第 5 章 基于句子语义与 Self-Attention 机制的中文和英文 NER 模型

5.1 概　　述

目前大多数命名实体识别模型通过上下文编码器(如 LSTM 和 CNN 等)获得的词的上下文状态表示,来预测最终的实体标签。尽管这些词可以学习到当前的上下文信息,但是这些词在句子中的语义信息仍然很贫乏。因此,在本章探索了一个携带句子语义信息的句子表示模型。通过将上下文词表示与句子表示进行拼接,使得每个词能获得丰富的语义信息,更好地进行实体标签预测。除此之外,还在模型中应用了 Self-Attention 机制,Self-Attention 可以直接捕捉句子中任意两个词的长距离依赖,更好的捕捉整个句子的全局依赖。本章所提出的模型不仅适用于英文数据集,同时也适用于中文数据集。

本章中模型的介绍总体分为 6 个部分:模型的总体结构、词嵌入层、第一层 BiLSTM、Self-Attention 层、词隐状态与句子级向量结合部分、第二层 BiLSTM。下面将分别详细介绍。

5.2 模型的总体结构

本章所提出的模型主要结合句子特征表示进行中文和英文的命名实体识别,同时还利用了 Self-Attention 机制来更好地捕获任意两个词之间长距离依赖关系。图 5.1 描述了该模型的整体结构,词嵌入通过第一层 BiLSTM 获得词的隐状态向量表示和句子的向量表示;然后词隐状态会送入到 Self-Attention 层增强捕获词之间关系的能力;最后将词隐状态拼接上句子级向量后送入到第二层的 BiLSTM 得到最终的隐状态向量以进行标签预测。考虑相邻标签之间的交互信息是很有限的,例如,标签 I-PER 不可能跟在 B-LOC 的后面。因此,最后使用 CRF 层联合地解码标签序列,CRF 使得模型从所有可能的标签序列中找到最优路径。

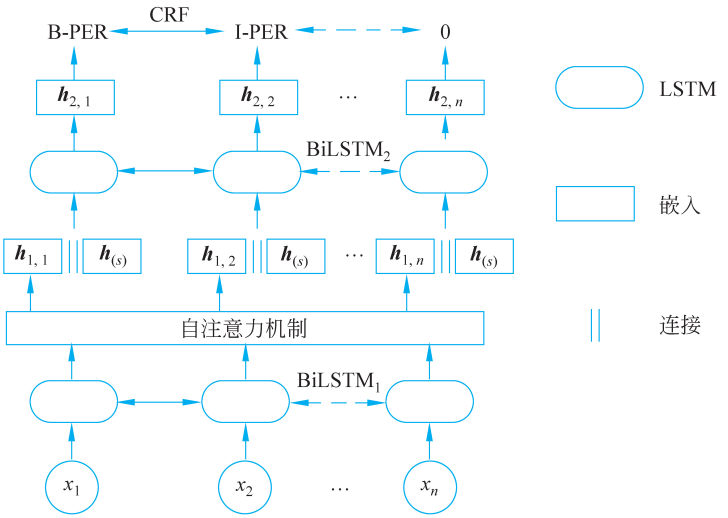

图 5.1　基于句子语义与 Self-Attention 机制的 NER 模型

5.3　词嵌入层

模型的第一步是将每一个输入的词或字符映射到分布式表示空间，它是低维稠密的向量表示空间，能够捕捉单词的语义和句法特性，由于本章节使用了中文和英文两个数据集，所以中文和英文的词嵌入层会因为语言特性而不同。

5.3.1　英文词嵌入层

在英文命名实体识别任务上，词嵌入通常会由词级向量与字符级向量组成。根据以往的工作[26]，使用预训练的词向量要比使用随机初始化的词向量能够获得很大性能的提升。因此，本章使用斯坦福大学[11]预训练好的公共可用的 300 维词向量作为词级向量。此外，不同的英文单词有表面或形态的相似性。例如，具有规则前后缀的单词可以共享一些字符级的特征。因此，使用字符级特征去处理未登录词。首先，采用 BiLSTM 建立字符级向量表示模型，公式为

$$\overrightarrow{\boldsymbol{h}_l} = \mathrm{LSTM}(\boldsymbol{c}_i, \overrightarrow{\boldsymbol{h}_{l-1}}) \tag{5-1}$$

$$\overleftarrow{\boldsymbol{h}_l} = \mathrm{LSTM}(\boldsymbol{c}_i, \overleftarrow{\boldsymbol{h}_{l+1}}) \tag{5-2}$$

$$\boldsymbol{h}_c = [\overrightarrow{\boldsymbol{h}_1}; \overleftarrow{\boldsymbol{h}_f}] \tag{5-3}$$

