# 命名实体识别调研报告

zekiye
zekiye@tencent.com

October 3, 2018

## 1 引言

命名实体识别（Named Entity Recognition，NER）为自然语言处理（NLP）的基础任务之一，其目标是提取文本中的命名实体并对这些实体进行分类，比如人名、地名、机构、时间、货币和百分比等，广泛用于信息提取、问答系统、句法分析、信息检索和情感分析等任务。

命名实体识别不仅需要找出实体的位置，还需要对实体进行分类。数据的标

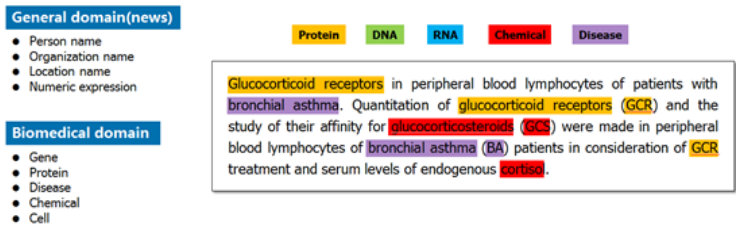



图 1: NER 的任务：找 + 分类

注体系有：IO、BIO、BMEWO 和 BMEWO+ 等。其中常用的是 BIO 和 BMEWO, 以 BIO 为例：

- **B** : Beginning of NE (B-) tag
- **I** : Inside of NE (I-) tag
- **O** : Outside of NE (O-) tag

训练数据的输入有两种形式，第一种是标注结果直接给在词的后面（词 1/tag1 词 2/tag2），第二种是将语料和标注结果分开，放在两个不同的文件中。值得注意的是，BIO 给出的是整体上的标注，是否是实体，还需要给出实体的具体类别，所以标注的 tag 实际中为“B-ORG B-LOC B-TIME I-TIME”的形式。

整体上，近年来 NER 研究的大体趋势如图 2。早期方法主要基于规则和词典构建 NER 系统，比如 Sheffield 大学的 LaSIEII[Humphreys et al., 1998] 和 ISOQuest 的 NetOwl[Krupka and Hausman, 1998]。到 2000 年初，CRF 等概率图模型得到广泛的应用。再之后，随着深度学习的兴起，Bi-LSTM+CRF 一度成

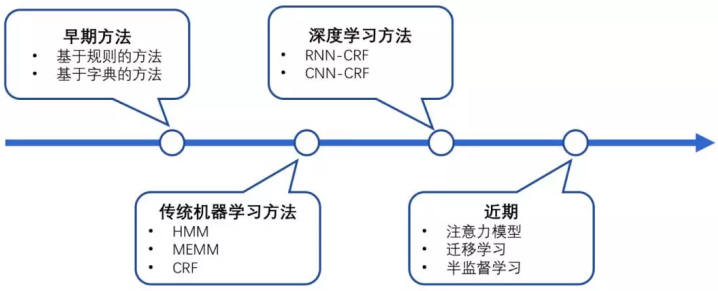



图 2: NER 发展趋势

为研究的热点，直到近年来很多方法都是在 Bi-LSTM-CRF 上的修改。LSTM 和 CRF 都需要合并序列中上下文的信息，注意力机制（Attention）自适应地计算不同上下文对象的权重，因此成为一个很有用的技巧。深度学习方法一般需要较大的数据量才能学好，但是实际生产中，往往出现数据集缺失和少量标注的情况，迁移学习和半监督在一定程度上能解决这些问题。

根据图 2，总结出本文将描述的内容：

1. 基于规则和词典
    - 基本流程
    - 词典与规则的构建
    - 规则的自动生成与维护
2. 特征工程
3. 基于机器学习方法
    - 无监督: 自动生成规则和关键词，学习模式
    - 有监督：CRF 和 Bi-LSTM-CRF 为代表，需要大量标注语料
    - 半监督：预训练 +Bi-LSTM-CRF+ 特征融合

## 2 基于规则和词典

采用机器学习方法做 NER 是近年来的热点，但是前提是需要有足够大的标注训练语料。在缺乏标注语料的时候，采用基本规则和词典不失为一种替代方法，而且基于规则的方法也能达到不错的性能。所以一般在生产环境中，能靠词典解决的问题就靠词典解决，这是最高效稳定的方法。其优点是，快速、可解释性强和不需要太多标注语料。其缺点也很明显，需要领域知识、规则和词典的构建与维护繁重、对未登陆词的匹配问题（召回会不够）。

首先给出一种实战中极简暴力的基于规则和词典的命名实体识别流程：

1. 构建词典：提取训练集中的所有命名实体，分词，取最后一个词（特证词），存入词典，去重
2. 标注序列

(a) 分词并得到 POS tag

(b) 如果这个词的词性为特定名词，则标为 S(start)

(c) 如果这个词在步骤 1 中的特证词词典中，则标为 E(end)

(d) 其他标为 O(outside)

3. 识别实体: 采用规则正则匹配

对于第三步识别实体，举一个例子，机构名的一般构成为：若干地名 + 若干其他成分 + 若干特征词。比如北京市腾讯科技有限公司，那么这里可采用的一个正则为：“S+O*E+”，即 1 个以上地名开头 (S+)，以 1 个以上特征词结尾 (E+)，中间成分数量就无所谓了, 这里有限公司为特征词。

接下来给出更加完整具体的流程 [Farmakiotou et al., 2000]：

1. 语言学上的预处理

(a) 标准化（tokenisation）

(b) 句子划分

(c) POS tagging

(d) 词干抽取（stemming）：减少词典大小和语法上的复杂性（ps. 此处可构建词典）

2. 命名实体抽取：识别命名实体的边界

(a) 初始化定界：采用 general patterns（词典匹配、拼写规则、特殊字符、特征词和标点符号等）

(b) 分隔

(c) 互斥（exclusion）：上下文、互斥表（名词缩写、常见命名实体词典）

3. 命名实体分类

(a) 应用分类规则

(b) 基于词典的分类

(c) 对于未登陆词，将未分类词与已分类词或词典部分匹配

预处理主要是分句与分词，中文分词可以采用jieba 分词、HanLP、中科院 NLPIR、哈工大 LTP 、FoolNLTK和 kcws 深度学习中文分词等。对于特定领域的特定实体，为了更好的分词，一般需提供特定领域的词典。如上文的流程所示，词典的使用有三个地方：第一，在分词时辅助分词；第二，实体抽取时匹配实体；第三，基于词典的分类（这一步其实也可以划为规则，只是规则只是一个词）。基于词典和规则的 NER 的核心在于词典和规则的构建与维护，下文对这些方面分开进行叙述。

## 2.1 词典的构建

词典构建最好的方法当然是全人工，但是全人工工作量较大，也容易出现覆盖不全的现象。所以，稍微好一点的方法时基于统计分析得到候选词典，然后再人工做筛选，同时人工提取领域中的重要术语和复用领域现有词典。现有综合中文语义词库：CSC（CWB 升级版）、hownet和Chinese Open Wordnet。

词典构建统计分析方法：

- 去停用词后词频统计，选取一定范围内的名词（不能太少也不能太多）
- 关键词抽取：TF-IDF、TextRank
- 借助维基百科页面分类系统
- 特征词分析：词共现、特定模式等（如，来自 xx）
- POS-tag 词法分析，从标记为人名 (nh)、组织 (ni)、时间 (nt)、日期 (nr)、专有名词 (nz) 或地名 (ns) 的词中抽取
- 依存句法分析：Is-a

