parsing Technology. Association for Computational Linguistics, 2005:125-132.
[9] Wang M, Sagae K, Mitamura T. A fast, accurate deterministic parser for Chinese[C]. International Conference on Computational Linguistics and the, Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, 2006:425-432.
[10] Zhang Y, Clark S. Transition-based parsing of the Chinese treebank using a global discriminative model[C]// International Conference on Parsing Technologies. Association for Computational Linguistics, 2009:162-171.
[11] Liu J, Zhang Y. Shift-Reduce Constituent Parsing with Neural Lookahead Features[J]. Transactions of the Association of Computational Linguistics (TACL), 2017, 5: 45-58.
[12] Cross J, Huang L. Span-Based Constituency Parsing with a Structure-Label System and Provably Optimal Dynamic Oracles[C]. Austin, Texas, USA: EMNLP6, 2016, 1-11.
[13] Socher R, Lin C Y, Ng A Y, et al. Parsing Natural Scenes and Natural Language with Recursive Neural Networks[C]. International Conference on International Conference on Machine Learning. Omnipress, 2012:129-136.
[14] Zhang H, Zhang M, Tan C L, et al. K-Best Combination of Syntactic Parsers[C]. Singapore :Conference on Empirical Methods in Natural Language Processing(EMNLP),2009,1552-1560.
[15] 朱慕华, 王会珍, 朱靖波. 向上学习方法改进移进-归约中文句法分析[J]. 中文信息学报, 2015, 29(2):33-39.
[16] 朱德熙编. 语法讲义[M]. 商务印书馆, 1982.
[17] 陆俭明. 现代汉语语法研究教程[M]. 北京大学出版社, 2005.
[18] Graves A. Supervised Sequence Labelling with Recurrent Neural Networks[M]. Springer Berlin Heidelberg, 2012.
[19] Elman J L. Finding structure in time.[J]. Cognitive Science, 1990, 14(2):179-211.
[20] Huang Z, Xu W, Yu K. Bidirectional LSTM-CRF Models for Sequence Tagging[J]. Computer Science, 2015.
[21] Hochreiter S, Schmidhuber J. Long Short-Term Memory[J]. Neural Computation, 1997, 9(8):1735-1780.
[22] Lafferty, John D., A. Mccallum, and F. C. N. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data[C]. Eighteenth International Conference on Machine Learning Morgan Kaufmann Publishers Inc. 2001:282-289.
[23] Abney S, Flickenger S, Gdaniec C, et al. Procedure for quantitatively comparing the syntactic coverage of English grammars[C]. Speech and Natural Language, Proceedings of A Workshop Held at Pacific Grove, California, Usa, February. DBLP, 1991:306-311.
[24] Ramshaw L A, Marcus M P. Text Chunking using Transformation-Based Learning[J]. Text Speech & Language Technology, 1995, 11:82--94.
[25] Sang E F T K. Memory-Based Shallow Parsing[J]. Journal of Machine Learning Research, 2002, 2(4):559-594.
[26]Collobert R, Weston J, Karlen M, et al. Natural Language Processing (Almost) from Scratch[J]. Journal of Machine Learning Research, 2011, 12(1):2493-2537.
[27] Wang Yu, Wang Ruibo, Jia Huichen, et al. Blocked 3×2 cross-validated t-test for comparing supervised classification learning algorithms[J]. Neural Computation, 2014, 26(1):208-235.
[28] Wang Ruibo, Wang Yu, Li Jihong, Yang Xinli, Yang Jing, Block-regularized mx2 Cross-validated Estimator of Generalization error. Neural Computation, 2017, Vol. 29, No. 2: 519-554
[29] Team T D, Alrfou R, Alain G, et al. Theano: A Python framework for fast computation of mathematical expressions[J]. 2016, arXiv:1605.02688.