人工智能研究杂志 57（2016）345-420 提交 9/15；发表 11/16

# 神经网络模型的基础
# 用于自然语言处理

**Yoav Goldberg** YOAV.GOLDBERG@GMAIL.COM
*计算机科学系*
*以色列巴伊兰大学*

## 摘要

在过去的几年里，神经网络重新成为强大的机器学习模型，在图像识别和语音处理等领域产生了最先进^结果。 最近，神经网络模型也开始应用于文本自然语言信号，再次取得了非常有希望的结果。 本教程从自然语言处理研究的角度对神经网络模型进行了调查，试图使自然语言研究人员加快神经技术的速度。 本教程涵盖自然语言任务的输入编码、前馈网络、卷积网络、递归网络和递归网络，以及用于自动梯度计算的计算图抽象。

## 1. 导言

十多年来，核心 NLP 技术以机器学习方法为主，这些方法使用线性模型，如支持向量机或 Logistic 回归，在非常高维但非常稀疏的特征向量上训练。

最近，该领域在从稀疏输入的线性模型切换到密集输入的非线性神经网络模型方面取得了一些成功。 虽然大多数神经网络技术很容易应用，有时几乎取代了旧的线性分类器，但在许多情况下存在很强的进入障碍。 在本教程中，我试图向 NLP 实践者（以及新来者）提供基本背景、行话、工具和方法，使他们能够理解神经网络模型背后的原理，并将它们应用于自己的工作。 本教程预期是独立的，同时在统一的表示法和框架下介绍不同的方法。 它重复了许多其他地方可用的材料。 它还指出了外部来源更先进的主题，在适当的时候。

这本入门书并不打算作为一个全面的资源，为那些将继续和发展神经网络机械的下一个进展（尽管它可能是一个很好的切入点）。 相反，它是针对那些有兴趣使用现有的、有用的技术并以有用和创造性的方式应用于他们最喜欢的 NLP 问题的读者。 对于更深入，一般性地讨论神经网络，其背后的理论，先进的优化方法和其他先进的主题，读者可以参考其他现有资源。 特别是，强烈推荐本吉奥、古德费罗和库维尔（2015 年）的书。

### 1.1 范围

重点是神经网络在语言处理任务中的应用。 然而，一些带有神经网络的语言处理子领域被故意排除在本教程的范围之外。 其中包括大量的语言建模和声学建模文献，使用神经网络进行机器翻译，以及结合语言和其他信号（如图像和视频）的多模态应用。 标题生成)。 对于高效运行时性能的缓存方法、具有大输出词汇表的高效训练方法和注意模型也没有讨论。 单词嵌入只在需要理解的范围内讨论，以便将它们用作其他模型的输入。 其他无监督的方法，包括自动编码器和递归自动编码器，也超出了范围。 虽然文中提到了神经网络在语言建模和机器翻译中的一些应用，但它们的处理并不全面。

### 1.2 术语说明

“特征”一词用于指具体的语言输入，如单词、后缀或词性标记。 例如，在一阶词性标记器中，特征可能是“当前单词、前一个单词、下一个单词、前一个词性部分”。 术语“输入向量”是指输入到神经网络分类器的实际输入。 同样，“输入向量条目”是指输入的特定值。 这与许多神经网络文献形成鲜明对比，其中“特征”一词在这两种用途之间超载，主要用于指输入向量条目。

### 1.3 数学符号

我用粗体大写字母表示矩阵(X，Y，Z)，用粗体小写字母表示向量(B)。 当有一系列相关矩阵和向量（例如，每个矩阵对应于网络中的不同层）时，使用上标索引($W^1$，$W^{12}$)。 对于我们想要指示矩阵或向量的幂的罕见情况，在项目周围添加一对括号，以便进行指数化：$(W)^2$，$(w)^{32}$。 除非另有说明，向量被假定为行向量。 我们使用[v1；v2]来表示向量连接。

选择使用行向量，它是右乘以矩阵(xW+b)有点不标准-许多神经网络文献使用列向量，这些列向量是左乘以矩阵(W x+b)。 我们相信读者在阅读文献时能够适应列向量表示法。[1]

1 使用行向量表示法的选择受到以下好处的启发：它与方法相匹配
输入向量和网络图通常在文献中绘制，它使层次/层次化
网络的结构更加透明，并将输入作为最左边的变量而不是嵌套；它导致完全连接的层维是 DINxDout 而不是 Doutxdi，它更好地映射到使用矩阵库(如 numpy)在代码中实现网络的方式。

## 2. 神经网络架构

神经网络是强大的学习模型。 我们将讨论两种神经网络结构，可以混合和匹配-前馈网络和递归/递归网络。 前馈网络包括具有完全连接层的网络，如多层感知器，以及具有卷积层和池层的网络。 所有的网络都充当分类器，但每个网络的强度不同。

完全连接的前馈神经网络（第 4 节）是非线性学习者，在大多数情况下，无论使用线性学习者的情况下，都可以用作下拉替换。 这包括二进制和多类分类问题，以及更复杂的结构化预测问题（第 8 节）。 网络的非线性，以及易于集成预先训练的单词嵌入的能力，往往导致优越的分类精度。 系列作品[2]通过简单地将解析器的线性模型替换为完全连接的前馈网络，获得了改进的句法分析结果。 前馈网络作为分类器替换的直接应用（通常与使用预先训练的词向量相结合）也为 CCG 超标记提供了好处，[3]对话框状态跟踪，[4]统计机器翻译的预先排序[5]以及语言建模。[6]伊耶尔、Manjunatha、Boyd-Graber 和 DaumeIII（2015）表明，多层前馈网络可以在情感分类和事实问题回答方面提供竞争结果。

具有卷积层和池层的网络（第 9 节）对于分类任务是有用的，在分类任务中，我们期望找到关于类隶属度的强局部线索，但这些线索可以出现在输入中的不同位置。 例如，在文档分类任务中，单个关键短语(或 ngram)可以帮助确定文档的主题(Johnson&Zhang，2015)。我们想了解的是，某些单词序列是该主题的良好指标，不一定关心它们在文档中的位置。卷积层和池层允许模型学习找到这样的局部指标，而不管它们的位置如何。 卷积和池结构在许多任务上显示出有希望的结果，包括文档分类，[7]短文本分类，[8]情感分类，[9]实体之间的关系类型分类，[10]事件检测，[IV V]释义识别，[12]语义角色标记，[13]回答问题，[14]预测票房 Rev-

2. 陈和曼宁（2014 年)、魏斯、阿尔伯蒂、柯林斯和彼得罗夫(2015 年)、裴、葛和张(2015 年)以及杜雷特和克莱因(2015 年）
3. 刘易斯和斯蒂德曼（2014 年）
4. 亨德森，汤姆森和杨（2013 年）
5. 德吉斯伯特、伊格莱西亚斯和伯尔尼（2015 年）
6. Bengio，Ducharme，Vincent，and Janvin（2003）and Vaswani，Zhao，Fossum，and Chiang（2013）
7. 约翰逊和张（2015）
8. 王，徐，徐，刘，张，王，郝(2015a)
9. Kalchbrenner、Grefenstette 和 Blunsom（2014 年）和 Kim（2014 年）
10. 曾，刘，赖，周，赵（2014)，多斯桑托斯，项，周(2015)
V 陈，徐，刘，曾，赵（2015)，阮和格里斯曼(2015)
12. 殷和舒策（2015）
13. Collobert、Weston、Bottou、Karlen、Kavukcuoglu 和 Kuksa（2011 年）
14. 董，魏，周，徐（2015）

[1516]基于评论家评论的电影场景，建模文本趣味性，并建模字符序列和部分语音标签之间的关系。[17]

在自然语言中，我们经常使用任意大小的结构化数据，如序列和树。 我们希望能够捕捉这种结构中的规律性，或者模拟这种结构之间的相似性。 在许多情况下，这意味着将结构编码为固定宽度向量，然后我们可以将其传递给另一个统计学习者进行进一步处理。 虽然卷积和池结构允许我们将任意大项编码为固定大小向量，捕捉它们最显著的特征，但它们通过牺牲大部分结构信息来实现。 另一方面，递归（第 10 节)和递归(第 12 节）体系结构允许我们在保存大量结构信息的同时使用序列和树。 递归网络(Elman，1990)被设计成对序列进行建模，而递归网络(Goller&Kuchler，1996)是可以处理树的递归网络的推广。 我们还将讨论递归网络的扩展，使它们能够建模堆栈(Dyer、Ballesteros、Ling、Matthews 和 Smith，2015 年；Watanabe 和 Sumita，2015 年)。

已经证明，递归模型对于语言建模以及序列标记、机器翻译、依赖分析产生了非常强的结果，[18192021 22232425]情感分析，噪声文本归一化，对话状态跟踪，响应生成，并建模字符序列和部分语音标签之间的关系。[26]

递归模型被证明能够产生最先进或接近最先进的结果，用于选区和依赖分析重新排序、话语解析、语义关系分类、基于解析树的政治意识形态检测、情感分类，[272829303132]目标依赖的情感分类和问题回答。[3334]

---

15 比特瓦伊和科恩（2015 年）
16 高，潘特尔，加蒙，何，邓（2014）
17 多斯桑托斯和扎德罗兹尼（2014 年）
18 一些著名的作品是 Mikolov、Karafiat、Burget、Cernocky 和 Khudanpur（2010 年）、Mikolov、Kombrink、Lukas Burget、Cernocky 和 Khudanpur（2011 年）、Mikolov（2012 年）、Duh、Neubig、Sudoh 和 Tsukada（2013 年）、Adel、Vu 和 Schultz（2013 年）、Auli、Galley、Quirk 和 Zweig（2013 年）以及 Auli 和 Gao（2014 年）
19 伊索和卡迪（2014 年)、徐、奥里和克拉克(2015 年）、凌、戴尔、布莱克、特兰索索、费尔南德斯、埃米尔、马鲁乔和路易斯(2015 年 b)
20 Sundermeyer、Alkhouli、Wuebker 和 Ney（2014 年）、Tamura、Watanabe 和 Sumita（2014 年）、Sutskever、Vinyals 和 Le（2014 年）和 Cho、van Merrienboer、Gulcehre、Bahdanau、Bougares、Schwenk 和 Bengio(2014 年 b)
21 Dyer 等人。 （2015 年）、渡边和 Sumita（2015 年）
22 王，刘，孙，王，王(2015b)
23 Chrupala（2014 年）
24 Mrksic，O Seaghdha，Thomson，Gasic，Su，Vandyke，Wen 和 Young（2015）
25 Sordoni，Galley，Auli，Brockett，Ji，Mitchell，Nie，Gao 和 Dolan（2015）
26 Ling 等人。 (2015 年 b)
27 索彻，鲍尔，曼宁和吴（2013 年）
28 勒和祖德马（2014），朱，邱，陈，黄(2015a)
29 李，李和霍维（2014）
30 桥本，米瓦，Tsuruoka 和 Chikayama（2013 年)，刘，魏，李，吉，周，王(2015 年）
31 伊耶尔、恩斯、博伊德-格雷伯和雷斯尼克(2014 年 b)
32 Socher、Perelygin、Wu、Chu、Manning、Ng 和 Potts（2013 年）、Hermann 和 Blunsom（2013 年）
33 董，魏，谭，唐，周，徐（2014）
34 伊耶尔、博伊德-格雷伯、克劳迪诺、索彻和 DaumaeIII(2014a)

3. 特征表示

在更深入地讨论网络结构之前，必须注意特征是如何表示的。 目前，我们可以认为前馈神经网络是一个函数 N(X)，它以 DIN 维向量 x 作为输入，并产生一个 dout 维输出向量。 函数通常用作分类器，在一个或多个 dout 类中分配输入 x 的隶属度。 函数可以是复杂的，并且几乎总是非线性的。 这一职能的共同结构将在第 4 节中讨论。 在这里，我们关注输入，x。 在处理自然语言时，输入 x 编码特征，如单词、词性标记或其他语言信息。 当从稀疏输入线性模型向基于神经网络的模型移动时，也许最大的概念跳跃是停止将每个特征表示为唯一维数（所谓的一热表示），并将它们表示为密集向量。 即每个核心特征嵌入到 d 维空间中，并在该空间中表示为向量。[35] 然后可以像函数 NN 的其他参数一样训练嵌入(每个核心特征的向量表示。 图 1 显示了两种特征表示方法。

特征嵌入（每个特征的向量条目的值）被视为需要与网络的其他组件一起训练的模型参数。 训练（或获取）特征嵌入的方法将在稍后讨论。 现在，考虑给定的特征嵌入。

因此，基于前馈神经网络的 NLP 分类系统的一般结构是：

1. 提取一组核心语言特征 这与预测有关 输出类。
2. 对于感兴趣的每个特征 FI，检索相应的向量 v(FI)。
3. 将向量（通过级联、求和或两者的组合）组合成输入向量 x。
4. 将 x 送入非线性分类器（前馈神经网络）。

那么，输入的最大变化是从稀疏表示，其中每个特征是自己的维度，移动到一个密集的表示，其中每个特征被映射到一个向量。 另一个区别是，我们只提取核心特征，而不是特征组合。 我们将简要阐述这两个变化。

### 3.1 密集向量与单点表示

将我们的特征表示为向量而不是唯一的 ID 有什么好处？ 我们应该始终将特征表示为密集向量吗？ 让我们考虑两种表示：

35 不同的特征类型可以嵌入到不同的空间中。 例如，一个可以使用 100 维表示单词特征，而部分语音特征使用 20 维。

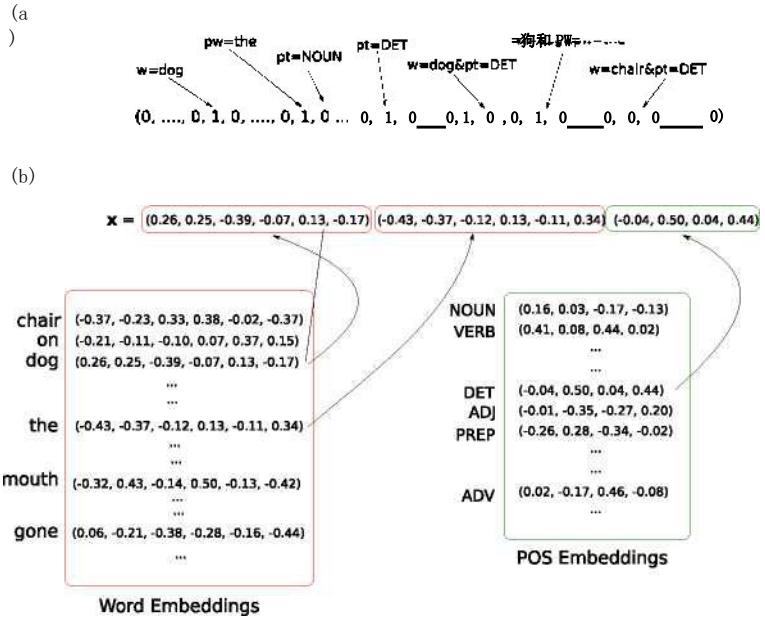



*图 1：稀疏与密集特征表示。信息的两个编码：当前单词是“狗”；以前的单词是“the”；以前的 pos 标签是“DET”。* (a) 稀疏特征向量。 每个维度代表一个特征。 特征组合接收自己的维度。 特征值是二进制的。 维度很高。 (b) 密集、基于嵌入的特征向量。 每个核心特征表示为向量。 每个特征对应于几个输入向量条目。 没有特征组合的显式编码。 维度较低。 特征到向量映射来自嵌入表。