对于最后一项，依存句法分析，与前面的特征词有一定概念上的重叠。举个例子，马化腾是腾讯的 CEO，其中存在雇佣关系 < 马化腾，腾讯公司 >，与这两个实体有依赖关系的词集是 < 是，的，CEO>，CEO 为触发词或者叫特征词。实体一般不是触发词，因此过滤掉触发词和停用词，剩下的就是实体词。

## 2.2 规则的构建

构建规则前，我们先看看规则长什么样。上文给出一个简单的不完整的规则“S+O*E+”，接下来给出两种形式的规则表达。

第一种表达形式为基于词性标注的推理规则，比如在识别人名中，有一条规则为“Na+Nb → Nb”，其中 Na 为普通名词，Nb 为专有名词，例如“CEO（Na）马化腾（Nb）”，提取出马化腾为人名。这里给出的例子比较简单，会提取出很多错误的实体。因此，需要根据上下文（2 步邻居词）来构建更加具体复杂的规则。

第二种表达采用一个特征集合来泛化表达命名实体 [Tatar and Cicekli, 2011]，以获得更灵活的表达和更好的覆盖率。简单来说，为每个词生成特征，然后在这些特征上做规则的构建与合并，规则反映的是特征的规律。其规则有 4 个部分，第一个部分简单给出规则的目标实体类别（NE class），后面三个部分分别为：

- PRE-FILLER (PRE)：试图感知和匹配目标 NE 之前的文本；
- FILLER (FILL)：试图感知和匹配目标 NE；
- POST-FILLER (POST)：试图感知和匹配目标 NE 之后的文本。

提取规则中的模式段可以被视为一系列模式元素，其类型可以是相似性（SIM）或可选性（OPT）。类型为 SIM 模式的元素需要准确匹配满足元素约束的文本中的一个标记（token）。另一方面，类型为 OPT 的模式元素可以匹配满足元素约束的标记，也可以不匹配。具体地，我们通过一个例子来详细的描述。图 3 给出了两个简单的规则。规则的各个部分之间采用冒号分隔，如图 3所示，第一条规

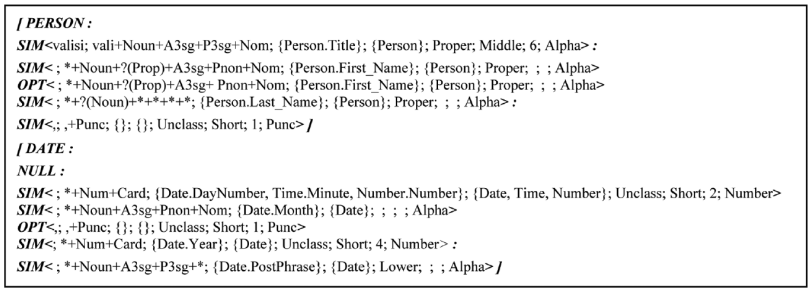



图 3: 规则示例

则的第一部分为“PERSON”，也就是人名这一 NE 类别，第二个部分 PRE，只有一条 SIM 模式元素，模式元素都具备 8 个以分号分隔的域：

1. token;
2. morphological tag（词法分析结果）;
3. low-level gazetteer set;
4. high-level gazetteer set;
5. case tag;
6. length class;
7. token length;
8. type class.

每个域中的模式代表一种约束，为了匹配文本中的 token，必须满足所有的模式约束。第一个域，为原子域，只准确匹配特定的文本，这里里面的元素为 token 文本 valisi（土耳其语，意为总督、统治者），表示目标 NE 必须以 valisi 作为前置元素。第二个域为词法分析的结果，其中的“*”和“?”与正则表达式中的意义相同。第三个和第四个域为词典集合的约束，当前其需要出现在词典中，但是为了发现未登录词，此处可做模糊匹配。剩下的均为原子域，均为正字特征（orthographic features）。规则剩余的部分类似，为空部分表示不匹配。应用以上两条规则，提取出实体的例子如图 4。

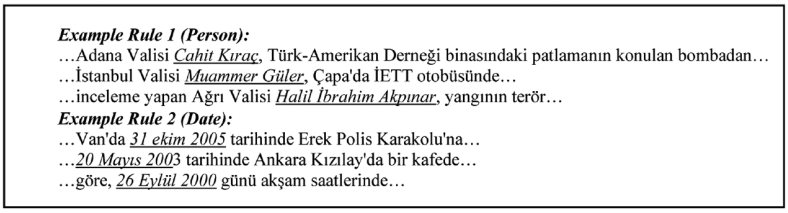
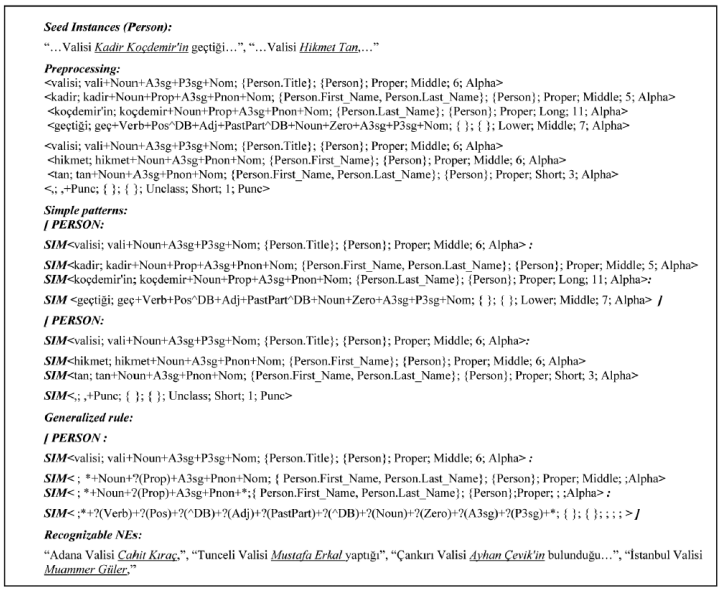



图 4: 使用图 3 匹配的结果示例

不难发现，规则的制定异常繁琐，而且规则制定得太细会造成召回很低，制定得太泛会造成准确率很低。因此，规则学习和后续泛化的能力是命名实

体识别系统中的关键功能之一。能否自动学习并维护规则呢？Tatar and Cicekli [2011] 给出了一种自动学习规则的方案。该方案的基本思想是对模式字符串做泛化，通过处理模式串之间的相似处（共同出现的子串）和不同处（同一个域不同的子串）来归纳两个规则。对于原子域（token, capitalization, length class, length, type class），归纳方式很直接，相同就保留，不同直接置空。对于句法域（morphological tag），采用Cicekli and Cicekli [2006] 中的方法来泛化字符串。对于句法域中的模式串，分析它们的相同之处和不同之处，保留相同之处，将不同之处替换为“任意（*）”或者是“可选（?）”。对于两个字典域，如果两个集合都非空，便合并集合，只要一个为空，便置为空。为了更加形象的解释规则归纳方式，图 5 给出了从两个句子“…Elazığ Valisi Kadir Koçdemir’in geçtiği…”和“…Van Valisi Hikmet Tan, konvoyuna …”，生成初始规则并合并的过程。


***Seed Instances (Person):***

"…Valisi *Kadir Koçdemir'in* geçtiği…", "…Valisi *Hikmet Tan*,…"

***Preprocessing:***

<valisi; vali+Noun+A3sg+P3sg+Nom; {Person.Title}; {Person}; Proper; Middle; 6; Alpha>
<kadir; kadir+Noun+Prop+A3sg+Pnon+Nom; {Person.First_Name, Person.Last_Name}; {Person}; Proper; Middle; 5; Alpha>
<koçdemir'in; koçdemir+Noun+Prop+A3sg+Pnon+Nom; {Person.Last_Name}; {Person}; Proper; Long; 11; Alpha>
<geçtiği; geç+Verb+Pos^DB+Adj+PastPart^DB+Noun+Zero+A3sg+P3sg+Nom; { }; { }; Lower; Middle; 7; Alpha>