其中：$\boldsymbol{c}_i$ 表示一个字符序列$(c_1, c_2, \cdots, c_f)$中的某一个；f 代表字符序列的长度；$\overrightarrow{\boldsymbol{h}_l}$和$\overleftarrow{\boldsymbol{h}_l}$分

别是前向 LSTM 与反向 LSTM 的隐状态向量；h_c 是最终要获得的字符级向量。

除此之外，本章还将研究如何使用 CNN 去构建字符级向量表示模型。CNN 在计算机视觉中已经有了广泛的应用。假如要识别一张分辨率为 64×64 的小图像是猫还是狗，第一步是要把它输入到模型中，实际上它的输入数量是 64×64×3，因为每张图像都有 3 个颜色通道，最后输入到模型中的特征维度是 12 288。这还只是一张 64×64 的小图像，一般的图像都要比它大。例如，一张 1000×1000 的图像，它的特征维度达到了 300 万，如果使用全连接层进行建模，在模型的第一个隐藏层中假如有 1000 个隐藏单元，参数矩阵大小将会是 1000×300 万，会有 30 亿个参数进行训练更新，这还只是一个隐藏层，在实际情况中，往往会有多个隐藏层，这是一个无比庞大的数据量。在如此大量参数的情况下，难以捕获足够多的数据来防止神经网络过拟合问题。此外，庞大的参数里所需求的巨大内存也让人无法接受。CNN 避免了对参数的过度依赖，减少大量参数，相比于全连接神经网络，能更好地识别超大图像。

文本卷积与图像上的卷积有所不同，首先是滤波器的大小，图像上滤波器的大小一般是 $n\times n$，是一个正方形，而在文本上一般是 $i\times j$，其中 j 是词向量的维度。在文本上滤波器是沿着句子中词的方向滑动，获得的是一个 N-gram 局部特征。图 5.2 描述了基于 CNN 生成字符级向量表示的过程。每个单词被分解为单个字符，模型的输入是一个字符序列。首先使用一个一维卷积层为每个字符捕捉与其相邻字符的信息，本质是获得字符 N-gram 特征。不同的滤波器大小可以获得不同种类的上下文信息 $\boldsymbol{c}_i$。然后，在卷积的结果上应用一个 max-over-time 池化层获得当前滤波器特征 $\boldsymbol{g}_i$。这种池化操作就是简单地从之前一维的特征中提取一个最大值，代表着最重要的信号或信息。此外，还可以解决可变长度的句子输入问题，因为不管有多少个特征值，只提取其中的最大值。最后，通过不同尺寸滤波器所获得的不同特征进行拼接作为最后词的字符级表示 $\boldsymbol{h}_c$，公式如下。

$$\boldsymbol{m}_i=\text{conv}([\boldsymbol{c}_{i-\frac{k}{2}},\cdots,\boldsymbol{c}_i,\cdots,\boldsymbol{c}_{i+\frac{k}{2}}]) \tag{5-4}$$

$$\boldsymbol{g}_i=\text{pooling}([\boldsymbol{m}_1,\boldsymbol{m}_2,\cdots,\boldsymbol{m}_f]) \tag{5-5}$$

$$\boldsymbol{h}_c=[\boldsymbol{g}_1;\boldsymbol{g}_2;\cdots;\boldsymbol{g}_o] \tag{5-6}$$

其中：k 是滤波器的尺寸大小；o 是滤波器种类的数量。

5.3.2 中文词嵌入层

与英文不同，中文由于缺乏天然的分隔符所以输入和分词有很大的关系。Peng 等[144]为中文命名实体识别提出了 3 种嵌入方法：词嵌入、字嵌入和字位置嵌入。相关最新的研究成果[145]分析了在中文 NLP 中是否需要分词，文中指出了分词的缺点。首先，词数据比较稀疏容易导致过拟合，而且大量的未登录词限制了模型的学习能力。其次，分词方法众多且不统一和分词效果不佳，错误的分词可能会对下游任务的使用产生错误的