每个功能都是它自己的维度。

- 单热向量的维数与特征个数相同。
- 特征完全相互独立。 “单词是‘狗’”的特征与“单词是‘思考’”的特征非常相似，而不是“单词是‘猫‘

密.每个特征都是 d 维向量。

- 向量的维数为 d。
- 模型训练将导致相似的特征具有相似的向量-信息在相似的特征之间共享。

使用密集和低维向量的一个好处是计算：大多数神经网络工具包不能很好地使用非常高维、稀疏的向量。 然而，这只是一个技术障碍，可以通过一些工程努力来解决。

密集表示的主要好处是泛化能力：如果我们认为某些特征可能提供类似的线索，则值得提供能够捕捉这些相似之处的表示。 例如，假设我们在训练过程中多次观察到“狗”这个词，但只观察了几次“猫”这个词，或者根本没有。 如果每个单词都与它自己的维度相关联，“狗”的出现就不会告诉我们任何关于“猫”发生的事情‘。 然而，在密集向量表示中，“狗”的学习向量可能类似于“猫”的学习向量，允许模型在两个事件之间共享统计强度。 这个论点假设“好”向量是以某种方式给我们的。 第 5 节描述了获取这种向量表示的方法。

如果我们在类别中有相对较少的不同特征，并且我们认为不同特征之间没有相关性，我们可以使用单热表示。然而，如果我们认为组中的不同特征之间存在相关性(例如，对于词性标记，我们可能认为不同的动词变化 VB 和 VBZ 在我们的任务中的行为可能类似)，那么让网络找出相关性并通过共享参数获得一些统计强度可能是值得的。 在某些情况下，当特征空间相对较小，训练数据丰富时，或者当我们不希望在不同的单词之间共享统计信息时，使用一热表示可以获得收益。 然而，这仍然是一个开放的研究问题，双方都没有强有力的证据。 大多数工作(首创于 Collobert&Weston，2008 年；Collobert 等人。 2011 年；Chen&Manning，2014 年)主张对所有特征使用密集、可训练的嵌入向量。 关于使用具有稀疏向量编码的神经网络体系结构的工作，请参阅 Johnson 和 Zhang（2015)的工作）。

最后，必须注意的是，将特征表示为密集向量是神经网络框架的组成部分，因此，使用稀疏和密集特征表示之间的差异比最初可能出现的要微妙。 事实上，在训练神经网络时，使用稀疏的、一热的向量作为输入，等于将网络的第一层用于根据训练数据为每个特征学习一个密集的嵌入向量。 我们在 4.6 节中讨论这个问题。

### 3.2 特征变量数：连续词袋

前馈网络假设一个固定的维数输入。 这可以很容易地适应提取固定数量特征的特征提取函数的情况：每个特征被表示为向量，并且向量被连接。 这样，得到的输入向量的每个区域对应一个不同的特征。 然而，在某些情况下，特征的数量事先不知道（例如，在文档分类中，句子中的每个单词都是特征）。 因此，我们需要使用固定大小向量来表示无界的特征数。 实现这一点的一种方法是通过所谓的连续词袋(CBOW)表示(Mikolov，Chen，Corrado，&Dean，2013 年)。 CBOW 与传统的词袋表示非常相似，在这种表示中，我们丢弃顺序信息，并通过对相应特征的嵌入向量进行求和或平均：[36]

$$CBOW(凡., 九) = Q\ v(FI) \quad (1)$$

“我=1

...。凡仇)。 *“ (2)

对 CBOW 表示的一个简单的变化是加权 CBOW，其中不同的向量接收不同的权重：

36 请注意，如果 v(FI)s 是一个热向量，而不是密集的特征表示，则 CBOW(Eq 1)和 WCBOW(Eq2)将减少到传统的（加权）字袋表示，这又相当于稀疏特征向量表示，其中每个二进制指示特征对应于唯一的“单词”。

在这里，每个特征 FI 都有一个相关的权重 a《》，表示特征的相对重要性。 例如，在文档分类任务中，特征 fi 可能对应于文档中的一个单词，相关的权重 a 可能是单词的 TF-IDF 评分。

### 3.3 距离和位置特征

[37]句子中两个单词之间的线性距离可以作为信息特征。 例如，在事件提取任务中，我们可能会得到一个触发词和一个候选参数词，并要求预测参数词是否确实是触发器的参数。 触发器与参数之间的距离（或相对位置）是此预测任务的强信号。 在“传统”NLP 设置中，距离通常是通过将距离绑定到几个组(即。 1，2，3，4，5-10，10+)，并将每个垃圾箱与一个热向量相关联。 在神经结构中，输入向量不是由二进制指示特征组成的，似乎很自然地将单个输入条目分配给距离特征，其中该条目的数值是距离。 然而，在实践中没有采取这种做法。 相反，距离特征的编码类似于

37 事件提取任务涉及从预定义的事件类型集合中识别事件。 例如，确定“购买”事件或“恐怖袭击”事件。 每个事件类型都可以由各种触发词（通常是动词）触发，并且有几个需要填充的插槽(参数。 谁买的？ 买了什么？ 多少钱？ )。

其他特征类型：每个 bin 与 d 维向量相关联，然后将这些距离嵌入向量训练为网络中的常规参数(Zeng 等人，2014 年；Dos Santos 等人，2015 年；朱等人，2015 年 a；Nguyen&Grishman，2015 年)。

### 3.4 特征组合

请注意，神经网络设置中的特征提取阶段只涉及核心特征的提取。 这与传统的基于线性模型的 NLP 系统形成鲜明对比，在这种系统中，特征设计人员不仅必须手动指定感兴趣的核心特征，而且还必须手动指定它们之间的交互(例如，不仅引入一个表示“Word 是 X”的特征和一个表示“标记是 Y”的特征，而且还引入一个表示“标记是 Y”的组合特征，表示“Word 是 X，标记是 Y”，有时甚至“Word 是 X，标记是 Y，以前的单词是 Z”)。 组合特征在线性模型中是至关重要的，因为它们为输入引入了更多的维数，将其转换为数据点更接近线性可分的空间。另一方面，可能的组合空间非常大，特征设计人员不得不花费大量的时间来想出一组有效的特征组合。 非线性神经网络模型的承诺之一是只需要定义核心特征。 由网络结构定义的分类器的非线性，预计将注意寻找指示性特征组合，减轻特征组合工程的需要。

核方法(Shawe-Taylor&Cristianini，2004)，特别是多项式核(Kudo&Matsumoto，2003)，也允许特征设计者只指定核心特征，将特征组合方面留给学习算法。 与神经网络模型相比，核方法是凸的，承认优化问题的精确解。 然而，核方法中分类的计算复杂度与训练数据的大小成线性关系，使得它们对于大多数实际目的来说太慢，不适合于大型数据集的训练。另一方面，使用神经网络分类的计算复杂度与网络的大小呈线性关系，而不考虑训练数据的大小。

### 3.5 维度

我们应该为每个特性分配多少维度？ 不幸的是，在这个空间中没有理论界，甚至没有既定的最佳实践。 显然，维度应该随着类中成员的数量而增长（您可能希望为单词嵌入分配更多的维度，而不是部分语音嵌入），但是多少就足够了？ 在目前的研究中，词嵌入向量的维数在 50 到几百之间，在某些极端情况下，在数千之间。 由于向量的维数对内存需求和处理时间有直接影响，一个好的经验法则是用几个不同的大小进行实验，并在速度和任务精度之间选择一个很好的权衡。

### 3.6 病媒分享

考虑一个例子，你有几个功能共享相同的词汇。 例如，当为给定的单词分配部分语音时，我们可能有一组考虑前一个单词的特征，以及一组考虑下一个单词的特征。 在构建分类器的输入时，我们将把前一个单词的向量表示连接到下一个单词的向量表示。 然后，分类器将能够区分这两个不同的指标，并对它们进行不同的处理。 但这两个特征应该共享相同的向量吗？ “狗：前一个词”的向量是否应该与“狗：下一个词”的向量相同”？ 或者我们应该给他们分配两个不同的向量？ 这又是一个经验问题。 如果你认为单词在不同的位置出现时表现不同(例如，单词 X 在前一个位置时表现为单词 Y，而 X 在下一个位置时表现为 Z)，那么使用两个不同的词汇表并为每个特征类型分配一组不同的向量可能是个好主意。 然而，如果您认为这些单词在两个位置的行为相似，那么可以通过对两种特征类型使用共享词汇表来获得一些东西。

### 3.7 网络的输出

对于 k 类的多类分类问题，网络的输出是一个 k 维向量，其中每个维度都表示特定输出类的强度。 也就是说，输出保持在传统的线性模型中-标量分数到离散集合中的项。 然而，正如我们将在第 4 节中看到的，有一个与输出层相关联的 dxk 矩阵。 这个矩阵的列可以被认为是输出类的 d 维嵌入。 k 类向量表示之间的向量相似性表示模型在输出类之间的学习相似性。

### 3.8 历史记录

将单词表示为输入到神经网络的密集向量是由 Bengio 等人推广的。 （2003 年）在神经语言建模方面。 它被介绍到 NLP 在 Collobert、Weston 和同事的开创性工作中的任务（2008 年、2011 年）。 使用嵌入来表示不仅是单词，而且是任意特征，这是继 Chen 和 Manning（2014)之后推广的）。[38]

## 4. 前馈神经网络

本节介绍前馈神经网络。 它开始于流行的大脑启发的隐喻，触发了它们，但很快又回到了使用数学符号。 讨论了前馈神经网络的结构、表示能力、常见的非线性和损耗函数。

### 4.1 大脑暗示隐喻

[39]顾名思义，神经网络受大脑的计算机制的启发，大脑由称为神经元的计算单元组成。 在隐喻中，神经元是一个具有标量输入和输出的计算单元。每个输入都有一个相关的权重。神经元将每个输入乘以其权重，然后求和，对结果应用非线性函数，并将其传递给输出。 神经元相互连接，形成网络：神经元的输出可能馈入一个或多个神经元的输入。 这些网络被证明是非常有能力的计算设备。 如果权值设置正确，具有足够神经元和非线性激活函数的神经网络可以近似于非常广泛的数学函数（稍后我们将对此更精确）。

38 虽然 Bengio、Collobert、Weston 和他的同事推广了这些方法，但它们并不是第一个使用它们的。 早期使用密集连续空间向量表示单词输入到神经网络的作者包括 Lee 等人。 （1992 年）和 Forcada 和 Neco（1997 年）。 同样，连续空间语言模型已经被 Schwenk 等人用于机器翻译。 （2006）。

39 虽然求和是最常见的操作，但其他函数，如 max，也是可能的

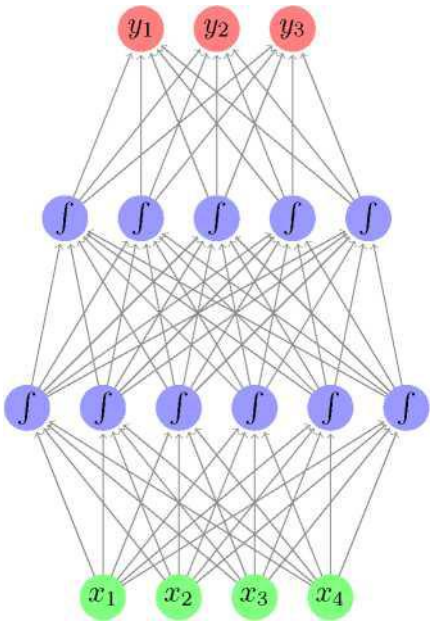



图 2：具有两个隐藏层的前馈神经网络。

一个典型的前馈神经网络可以绘制如图 2 所示。 每个圆都是一个神经元，传入的箭头是神经元的输入，传出的箭头是神经元的输出。 每个箭头都有一个重量，反映了它的重要性（未显示）。 神经元层层排列，反映信息的流动。 底层没有传入箭头，是网络的输入。 最顶层没有传出箭头，是网络的输出。 其他层被认为是“隐藏的”。 中间层神经元内的乙状结肠形状代表一个非线性函数(即 Logistic 函数 1/(1+e$^{-xa}$在将其传递给输出之前应用于神经元的值。在图中，每个神经元连接到下一层的所有神经元-这被称为完全连接层或仿射层。

虽然大脑隐喻是性感和有趣的，但它也是分散注意力和繁琐的数学操作。 因此，我们转向使用更简洁的数学表示法。 网络中每一行神经元的值可以看作是一个向量。 在图 2 中，输入层是一个 4 维向量(X)，它上面的层是一个 6 维向量(H$^{1}$)。 完全连接的层可以被认为是从 4 维到 6 维的线性变换。 一个完全连接的层实现了向量矩阵乘法，h=xW，其中从输入行中的 ith 神经元到输出行中的 JT H 神经元的连接的权重为 Wij。[40] 然后，h 的值由一个非线性函数 g 转换，该函数 g 应用于每个值，然后传递给下一个输入。 从输入到输出的整个计算可以写成：(g(xW$^{1}$)w$^{2}$ 在哪里 $^{1}$ 是第一层和 W 的权重 $^{2}$是第二个的重量。

### 4.2 数学符号

从这一点开始，我们将放弃大脑隐喻，只用向量矩阵运算来描述网络。
最简单的神经网络是感知器，它是其输入的线性函数：

神经感受器(X)=xW+b (3)

40 要了解为什么会出现这种情况，请将 h 中 JTH 神经元的 ith 输入的权重表示为 Wij。 然后 hj 的值是 hj=£4=1xi-wij。

x e R 丁丁 Xdout b e R 达特

式中，W 为权矩阵，b 为偏置项。[41]为了超越线性函数，我们引入了一个非线性隐藏层（图 2 中的网络有两个这样的层），从而产生了一个隐藏层(MLP1)的多层感知器)。 具有一个隐层的前馈神经网络的形式是：

$$NN_{MLPI}(X)=G(XW^{1}+b^{1})w^{2}+b^{2} \tag{4}$$

x G Rdin，W¹G Rdin Xdi，b¹ g，w²g r$^{d1Xd_2}$, $_{b}{}^{2}$g

在这里 $^{1}$ 和 $b^{1}$ 是输入的第一个线性变换的矩阵和偏置项，g 是一个非线性函数，它是应用于元素的（也称为非线性或激活函数）和 $W^{2}$ 和 $b^{2}$ 是第二个线性变换的矩阵和偏置项。

打破它，xW¹+b]是输入 x 从 DIN 维到 di 维的线性变换。 然后将 g 应用于每个 di 维和矩阵 $W^{2}$ 连同偏置向量 $b^{2}$ 然后用于将结果转换为 dˆ维数输出向量。 非线性激活函数 g 在网络表示复杂函数的能力中起着至关重要的作用。 没有 g 中的非线性，神经网络只能表示输入的线性变换。[42]

我们可以添加额外的线性变换和非线性，从而产生具有两个隐藏层的 MLP（图 2 中的网络是这种形式）：

$$=(G^{2}(g^{1}(\mathbf{x}\ \mathbf{W}^{1}+b^{1})w^{2}+b^{2})w^{3} \tag{5}$$

也许更清楚的是，使用中介变量来编写这样更深层次的网络：

41 图 2 中的网络不包括偏置项。 一个偏置项可以通过添加一个没有任何传入连接的附加神经元来添加到一个层中，其值总是 1。

42 要了解原因，请考虑线性变换序列仍然是线性变换。

$$\mathrm{NNMLP2}(X)=Y$$
$$\mathbf{h}^1=g^1(\mathbf{x}\ \mathbf{W}^1+b^1)\ h^2=g^2(h^1\mathbf{w}^2+b^2)\ y=h^2\mathbf{w}^3$$