<valisi; vali+Noun+A3sg+P3sg+Nom; {Person.Title}; {Person}; Proper; Middle; 6; Alpha>
<hikmet; hikmet+Noun+A3sg+Pnon+Nom; {Person.First_Name}; {Person}; Proper; Middle; 6; Alpha>
<tan; tan+Noun+A3sg+Pnon+Nom; {Person.First_Name, Person.Last_Name}; {Person}; Proper; Short; 3; Alpha>
<,; ,+Punc; { }; { }; Unclass; Short; 1; Punc>

***Simple patterns:***

***[ PERSON:***

***SIM***<valisi; vali+Noun+A3sg+P3sg+Nom; {Person.Title}; {Person}; Proper; Middle; 6; Alpha> ***:***

***SIM***<kadir; kadir+Noun+Prop+A3sg+Pnon+Nom; {Person.First_Name, Person.Last_Name}; {Person}; Proper; Middle; 5; Alpha>
***SIM***<koçdemir'in; koçdemir+Noun+Prop+A3sg+Pnon+Nom; {Person.Last_Name}; {Person}; Proper; Long; 11; Alpha>***:***

***SIM*** <geçtiği; geç+Verb+Pos^DB+Adj+PastPart^DB+Noun+Zero+A3sg+P3sg+Nom; { }; { }; Lower; Middle; 7; Alpha> ***]***

***[ PERSON:***

***SIM***<valisi; vali+Noun+A3sg+P3sg+Nom; {Person.Title}; {Person}; Proper; Middle; 6; Alpha>***:***

***SIM***<hikmet; hikmet+Noun+A3sg+Pnon+Nom; {Person.First_Name}; {Person}; Proper; Middle; 6; Alpha>
***SIM***<tan; tan+Noun+A3sg+Pnon+Nom; {Person.First_Name, Person.Last_Name}; {Person}; Proper; Short; 3; Alpha>

***SIM***<,; ,+Punc; { }; { }; Unclass; Short; 1; Punc>

***Generalized rule:***

***[ PERSON :***

***SIM***<valisi; vali+Noun+A3sg+P3sg+Nom; {Person.Title}; {Person}; Proper; Middle; 6; Alpha> ***:***

***SIM***< ; *+Noun+?(Prop)+A3sg+Pnon+Nom; { Person.First_Name, Person.Last_Name}; {Person}; Proper; Middle; ;Alpha>
***SIM***< ; *+Noun+?(Prop)+A3sg+Pnon+*;{ Person.First_Name, Person.Last_Name}; {Person};Proper; ; ;Alpha> ***:***

***SIM***<;*+?(Verb)+?(Pos)+?(^DB)+?(Adj)+?(PastPart)+?(^DB)+?(Noun)+?(Zero)+?(A3sg)+?(P3sg)+*; { }; { }; ; ; ; > ***]***

***Recognizable NEs:***

"Adana Valisi *Cahit Karaç*,", "Tunceli Valisi *Mustafa Erkal* yaptığı", "Çankırı Valisi *Ayhan Çevik'in* bulunduğu…", "İstanbul Valisi *Muammer Güler*,"

图 5: 规则归纳示例

整体的算法流程见图 6。规则集合 RULES 初始化为空集，然后算法在每个正样本上（有效样本，包含目标实体）上生成规则，并将非重复的规则添加到 RULES 中。规则的生成方式如上文所述，先将文本预处理为特征表示，然后在该特征表示上生成规则。下一步便是尝试对两两规则做合并，按相似性排序，保证相似性高的规则能添加到 RULES 中。紧接着，直到将 k 个广义规则添加到 RULES 中或者 $RULES_{temp}$ 中的每个规则均被验证，将 $RULES_{temp}$ 中的每条规则在训练集上做验证，得到其置信度，将置信度大的规则加入 RULES，同时删除置信度小的。

此外，再介绍一种自动构建规则和弱标注的思想：

1. 初始词典种子；
2. 提供少量的一批样本，寻找包含词典种子词的句子，识别出通用的上下文

```
(1)   RULES ← { }.
(2)   For each positive example e in the example space E:
(3)         R ← e.
(4)         If R ∉ RULES Add R into RULES.
(5)   hasMoreGeneralization ← TRUE
(6)   while (hasMoreGeneralization = TRUE)
(7)         hasMoreGeneralization ← FALSE
(8)         For each rule R1 in RULES:
(9)               RULES_temp ← { }.
(10)              For each rule R2 in RULES (where R1 ≠R2):
(11)                    R ← GEN(R1, R2).
(12)                    If R ∉ RULES_temp Add R into RULES_temp.
(13)              Sort RULES_temp in descending order of the similarity factor
(14)              Until k rules added into RULES or every rule in RULES_temp validated:
(15)                    Test every rule R in RULES_temp on the training dataset and Calculate CF_R (the confidence factor of R).
(16)                    If CF_R >= T (confidence factor threshold) and R ∉ RULES
(17)                          Add R into RULES.
(18)                          Drop rules R1 and R2 , from which R generalized, from RULES
(19)                          hasMoreGeneralization ← TRUE
(20)  Sort RULES in ascending order of the coverage
(21)  For each rule R in RULES:
(22)        If there exists another rule that covers E_R (the positive examples covered by R)
(23)              Drop R from RULES
(24)        If every example in E_R is also covered by another rule in RULES
(25)              Drop R from RULES
```

图 6: 规则归纳算法流程

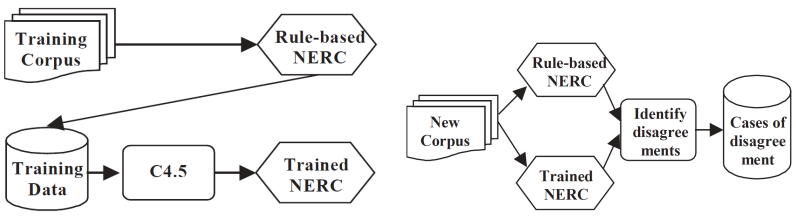
模式（初始规则）；

3. 在语料中寻找具备相识上下文模式的句子；
4. 学习过程应用在新提取出来的句子上，发现新的上下文模式（规则）；
5. 循环以上过程，得到大量的目标实体词及其上下文。

模式存再对 spelling, context 中，spelling 对应实体，context 对应其上下文，最频繁出现的上下文被总结生成上下文规则。

## 2.3 规则维护

以上方法时在给定数据集上自动生成并合并规则，数据集是给定的，如果又来了一批新的语料，那么如何调整规则呢？Petasis et al. [2001] 给出了一种基于机器学习长期维护规则的框架。图 7 给出了该方法的示意图。该方法有两个阶段，训练阶段和运用阶段。训练阶段上，首先在给定训练语料上生成规则，提取出规则匹配到的样本，抽取特征，使用分类器进行分类。注意这里样本的类别是规则匹配的结果，也就是说，分类器拟合的是规则的分类模式。在运用阶段，同时采用规则和分类来对样本进行分类，找出分类结果不一致的样本，重新进行规则的生成与归纳。




(a) training state (b) deployment state

图 7: 一种基于机器学习长期维护规则的框架。

# 3 特征工程

不管是基于规则还是基于机器学习方法，都需要先将原始数据表示为特征，在特征上做进一步的处理。在本节，我们列举出在命名实体识别上常用的特征，大体可以分为三类：Word-level features，List lookup features 和 Document and corpus features[Nadeau and Sekine, 2007]。