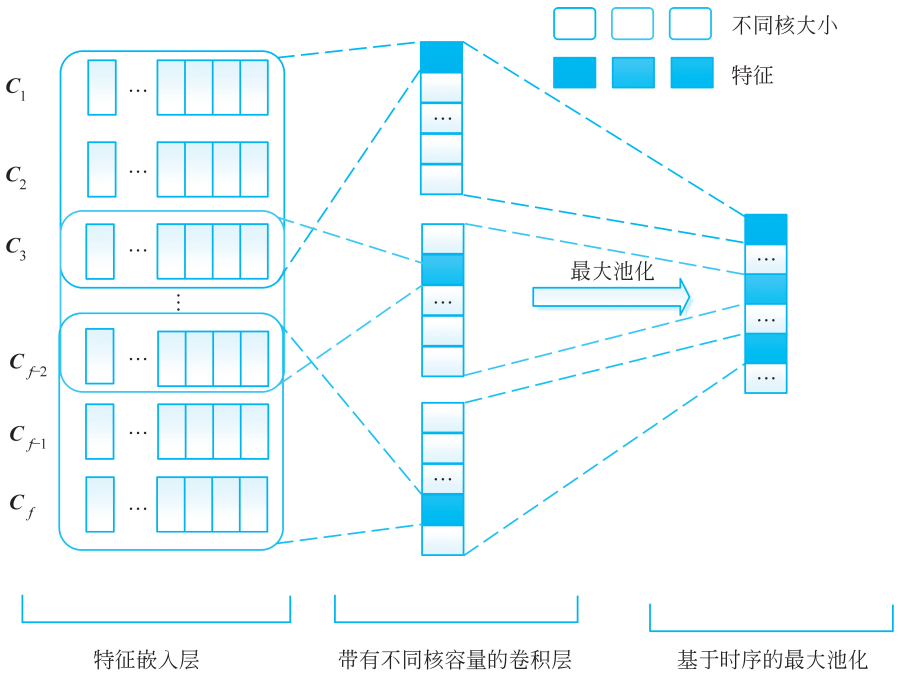

图 5.2　基于 CNN 字符级表示模型

影响，从而使得最终模型的性能不佳。最后，分词所带来的收益并不明确，尽管站在人的角度直观观察，词所带的语义信息要比字更加丰富，但是对于神经网络而言，词不一定有利于提高表现。上述相关论文中在 4 个 NLP 任务上分别对词级别和字级别的模型进行了实验得出以下结论。首先，在神经网络模型下进行中文 NLP 任务，字级别的表现几乎总是优于词级别的表现。其次，对于大多数中文 NLP 任务不需要额外的分词。最后，对于部分任务而言，单使用字级别表示就可以达到最佳的表现，加入词会适得其反，有负作用。因此，使用预训练中文 100 维字向量作为模型的输入。

5.4　Self-Attention 机制

相比较于 RNN，BiLSTM 可以处理一些较长的句子，然而当句子长度越长，距离越远的词之间的依赖信息就越少，因此，词之间距离上的依赖信息并没有被很好地捕捉。Self-Attention 不管它们之间的距离如何，可以直接捕捉句子中两个词之间的关系，同时也可以很好的捕捉句子中句法和语义特征，这对 BiLSTM 来说是一个很好的补充。

5.4.1 Attention 机制

Attention 机制是对于某个时刻的输出 y，它在输入 x 上的各个部分上的注意力，这里的注意力也就是权重，是输入 x 的各个部分对某时刻输出 y 贡献的权重。Attention 的起源借鉴了人的注意力模式，通常人类在观察一张图像时会通过快速扫描全局图像获得需要重点关注的目标区域，这叫作注意力焦点。人会对注意力焦点投入更多的精力以获得关注目标的更多、更细致的信息，抑制其他无用的信息。这种注意力模式利用有限的资源从大量的信息中筛选出高价值有用的信息，极大地提升了信息处理的效率与准确性。深度学习的注意力机制本质上与人类的注意力机制相似，核心目标也是从大量信息中选取对当前任务最有价值的信息。目前大多数 Attention 模型附着在 Encoder-Decoder 框架下，这种框架可以看作是一种深度学习模式，应用场景很广泛。