每个线性变换产生的向量称为层。 最外层的线性变换导致输出层，其他线性变换导致隐藏层。每个隐藏层后面都有一个非线性激活。在某些情况下，例如在示例的最后一层中，偏置向量被迫为 0（“下降”）。

由线性变换产生的层通常被称为完全连接，或仿射。 存在其他类型的体系结构。 特别是，图像识别问题受益于卷积层和池层。 这些层在语言处理中也有用途，将在第 9 节中讨论。有几个隐藏层的网络被称为深度网络，因此被称为深度学习。

在描述神经网络时，应该指定层的尺寸和输入。 层将期望 DIN 维向量作为其输入，并将其转换为 DUT 维向量。 层的维数被认为是其输出的维数。 对于具有输入维数 din 和输出维数 dout 的完全连接层 l(X)=xW+b，x 的维数为 lxdin，W 的维数为 dinxdout，b 的维数为 $^{x\ d}_{1}$ 出去。

网络的输出是 dout 维向量。 在 dout=1 的情况下，网络的输出是标量。 这种网络可以通过考虑输出的值来进行回归（或评分），也可以通过咨询输出的符号来进行二进制分类。具有 dout=k>1 的网络可以用于 k 类分类，通过将每个维度与类关联，并寻找具有最大值的维度。 类似地，如果输出向量条目是正的，并且和为一个，则输出可以解释为类分配的分布(这种输出归一化通常是通过在输出层上应用 Softmax 变换来实现的，参见 4.5 节)。

定义线性变换的矩阵和偏置项是网络的参数。 通常将所有参数的集合称为 0。 参数与输入一起决定网络的输出。 训练算法负责设置它们的值，使网络的预测是正确的。 培训在第 6 节中讨论。

### 4.3 代表权力

在表示能力方面，Hornik、Stinchcombe 和 White（1989）和 Cybenko（1989）表明，MLP1 是一个通用逼近器-它可以用任何期望的非零误差量来逼近一个包含 R 的封闭和有界子集上的所有连续函数族[43]以及从任何有限维离散空间到另一个空间的任何函数映射。 这可能表明，没有理由超越 MLP1 到更复杂的体系结构。 然而，理论结果并没有讨论神经网络的可学习性（它说一个表示存在，但没有说基于训练数据和特定的学习算法来设置参数是多么容易或困难）。 它也不能保证训练算法会找到正确的函数来生成我们的训练数据。 最后，它没有说明隐藏层应该有多大。 事实上，Telgarsky（2016）表明，存在着许多有界大小层的神经网络，这些神经网络不能被层数较少的网络近似，除非它们的层是指数大的。

在实践中，我们使用随机梯度下降的变体等局部搜索方法，在相对少量的数据上训练神经网络，并使用相对适度大小的隐藏层（高达数千）。 由于普遍近似定理在这些非理想的、真实的条件下没有给出任何保证,因此在尝试比 MLP1 更复杂的体系结构方面肯定有好处。然而，在许多情况下，MLP1 确实提供了强有力的结果。 关于前馈神经网络的表示能力的进一步讨论，见 Bengio 等人的书。 （2015 年，第 6.5 款）。

43 具体来说，具有线性输出层和至少一个具有“压缩”激活函数的隐藏层的前馈网络可以近似于从一个有限维空间到另一个有限维空间的任何 Borel 可测量函数。

### 4.4 常见的非线性

非线性 g 可以采取多种形式。 目前还没有很好的理论来解释在哪些条件下应用哪些非线性，以及为给定的任务选择正确的非线性在很大程度上是一个经验问题。 我现在将从文献中回顾常见的非线性：乙状结肠、TANH、硬 TANH 和校正线性单元(ReLU)。 一些 NLP 研究人员还对其他形式的非线性进行了实验，如立方体和 tanh-cube。

#### 4.4.1 乙状结肠

乙状结肠激活函数 $a(X)=1/(1+e^{-x})$，又称逻辑函数，是一个 S 形函数，将每个值 x 转换为范围[0,1]。 乙状结肠是神经网络自成立以来的典型非线性，但目前被认为是不推荐用于神经网络的内部层，因为下面列出的选择证明在经验上工作得更好。

#### 4.4.2 双曲汤(TANH)

双曲正切 tanh(x)=氖$^{=}$激活函数是一个 S 形函数，将值 x 转换为范围[-1，1]。

### 4.4.3 很硬的

硬-tanh 激活函数是 tanh 函数的近似，计算和取导数更快：

$$\text{哈德坦(X)}=\begin{cases}-1 & x<-1\\ 1 & x>1\\ x & \text{否则}\end{cases}$$

### 4.4.4 整流器(RELU)

整流激活函数(Glorot，Bordes，&Bengio，2011)，也称为整流线性单元，是一种非常简单的激活函数，易于使用，并多次显示以产生优异的结果。[44]ReLU 单元在 0 时剪辑每个值 x<0。尽管它简单,但它在许多任务中表现良好,特别是当与辍学正则化技术相结合时（见 6.4 节）。

$$\begin{cases}0 & x<0\\ \text{否则} & \end{cases} \qquad (8)$$

.1 > \ /

根据经验，ReLU 单位比 tanh 工作得更好，tanh 工作得比乙状结肠好。[45]

## 4.5 输出变换

在许多情况下，输出层向量也被转换。 常见的变换是 softmax：

$$\text{Softmax(X I)}=\frac{e^{x_i}}{\sum_k e^{x_k}}$$

---

*44* 相对于 Sigmoid 和 tanh 激活函数,ReLU 的技术优势是它不涉及昂贵的计算函数,更重要的是它不饱和。乙状结肠和 tanh 激活被限制在 1,并且函数的这个区域的梯度接近于零,驱动整个梯度接近于零。ReLU 激活不存在这个问题，使得它特别适合于具有多层的网络，当使用饱和单元进行训练时，这些网络容易受到消失梯度问题的影响。

45 除了这些激活函数外，NLP 社区最近的工作与其他形式的非线性实验并报告成功。 立方体激活函数，$g(X)=(X)^3$陈和曼宁（2014）提出，他发现它比前馈网络中的其他非线性更有效，前馈网络用于预测基于贪婪过渡的依赖解析器中的操作。 tanh 立方体激活函数 $g(x)=\tanh(x)^3+x)$是由 Pei 等人提出的。（2015 年），他发现它比前馈网络中的其他非线性更有效，该网络被用作基于结构化预测图的依赖解析器的组件。

立方体和 tanh-cube 激活函数的动机是希望更好地捕捉不同特征之间的交互。 虽然这些激活函数被报道在某些情况下可以提高性能，但它们的普遍适用性仍有待确定。

结果是一个非负实数的向量，求和为一个，使它在 k 个可能的结果上是一个离散的概率分布。

当我们有兴趣在可能的输出类上建模概率分布时，使用 Softmax 输出转换。 为了有效，它应该与概率训练目标（如交叉熵)一起使用(见下文 4.7.4 节）。

当将 softmax 变换应用于没有隐藏层的网络的输出时，其结果是众所周知的多项式 Logistic 回归模型，也称为最大熵分类器。

### 4.6 嵌入层

到目前为止，讨论忽略了 x 的来源，将其视为任意向量。 在 NLP 应用程序中，x 通常由各种嵌入向量组成。 我们可以明确说明 x 的来源，并将其包含在网络的定义中。 我们介绍了 c（-），一个从核心特征到输入向量的函数。

通常 c 提取与每个特征相关联的嵌入向量，并将它们连接起来：

$$\begin{aligned} x &= c(f_1, f_2, f_3) = [五(f_1); 五(f_2); 五(f_3)] \\ NN_{MLPI}(X) &= NN_{MLPI}(C(/I, F\text{‘}2, F\text{’}3)) \\ &= NNMLP1([v(F1); v(F2); v(F3)]) = (g([v(FI); v(F_2); \\ &\quad v(f3)]w1+b1)w^2 + b^2 \end{aligned} \tag{10}$$

另一个常见的选择是 c 和嵌入向量（这假设嵌入向量都具有相同的维数）：

$$\begin{aligned} x &= c(f_1, f_2, f_3) = 五(f_1) + 五(f_2) + 五(f_3) \\ NN_{MLPI}(X) &= NN_{MLP1}(C(/1, F\text{‘}2, F\text{’}3)) \\ &= NNMLP1(v(/1) + v(f2) + v(f3)) \\ &= (g((v(f1) + v(f2) + v(f3))W1+b1)W^2 + b^2 \end{aligned} \tag{11}$$

c 的形式是网络设计的重要组成部分。 在许多论文中，将 c 称为网络的一部分是很常见的，同样地，将单词嵌入 v(FI)视为由“嵌入层”或“查找层”产生的”。 考虑\V|单词的词汇表，每个单词嵌入为 d 维向量。 然后，向量的集合可以被认为是|V|xd 嵌入矩阵 E，其中每行对应于嵌入特征。 设 f i 为|V|维向量，它是除一个索引外的所有零，对应于第 i 个特征的值，其中值为 1（这称为一个热向量）。 然后乘法 FIE 将“选择”相应的 E 行。因此，v(FI)可以用 E 和 f 定义 $_i$:

$$第五(FI) = FIE \tag{12}$$

同样：

$$E(FI\ E) \qquad \sum_{我=1}^{k} (fi)E \qquad (13)$$

然后，网络的输入被认为是一个单热向量的集合。 虽然这是优雅和良好的数学定义，一个有效的实现通常涉及一个基于哈希的数据结构映射特征到他们相应的嵌入向量，而不经过一个热表示。

在本教程中，我们将 c 与网络体系结构分开：网络的输入总是密集的实值输入向量，在输入通过网络之前应用 c，类似于熟悉的线性模型术语中的“特征函数。 然而，当训练网络时，输入向量 x 确实记得它是如何构造的，并且可以酌情将误差梯度传播回其分量嵌入向量（误差传播在第 6 节中讨论）。

### 4.6.1 注记

当描述将级联向量 x、y 和 z 作为输入的网络层时，一些作者使用显式级联([x；y；z]W+b)，而另一些作者使用仿射变换(xU+yV+zW，b)。 如果仿射变换中的权重矩阵 U、V、W 彼此不同，则这两个符号是等价的。

### 4.6.2 关于稀疏与稀疏的注释。 密集的特征

考虑一个网络，它使用“传统”稀疏表示为其输入向量，而没有嵌入层。 假设所有可用特性的集合是 V，并且我们有 k 个“on”特性 FI，...，fk，fi∈V，网络的输入是：

$$x=\sum_{我=1}^{k} £fi \qquad 十∈邱^{1} \qquad (14)$$

因此，第一层（忽略非线性激活）是：

$$xW+b=\sum_{我=1}^{k} (£fi)W \qquad (15)$$

$$w∈r^{|诉}伙^{d}b∈$$

该层选择与 x 中的输入特征相对应的 W 行并对它们进行求和，然后添加一个偏置项。 这与在特征上产生 CBOW 表示的嵌入层非常相似，其中矩阵 W 充当嵌入矩阵。 主要的区别是引入了偏置向量 b，并且嵌入层通常不经历非线性激活，而是直接传递到第一层。另一个区别是，这种场景迫使每个特征接收一个单独的向量(W 中行)，而嵌入层提供了更多的灵活性，例如，允许“下一个单词是狗”和“前一个单词是狗”共享相同的向量。

然而，这些差异是微小而微妙的。 当涉及到多层前馈网络时，密集和稀疏输入之间的差异比乍看起来要小。

## 4.7 丧失功能

当训练神经网络时（更多关于下面第 6 节的训练），就像训练线性分类器时，定义了一个损

失函数 L(y，y)，说明当真实输出为 y 时预测 y 的损失。 然后，培训目标是最小化不同培训示例的损失。 损失 L(y，y)为网络的输出 y 分配一个数值分数（标量），给定真实的期望输出 y。[46]损失函数应该从下面有界，只有在网络输出正确的情况下才能达到最小值。

网络的参数(矩阵 W[1]，偏见 b[1] 通常，嵌入 E)被设置为最小化训练示例上的损失 L（通常，它是被最小化的不同训练示例上的损失之和）。

损失可以是将两个向量映射到标量的任意函数。 为了优化的实际目的，我们将自己限制在我们可以轻松计算梯度（或子梯度)的函数上）。 在大多数情况下，依靠共同的损失函数而不是定义自己的损失函数是足够和可取的。 关于神经网络损失函数的详细讨论，请参阅 LeCun、Chopra、Hadsell、Ranzato 和 Huang（2006 年）、LeCun 和 Huang（2005 年）和 Bengio 等人的工作。 （2015）。 我们现在讨论了 NLP 神经网络中常用的一些损失函数。

### 4.7.1 铰链（二进制）

对于二进制分类问题，网络的输出是单个标量 y，预期的输出 y 在{+1，-1}中}。 分类规则是符号(y')，如果 y-y>0，则分类被认为是正确的，这意味着 y 和 y 共享相同的符号。 铰链损失又称边缘损失或 SVM 损失，定义为：

$$L_{铰链（二进制）}(y, y) = 最大值(0, 1-y \cdot y) \tag{16}$$

当 y 和 y 共享相同的符号时，损失为 0|>1。 否则，损失是线性的。 换句话说，二进制铰链损失试图实现正确的分类，边缘至少为 1。

### 4.7.2 铰链（多类）

铰链损耗由 Crammer 和 Singer（2002)扩展到多类设置）。 让你=易，...，你 $_n$是网络的输出向量，y 是正确输出类的一个热向量。

分类规则定义为选择得分最高的类：

$$预测=\underset{i}{Argmax} yi \tag{17}$$

用 t=argmax 表示正确的类，用 k=argmax 表示最高的评分类，使得 k=t。多类铰链损失定义为：

$$L_{铰链（多类）}(y, y) = 马克斯(0, 1 - y不 \cdot yk)) \tag{18}$$

多类铰链损耗试图以至少 1 的裕度将正确的类得分高于所有其他类。

二进制和多类铰链损耗都打算与线性输出层一起使用。 每当我们需要一个硬决策规则时，铰链损失是有用的，并且不要试图建模类隶属概率。

### 4.7.3 日志丢失

对数损失是铰链损失的一个常见变化，可以看作是铰链损失的一个“软”版本，具有无限的裕度(LeCun 等人，2006 年)。

---

46 在我们的表示法中，模型的输出和预期输出都是向量，而在许多情况下，更自然地将预期输出看作标量（类赋值）。 在这种情况下，y 只是相应的一个热向量。

$$=日志(1+exp(-(yt-y_{FC})) \tag{19}$$

### 4.7.4 范畴交叉熵损失

当需要对分数进行概率解释时，使用分类交叉熵损失(也称为负对数似然。

*让你=易，...，你*$_n$是一个向量，表示标签 1，...，n 上的真实多项式分布，让 y=yi，...，$y_n$ 是网络的输出，由 Softmax 激活函数转换，表示类成员条件分布 YI=P(y=I|x)。 分类交叉熵损失度量真实标签分布 y 和预测标签分布 y 之间的差异，定义为交叉熵：

$$^{1}交叉熵^{(y,\ y)}=\ \overline{\quad}\ £易原木（易） \tag{20}$$

对于每个训练示例都有一个正确的类分配的硬分类问题，y 是一个表示真实类的单热向量。 在这种情况下，交叉熵可以简化为：

$$^{1}交叉熵（硬分类）^{(y,\ y)}\ =\ ^{日志(y}t) \tag{21}$$

其中 t 是正确的课堂作业。 这试图将分配给正确类 t 的概率质量设置为 1。 由于分数 y 已经使用 Softmax 函数进行了转换，并表示了一个条件分布，因此增加分配给正确类的质量意味着减少分配给所有其他类的质量。