## 3.1 Word-level features

Word-level feature 与单词的字符构成有关。它们具体描述单词大小写，标点符号，数值和特殊字符。图 8 列出了其子类别。数字可以表达各种有用的信息，

| Features | Examples |
|---|---|
| Case | - Starts with a capital letter<br>- Word is all uppercased<br>- The word is mixed case (e.g., ProSys, eBay) |
| Punctuation | - Ends with period, has internal period (e.g., St., I.B.M.)<br>- Internal apostrophe, hyphen or ampersand (e.g., O'Connor) |
| Digit | - Digit pattern<br>- Cardinal and Ordinal<br>- Roman number<br>- Word with digits (e.g., W3C, 3M) |
| Character | - Possessive mark, first person pronoun<br>- Greek letters |
| Morphology | - Prefix, suffix, singular version, stem<br>- Common ending |
| Part-of-speech | - proper name, verb, noun, foreign word |
| Function | - Alpha, non-alpha, n-gram<br>- lowercase, uppercase version<br>- pattern, summarized pattern<br>- token length, phrase length |

图 8: word-level features

如日期，百分比，间隔，标识符等。例如，两位数和四位数可以表示年，如果后面紧接着一个“s”，便能表示年代。共同的单词结尾是指词根词缀。例如，人类职业通常以“ist”（journalist, cyclist）结束，或者国籍和语言通常以“ish”和“an”结尾（Spanish, Danish, Romanian）。“Functions over words”是指定义在词上的函数，这个函数一般是用来提取词的模式，比如将词中的非字母字符提取出来（例如，nonalpha(A.T.&T.) = ..&.），将词中的大写字母转为‘A’，小写字母转为’a’，其他字符转为‘-’（GetSummarizedPattern(”Machine-223”) = ”Aa-0”）和定义在上下文上的函数（n-gram 等）。

## 3.2 List lookup features

词典在 NER 中的作用很多，论文中的术语“gazetteer”、“lexicon”、“dictionary”和“list”都是指词典。词典包含一般表达的是一种“is a”的关系，比如“Paris is a city”，如果一句话中出现词“Paris”那么这个词是城市的概率就很

大，也只是概率很大，因为还存在一词多义的现象，这是就需要考虑句子中出现的其他词来判断。使用词典一般就采用词袋来表达特征，图 9 给出了常用的词典，具体需要词典需要根据实际的任务（目标实体）来定。大多数方法要求

| Features | Examples |
|---|---|
| General list | - General dictionary<br>- Stop words (function words)<br>- Capitalized nouns (e.g., January, Monday)<br>- Common abbreviations |
| List of entities | - Organization, government, airline, educational<br>- First name, last name, celebrity<br>- Astral body, continent, country, state, city |
| List of entity cues | - Typical words in organization<br>- Person title, name prefix, post-nominal letters<br>- Location typical word, cardinal point |

图 9: list lookup features

目标实体词需要精确匹配预定义好的词典，但是词典终归是有限的，在有些情况下，需要进行灵活弹性的匹配。下面具体描述三种这样的方法。

第一，在词进行匹配前，对词进行预处理，进行词目化（lemmatized）。比如过去式、将来时都转化为现在时，复数变为单数，去掉词缀只保留词根等。

第二，目标词可以采用“模糊匹配的方式”与词典进行比对，比如采用一定阈值下的编辑距离和 Jaro-Winkler[Cohen and Sarawagi, 2004]。这允许捕获不一定是派生的单词中的小词汇变化。例如，Frederick 可以匹配 Frederik，因为两个单词之间的编辑距离非常小（仅抑制一个字符，’c’）。Jaro-Winkler 的度量标准专门用于匹配专有名称，因为观察到第一个字母往往是正确的，而名称结尾往往是变化的。

第三，可以使用 Soundex 算法 [Raghavan and Allan, 2004] 访问预定义词典，该算法将目标候选词归一化为它们各自的 Soundex 码。此代码是一个单词的第一个字母加上代表其语音的三位数代码的组合。因此，类似的声音名称如 Lewinskey（soundex = l520）和 Lewinsky（soundex = l520）在其 Soundex 码上是相同的。

## 3.3 Document and corpus features

文档特征时定义在文档内容和文档结构上的特征，主要是指有关文档和语料库统计信息的元信息。图 10 列出了一些文档特征。对于多词共现，在一些文档中经常会出现大小写同时出现的情况，一般词在句首会大写，通过这种方式便可以判断这个词为并非人名，人名在任何地方都会大写。大多数关于文档的元信息都可以直接使用：电子邮件标题是人名的良好指标，新闻通常以位置名称开头等。以上 feature 是一种从粒度上的划分方式，条目也很多，实际中常用的 feature 列举如下：

1. **Morphological**: n-gram character, n-gram word, suffixes and prefixes
2. **Orthographic**: capitalization, symbols

| Features | Examples |
|---|---|
| Multiple occurrences | - Other entities in the context<br>- Uppercased and lowercased occurrences<br>- Anaphora, coreference |
| Local syntax | - Enumeration, apposition<br>- Position in sentence, in paragraph, and in document |
| Meta information | - Uri, Email header, XML section,<br>- Bulleted/numbered lists, tables, figures |
| Corpus frequency | - Word and phrase frequency<br>- Co-occurrences<br>- Multiword unit permanency |

图 10: 文档特征

3. **Linguistic**: lemmatization, stemming, POS tag, chunking, syntactic parsing
4. **Context**: windows, conjunctions
5. **Domain knowledge**: lexicons, exiting NER tools

# 4 基于机器学习方法

采用机器学习方法构建模型是目前命名实体识别的主流。命名实体识别本质上可以视为一个序列标注问题，因此传统方法大多采用 HMM、ME 和 CRF 等概率图模型。随着 RNN 在序列上的惊人表现，今年来 NER 上基于 Bi-LSTM-CRF 的深度学习方法成为了主流。如果从模型的学习任务上分，NER 方法可以分为有监督、半监督和迁移学习三类，下文分别进行描述。

## 4.1 有监督

有监督方法具体有：

- SVM
- 决策树
- HMM
- MEMM
- CRF
- NN/CNN-CRF
- Bi-LSTM-CRF
- Bi-LSTM-CRF+Attention

SVM 和决策树等将 NER 视为一个分类问题，模型以词为单位，根据提取的特征和标注好的类别来进行训练。这样存在的问题是对分词要求较高，而且捕获上下文关系（序列关系）的能力很弱（只能通过上下文特征），所以之后更多的是将 NER 视为序列标注问题。HMM 和 MEMM 都是最基础的序列模型，这里不再赘述。CRF 是除了深度学习应用最广而且效果最好的方法。图 11 形象描述了 CRF 的模型定位，图中给出的是这些模型的图模型表示。图 11 中，上面一排都属于生成式模型，对应的，下面一排都属于判别式模型。就像 NB 和 LR 是一