图 5.3 是一种文本领域常用的 Encoder-Decoder 框架，可以把它看作输入一个句子生成另外一个句子，即句子对(Source，Target)，在机器翻译任务上，Source 和 Target 可以是不同的语言。假设输入句子 Source 为(x_1，x_2，…，x_m)，Target 为(y_1，y_2，…，y_m)。Encoder 就是对输入句子 Source 编码，进行非线性转换为中间语义向量表示 $\boldsymbol{C}$ 的公式为

$$\boldsymbol{C}=f(x_1,x_2,\cdots,x_m) \tag{5-7}$$

其中：f 是非线性转换函数。

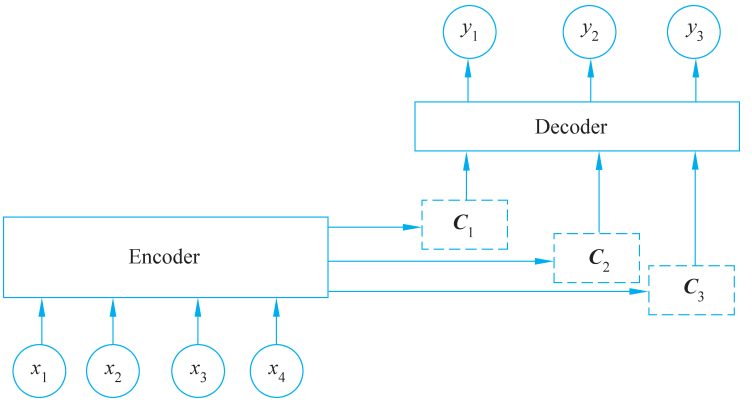

图 5.3 文本领域的 Encoder-Decoder 框架

Decoder 根据 Source 的中间语义向量表示 $\boldsymbol{C}$ 和已经生成的 y_1，y_2，…，y_{i-1} 来生成 y_i，其公式为

$$y_i=\partial(\boldsymbol{C},y_1,y_2,\cdots,y_{i-1}) \tag{5-8}$$

其中：∂ 与 f 类似，是非线性转换函数。

如果没有引入 Attention 机制时，无论生成目标句子 Target 中的哪个词，它们使用的

输入句子 Source 的语义编码 $\boldsymbol{C}$ 都是一样的。但是，Source 中的任意单词 x_i 对生成目标单词 y_i 来说影响力是不同的。这类似于人类观察图像时，人眼中没有注意力焦点一样。以机器翻译为例，输入句子 Source 为“Tom chases Jerry”，目标句子 Target 为“汤姆追逐杰瑞”。在没有引入 Attention 时，翻译“杰瑞”这个单词时 Source 中每个英文单词对“杰瑞”的影响是相同的，这显然是不合理的，Jerry 一词对于翻译“杰瑞”应该更重要。在引入了 Attention 机制之后，会体现出英文单词对于翻译当前中文单词不同的影响程度。当翻译“杰瑞”时，注意力机制会给不同的英文单词分配不同的注意力权重，对正确翻译当前词有很大的积极作用。这意味着生成每个目标词 y_i 时，由之前固定的中间语义表示 $\boldsymbol{C}$ 换成了根据当前输入词不断变化的语义表示 $\boldsymbol{C}_i$。生成目标单词过程如下。

$$y_1 = \partial(\boldsymbol{C}_1) \tag{5-9}$$

$$y_2 = \partial(\boldsymbol{C}_2, y_1) \tag{5-10}$$

$$y_3 = \partial(\boldsymbol{C}_3, y_1, y_2) \tag{5-11}$$

每个 $\boldsymbol{C}_i$ 对应着不用 Source 中单词的注意力权重，公式为

$$\boldsymbol{C}_i = \sum_{j=1}^{L_x} a_{ij} \boldsymbol{h}_j \tag{5-12}$$

其中：L_x 代表输入句子 Source 的长度；a_{ij} 代表 Target 输出第 i 个单词时 Source 中第 j 个单词的注意力权重；$\boldsymbol{h}_j$ 是 Source 中第 j 个单词的语义编码。