交叉熵损失在神经网络文献中是非常常见的，它产生了一个多类分类器，它不仅预测了一个最好的类标签，而且还预测了可能的标签上的分布。 当使用交叉熵损失时，假设网络的输出是使用 Softmax 变换进行的。

### 4.7.5 排名损失

在某些设置中，我们没有得到标签方面的监督，而是作为对正确和不正确的项目 x 和 x ‘，我们的目标是在不正确的项目之上得分正确的项目。 当我们只有正面的例子时，就会出现这种训练情况，并通过腐蚀一个正面的例子来产生负面的例子。 在这种情况下，一个有用的损失是基于保证金的排名损失，定义为一对正确和不正确的例子：

$$1_{r\ 边缘}(x，X)=最大值(0，1-(N\ N(X)-NN(X’)) \tag{22}$$

其中 NN(X)是网络为输入向量 x 分配的分数。 目标是对不正确的输入进行评分（排名），其差值至少为 1。

一个常见的变化是使用日志版本的排名损失：

$$排序(Iog)(x，x\ ‘)=日志(1+exp(-(N\ N(x)-NN(x\ ‘))) \tag{23}$$

在语言任务中使用排序铰链损失的例子包括用于导出预先训练的单词嵌入的辅助任务的训练（见第 5 节），其中我们得到了一个正确的单词序列和一个损坏的单词序列，我们的目标是在损坏的序列之上得分正确的序列(Collobert&Weston，2008)。 同样，Van de Cruys（2014 年）在选择偏好任务中使用了排序损失，其中网络被训练为将正确的动宾对排序高

于不正确的、自动派生的对，Weston、Bordes、Yakhnenko 和 Usunier（2013 年)训练了一个模型，在信息提取设置中，将正确的(头部、关系、跟踪）三胞胎评分高于损坏的三胞胎。在高等人的工作中可以找到使用排序日志丢失的一个例子。 （2014）。 在 DosSantos 等人的工作中，给出了排序日志损失的变化，允许负类和正类有不同的裕度。 （2015）。

5. 字嵌入

神经网络方法的一个主要组成部分是使用嵌入-将每个特征表示为低维空间中的向量。 但是向量来自哪里？ 本节将调查共同的方法。

### 5.1 随机初始化

当有足够的监督训练数据可用时，可以将特征嵌入与其他模型参数相同：将嵌入向量初始化为随机值，并让网络训练过程将它们调整为“好”向量。

在执行随机初始化的过程中必须注意一些问题。 有效的 Word2vec 实现使用的方法(Mikolov 等人，2013 年；Mikolov、Sutskever、Chen、Corrado 和 Dean，2013 年)是将单词向量初始化为在范围[-&，&]中均匀采样的随机数，其中 d 是维数。 另一种选择是使用 Xavier 初始化（见 6.3.1 节），并使用统一采样值初始化[从]-将的尚。

在实践中，人们通常使用随机初始化方法来初始化常见特征的嵌入向量，例如词性标记或单个字母，同时使用某种形式的监督或无监督的预训练来初始化潜在的罕见特征，例如单个单词的特征。 然后，预先训练的向量可以在网络训练过程中被视为固定的，或者更常见的是，被视为随机初始化的向量，并进一步调整到手头的任务。

### 5.2 监督特定任务的预训练

如果我们对任务 A 感兴趣，对于它，我们只有有限数量的标记数据（例如，句法解析），但有一个辅助任务 B（例如，词性标注），我们有更多的标记数据，我们可能希望预先训练我们的单词向量，以便它们作为任务 B 的预测器，然后使用训练的向量来训练任务 A。 这样，我们就可以利用任务 B 中的大量标记数据。 当训练任务 A 时，我们可以将预先训练的向量视为固定的，也可以为任务 A 进一步调整它们。 另一种选择是为两个目标联合培训，详见第 7 节。

### 5.3 无监督的预训练

常见的情况是，我们没有一个具有足够多注释数据的辅助任务（或者我们可能希望帮助引导具有更好的向量的辅助任务训练）。 在这种情况下，我们采用“无监督”的方法，可以在大量未注释的文本上进行训练。

训练单词向量的技术本质上是监督学习的技术，但我们不是对我们关心的任务进行监督，而是从原始文本中创建几乎无限数量的监督训练实例，希望我们创建的任务与我们关心

的最终任务相匹配(或足够接近。[47]

无监督方法背后的关键思想是，人们希望“相似”字的嵌入向量具有相似的向量。 虽然单词相似性很难定义，而且通常非常依赖于任务，但目前的方法来源于分布假设(Harris，1954)，指出如果单词出现在相似的上下文中，它们是相似的。 不同的方法都创建了有监督的训练实例，其中的目标是从上下文中预测单词，或者从单词中预测上下文。

在大量未注释数据上训练词嵌入的一个重要好处是它为不出现在监督训练集中的单词提供向量表示。 理想情况下，这些单词的表示将类似于训练集中出现的相关单词，从而使模型能够更好地概括看不见的事件。 因此，期望由无监督算法学习的单词向量之间的相似性捕捉到相同的相似性方面，这些相似性对于执行网络的预期任务是有用的。

[48]常见的无监督词嵌入算法包括 Word2vec(Mikolov 等人，2013 年，2013 年)、GloVe(Pennington，Socher，&Manning，2014 年)和 Collobert 和 Weston（2008 年，2011 年）嵌入算法。 这些模型受神经网络的启发，基于随机梯度训练。 然而，它们与 NLP 和 IR 社区进化的另一系列算法有着深刻的联系，这些算法是基于矩阵因式分解的(讨论见 Levy&Goldberg，2014b；Levy 等人，2015 年)。

可以说，辅助问题的选择（基于什么样的上下文正在预测什么）对结果向量的影响远远大于用于训练它们的学习方法。 因此，我们关注可用的辅助问题的不同选择，只浏览培训方法的细节。 有几个用于导出单词向量的软件包，包括 Word2vec 和 Gensim 实现基于 Word 窗口上下文的 Word2vec 模型，Word2vecf 是 Word2vec 的修改版本，允许使用任意上下文，GloVe 实现 GloVe 模型。 许多预先训练过的词向量也可在网上下载。[49][50][51][52]

虽然超出了本教程的范围，但值得注意的是，由无监督训练算法导出的单词嵌入在 NLP 中有着广泛的应用，而不仅仅是使用它们初始化神经网络模型的单词嵌入层。

### 5.4 培训目标

给定一个词 w 及其上下文 c，不同的算法制定不同的辅助任务。 在所有情况下，每个单词都表示为一个 d 维向量，它被初始化为一个随机值。 训练模型很好地执行辅助任务将导致良好的单词嵌入，将单词与上下文联系起来，这反过来又将导致类似单词的嵌入向量相互相似。

语言建模启发的方法，如 Mikolov 等人采取的方法。（2013 年）、Mnih 和 Kavukcuoglu（2013 年）以及 Glo Ve(Pennington 等人，2014 年)使用辅助任务，目标是根据上下文预测单词。 这是在概率设置中提出的，试图对条件概率 P(w|c)进行建模)。

其他方法将问题减少到二进制分类的问题。除了观察到的单词-上下文对的集合 D 之外，还从随机单词和上下文对创建集合 D。 二元分类问题是：给定的(w，c)对是否来自 D？ 这些方法不同于集合 D 是如何构造的，分类器的结构是什么，目标是什么被优化。 Collobert

47 从原始文本中创造辅助问题的解释受到 Ando 和 Zhang(2005a)和 Ando 和 Zhang(2005b)的启发)。

48 虽然通常被视为一个单一的算法，但 Word2vec 实际上是一个软件包，包括各种训练目标、优化方法和其他超参数。 请参阅荣（2014）和 Levy、Goldberg 和 Dagan（2015）的工作进行讨论。

49 https://code.google.eom/p/word2vec/

50 https://radimrehurek.com/gensim/

51 https://bitbucket.org/yoavgo/word2vecf

52 http://nlp.stanford.edu/projects/glove/

和 Weston（2008，2011）采用基于边际的二进制排序方法，训练前馈神经网络对不正确的(w，c)对进行评分。 Mikolov 等人。 （2013 年，2014 年）采用概率版本，训练对数双线性模型来预测这对来自语料库而不是随机样本的概率 P((w，c)、eD|w，c。

### 5.5 语境的选择

在大多数情况下，一个词的上下文被认为是出现在它周围的其他词，要么出现在它周围的一个短窗口中，要么出现在同一个句子、段落或文档中。 在某些情况下，文本由句法分析器自动解析，上下文由自动解析树诱导的句法邻域导出。 有时，单词和上下文的定义也会改变，包括单词的一部分，如前缀或后缀。

神经词嵌入起源于语言建模的世界，其中一个网络被训练以根据前面单词的序列来预测下一个单词(Bengio 等人，2003 年)。 在那里，文本用于创建辅助任务，其目的是根据上下文预测一个单词，即 k 个以前的单词。 虽然语言建模辅助预测问题的训练确实产生了有用的嵌入，但这种方法不必要地受到语言建模任务的约束的限制，其中允许只看前面的单词。如果我们不关心语言建模，而只关心由此产生的嵌入，我们可以通过忽略这个约束并将上下文作为焦点字周围的对称窗口来做得更好。

#### 5.5.1 窗户通道

最常见的方法是滑动窗口方法，其中辅助任务是通过查看 2k+1 个单词的序列来创建的。 中间的单词被称为焦点词，每边的 k 个单词都是上下文。 然后，创建一个任务，其中目标是基于所有上下文词预测焦点词(使用 CBOW 表示，参见 Mikolov 等人，2013 年或向量连接，参见 Collobert&Weston，2008 年)，或者创建 2k 个不同的任务，每个任务将焦点词与不同的上下文词配对。 由 Mikolov 等人推广的 2k 任务方法。 （2013 年）被称为跳图模式。 基于 Skipgram 的方法被证明是稳健和有效的培训(Mikolov 等人，2013 年；Pennington 等人，2014 年)，并经常产生最先进的结果。

窗口大小的影响滑动窗口的大小对得到的向量相似性有很强的影响。 较大的窗口往往会产生更多的主题相似性(即。“狗”、“树皮”和“皮带”将组合在一起，以及“走”、“跑”和“走^)，而较小的窗口往往产生更多的功能和句法相似性(即。“Poodle”、“Pitbull”、“Rottweiler”或“步行”、“跑步”、“接近”)。

当使用 CBOW 或跳图上下文表示时，窗口中的所有不同上下文词都被平等对待。 在接近焦点词的上下文词和离焦点词更远的上下文词之间没有区别，同样，在焦点词之前出现的上下文词和在焦点词之后出现的上下文词之间也没有区别。 通过使用位置上下文可以很容易地将这些信息考虑在内：为每个上下文词指示其相对于焦点词的相对位置(即。而不是上下文词是“the”，它变成了“the：+2”，表示该词出现在焦点词右侧的两个位置)。 位置上下文和较小的窗口的使用往往会产生更多句法上的相似之处，有一种强烈的倾向，即将共享词性的单词组合在一起，以及在语义上功能上相似。 位置向量由 Ling、Dyer、Black 和 Trancoso(2015a)显示，当用于初始化用于词性标记和句法依赖分析的网络时，位置向量比基于窗口的向量更有效。

窗口方法上的许多变体是可能的。 人们可以在学习之前对单词进行分层，应用文本规范化，过滤太短或太长的句子，或者删除大写(例如，参见 DosSantos&Gatti，2014 年描述的预处

理步骤)。 一个人可以对语料库的部分进行子样本，以某种概率跳过从窗口创建任务，这些窗口有太常见或太罕见的焦点词。 窗口大小可以是动态的，在每个回合使用不同的窗口大小。 人们可能会对窗口中的不同位置进行不同的权衡，更多地关注于试图预测正确地关闭单词-上下文对，而不是更远的。 这些选择中的每一个都将影响结果向量。 其中一些超参数（和其他）由 Levy 等人讨论。 （2015）。

### 5.5.2 句子、段落或文件

使用跳过图(或 CBOW)方法，人们可以认为一个词的上下文是在同一个句子、段落或文档中出现的所有其他单词。 这相当于使用非常大的窗口大小，并有望导致捕获局部相似性的单词向量(来自同一主题的单词，即。 一个人期望出现在同一文档中的单词，很可能会收到类似的向量)。

### 5.5.3 句法窗口

一些工作将句子中的线性上下文替换为句法上下文(Levy&Goldberg，2014 年 a；Bansal，Gimpel，&Livescu，2014 年)。 使用依赖解析器自动解析文本，并将单词的上下文视为在解析树中接近的单词，以及它们连接的句法关系。 这种方法产生了高度的功能相似性，将单词组合在一起，而不是在句子中填充相同的作用(例如。颜色，学校名称，运动动词)。分组也是句法，将共享拐点的单词分组在一起(Levy&Goldberg，2014a)。

### 5.5.4 多语种

另一种选择是使用基于多语种翻译的上下文(Hermann&Blunsom，2014 年；Faruqui&Dyer，2014 年)。例如，给定大量的句子对齐并行文本，可以运行双语对齐模型，如 IBM 模型 1 或模型 2(即。 使用 GIZA++软件)，然后使用生成的对齐来导出单词上下文。 在这里，单词实例的上下文是与它对齐的外语单词。 这种对齐往往导致同义词词接收相似的向量。 一些作者转而在句子对齐水平上工作，而不依赖于单词对齐(Gouws，Bengio，&Corrado，2015)或训练端到端的机器翻译神经网络，并使用由此产生的单词嵌入(Hill，Cho，Jean，Devin，&Bengio，2014)。 一种吸引人的方法是将基于单语文窗口的方法与多语文方法混合，创建两种辅助任务。 这可能会产生类似于基于窗口的方法的向量，但会减少窗口的一些不希望的效果-

基于反义词的方法(例如。冷热、高和低)倾向于接收类似的向量(Faruqui&Dyer，2014 年)。

### 5.5.5 基于字符和子词的表示

一系列有趣的工作试图从组成单词的字符中导出单词的向量表示。 这种方法可能对具有句法性质的任务特别有用，因为单词中的字符模式与其句法功能密切相关。 这些方法还具有产生非常小的模型大小的好处（只需要为字母表中的每个字符存储一个向量，以及需要存储少量的小矩阵），并且能够为可能遇到的每个单词提供嵌入向量。多斯桑托斯和加蒂（2014 年)、多斯桑托斯和扎德罗兹尼(2014 年）和 Kim 等人。（2015 年)使用卷积网络(见第 9 节）对字符的嵌入进行建模。 Ling 等人。 (2015b)使用两个 RNN(LSTM)编码器的最终状态的级

联（第 10 节）对单词的嵌入进行建模，一个从左到右读取字符，另一个从右到左。 两者都产生了非常强的结果，部分语音标记。 巴列斯特罗斯等人的工作。 （2015 年）表明，Ling 等人的双 LSTM 编码。 (2015b)也有利于在形态丰富的语言的依赖分析中表示单词。

从字符的表示中导出单词的表示是由未知单词问题驱动的-当你遇到一个没有嵌入向量的单词时，你会怎么做？ 在字符水平上的工作在很大程度上缓解了这个问题，因为可能的字符的词汇量远远小于可能的单词的词汇量。 然而，在字符级别上工作是非常具有挑战性的，因为语言中的形式（字符)和功能(语法、语义）之间的关系是相当松散的。 限制自己停留在字符级别上可能是一个不必要的硬约束。 一些研究人员提出了一个中间地带，其中一个词被表示为词本身的向量与构成它的子词单位的向量的组合。 子词嵌入随后有助于在具有相似形式的不同单词之间共享信息，以及在单词未被观察到时允许回退到子词级别。同时，当对单词有足够的观察时，模型不会被迫仅仅依赖于形式。 Botha 和 Blunsom（2014 年）建议将一个词的嵌入向量建模为一个词特定向量的总和，如果有这种向量，则使用包含它的不同形态分量的向量(这些分量是使用 Morfessor(Creutz&Lagus，2007 年)导出的，这是一种无监督的形态分割方法)。 Gao 等人。 （2014 年）建议不仅使用单词形式本身，而且为单词中的每个字母图表使用一个独特的特征(因此是一个独特的嵌入向量。

## 6. 神经网络培训

神经网络训练是通过使用基于梯度的方法试图最小化训练集上的损失函数来完成的。 粗略地说，所有的训练方法都是通过反复计算数据集上的误差估计，计算相对于误差的梯度，然后将参数移动到梯度的相反方向来工作的。 模型不同于如何计算误差估计，以及如何定义“向梯度的相反方向移动。 我们描述了基本的算法，随机梯度下降(SGD)，然后简要地提到了其他的方法与指针，以供进一步阅读。 梯度计算是该方法的核心。 梯度可以有效和自动地计算使用反向模式微分的计算图-一个一般的算法框架，自动计算任何网络的梯度和损失函数，将在 6.2 节中讨论。

### 6.1 随机梯度训练

训练神经网络的常用方法是使用随机梯度下降(SGD)算法(Bottou，2012 年；LeCun，Bottou，Orr，&Muller，1998a)或它的变体。 SGD 是一种通用的优化算法。 它接收由 0 参数化的函数 f、损失函数以及所需的输入和输出对。 然后，它试图设置参数 0，使 f 相对于训练示例的损失很小。 算法工作如下：

算法 1 在线随机梯度下降训练

1：输入：函数 f(x；0)参数为 0。

2：输入：输入 xi，...，x 的训练集。以及期望的输出。.