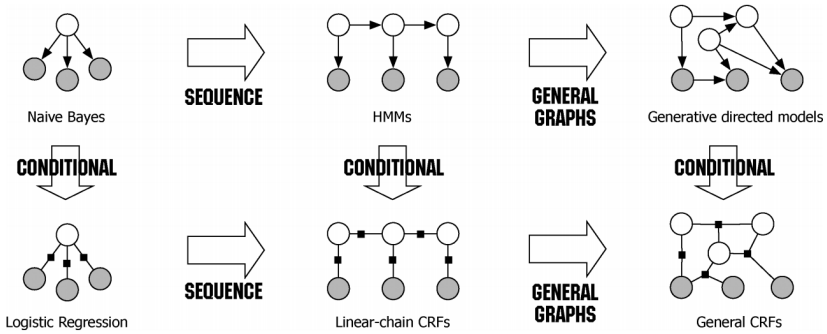



图 11: CRF 与 NB、LR 和 HMM 的关系

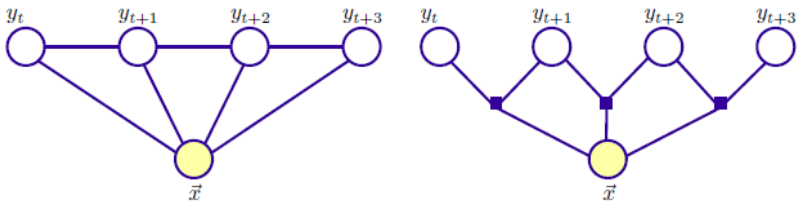
对生成式-判别式对，HMM 和 CRF 也同样是一对，当联合分布为 HMM 的形式时，相应的条件概率分布为线性链式的 CRF。具体 CRF 和 HMM 之间关系详细推到可参考Sutton et al. [2012]。下面以线性链条件随机场为例来介绍 CRF。

线性链条件随机场可以采用因子图模型 (TODO: 回去找概率图模型的 pdf) 的形式直观的表达：

$$p(y|x) = \frac{1}{Z(x)} \prod_{t=1}^{T} \Psi_t(y_t, y_{t-1}, x_t) \tag{1}$$

$$\Psi_t(y_t, y_{t-1}, x_t) = \exp\{\sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t)\} \tag{2}$$

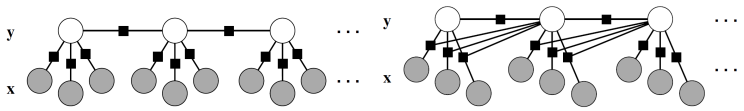
图 12 为其对应的因子图模型的形式。因子图中有两种类型的节点，variable node




(a) Independency graph (b) Factor graph

图 12: Linear chain CRF

和 factor node。在图 11 分别以圆圈和黑方块表示。variable node 定义的是数值特征向量，factor node 为定义在其连接的 variable node 上的函数。这里将输入特征 $x$ 和各个类别 $y$ 写到一个函数中的定义方式是为了兼容不同的特征函数 $\Psi_t(\cdot)$。图 13 列举了另外两种常见的定义方式。在训练时可以使用 SGD 学习模型参数。在已知模型时，给输入序列求预测输出序列即求使目标函数最大化的最优序列，是一个动态规划问题，可以使用维特比算法进行解码。CRF 是在整个序列上计算全局最优，所以计算复杂度相对于 HMM 和 MEMM 会高一些，速度较慢，但

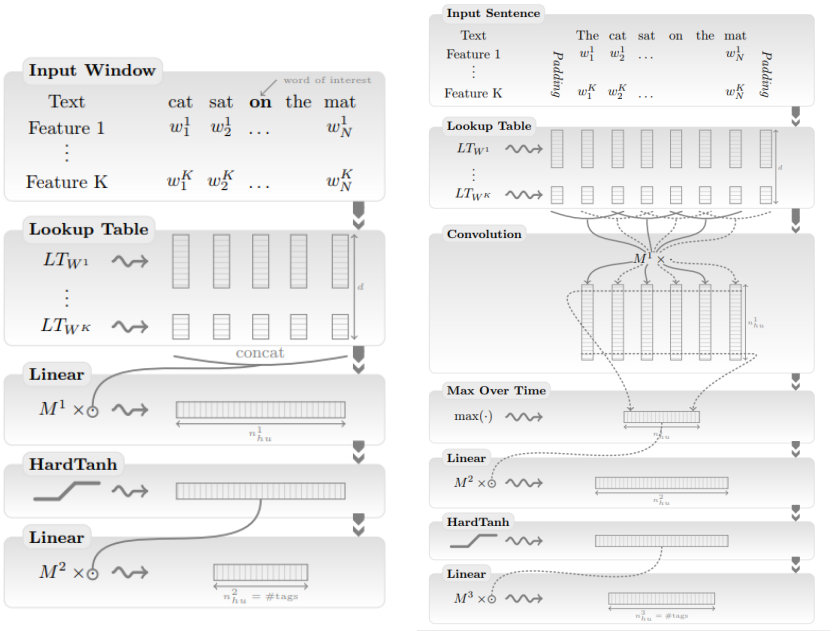



(a) (b)

图 13: 线性链 CRF 的其他定义方式

优点是特征设计灵活，可以容纳较多的上下文信息，能够做到全局最优。

随着神经网络的复兴，深度学习方法逐渐成为 NER 的热点。Collobert et al. [2011] 是较早使用 NN/CNN-CRF 神经网络模型的代表作之一。在这篇论文中，作者提出了窗口方法与句子方法两种网络结构来进行 NER。窗口方法仅使用当前预测词的上下文窗口进行输入，然后使用传统的 NN 结构；而句子方法是以整个句子作为当前预测词的输入，加入了句子中相对位置特征来区分句子中的每个词，然后使用了一层卷积神经网络 CNN 结构。这里的输入采用的是词袋的




(a) window approach (b) sentence approach

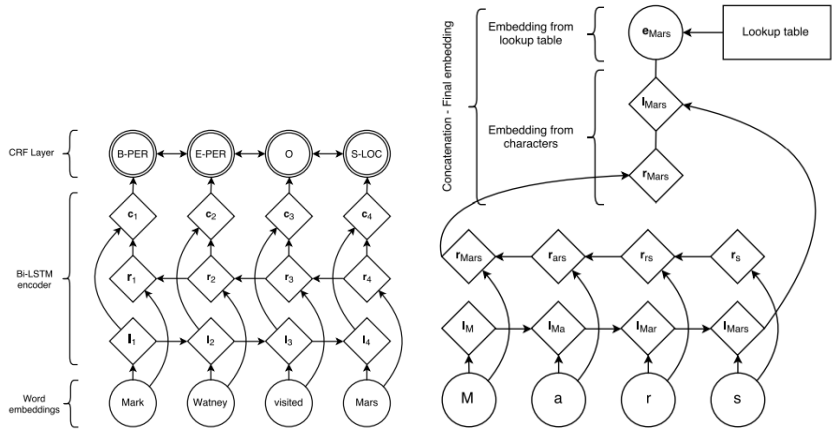
图 14: NN/CNN-CRF 模型结构

形式，采用 Lookup Table 来学习隐层特征。在训练阶段，作者也给出了两种目标函数：一种是词级别的对数似然，即使用 softmax 来预测标签概率，当成是一个传统分类问题；另一种是句子级别的对数似然，其实就是考虑到 CRF 模型在序列标注问题中的优势，将标签转移得分加入到了目标函数中。后来许多相关

工作把这个思想称为结合了一层 CRF 层。在作者的实验中，上述提到的 NN 和 CNN 结构效果基本一致，但是句子级别似然函数即加入 CRF 层在 NER 的效果上有明显提高。

借鉴 CRF 的思路，后来出现了一系列 RNN+CRF 进行 NER 的优秀工作[1]。RNN 中的主要代表是 LSTM，通过使用双向的 LSTM 使得编码结果能捕获序列信息。这些工作根据编码对象又可以分为词级别 LSTM-CRF、字符级别 LSTM-CRF 和词 + 句子级别 LSTM-CRF。

Lample et al. [2016] 给出了词级别和字符级别的 NER 神经网络框架，如图 15。在词级别中，输入的 word embedding 是在语料上预训练好的，通过双向