把 Attention 机制进一步抽象，其本质思想如图 5.4 所示，可以看出将 Source 中的元素变成(Key，Value)数据对，一般情况下 Key 等于 Value 存储在 Source 中的元素。Target 中的元素为 Query，给定一个 Query，通过计算 Query 与各个 Key 之间的相似性得到 Key 对应 Value 的权重系数，最后对 Value 进行加权求和，得到最终的 Attention 数值，其公式为

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{Softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{V} \tag{5-13}$$

其中：$\boldsymbol{Q}$、$\boldsymbol{K}$ 和 $\boldsymbol{V}$ 分别是 Query、Key 和 Value 的缩写；$\sqrt{d_k}$ 是一个调节因子，保证内积不会太大。Attention 机制的计算过程可以分为三步。首先，根据 Query 与各个 Key 的内积计算它们之间的相似性。其次，引入 Softmax 函数对内积结果进行归一化，将原始数值变成所有数值之和为 1 的概率分布，此外，根据 Softmax 特有的机制能更加突显出重要元素的权重。最后，根据得到的权重系数对 Value 进行加权求和。

现在大多数 Attention 机制的计算方法都是根据以上 3 个步骤求出 Query 的 Attention 数值。

5.4.2　Multi-Head Attention

如图 5.5 所示，Multi-Head Attention[21] 是谷歌公司提出的概念，是 Attention 的提

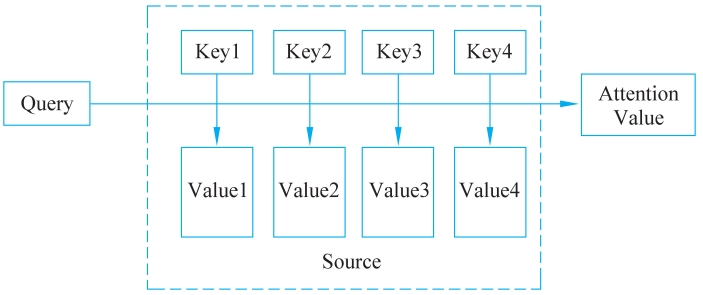

图 5.4　Attention 机制的本质思想

升版。其原理很简单，就是把 $\boldsymbol{Q}$、$\boldsymbol{K}$ 和 $\boldsymbol{V}$ 通过参数矩阵进行线性映射，然后再做 Attention，这个过程重复 h 次，最后把 h 个 Attention 结果拼接在一起再一次通过参数矩阵进行映射得到一个新的向量表示，公式如下。

$$\text{head}_i = \text{Attention}(\boldsymbol{QW}_i^Q, \boldsymbol{KW}_i^K, \boldsymbol{VW}_i^V) \tag{5-14}$$

$$\text{MultiHead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{Concat}(\text{head}_1, \text{head}_2, \cdots, \text{head}_h)\boldsymbol{W}^O \tag{5-15}$$

其中：h 是 Attention 重复地次数，也是 head 的数量；$\boldsymbol{W}_i^Q$、$\boldsymbol{W}_i^K$、$\boldsymbol{W}_i^V$ 和 $\boldsymbol{W}^O$ 是可训练的参数。Multi-Head 就是多做了几次同样的事情，参数不共享，之后把结果进行拼接。

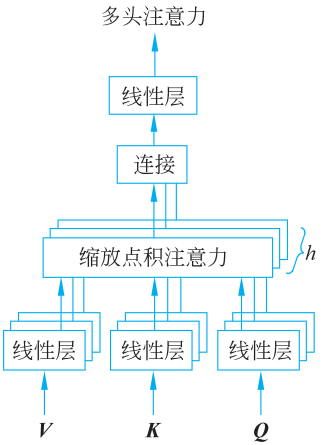

图 5.5　Multi-Head Attention 模型

5.4.3　Self-Attention

Self-Attention 即自注意力或称作内部注意力，它是一种特殊的 Attention 机制，输入序列即输出序列，在序列内部做 Attention，计算序列对其本身的 Attention 权重，寻找序列内部的联系。Query = Key = Value，即 Attention($\boldsymbol{X}$, $\boldsymbol{X}$, $\boldsymbol{X}$)，最后的输出就是 $\boldsymbol{Y}$ =