3：输入：损失函数 L。

4：虽然停止标准不符合做

5：示例一个训练示例 x；，y；

6：计算损失 L(f(x；0)，y；)

7：-L(f(x；0)，y；)w.r.t0 的梯度

8： 0-0-ntg

**9：返回 0**

该算法的目标是设置参数 0，使总损失最小化

在训练集上的 L(f(x；0)，y。 它通过重复采样训练示例和计算示例上的误差相对于参数 0（第 7 行）的梯度来工作-假定输入和期望输出是固定的，损失被视为参数 0 的函数。然后在梯度的相反方向上更新参数 0，按学习速率 NT（第 8 行)缩放）。 学习速率可以在整个训练过程中固定，也可以作为时间步长 t 的函数衰减。[53]关于设置学习率的进一步讨论，见第 6.3 节。

请注意，第 6 行中计算的错误是基于一个单一的训练示例，因此只是我们旨在最小化的语料库范围损失的粗略估计。 损失计算中的噪声可能导致梯度不准确。 减少这种噪声的一种常见方法是基于 m 个例子的样本来估计误差和梯度。 这就产生了小舱口 SGD 算法：

在第 6-9 行中，该算法基于最小块估计语料库损失的梯度。 在循环之后，g 包含梯度估计，参数 0 被更新为 g。小舱口的大小可以从 m=1 到 m=n。 较高的值提供了更好的语料库范围梯度估计，而较小的值允许

算法 2 小舱口随机梯度下降训练

1：输入：函数 f(x；ff)参数为 0。
2：输入：输入 xi，...，x 的训练集。以及期望的输出。.
3：输入：损失函数 L。
4：虽然停止标准不符合做
5：示例 m 个示例{(xi，yi)、...、(x。，是的。)}
6：g-0
7：我=1 到我做
8： 计算损失 L(f(x；ff)，易)
9： g-g+土 L 梯度(f(x；ff)，y；)w.r.t ff
io：ff-ff-ntg
11：返回 ff 更多的更新，进而更快地收敛。 除了提高梯度估计的精度外，最小匹配算法还为提高训练效率提供了机会。 对于 m 的适度大小，一些计算体系结构(即。 GPU)允许在第 6-9 行中有效地并行实现计算。 在适当降低学习速率的情况下，如果函数是凸的，SGD 保证收敛到全局最优。 然而，它也可以用来优化非凸函数，如神经网络。 虽然不再保证寻找全局最优，但该算法证明是稳健的，并在实践中表现良好。[54]

训练神经网络时，参数化函数 f 是神经网络，参数 ff 是线性变换矩阵、偏置项、嵌入矩阵等。 梯度计算是 SGD 算法以及所有其他神经网络训练算法中的关键步骤。 问题是，如何计算网络误差相对于参数的梯度。 幸运的是，有一个简单的解决方案，形式是反向传播算法(Rumelhart，Hinton，&Williams，1986；LeCun，Bottou，Bengio，&Haffner，1998b)。反向传播算法是一个花哨的名称，用于使用链式规则有条不紊地计算复杂表达式的导数，同时缓存中介结果。 更广泛地说，反向传播算法是反向模式自动微分算法的特例(Neidinger，2010 年，第 7 节；Baydin，Pearlmutter，Radul，&Siskind，2015 年；Bengio，2012 年)。

53 为了证明 SGD 的收敛性，需要学习速率衰减。

54最近神经网络文献的工作表明，网络的非凸性表现为鞍点的扩散，而不是局部极小值(Dauphin，Pascanu，Gulcehre，Cho，Ganguli，&Bengio，2014 年)。 这可能解释了尽管使用局部搜索技术，但训练神经网络的一些成功。

下面的部分描述了计算图抽象上下文中的反向模式自动区分。

### 6.1.1 超越 SGD

虽然 SGD 算法可以而且经常产生良好的结果，但更先进的算法也是可用的。SGD+Momentum(Polyak，1964 年)和 Nesterov Momentum(Sutskever，Martens，Dahl，&Hinton，2013 年；Nesterov，1983，2004 年)算法是 SGD 的变体，其中以前的梯度被累积并影响当前的更新。 自适应学习速率算法，包括 AdaGrad(Duchi，Hazan，&Singer，2011 年)、AdaDelta(Zeiler，2012 年)、RMSProp(Tieleman&Hinton，2012 年)和 Adam(Kingma&Ba，2014 年)，旨在选择每个小型比赛的学习速率，有时是在一个协调的基础上进行的，这可能会减轻对学习速率调度的需求。有关这些算法的详细信息，请参阅 Bengio 等人的原始论文或书籍。 （2015 年，第 8.3、8.4 节）。 由于许多神经网络软件框架提供了这些算法的实现，因此尝试不同的变体是容易的，有时也是值得的。

## 6.2 计算图形抽象

虽然可以手工计算网络各种参数的梯度，并在代码中实现它们，但这一过程繁琐且容易出错。就大多数目的而言，最好使用自动工具进行梯度计算(Bengio，2012 年)。 计算-图抽象允许我们轻松地构造任意网络，评估它们对给定输入的预测（前向传递），并计算它们的参数相对于任意标量损失(后向传递)的梯度）。

计算图是任意数学计算的表示形式。 它是一个有向无环图(DAG)，其中节点对应数学运算或（绑定）变量和边对应于节点之间的中介值的流动。 图结构根据不同组件之间的依赖关系来定义计算的顺序。图是 DAG 而不是树，因为一个操作的结果可以是几个连续的输入。例如，考虑用于计算(a*b+1)*(a*b+2)的图形)：

*

1a b2

共享 a*b 的计算。 我们只限于计算图连接的情况。

由于神经网络本质上是一个数学表达式，所以它可以表示为一个计算图。

例如，图 3a 显示了具有一个隐藏层和一个 Softmax 输出转换的 MLP 的计算图。 在我们的表示法中，椭圆节点表示数学运算或函数，阴影矩形节点表示参数（绑定变量）。 网络输入被视为常量，并在没有周围节点的情况下绘制。 输入和参数节点没有传入弧，输出节点没有传出弧。 每个节点的输出是一个矩阵，其维数在节点上方表示。

这个图是不完整的：如果不指定输入，我们就无法计算输出。 图 3b 显示了一个 MLP 的完整图，它以三个单词作为输入，并预测了第三个单词的部分语音标签上的分布。 这个图可以用于预测，但不能用于训练，因为输出是向量（不是标量），并且图没有考虑正确的答案或损失项。 最后，3c 中的图显示了

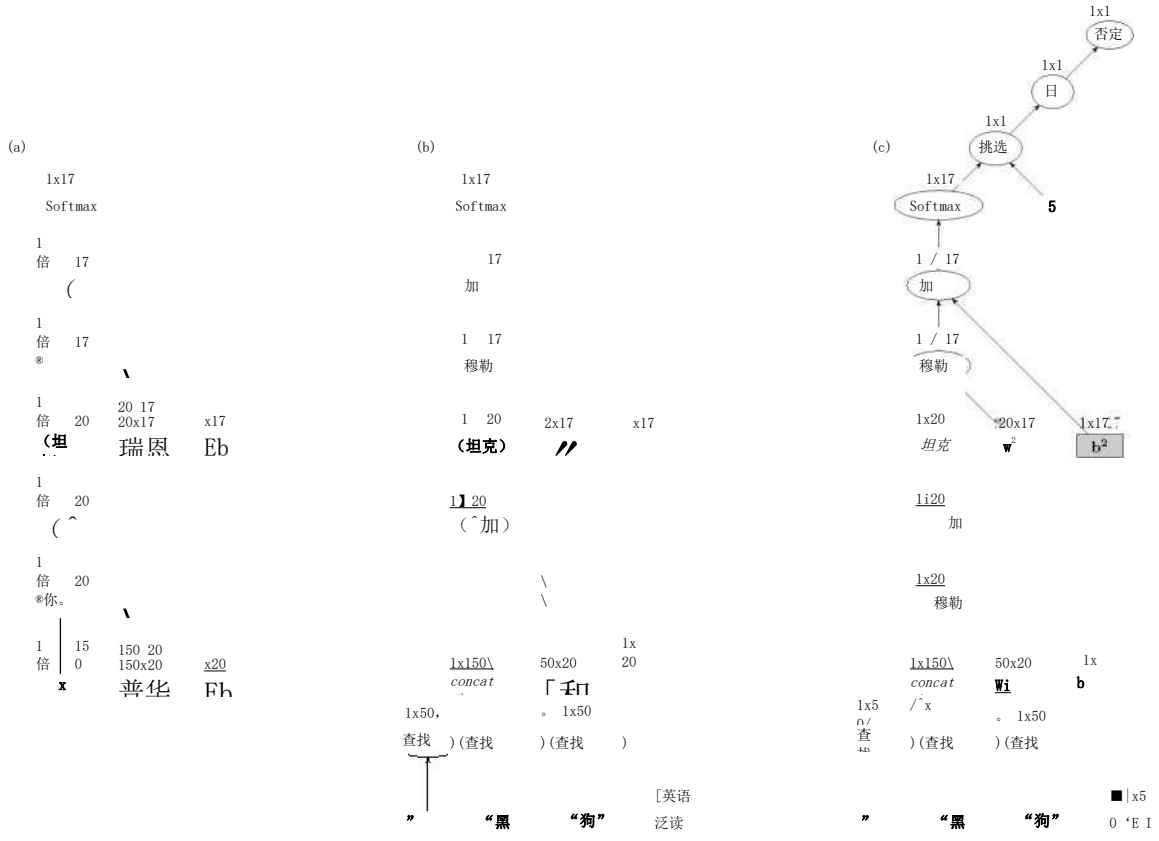



**图 3：MLP1 的计算图。** (a)无约束输入图。 (b)有具体投入的图表。 (c)具体投入、预期产出和损失节点图。

特定训练示例的计算图，其中输入是“the”、“black”、“dog”等词的（嵌入），预期输出是“NOUN”（其索引为 5）。选择节点实现索引操作，接收向量和索引（在本例中为 5）并返回向量中的相应条目。

一旦构建了图形，就可以很容易地运行正向计算(计算计算结果)或反向计算(计算梯度)，如下所示。 构建图表可能会让人望而生畏，但实际上使用专用软件库和 API 非常容易。

### 6.2.1 前向计算

前向传递计算图中节点的输出。 由于每个节点的输出只依赖于它本身及其传入的边缘，因此通过按拓扑顺序遍历节点并计算每个节点的输出来计算所有节点的输出是微不足道的，因为给定其前辈已经计算过的输出。

更正式地说，在 N 个节点的图中，我们根据它们的拓扑排序将每个节点与索引 I 关联起来。设 fi 是节点 i(例如。乘法。此外，。。。)。设 n(I)为节点 i 和 n 的父节点$^{-1}$={j|iGn(J)}节点 I 的子节点(这些是 FI 的参数)。 用 v(i)表示节点 i 的输出，即 fi 对其参数 n 的输出值的应用$^{-1}$(i)。 对于变量和输入节点，fi 是常数函数，$n^{-1}$(i)是空的。 前向算法计算所有 iG[1，N]的值 v(I]。

算法 3 计算图形前向传递

1：我=1 到 N 做

*2：让艾，...，$a_m$ =$^{-1}$(i)*

3： v(i)-fi(v(a1)、.、v(am))

### 6.2.2 反向计算（导数，反向传播）

向后传递首先指定一个节点 N，其中标量(1x1)输出为损失节点，
并向前计算到该节点。 向后计算计算
相对于该节点的值的梯度。 用 d(i)表示数量 $\frac{d\ N}{}$ 的
反向传播算法用于计算所有节点 i 的值 d(I。

向后传递填充表 d(I)如下：

算法 4 计算图后向传递（反向传播）1：d(N)-1

2：我=N-1 到 1 做

3： $^{d(i)}$-乙 *g)*$^{d(j}$）‘次

DF-
量是 fj(n)的偏导数$^{-1}$(j)w.r.t 论点$^{-1}$(j)

这个值取决于函数 fj 和值 v(A1)、...、v($a_m$)(A1，...，$a_m$=$n^{-1}$(j)其参数，在前传中计算。

因此，为了定义一种新的节点，需要定义两种方法：一种用于计算基于节点输入的前向值 v(I)，另一种用于计算 DFI
对于每个 xGn$^{-1}$(i)。
DX

有关自动区分的进一步信息，请参阅 Neidinger（2010 年，第 7 节）和 Baydin 等人的工作。（2015）。有关反向传播算法和计算图（也称为流图）的更多深入讨论，请参见 Bengio 等人的工作。 （2015 年，第 6.4 节），Le Cun 等人。 (1998 年 b)和本吉奥（2012 年）。关于一个受欢迎的技术演示，请参阅 Olah(2015a)的在线帖子)。