(a) word-level (b) character-level

图 15: 词级别和字符级别的 NER 神经网络框架

LSTM 后拼接前向跟后向的编码结果作为最终词编码，最后通过 CRF 分类。在字符级别中，双向 LSTM 作用在单词中的字符上，来捕捉一个词语内部字符级别的信息。最终的编码结果除了双向 LSTM 的编码，还添加了 Lookup table 的结果，即词级别的预编码。在训练阶段，模型的目标是最大化标注结果序列 $y$ 的对数概率：

$$\log(p(y|X)) = \log(\frac{e^{s(X,y)}}{\sum_{\widetilde{y}\in Y_X} e^{s(X,\widetilde{y})}}) \tag{3}$$

$$= s(X,y) - \log(\sum_{\widetilde{y}\in Y_X} e^{s(X,\widetilde{y})}) \tag{4}$$

其中 s(X,y) 为标注结果序列 $y$ 的得分。

$$s(X,y) = \sum_{i=0}^{n} A_{y_i,y_{i+1}} + \sum_{i=1}^{N} P_{i,y_i} \tag{5}$$

[1] https://github.com/sebastianruder/NLP-progress/blob/master/named_entity_recognition.md

其中 $A$ 为标签之间的概率转移矩阵，$P \in \mathcal{R}^{n \times k}$ 为 Bi-LSTM 的输出结果，$k$ 为标签的个数。在预测时，得到序列的序列通过最大化得分得到：

$$y^* = \text{argmax}_{\widetilde{y} \in Y_X} s(X, \widetilde{y}) \tag{6}$$

在字符级别上，今年 COLING 上的一篇文章”Contextual String Embeddings for Sequence Labeling“[Akbik et al., 2018] 是一个很好的代表。通过学习基于先前字符预测下一个字符，这些模型已经被证明能自动内化（internalize）语言概念，例如单词，句子，子句甚至情感。在这篇文章中，他们利用训练好的语言模型的内部状态来产生一种新颖的单词编码模型，称为上下文字符串编码。他们指出模型有两个不同之处，1) 在没有任何明确的单词概念的情况下被训练，因此从根本上将单词建模为字符序列；2) 通过其周围文本进行语境化，这意味着相同的单词将具有不同的编码，取决于其上下文。其模型的整体架构见图 16。 模型

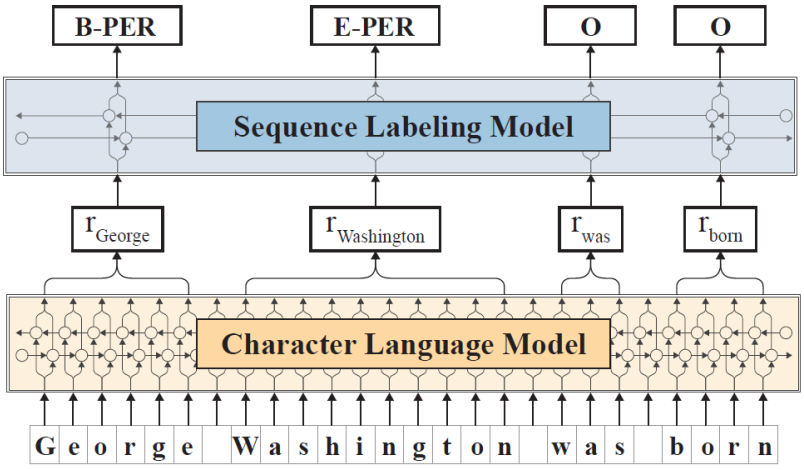



图 16: Contextual string embedding

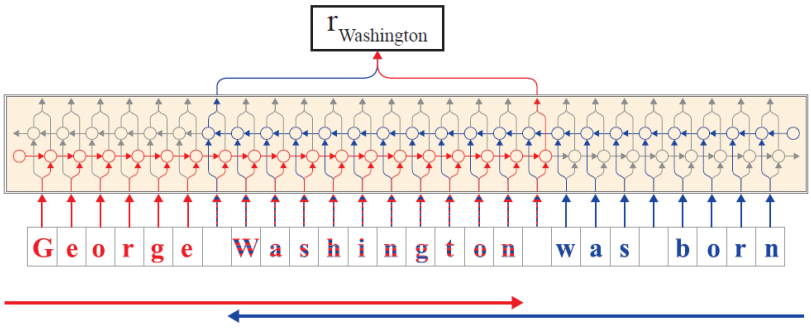
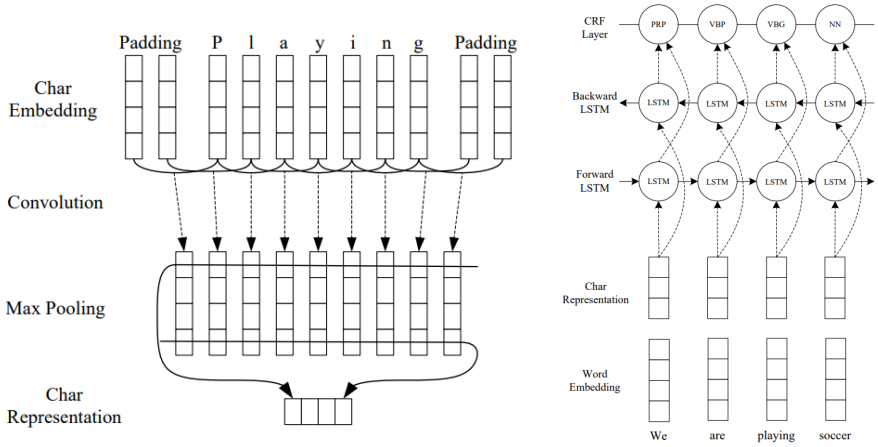



图 17: 预训练语言模型

首先通过语言模型预训练，然后提取出词级别的特征，后面接上 Bi-LSTM+CRF。

这里的特点在于前期的预训练语言模型和词级别特征的生成。字符级别的语言模型其实就是定义在句子中字符级别上的无监督 Bi-LSTM，每个字符为一个单元。这样相同的字符在句子中不同的位置会有不同的编码结果。字符编码训练完成后，根据分词结果得到具体每个词的用于序列标注模型的输入特征。具体取法见图 17 。不难发现，这与图 15 中的结构很相似，其实也都是这些套路。

最后是混合方法，这里我们以 BLSTM-CNNs-CRF[Ma and Hovy, 2016] 模型为代表进行描述。其核心特点是用 CNN 将字符编码转化为词级别的字符特征表达。这些都预训练好之后，在输入 Bi-LSTM 之前，将句子中词的字符特征表达和使用 word2vec 等预训练好的 Word Embedding 拼接，作为序列标注模型的输入。




(a) 使用 CNN 提取字符级别表达　　(b) BLSTM-CNN-CRF 模型结构

图 18: BLSTM-CNN-CRF 模型

总结来说，混合方法的核心是构造序列标注模型 Bi-LSTM-CRF 的输入，通过融合多方面的特征。下图给出了一个这类方法的总结。这类方法一般都是先在

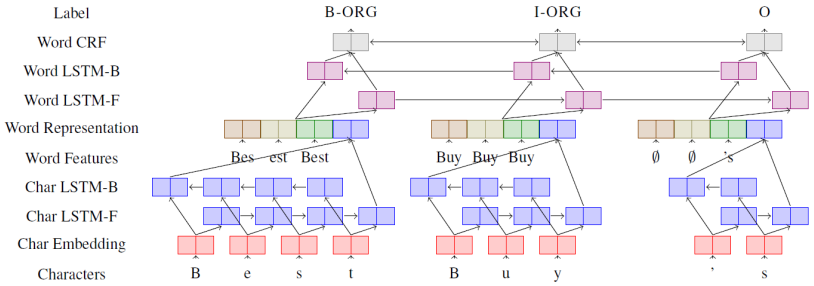



图 19: 词 + 字符 + 词缀等混合模型结构

字符级别做预训练，不同之处在于，是以词为单位还是以句子为单位。图 19 所

示是以词为单位，图 17 是以整个句子为一条输入。得到字符编码结果后，取词两端的 LSTM 的输出作为词的编码结果，然后再拼接上其他方面的编码，比如，word embedding、词缀和音韵 [Bharadwaj et al., 2016] 等。最近的很多工作都在构造这个最终用于输入的词向量上做文章。