MultiHead($\boldsymbol{X}$,$\boldsymbol{X}$,$\boldsymbol{X}$)。

5.5　句子表示模型

尽管双向 LSTM 可以学到当前词的上下文信息，但是当前词的全局语义信息仍然很贫乏。本节重点探索了两个具有全局语义信息的句子表示模型，分别是基于双向 LSTM 的句子表示模型和基于多通道卷积神经网络的句子表示模型。将句子表示与经过 BiLSTM 获得的上下文词状态进行连接可以很好地利用全局上下文信息。

5.5.1　基于双向 LSTM 的句子表示模型

图 5.6 描述了基于双向 LSTM 神经网络生成句子表示的过程，输入是句子中的词序列，每个词被前向的 LSTM 与反向的 LSTM 分别捕捉过去和未来的信息，最后，每个方向的 LSTM 隐状态连接在一起生成句子表示，公式如下。

$$\overrightarrow{\boldsymbol{h}_l} = \mathrm{LSTM}(x_i, \overrightarrow{\boldsymbol{h}_{l-1}}) \tag{5-16}$$

$$\overleftarrow{\boldsymbol{h}_l} = \mathrm{LSTM}(x_i, \overleftarrow{\boldsymbol{h}_{l+1}}) \tag{5-17}$$

$$\boldsymbol{h}_S = [\overrightarrow{\boldsymbol{h}_1}; \overleftarrow{\boldsymbol{h}_f}] \tag{5-18}$$

其中：x_i 表示输入句子($x_1, x_2, \cdots, x_n$)中的一个词；n 代表词序列的长度；$\overrightarrow{\boldsymbol{h}_l}$和$\overleftarrow{\boldsymbol{h}_l}$分别是前向 LSTM 与反向 LSTM 的隐状态向量；$\boldsymbol{h}_S$ 是最终要获得的句子级向量。

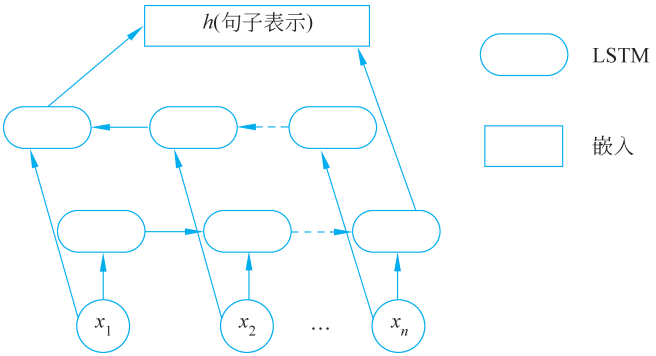

图 5.6　基于双向 LSTM 的句子表示模型

5.5.2　基于多通道 CNN 的句子表示模型

如何使用 CNN 构建句子表示模型？CNN 编码句子通常需要使用一个通道结构，类似于 5.3.1 节中的字符卷积模型，使用一个 1-D 卷积层生成一个句子表示，本节中还重点

研究了一个多通道 CNN 用于建立句子的特征表示模型。如图 5.7 所示，嵌入层有两个通道，每个通道都是一组向量，第一个通道是字符级向量，第二个通道是词级向量。每个滤波器应用到两个通道上捕捉相邻词信息，然后应用了 max-pooling 层选取一个最强信息作为当前的句子表示。公式类似于 5.3.1 节中字符卷积公式，但模型输入有所不同。

$$\boldsymbol{m}_i = \text{conv}([x_{i-\frac{k}{2}}, \cdots, x_i, \cdots, x_{i+\frac{k}{2}}]) \tag{5-19}$$

$$\boldsymbol{g}_i = \text{pooling}([\boldsymbol{m}_1, \boldsymbol{m}_2, \cdots, \boldsymbol{m}_f]) \tag{5-20}$$

$$\boldsymbol{h}_S = [\boldsymbol{g}_1; \boldsymbol{g}_2; \cdots; \boldsymbol{g}_o] \tag{5-21}$$

其中：k 是滤波器的尺寸大小；o 是滤波器种类的数量；$\boldsymbol{g}_i$ 是不同种类滤波器获得的特征；x_i 表示输入句子$(x_1, x_2, \cdots, x_n)$中的一个词；$\boldsymbol{h}_S$ 表示输入句子的句子级表示。