### 6.2.3 软件

[55][56][57]几个软件包实现了计算图模型，包括 Theano、Chainer、Penne 和 CNN/py CNN。 所有这些包都支持定义广泛的神经网络体系结构的所有基本组件（节点类型），包括本教程中描述的结构等等。 通过使用运算符重载，图形创建几乎是透明的。 该框架定义了表示图节点的类型（通常称为表达式），用于为输入和参数构造节点的方法，以及一组以表达式为输入并导致更复杂表达式的函数和数学操作。 例如，使用 py CNN 框架从图(3c)创建计算图的 python 代码是：

```
将 pycnn 导入 PC
#模型初始化。
模型=PC。 模型()
p W1=模型.add_parameters（20,150））
pb1=模型.add_parameters（20）
p W2=模型.add_parameters（17,20））
=模型 add_parameters（17）
单词=模型。add_lookup_parameters（100，50））

#  建立计算图：pc.renew_cg()#创建一个新的图形。
#  将模型参数包装为图形节点。 W1=pc.参数(p W1)
b1=pc.参数(pb1)
W2=pc.参数(p W2)
b2=pc.参数(pb2)
                                        防御 get_index(X)：返回 1#位置保持器层。 get_index（“黑”）
生成嵌入 vthe=pc.lookup(单词、get_index（“狗”）
vblack=pc.lookup(  单  词  、
vdog=pc.lookup(单词，          将叶节点连接到一个完整的图中。x=pc.concatenate([vthe，vblack，
                               vdog])输出=pc.softmax(W2*(pc.tanh(W1*x)+b1)+b2)损失
=-pc.log(pc.pick（输出，5））loss_value=损失。向前()损失。向后，#梯度被计算
                    #  并存储在相应的
                    #  参数。
```

大多数代码涉及各种初始化：第一个块定义了模型参数，这些参数在不同的计算图之间共享

55 http://deeplearning.net/software/theano/
56 http://chainer.org
57 https://bitbucket.org/ndnlp/penne

这里，build_computation_graph 是一个用户定义的函数，它为给定的输入、输出和网络结构构建计算图，返回一个损失节点。 update_parameters 是一个特定于优化器的更新规则。 配方指定为每个培训示例创建一个新的图形。 这适应了网络结构在训练示例之间变化的情况，例如递归和递归神经网络，将在第 10-12 节中讨论。 对于具有固定结构的网络，例如 MLP，创建一个基本计算图并在示例之间只更改输入和预期输出可能更有效。

### 6.2.5 网络组成

只要网络的输出是一个向量(1xk 矩阵)，通过使一个网络的输出成为另一个网络的输入来组成网络是微不足道的，从而创建任意的网络。 计算图抽象使这种能力变得显式：计算图中的节点本身可以是具有指定输出节点的计算图。 就可以了

（回想一下每个图对应于特定的训练示例）。 第二块将模型参数转换为图形节点（表达式）类型。 第三个块检索输入单词嵌入的表达式。 最后，第四块是创建图形的地方。 请注意图形创建是多么透明——创建图形和数学描述图形之间几乎是一一对应的。 最后一个块显示向前和向后通过。其他软件框架遵循类似的模式。

Theano 涉及到计算图的优化编译器，这既是祝福，也是诅咒。 一方面，一旦编译，大图可以在 CPU 或 GPU 上有效地运行，这使得它非常适合具有固定结构的大图，其中只有输入在实例之间发生变化。 然而，编译步骤本身可能会花费很大的费用，它使接口的工作变得有点麻烦。 相反，其他包侧重于构建大型和动态计算图，并在没有编译步骤的情况下“立即”执行它们。 虽然执行速度可能会受到 Theano 优化版本的影响，但这些包在处理第 10、12 节中描述的递归和递归网络以及第 8 节中描述的结构化预测设置时特别方便。

### 6.2.4 实施配方

利用计算图抽象，在算法 5 中给出了网络训练算法的伪码。

算法 5 神经网络训练与计算图抽象（使用小样本的大小 1）

```
1：定义网络参数。
2：迭代=1 到 N 做
3：训练示例 XI，YI 在数据集中做
4：      loss_node-build_computation_graph(十一，易，  参数)
5：      loss_node，向前()
6：      梯度-loss_node（）。向后()
7：      参数-update_parameters(参数，                梯度)
8：返回参数。
```

设计任意深度和复杂的网络，并且能够通过自动正向和梯度计算轻松地评估和训练它们。这使得我们很容易定义和训练结构化输出和多目标训练的网络，正如我们在第 7 节中所讨论的，以及复杂的递归和递归网络，如第 10-12 节所讨论的。

### 6.3 优化问题

一旦考虑到梯度计算，网络就会使用 SGD 或另一种基于梯度的优化算法进行训练。 被优化的函数不是凸的，长期以来，神经网络的训练被认为是一门“黑艺术”，只能由少数人来完成。 事实上，许多参数影响优化过程，必须注意调整这些参数。 虽然本教程不打算作为成功训练神经网络的全面指南，但我们确实在这里列出了一些突出的问题。 有关神经网络优化技术和算法的进一步讨论，请参阅 Bengio 等人的著作。 (2015 年，ch。 8)。 有关一些理论讨论和分析，请参阅 Glorot 和 Bengio（2010)的工作）。 有关各种实用提示和建议，请参阅 LeCun 等人的工作。 (1998a)和 Bottou（2012 年）。

#### 6.3.1 初始化

损失函数的非凸性意味着优化过程可能卡在局部最小点或鞍点，并且从不同的初始点(例如。参数的不同随机值)可能导致不同的结果。 因此，建议从不同的随机初始化开始运行训练的几个重新启动，并根据开发集选择最佳的一个。 对于不同的网络公式和数据集，结果中的方差量是不同的，不能预先预测。[59]

随机值的大小对训练的成功有重要影响。 由于 Glorot 和 Bengio（2010）的一个有效方

$$\left[-\frac{\sqrt{6}}{\sqrt{d_{\text{in}}+d_{\text{out}}}},+\frac{\sqrt{6}}{\sqrt{d_{\text{in}}+d_{\text{o}}}}\right. \tag{24}$$

案，在 Glorot 的名字之后被称为 Xavier 初始化，建议初始化一个权重矩阵 $W\in\mathbb{R}^{d_{\text{in}}\times d_{\text{out}}}$as：

$$\mathbf{wu}$$

ut

59 调试时，为了结果的重现性，建议使用固定的随机种子。

其中 U[a，b]是在[a，b]范围内均匀采样的随机值]。 该建议基于 tanh 激活函数的属性，在许多情况下工作良好，并且是许多人首选的默认初始化方法。

他等人的分析。 （2015）建议，当使用 ReLU 非线性时，应该通过从标准差为 • 的零均值高斯分布中采样来初始化权重，He 等人发现这种初始化比图像分类任务中的 Xavier 初始化更好，特别是当涉及深度网络时。

### 6.3.2 消失和爆炸梯度

在深层网络中，通常误差梯度要么消失（变得非常接近 0)，要么爆炸(变得非常高），因为它们通过计算图传播回来。 在更深的网络中，问题变得更加严重，特别是在递归和递归网络中(Pascanu，Mikolov，&Bengio，2012 年)。 解决梯度消失问题仍然是一个开放的研究问题。 解决方案包括使网络变浅，逐步训练（首先根据一些辅助输出信号训练第一层，然后修复它们，并根据实际任务信号训练完整网络的上层），执行批归一化(Ioffe&Szegedy，2015 年)（对于每一个小批，规范每个网络层的输入，使其具有零均值和单位方差），或使用专门的体系结构来帮助梯度流动(例如,第 11 节讨论的递归网络的 LSTM 和 GRU 体系结构)。处理爆炸梯度有一个简单但非常有效的解决方案：如果梯度的范数超过给定的阈值，则裁剪梯度。 设 g 是网络中所有参数的梯度，| | 是它们的 NORˆ，Pascanu 等人（2012）建议设置：g%_____ 阈值 g 如果网络，||g||是它们的 nor 丄丄丄。丄阿斯卡努。等人。（2012 年）建议设定。如果||g||> 阈值，则 g。

### 6.3.3 饱和和死亡神经元

具有 tanh 和 Sigmoid 激活的层可以变得饱和-导致该层的输出值都接近一个,即激活函数的上限。饱和神经元的梯度很小，应该避免。具有 ReLU 激活的层不能饱和，但可以“死亡”一大多数或所有值都是负的,因此对于所有输入,在零处被裁剪,从而导致该层的梯度为零。如果你的网络不能很好地训练，最好监测网络中有许多饱和或死亡神经元的层。

饱和神经元是由太大的值进入层引起的。 这可以通过改变初始化、缩放输入值的范围或改变学习速率来控制。 死神经元是由进入该层的所有信号都是负的（例如，在大的梯度更新之后，这可能会发生）。 在这种情况下，降低学习率将有所帮助。 对于饱和层，另一种选择是将 qa 十 11rd 十 arlloitc\paf+ar 十 Inalc\十 aarl cfn(Li-十 an， 11q1n cr n(Li-Tianh(h)) 活化后的饱和层，即。 而不是 g（丄丄）-Tanh（丄丄） 使用 g（丄丄）-卩 tanh(h)“♦

层归一化是对抗饱和的有效措施，但在梯度计算方面也是昂贵的。 一种相关的技术是批归一化，这是由于 Ioffe 和 Szegedy（2015），其中每个层的激活被归一化，使得它们在每个小批中的均值为 0，方差为 1。批量归一化技术成为计算机视觉深度网络有效训练的关键组成部分。 在这篇文章中，它在自然语言应用中不太受欢迎。

### 6.3.4 扭来扭去

将训练示例呈现给网络的顺序是重要的。 上面的 SGD 公式指定在每个回合中选择一个随机示例。 在实践中，大多数实现都按顺序遍历训练示例。 建议在每次通过数据之前洗牌训练示例。

### 6.3.5 学习率

学习率的选择很重要。 过大的学习率将阻止网络收敛到一个有效的解决方案。 太小的学习率需要很长时间才能收敛。 作为一个经验法则，一个人应该尝试在范围[0,1]的一系列初始学习率，例如。 0.001,0.01,0.1,1. 随着时间的推移，监控网络的损失，一旦损失停止改善，学习率就会降低。 学习速率调度降低了速率，这是观察到的小样本数的函数。 一个常见的时间表是将初始学习速率除以迭代次数。Leon Bottou(2012)建议使用表格%=NO(1+NOAT)的学习率)$^{-1}$ 其中 NO 是初始学习速率，NT 是在 TTH 训练示例中使用的学习速率，A 是一个额外的超参数。 他进一步建议在整个数据集上运行之前，根据数据的一个小样本来确定 NO 的良好值。

### 6.3.6 迷你表

参数更新发生在每个训练示例（大小为 1 的小样本）或每个 k 个训练示例中。 一些问题受益于更大的小舱口尺寸的培训。 在计算图抽象方面，可以为每个 k 个训练示例创建一个计算图，然后在平均节点下连接 k 个损失节点，其输出将是最小块的损失。 大型的小型训练在 GPU 等特殊计算体系结构的计算效率方面也是有益的,并可以用矩阵矩阵运算代替矢量矩阵运算。 这超出了本教程的范围。

## 6.4 正规化

神经网络模型参数多，容易发生过拟合。 通过正则化可以在一定程度上减轻过度拟合。 一种常见的正则化方法是 L2 正则化,通过添加加性*|时|,对具有大值的参数施加平方惩罚$^{2}$ 要最小化的目标函数的项，其中 0 是模型参数集，||-||$^{2}$是平方 L2 范数（值的平方和），A 是控制正则化量的超参数。

最近提出的另一种正则化方法是辍学(Hinton，Srivastava，Krizhevsky，Sutskever，&Salakhutdinov，2012 年)。辍学方法是为了防止网络学习依赖特定的权重。 它通过在每个训练示例中随机丢弃（设置为 0)网络中(或特定层中）的一半神经元来工作。 Wager 等人的工作。 （2013 年）在辍学方法和 L 之间建立了牢固的联系$_{2}$正规化。

丢包技术是神经网络方法在图像分类任务(Krizhevsky，Sutskever，&Hinton，2012) 特别是当与 ReLU 激活单元(Dahl，Sainath，&Hinton，2013) 。 丢包技术在神经网络的 NLP 应用中也是有效的。

## 7. 级联和多任务学习

在线训练方法与使用计算图抽象的自动梯度计算相结合，可以方便地实现模型级联、参数共享和多任务学习。

### 7.1 模型级联

是一种强大的技术，其中大型网络是通过将它们从较小的组件网络中组合而成的。 例如，我们可能有一个前馈网络来预测一个词的词性，基于它的相邻词和/或组成它的字符。 在流水线方法中，我们将使用这个网络来预测词性，然后将预测作为输入特征输入到进行句法分块或解析的神经网络中。 相反，我们可以认为这个网络的隐藏层是一种编码，它捕捉相关信息来预测词性。 在级联方法中，我们将该网络的隐藏层连接起来（而不是语音预测本身）作为句法网络的输入。 我们现在有一个更大的网络，它以单词和字符作为输入序列，并输出句法结构。 计算图抽象允许我们轻松地将错误梯度从句法任务损失传播到字符。

为了解决深层网络的消失梯度问题，以及更好地利用可用的训练材料，可以通过在相关任务上分别训练单个组件网络的参数来引导它们，然后将它们插入到更大的网络中进行进一步的调优。 例如，在将其隐藏层插入到可用训练数据较少的句法分析网络之前，可以训练语音部分预测网络，以便在一个相对较大的注释语料库上准确地预测语音部分。 如果训练数据为这两个任务提供直接监督，我们可以在训练期间利用它，方法是创建一个具有两个输出的网络，每个任务一个输出，为每个输出计算一个单独的损失，然后将损失求和成一个节点，从中反向传播误差梯度。

模型级联在使用卷积、递归和递归神经网络时非常常见，例如，递归网络用于将句子编码成固定大小的向量，然后将其用作另一个网络的输入。 递归网络的监督信号主要来自上层网络，它在输入时消耗递归网络的输出。

### 7.2 多任务学习

当我们有相关的预测任务时使用，这些任务不一定相互反馈，但我们确实相信，对于一种类型的预测有用的信息也可以对其他一些任务有用。 例如，集群、命名实体识别(NER)和语言建模是协同任务的例子。 用于预测块边界、命名实体边界和句子中下一个单词的信息都依赖于一些共享的底层句法语义表示。 而不是为每个任务训练一个单独的网络，我们可以创建一个具有多个输出的单一网络。 一种常见的方法是拥有一个多层前馈网络，它的最终隐藏层(或所有层的连接

然后将隐藏层)传递给不同的输出层。这样，网络的大部分参数在不同的任务之间共享。从一个任务中学到的有用信息可以帮助消除其他任务的歧义。 同样，计算图抽象使得构造这样的网络和计算它们的梯度非常容易，方法是为每个可用的监督信号计算一个单独的损失，然后将损失求和为一个用于计算梯度的损失。 如果我们有几个语料库，每个语料库都有不同的监督信号(例如。 我们有一个用于 NER 的语料库，另一个用于集群)，训练过程将洗牌所有可用的训练示例，执行梯度计算和更新，每次都有不同的损失。 在 Collobert 等人的工作中，介绍并讨论了语言处理背景下的多任务学习。 （2011）。 关于前馈网络中级联多任务学习的例子，请参阅 Zhang 和 Weiss（2016)的工作）。 在递归神经网络的背景下，请参阅 Luong、Le、Sutskever、Vinyals 和 Kaiser（2015）以及 Sogaard 和 Goldberg（2016)

的工作）。

8. 结构化输出预测

NLP 中的许多问题涉及结构化输出：期望输出不是类标签或类标签上的分布，而是序列、树或图等结构化对象的情况。 典型的例子是序列标记(例如。 词性标注)序列分割(集群、NER)和句法分析。 在本节中，我们将讨论前馈神经网络模型如何用于结构化任务。 在后面的章节中，我们将讨论处理序列（第 10 节)和树(第 12 节)的专门神经网络模型）。