## 4.2 半监督

有监督模型需要大佬高质量的训练数据，也就是所谓的强监督。但是实际中往往大量缺失标注数据，是弱监督，弱监督也是未来研究的一个重点方向。弱监督主要有两类方法：半监督和迁移学习。本节介绍在 NER 任务上的半监督模型，下一小节介绍迁移学习。

TagLM[Peters et al., 2017] 是 ACL2017 录用的最近比较火的工作。该工作使用海量无标注语料库训练了一个 Bi-LSTM，然后使用这个训练好的语言模型来获取当前要标注词的语言模型向量（LM embedding），然后将该向量作为特征加入到原始的双向 RNN-CRF 模型中。实验结果表明，在少量标注数据上，加入这个语言模型向量能够大幅度提高 NER 效果，即使在大量的标注训练数据上，加入这个语言模型向量仍能提供原始 RNN-CRF 模型的效果。TagLM 的主要流程如图 20。仔细看这个流程会发现，这个流程与上文有监督描述的方法有

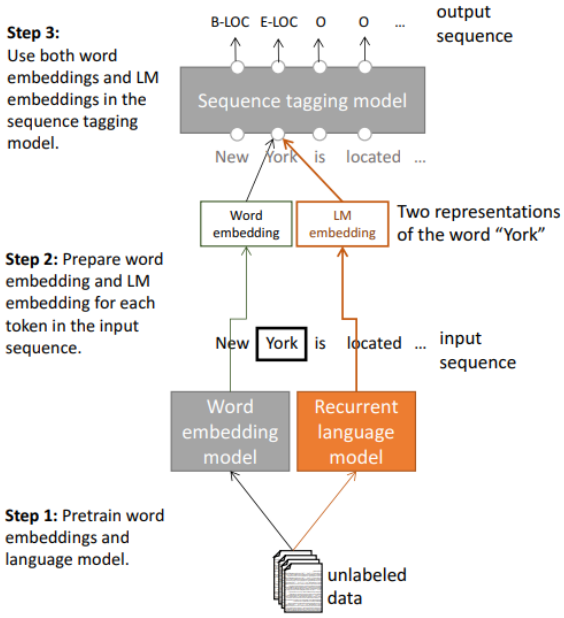



图 20: TagLM 的主要流程

着不清不楚的关系，都具备一个在大量语料上的预训练过程和编码拼接过程。事实上也确实如此，深度学习中预训练已经成为一个 trick，本身就包含了迁移

和半监督的概念。如果非要说与上小节的不同之处的话，TagLM 的预训练是在目标任务数据集上进行的，也就是说任务本身的特定领域。上小节的预训练可能是在更加 general 的语料库上做得训练，或者说是直接拿的 google 已训练好的 word2vec 等。这时预训练不在特定领域进行，而是更加一般化的语料，这对于数据较少（不单单标注数据少）的情况比较有效。

注意，TagLM 的工作不止于此，这只是半监督的流程，其模型的具体细节如下图：这里需要注意的有两个地方。第一，在输入序列标注模型 bi-RNN 之前，

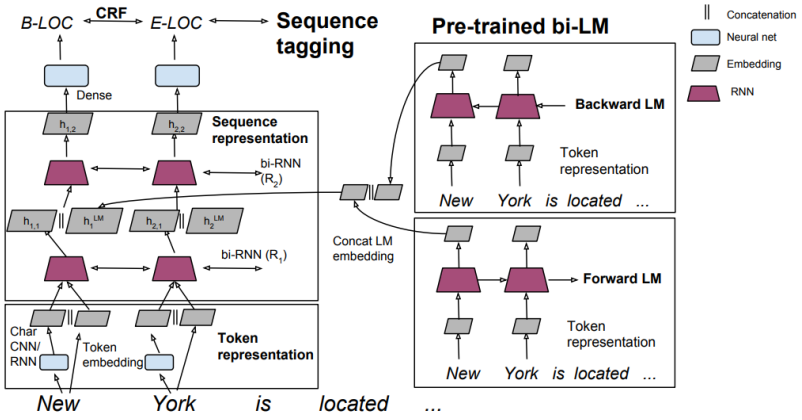



图 21: TagLM 模型结构

词的特征是由 Token embedding 和 character-level 的编码得到，这个步骤与上小节所述的方法一致。注意，这里并没有加入在当前语料上预训练的词编码，这便是与上小节的不同之处。那么加到哪里了呢，这就是需要注意的第二点，加到了第一层 bi-RNN 之后。TagLM 采用了两个 bi-RNN 层，这两个 trick 的使用并非空穴来风。之前的研究表明，深度 bi-RNN 的不同隐层编码者不同的信息。例如，在深度 LSTM 模型中的浅层引入多任务语法监督（如 POS tags）能提高更高层任务的整体性能，比如依赖解析 [Hashimoto et al., 2016] 或者 CCG super tagging[Søgaard and Goldberg, 2016]。Belinkov et al. [2017] 指出在两层 LSTM 中，第一层的编码比第二层的在预测 POS tags 上效果更好。所以，TagLM 这篇文章指出实验中也发现将预训练的结果拼接到第一层 bi-RNN 后表现最好。

## 4.3 迁移学习

如上文所述，深度学习需要大量的标注语料用于训练，那么现有模型在没有足够训练数据的任务上是否有效呢？通过迁移学习，Yang et al. [2017] 给出了肯定的回答。他们给出了三种不同的迁移学习架构。基本的序列标注模型在上文已经进行了详细的描述，这里主要对这三种迁移模型进行解释。

由于不同的领域是具有领域特定规则性的“子语言”，因此在一个领域上训练的序列标记模型可能在另一个域上不具有最佳性能。跨域迁移的目标是学

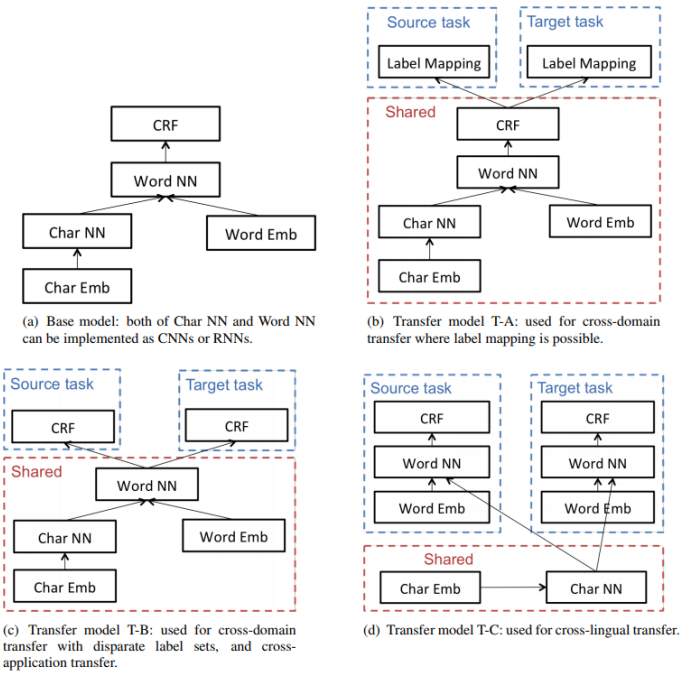



图 22: 序列标注迁移学习架构

习将知识从源域转移到目标域的序列标记器。我们假设目标域中有少量标签可用。有两种情况的跨域迁移，这两个域可能具有可以相互映射的标签集，也可能时两个相互独立的标签集。如果这两个域有可以相互映射的标签集，那么就可以神经网络模型中的所有参数和特征表达，包括词和字符编码、word-level 层、character-level 层和 CRF 层。只是在 CRF 后面多加一个标签映射过程，这种模型为 b) 中 T-A。如果属于后一种情况，便不在 CRF 层做参数共享，每个域单独训练 CRF 层，这种模型为 c) 中 T-B。