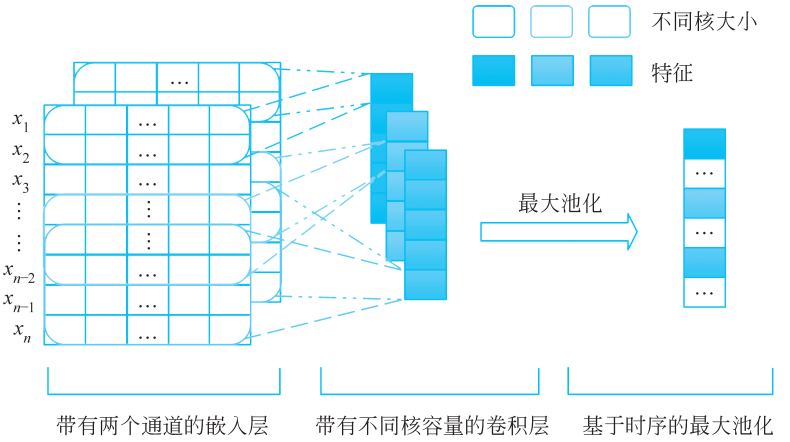

图 5.7　基于多通道 CNN 的句子表示模型

5.6　实验与分析

5.6.1　数据集

为了评估本章所提出的 NER 模型，在两个不同语言的数据集上进行了验证实验，分别是 CoNLL 2003 NER 英文数据集和 Original Weibo NER 中文数据集。中文数据集来自新浪微博数据，总共有 4 种实体类型：人名（person）、地名（location）、组织名（organization）和地缘政治实体（geo-political entity）。训练集有 1350 条句子，测试集有 270 条句子，验证集有 270 条句子。英文数据集来自 Reuters 语料库的新闻数据组成，有与中文数据集相同的 4 种实体类型。训练集有 14 987 条句子，测试集有 3466 条句子，验证集有 3684 条句子。表 5.1 列出了数据集的详细统计。

表 5.1　数据集统计

数　据　集	类型	训练集	验证集	测试集
CoNLL 2003 NER	句子	14 987	3466	3684
	词	203 621	51 362	46 435
	实体	23 499	5942	5648
Original WeiboNER	句子	1400	270	270
	字	73 800	14 500	14 800
	实体	1890	390	420

5.6.2　超参数

在英文任务上，词嵌入维度是 300，字符嵌入维度是 100，在模型训练过程进行微调。字符级 BiLSTM 的隐藏单元为 150，词级 BiLSTM 的隐藏单元为 300。采用 Adam 优化器，初始学习率为 0.001，衰减率为 0.9。为了防止过拟合，在 BiLSTM 隐藏单元处应用 dropout，值为 0.5，批次大小为 20，梯度裁剪为 3。在总共 100 个 epoch 的训练过程中，如果 20 个 epoch 内的结果没有提升会提前终止训练。

在中文任务上，通过 look-up 表从预训练的字嵌入矩阵中获得每个字嵌入，其维度为 100。BiLSTM 的隐藏单元为 120，采用 Adam 优化器，初始学习率为 0.001，dropout 值为 0.7。批次大小为 20，梯度裁剪为 5，模型总共训练 140 个 epoch，采取 F 作为评价指标对实验结果进行评测。

5.6.3　模型探索

如 5.2.1 节所示，在英文 NER 任务上探索了两个字符级表示模型，分别是 Char-BiLSTM 和 Char-CNN。本章实现了一个 Baseline 模型：BiLSTM＋CRF。将 Char-BiLSTM 和 Char-CNN 分别和 BiLSTM＋CRF 结合，称之为 BiLSTM＋CRF＋char-BiLSTM 和 BiLSTM＋CRF＋char-CNN。如表 5.2 所示，不同的字符级表示模型在 CoNLL 2003 NER 数据集上模型性能比较。BiLSTM＋CRF＋char-CNN 是使用 1 个滤波器大小为 3 的卷积层，BiLSTM＋CRF＋char-CNN(multi-size)是使用 3 个不同滤波器大小的卷积层，kernel size 分别为 3、4、5。通过实验结果观察，3 个模型的实验结果大致相同。尽管使用多个不同大小的滤波器 CNN 可以捕获到不同种类的上下文信息，但是这对构建字符级向量表示并没有太大的提升。此外，考虑到模型的复杂性与相对较小的性能差距，使用 char-BiLSTM 是一种安全的选择。因此，在英文 NER 任务上使用