### 8.1 贪婪的结构化预测

结构预测的贪婪方法是将结构预测问题分解为一系列局部预测问题,并训练分类器来执行每个局部决策。 在测试时，训练的分类器以贪婪的方式使用。 这种方法的例子是左到右标记模型(Gimenez&Marquez，2004 年)和基于贪婪的过渡解析(Nivre，2008 年)。 这种方法很容易适应于使用神经网络，简单地将局部分类器从线性分类器(如 SVM 或 Logistic 回归模型)替换为神经网络，如 Chen 和 Manning（2014）和 Lewis 和 Steedman（2014)所示）。

贪婪的方法受到错误传播的影响，早期决策中的错误会影响到以后的决策。 非线性神经网络分类器的整体高精度有助于在一定程度上抵消这一问题。 此外，还提出了通过尝试在较困难的预测之前采取更容易的预测(Goldberg&Elhadad，2010 年的第一种方法)或通过将培训程序暴露于可能的错误造成的投入，使培训条件更类似于测试条件(Hal Daume III，Langford，&Marcu，2009 年；Goldberg&Nivre，2013 年)来减轻错误传播问题的培训技术)。这些对于训练贪婪神经网络模型也是有效的,如 Ma、Zhang 和 Zhu(2014)(Easy-FirstTagger)和 Ballesteros、Goldberg、Dyer 和 Smith（2016）(贪婪依赖解析的动态 Oracle 训练)所示)。

### 8.2 基于搜索的结构化预测

预测自然语言结构的常用方法是基于搜索。 关于 NLP 中基于搜索的结构预测的深入讨论，请参阅 Smith（2011)的书）。 这些技术可以很容易地适应于使用神经网络。 在神经网络文献中，这些模型是在基于能量的学习框架下讨论的(LeCun 等人，2006 年，第 7 节)。 它们在这里使用 NLP 社区熟悉的设置和术语来呈现。

基于搜索的结构化预测被描述为对可能结构的搜索问题：

$$\text{预测}(X)=\underset{y\ 0(x)}{\text{ARG 最大评分}}(x,\ y) \tag{25}$$

其中 x 是输入结构，y 是 x 上的输出(在一个典型的例子中，x 是一个句子，y 是句子上的标记分配或解析树)，y(X)是 x 上所有有效结构的集合，我们正在寻找一个输出 y，它将使 x，y 对的分数最大化。

评分函数定义为线性模型：

$$\text{分数}(x,\ y)=w\text{-}①(x,\ y) \tag{26}$$

其中①是特征提取函数，w 是权向量。

为了使搜索最优 y 可处理，将结构 y 分解为零件，并根据零件定义特征函数，其中©(P)是局部特征提取函数：

$$\text{以}^{+,}\ y)=\underset{\text{∈零件}(x,\ y)}{£©^{\prime}p)} \tag{27}$$

$$\text{分数}(x,\ y)=w\text{-}(x,\ y)\quad \mathcal{O}p)\quad \underset{p\ Cy}{\text{桓}^{w}}\ W(p)=\quad \text{得分}(P) \tag{28}$$

其中 pGy 是 pe 部分(x，y)的速记)。 将 y 分解为部分，使得存在一种推理算法，该算法允许有效地搜索给定单个部分分数的最佳评分结构。

现在我们可以简单地用神经网络代替部分上的线性评分函数：

$$(x.y)-\sum_{p\in y}\qquad \underset{p\ Cy}{\text{ENN}(c(p))} \tag{29}$$

各部分分别计分，结构得分为各组成部分得分之和：

其中 c(P)将 p 部分映射为 $di_{n}$ 维向量。 在一个隐藏层前馈网络的情况下：



得分　　得分(P)

$$\text{分数}(x,\ y)=\pounds nN\ \underset{p\ Cy}{\text{mlpi}}(c(P)= \qquad \underset{p\pounds y}{22(g(c(p)W^{\sharp}+b^{1})w} \tag{30}$$

c(p)e $R^{\top}$，$W^{1}$e $R^{\top XD1}$$b^{1}$e $R^{d1d1}$.结构化预测的一个共同目标是使黄金结构 y 得分高于任何其他结构 $y^{z}$ 导致以下（广义感知器）损失：

$$\underset{寸}{\text{最大分数}}(x,\ y^{z}\text{得分}(x,\ y) \tag{31}$$

在实现方面，这意味着：为每个可能的部分创建一个计算图 CGP，并计算其分数。 然后，对得分部分进行推理，找出最佳得分结构 $y^{r}$。 将与金（预测）结构 y(y ‘)中的部分对应的计算图的输出节点连接到求和节点 CGy($CG^{r}_{y}$)。 连接 CGY 和 $CG^{r}_{y}$ 使用“负”节点，CGI，并计算梯度。

正如 LeCun 等人所说。 （2006 年，第 5 节），当训练结构化预测神经网络时，广义感知器损失可能不是一个很好的损失函数，因为它没有裕度，并且首选基于裕度的铰链损失：

$$\underset{你=你}{\text{最大值}}(0,\ m+\text{最大分数}(x,\ y\ ‘)-\text{分数}(x,\ y)) \tag{32}$$

修改上面的实现以处理铰链损耗是微不足道的。

请注意，在这两种情况下，我们都失去了线性模型的良好性质。 特别是，模型不再是凸的。 这是可以预料的，因为即使是最简单的非线性神经网络也已经是非凸了。 尽管如此，我们仍然可以使用标准的神经网络优化技术来训练结构化模型。

训练和推理较慢，因为我们必须|次评估神经网络（并采取梯度）|部分(x，y。

结构化预测是一个广阔的领域，超出了本教程的范围，但损失函数、正则化和方法，例如 Smith（2011）所描述的方法，如成本增强解码，可以很容易地应用或适应神经网络框架。[60]

### 8.2.1 概率目标(CRF)

在概率框架(条件随机字段，“CRF”)中，我们将每个部分的分数视为一个团势(见 Smith，2011 年和 Lafferty，McCallum，&Pereira，2001 年的讨论)，并将每个结构 y 的分数定义为：

$$R_{f}(x,\ y)=P(yx)=\ \underset{Ey/_{,}y(x)\exp^{(}\pounds\text{佩伊，得分®})=\ \underline{\exp}\ \underline{S/NN^{(}©^{(}p)))}}{(、\cdots、\ \ \exp(\pounds\ \ \text{得分}(P))\ \ \text{---------- ----------}\ \ -Ey/_{,}y(x)\exp^{(}\text{安永，}NN^{(}©^{(}p)))} \tag{33}$$

评分函数定义了条件分布 P(y\x)，我们希望设置网络的参数，使语料库条件日志似然£(。

60人们应该记住，由此产生的目标不再是凸的，因此缺乏与凸优化问题相关的形式保证和界限。 同样，与算法相关的理论、学习边界和保证也不会自动转移到神经版本。

勿)培训日志 P(y\ux ‘)最大化。

给定训练示例(x，y)的损失是：-log 评分 cRr(x，y)。 与建立相关的计算图一样，考虑损失的梯度。 棘手的部分是分母（划分函数），它需要对 Y • 中潜在的指数多个结构进行求和。然而，对于一些问题，存在一种动态规划算法来有效地求解多项式时间的求和(即。 序列的前向后维特比递归和树结构的 CKY 内外递归)。 当这种算法存在时，它也可以适应于创建多项式大小的计算图。

当没有足够有效的算法来计算分区函数时，可以使用近似方法。 例如，人们可以使用波束搜索来推断，而不是在指数大 Y(X)上对剩余在波束中的结构进行分割函数和)。

具有神经网络群势的序列级 CRF 由 Peng、Bo 和 Xu（2009）和 Do、Arti 等人（2010）讨论，它们被应用于生物数据、OCR 数据和语音信号的序列标记，Wang 和 Manning（2013）将它们应用于传统的自然语言标记任务(集群和 NER)。 裴等人采用了一种基于铰链的方法。（2015 年）用于弧形依赖分析，以及 Durrett 和 Klein（2015 年）用于 CRF 选区解析器的概率方法。 周等人有效地利用了基于波束的近似分割函数。 （2015 年）基于转换的解析器。

### 8.2.2 打屁股

当对所有可能的结构进行搜索是难以解决的、低效的或难以集成到模型中时，经常使用重新排序方法。 在重新排序框架(Charniak&Johnson，2005；Collins&Koo，2005)中，使用一个基本模型来生成 kest 评分结构的列表。 然后训练一个更复杂的模型，在 k-最佳名单中给候选人打分，这样黄金的最佳结构得分最高。 由于搜索现在是在 k 项上执行的，而不是在指数空间上执行的，复杂模型可以条件（从）评分结构的任意方面提取特征。 递归方法是使用神经网络模型进行结构化预测的自然候选方法，因为它们允许建模者专注于特征提取和网络结构，同时消除了将神经网络评分集成到解码器中的需要。 事实上，重新排序方法通常用于实验神经模型，这些模型不容易集成到解码器中，例如卷积、递归和递归网络，这将在后面的章节中讨论。 使用重新排序方法工作

包括 Schwenk 等人的那些。 （2006 年），Socher 等人。 （2013 年），Auli 等人。 （2013 年）、Le 和 Zuidema（2014 年）和 Zhu 等人。 (2015a)。

### 8.2.3MEMM 和混合方法

当然，其他配方也是可能的。 例如，MEMM(McCallum，Freitag，&Pereira，2000)可以通过用 MLP 替换 Logistic 回归（“最大熵”）组件来简单地适应神经网络世界。

还探讨了神经网络和线性模型之间的混合方法。 特别是 Weiss 等人。 （2015 年）报告了两阶段模型中基于过渡的依赖解析的强大结果。 在第一阶段，对静态前馈神经网络(MLP2)进行训练，以孤立地对结构化问题的每个单独决策执行良好。 在第二阶段，神经网络模型被固定，然后将每个输入的不同层（输出以及隐藏层向量）连接起来，并用作线性结构感知器模型(Collins，2002)的输入特征，该模型被训练以执行波束搜索最佳结果结构。 虽然目前尚不清楚这种训练机制是否比训练单一结构预测神经网络更有效，但使用两个更简单、孤立的模型使研究人员能够进行更广泛的超参数搜索(例如。 每个模型的调谐层大小、激活函数、学习速率等都比在更复杂的网络中可行。

9. 卷积层

有时，我们感兴趣的是根据有序的项目集进行预测(例如。 句子中的单词序列、文档中的句子序列等等)。 例如，考虑预测句子的情绪(积极、消极或中性。 一些句子单词对情感信息非常丰富，其他单词信息较少，而且在很好的近似下，一个信息丰富的线索是信息丰富的，不管它在句子中的位置如何。 我们想把所有的句子单词都反馈给一个学习者，让训练过程找出重要的线索。一个可能的解决方案是将 CBOW 表示输入一个完全连接的网络，例如 MLP。然而，CBOW 方法的一个缺点是它完全忽略了排序信息，将句子分配为“它不是好的，它实际上是相当糟糕的”和“它不是坏的，它实际上是相当好的”完全相同的表示。 虽然指标的全球位置“不好”和“不坏”对分类任务并不重要,但单词的局部排序(“不”一词出现在“坏”一词之前）是非常重要的。 一种天真的方法会建议嵌入单词对（双图）而不是单词，并在嵌入的 bigrams 上构建一个 CBOW。 虽然这样的体系结构可能是有效的，但它将导致巨大的嵌入矩阵，不会扩展到更长的 ngram，并且会受到数据稀疏问题的影响，因为它不能在不同的 ngram 之间共享统计强度（“相当好的”和“非常好的”的嵌入是完全相互独立的，因此如果学习者在培训期间只看到其中一个，它将无法根据其组件词推断出任何关于另一个的信息）。 卷积和池(也称为卷积神经网络，或 CNN)体系结构是解决这个建模问题的优雅和健壮的解决方案。 设计了一个卷积神经网络来识别大结构中的指示性局部预测器，并将它们组合起来，生成结构的固定大小向量表示，捕获这些对手头的预测任务信息最丰富的局部方面。

卷积和池结构(LeCun&Bengio，1995)是在神经网络视觉社区中进化而来的，在那里，它们作为对象检测器显示出巨大的成功-从预定义的类别（“猫”、“自行车”）中识别一个对象，而不管它在图像中的位置如何(Krizhevsky 等人，2012 年)。 当应用于图像时，该体系结构使用二维（网格）卷积。 当应用于文本时，我们主要关注的是一维（序列）卷积。 卷积网络是在 Collobert、Weston 和他的同事（2011 年）的开创性工作中引入到 NLP 社区的，他们使用它们进行语义角色标记，后来由 Kalchbrenner 等人引入。（2014 年）和 Kim（2014 年），他们将它们用于情感和问题类型分类。

### 9.1 基本卷积+池化

语言任务的卷积和池架构背后的主要思想是在句子上的 k 字滑动窗口的每个实例化上应用非线性（学习）函数。这个函数（也称为“过滤器”）将 k 个单词的窗口转换为一个 d 维向量，它捕获窗口中单词的重要属性（每个维度有时在文献中被称为“通道”）。然后，使用“池”操作将来自不同窗口的向量组合成一个 d 维向量，方法是取在不同窗口上的每个 d 通道中观察到的最大值或平均值。目的是关注句子中最重要的“特征”，而不管它们的位置如何。然后将 d 维向量进一步输入用于预测的网络。 在训练过程中，从网络损失中传播回来的梯度被用来调整滤波器函数的参数，以突出显示对网络训练的任务重要的数据方面。 直观地说，当滑动窗口运行在序列上时，过滤器函数学会识别信息丰富的 k-gram。

更正式地说，考虑一个单词序列 x=xi，...，$x_n$每个都有相应的 d $_{阿姆}$[61]二维词嵌入 v(x《)。宽度 k 的一维卷积层通过在句子上移动一个大小为 k 的滑动窗口来工作，并将相同的“滤波器”应用于序列中的每个窗口[v(x ‘)；v(x ‘+i)；v(x ‘+k_i)]。 滤波器函数通常是线性变

61 这里的一维是指在一维输入（如序列）上工作的卷积，而不是应用于图像的二维卷积。

换，然后是非线性激活函数。

设第 i 个窗口的级联向量为 w；=[v(x《)；v(x《+i)；.；v(x《+k_i)]，wi。 根据我们是否将句子每边用 k-1 个单词标记，我们可以得到 m=n-k+1（窄卷积）或 m=n+k（宽卷积）(Kalchbrenner 等人，2014 年)。 卷积层的结果是 m 向量 pi，...，pm，pieR$^{dconv}$ 地点：

$$PI=g(wiW+b) \tag{34}$$

是一个非线性激活函数，它是应用元素的，WER$^{K,\ Demb\ Xdconv}$ 和 b$^{dconv}$ 是网络的参数。 每个 pi 是一个 dconv 维向量，编码