以上都假设在同一种语言下做相同的任务（比如都在英语下做序列标注任务），对于不同任务的情况，也采用 T-B。在跨语言的情形下，虽然跨语言转移通常通过额外的多语言资源来完成，但这些方法对其他资源的大小和质量很敏感。所以 d) 中给出的模型 T-C，侧重于具有相似字母表的语言之间的转移学习，例如英语和西班牙语，对于英语和中文就无法迁移。图 23 指出，迁移后确实有所提升。

## 4.4 小结

深度学习 NER 模型都处于 Bi-LSTM-CRF 框架下，不同的地方在于编码的学习，词编码、字符编码和句子编码等。然后就是有很多模型设计上的技巧和规律被发现和解释。CoNLL 2003 NER 任务包括来自路透社 RCV1 语料库的新闻专线文本，标记有四种不同的实体类型（PER，LOC，ORG，MISC），模型采

| Source | Target | Model | Setting | Transfer | No Transfer | Delta |
|---|---|---|---|---|---|---|
| PTB | Twitter/0.1 | T-A | dom | 83.65 | 74.80 | 8.85 |
| CoNLL03 | Twitter/0.1 | T-A | dom | 43.24 | 34.65 | 8.59 |
| PTB | CoNLL03/0.01 | T-B | app | 74.92 | 68.64 | 6.28 |
| PTB | CoNLL00/0.01 | T-B | app | 86.73 | 83.49 | 3.24 |
| CoNLL03 | PTB/0.001 | T-B | app | 87.47 | 84.16 | 3.31 |
| Spanish | CoNLL03/0.01 | T-C | ling | 72.61 | 68.64 | 3.97 |
| CoNLL03 | Spanish/0.01 | T-C | ling | 60.43 | 59.84 | 0.59 |
| PTB | Genia/0.001 | T-A | dom | 92.62 | 83.26 | 9.36 |
| CoNLL03 | Genia/0.001 | T-B | dom&app | 87.47 | 83.26 | 4.21 |
| Spanish | Genia/0.001 | T-C | dom&app&ling | 84.39 | 83.26 | 1.13 |
| PTB | Genia/0.001 | T-B | dom | 89.77 | 83.26 | 6.51 |
| PTB | Genia/0.001 | T-C | dom | 84.65 | 83.26 | 1.39 |

图 23: 迁移前后效果对比

用 span-based F1 来进行评估。下面列举出在这个任务上的一些 state-of-the-art 模型的结果，其中有一些在上文进行了介绍，放在这里作为一个小结。

表 1: CoNLL 2003 NER 上 state-of-the-art 模型

| Model | F1 | Paper | Code |
|---|---|---|---|
| Flair embeddings[Akbik et al., 2018] | 93.09 | Contextual String Embeddings for Sequence Labeling | Flair framework |
| BiLSTM-CRF+ELMo[Peters et al., 2018] | 92.22 | Deep contextualized word representations | AllenNLP GitHub |
| Peters et al. [2017] | 91.93 | Semi-supervised sequence tagging with bidirectional language models | |
| LM-LSTM-CRF[Liu et al., 2017] | 91.71 | Empowering Character-aware Sequence Labeling with Task-Aware Neural Language Model | LM-LSTM-CRF |
| Yang et al. [2017] | 91.26 | Transfer Learning for Sequence Tagging with Hierarchical Recurrent Networks | |
| Ma and Hovy [2016] | 91.21 | End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF | |
| LSTM-CRF[Lample et al., 2016] | 90.94 | Neural Architectures for Named Entity Recognition | |

# 参考文献

Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, 2018.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. What

do neural machine translation models learn about morphology? *arXiv preprint arXiv:1704.03471*, 2017.

Akash Bharadwaj, David Mortensen, Chris Dyer, and Jaime Carbonell. Phonologically aware neural model for named entity recognition in low resource transfer settings. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1462–1472, 2016.

Ilyas Cicekli and Nihan Kesim Cicekli. Generalizing predicates with string arguments. *Applied Intelligence*, 25(1):23, 2006.

William W Cohen and Sunita Sarawagi. Exploiting dictionaries in named entity extraction: combining semi-markov extraction processes and data integration methods. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98. ACM, 2004.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537, 2011.

Dimitra Farmakiotou, Vangelis Karkaletsis, John Koutsias, George Sigletos, Constantine D Spyropoulos, and Panagiotis Stamatopoulos. Rule-based named entity recognition for greek financial texts. In *Proceedings of the Workshop on Computational lexicography and Multimedia Dictionaries (COMLEX 2000)*, pages 75–78, 2000.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. A joint many-task model: Growing a neural network for multiple nlp tasks. *arXiv preprint arXiv:1611.01587*, 2016.

Kevin Humphreys, Robert Gaizauskas, Saliha Azzam, Chris Huyck, Brian Mitchell, Hamish Cunningham, and Yorick Wilks. University of sheffield: Description of the lasie-ii system as used for muc-7. In *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29-May 1, 1998*, 1998.

George R Krupka and Kevin Hausman. Isoquest inc.: Description of the netowl (tm) extractor system as used for muc-7. In *Proceedings of MUC*, volume 7, 1998.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

Liyuan Liu, Jingbo Shang, Frank Xu, Xiang Ren, Huan Gui, Jian Peng, and Jiawei Han. Empower sequence labeling with task-aware neural language model. *arXiv preprint arXiv:1709.04109*, 2017.

Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*, 2016.

David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

Georgios Petasis, Frantz Vichot, Francis Wolinski, Georgios Paliouras, Vangelis Karkaletsis, and Constantine D Spyropoulos. Using machine learning to maintain rule-based named-entity recognition and classification systems. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 426–433. Association for Computational Linguistics, 2001.

Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. Semi-supervised sequence tagging with bidirectional language models. *arXiv preprint arXiv:1705.00108*, 2017.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

Hema Raghavan and James Allan. Using soundex codes for indexing names in asr documents. In *Proceedings of the Workshop on Interdisciplinary Approaches to Speech Indexing and Retrieval at HLT-NAACL 2004*, pages 22–27. Association for Computational Linguistics, 2004.

Anders Søgaard and Yoav Goldberg. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 231–235, 2016.

Charles Sutton, Andrew McCallum, et al. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373, 2012.

Serhan Tatar and Ilyas Cicekli. Automatic rule learning exploiting morphological features for named entity recognition in turkish. *Journal of Information Science*, 37(2): 137–151, 2011.

Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. Transfer learning for sequence tagging with hierarchical recurrent networks. *arXiv preprint arXiv:1703.06345*, 2017.