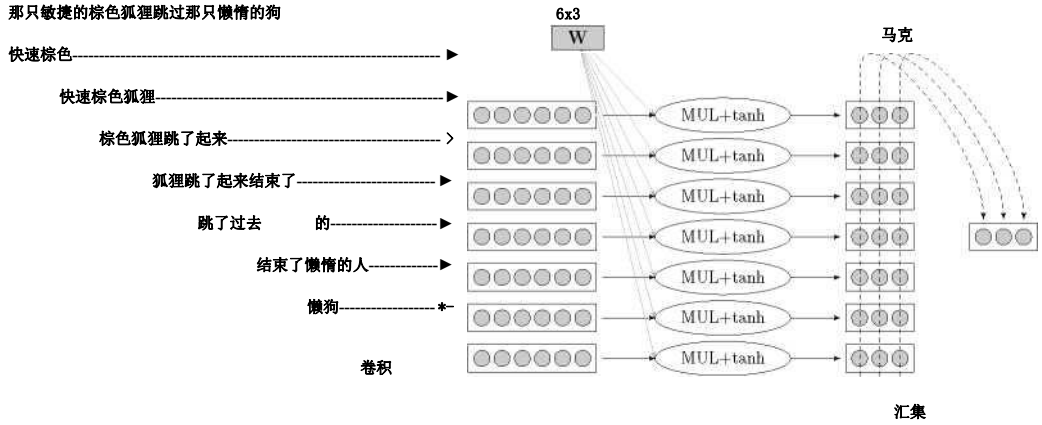



图 4：一维卷积+汇集在句子“快速棕色狐狸跳过懒惰的狗”上”。 这是一个窄的卷积（句子中不添加填充），窗口大小为 3。 每个单词被翻译成一个二维嵌入向量（未显示）。 然后将嵌入向量连接起来，得到 6-dim 窗口表示。 七个窗口中的每个都通过一个 6x3 滤波器(线性变换，然后是元素级 tanh)进行传输，从而产生七个三维滤波表示。 然后，应用最大池操作，在每个维度上取最大值，最终得到一个三维集合向量。

理想情况下，每个维度捕获不同类型的指示性信息。 然后使用最大池层组合 m 向量，从而产生单个 dconv 维向量 c。

$$CJ=\max_{1<我<m} pi[j] \tag{35}$$

π[j]表示 π 的 jth 分量。最大池操作的效果是在窗口位置获得最显著的信息。理想情况下，每个维度将“专门化”于特定类型的预测器，而最大操作将选择每种类型的最重要预测器。

图 4 提供了过程的说明。

得到的向量 c 是句子的表示，其中每个维度反映了与某些预测任务有关的最突出的信息。然后将 c 馈送到下游网络层，可能与其他向量并行，最终形成用于预测的输出层。 网络的训练过程计算相对于预测任务的损失，误差梯度通过池层和卷积层以及嵌入层一路传播回来。[62]

62除了对预测有用外，训练过程的一个副产品是一组参数 W、B 和嵌入 v()，这些参数可以用于卷积和池体系结构来编码任意长度

虽然最大池是文本应用程序中最常见的池操作，但其他池操作也是可能的，第二个最常见的操作是平均池，取每个索引的平均值而不是最大值。

### 9.2 动态、层次和 k-max 池

与其在整个序列上执行单个池操作，我们可能希望根据我们对手头预测问题的域理解保留一些位置信息。 为此，我们可以将向量 pi 分成/不同的组，在每个组上分别应用池，然后连接/结果 d 康v 维向量 ci，...，5 在域知识的基础上，将 pi 分成组。 例如，我们可以推测，在句子中出现的早期单词比出现晚的单词更具指示性。 然后，我们可以将序列分割成/大小相等的区域，对每个区域应用一个单独的最大池。 例如，Johnson 和 Zhang（2015)发现，在将文档分类为主题时，有 20 个平均池区域是有用的，可以清楚地将初始句子(通常介绍主题的地方)与后面的句子分开，而对于情感分类任务，对整个句子的单个最大池操作是最佳的(表明一个或两个非常强的信号足以确定情感，而不管句子中的位置如何）。

同样，在关系提取类任务中，我们可以得到两个单词，并被要求确定它们之间的关系。我们可以认为，第一个单词之前的单词、第二个单词之后的单词以及它们之间的单词提供了三种不同的信息(Chen 等人，2015 年)。 因此，我们可以相应地拆分 pi 向量，将每个组产生的窗口分开汇集。

另一个变化是使用卷积层的层次结构，其中我们有一系列的卷积和池层，其中每个阶段对一个序列应用卷积，池每个 k 个相邻向量，对生成的集合序列执行卷积，应用另一个卷积等等。 这种结构允许对越来越大的结构敏感。

最后，Kalchbrenner 等人。 （2014 年）引入了 k-max 池操作，其中保留每个维度中的顶部 k 值，而不是仅保留最佳值，同时保留它们在文本中出现的顺序。 例如 a，考虑以下矩阵：

1 2 3’
2 3 1
7 8 1
341

将导致以下矩阵：
96
78
然后将其行连接到列向量上的 1-max 池将导致[985]，而 2-max 池
963785

句子变成固定大小的向量，这样共享相同类型预测信息的句子就会彼此接近。

k-max 池操作使得可以将 k 个最活跃的指示器集合起来，这些指示器可能是若干位置分开的；它保留了特征的顺序，但对它们的特定位置不敏感。 它还可以更精细地识别特征被高度激活的次数(Kalchbrenner 等人，2014 年)。

### 9.3 差异

而不是单个卷积层，几个卷积层可以并行应用。 例如，我们可能有四个不同的卷积层，每

个层的窗口大小在 2-5 范围内不同，捕获不同长度的 n-gram 序列。 然后将每个卷积层的结果汇集起来，并将得到的向量连接起来并馈送到进一步处理(Kim，2014 年)。

卷积结构不需要限制在句子的线性排序中。 例如，Ma 等人。 （2015 年）将卷积运算推广到句法依赖树上。 在那里，每个窗口都围绕着句法树中的一个节点，池在不同的节点上执行。 同样，Liu 等人。（2015 年）在从依赖树提取的依赖路径之上应用卷积体系结构。 Le 和 Zuidema（2015）建议在图表解析器中对表示导致相同图表项的不同导子的向量执行最大池。

10. 递归神经网络-建模序列和堆栈

在处理语言数据时，通常使用序列，如单词（字母序列)、句子(单词序列）和文档。 我们看到了前馈网络如何通过使用向量级联和向量加法(CBOW)来适应序列上的任意特征函数)。特别是，CBOW 表示允许将任意长度序列编码为固定大小的向量。 然而，CBOW 表示是相当有限的，并且迫使一个人忽略特征的顺序。卷积网络还允许将序列编码成固定大小的向量。虽然从卷积网络导出的表示比 CBOW 表示更好，因为它们对语序具有一定的敏感性，但它们的顺序敏感性仅限于大多数局部模式，而忽略了序列中相距很远的模式的顺序。

递归神经网络(RNNs)(Elman，1990)允许在固定大小的向量中表示任意大小的结构化输入，同时注意输入的结构化特性。

### 10.1 RNN 抽象

我们用习$_{:}$ 表示向量 xi，...，xj 的序列。 RNN 抽象以输入向量 xi、...、x 的有序列表作为输入$_{n}$ 与初始状态向量 s° 一起，返回一个有序的状态向量 si，...，sn 列表，以及输出向量 yi，...，y 的有序列表-输出向量 yi 是相应的状态向量 si 的函数。 输入向量 xi 以顺序的方式呈现给 RNN，状态向量 $s_{i}$ 输出向量 $y_{i}$在观察输入 xi：i 后表示 RNN 的状态。 然后使用输出向量 yi 进行进一步的预测。 例如，一个模型

预测事件 e 的条件概率给定序列 MI。我可以定义为 p(e=j|xi：i)=Softmax(YiW+b)[j]，是 Softmax 操作产生的输出向量中的 JTH 元素。 RNN 模型提供了一个框架来调节整个历史 xi，...，xi，而不求助于传统上用于建模序列的马尔可夫假设。[63]事实上，与基于 ngram 的模型相比，基于 RNN 的语言模型会产生很好的困惑评分。

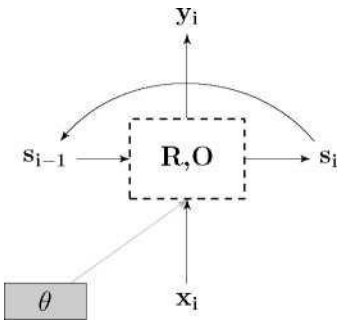
在数学上，我们有一个递归定义的函数 R，它以状态向量 Si 和输入向量 Xi 作为输入，并产生一个新的状态向量 si，[64]i. 附加函数 O 用于将状态向量 Si 映射到输出向量 y，当构造 RNN 时，就像构造前馈网络时一样，必须指定输入 x 的维数 i 以及输出 y 的维数 i。 状态 Si 的维数是输出维数的函数。[65]

$$RNN(so, xi: n)=si: n, yi: n$$

$$s我=R(sI-1, xi)=O(S\ i) \quad (36)$$

Xie e R ，易达梯，sie r f(dout)

函数 R 和 O 在序列位置上是相同的，但是 RNN 通过 R 的调用之间保持和传递的状态向量来跟踪计算状态。




从图形上看，RNN 传统上如图 5 所示。

63 kth-orderMarkov 假设表明，时间 i 的观测与观测无关
有时 i-(k+j)Vj>0，　　我-k。 这个假设是在
许多序列建模技术的基础，如 n-gram 模型和隐马尔可夫模型。

64 使用 O 函数有点不标准，用于统一下一节中要呈现的不同 RNN 模型。 对于简单 RNN(ElmanRNN)和 GRU 体系结构，O 是标识映射，对于 LSTM 体系结构，O 选择状态的固定子集。

65 虽然状态维度与输出维度无关的 RNN 体系结构是可能的，但目前流行的体系结构，包括简单的 RNN、LSTM 和 GRU，并不遵循这种灵活性。

图 5：RNN（递归）的图形表示）。

此表示遵循递归定义，对于任意长序列是正确的。 然而，对于有限大小的输入序列（我们处理的所有输入序列都是有限的），可以展开递归，从而得到图 6 中的结构。

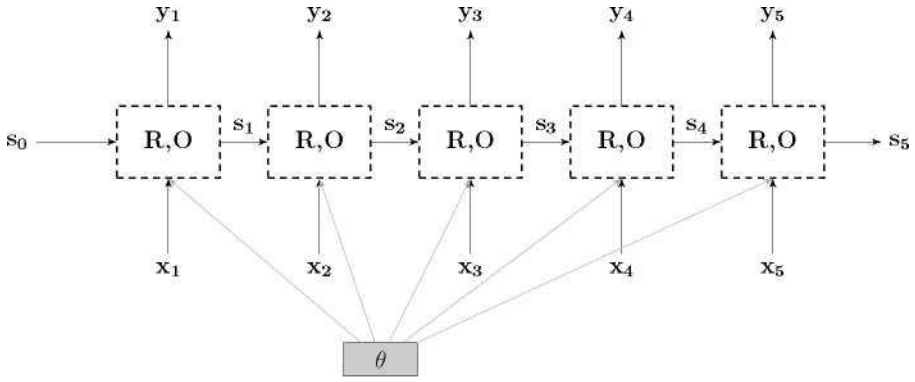



Figure 6: Graphical representation of an RNN (unrolled).

虽然通常不显示在可视化中，但我们在这里包含参数 Q，以突出显示在所有时间步骤中共享相同参数的事实。 不同的 R 和 O 实例化将导致不同的网络结构，并且在运行时间和使用基于梯度的方法有效训练的能力方面将表现出不同的属性。 然而，它们都坚持相同的抽象接口。 我们将在第 11 节中提供 R 和 O 的具体实例化细节-简单的 RNN、LSTM 和 GRU。 在此之前，让我们考虑使用 RNN 抽象进行建模。

首先，我们注意到 s；的值是基于整个输入 xi，例如，通过扩展 I=4 的递归，我们得到：

$$
\begin{aligned}
\text{第 } 4 &= r(S3,\ x4) \\
&\overset{S3}{=} r(s2,\ x3),\ x4) \\
&= R(R(Si,\ X2),\ X3),\ X4) \\
&\overset{S1}{=} R(R(So,\ Xi),\ X2),\ X3),\ X4)
\end{aligned}
\tag{37}
$$

因此，$s_n$（还有 $y_n$可以被认为是编码整个输入序列。[66]编码有用吗？ 这取决于我们对有用性的定义。 网络训练的工作是设置 R 和 O 的参数，使状态为我们要解决的任务传递有用的信息。

### 10.2 RNN 培训

从图 6 中可以很容易地看到，未滚动的 RNN 只是一个非常深的神经网络（或者更确切地说，

66 请注意，除非 R 是专门针对这一点设计的，否则输入序列的后期元素对 Sn 的影响可能比早期元素更强。

是一个具有一些复杂节点的非常大的计算图），其中相同的参数在计算的许多部分共享。为了训练 RNN 网络，我们所需要做的就是为给定的输入序列创建未滚动的计算图，在未滚动的图中添加一个损失节点，然后使用反向（反向传播）算法计算与该损失有关的梯度。 这一程序在 RNN 文献中被称为通过时间的反向传播，或 BPTT(Werbos，1990)。[67]监理信号的应用方式多种多样。

### 10.2.1 接受者

一种选择是将监督信号仅基于最终输出向量 $y_n$。 从这个角度来看，RNN 是一个受体。 我们观察最后的状态，然后决定结果。[68] 例如，考虑培训一个 RNN 逐个阅读一个单词的字符，然后使用最终状态来预测该单词的词性(这是由 Ling 等人提出的，2015 年 b)，一个 RNN 在一个句子中阅读，并根据最终状态决定它是传达积极的还是消极的情绪(这是由 Wang 等人提出的，2015 年 b)，还是一个 RNN，它以一系列的单词阅读并决定它是否是一个有效的名词短语。 在这种情况下，损失是用 yn=O(Sn)的函数定义的，误差梯度将通过序列的其余部分反向传播（见图 7）。[69] 损失可以采取任何熟悉的形式-交叉熵、铰链、边缘等。

### 10.2.2 编码器

类似于受主情况，编码器监督只使用最终输出向量 y-然而，与受主不同的是，预测完全基于最终向量，这里将最终向量视为序列中信息的编码，并与其他信号一起用作附加信息。例如，提取文档摘要系统可能首先使用 RNN 运行文档，结果

---

67 BPTT 算法的变体包括每次只对固定数量的输入符号展开 RNN：首先对输入 xi：k 展开 RNN，导致 si：k。计算损失，并通过网络反向传播错误(k 步回)。 然后，打开输入 Xk+i：2k，这次使用 Sk 作为初始状态，然后再次反向传播 k 步骤的错误，等等。 这种策略是基于观察到的，对于 Simple-RNN 变体，k 步后的梯度倾向于消失(对于足够大的 k)，因此省略它们是可以忽略不计的。 这个过程允许训练任意长的序列。 对于 RNN 变体，如 LSTM 或 GRU，它们是专门为减轻消失梯度问题而设计的，这种固定大小的展开动机较小，但它仍然被使用，例如，当在一本书上进行语言建模时，而不将其分解为句子。类似的变体在前进步骤中为整个序列展开网络，但只从每个位置将梯度传播回 k 个步骤。

68 术语是从有限国家接受方借用的。 然而，RNN 具有潜在的无限数量的状态，因此需要依赖查找表以外的函数将状态映射到决策。

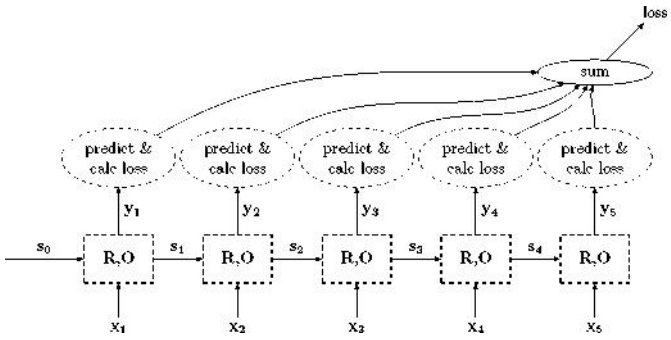
69 这种监督信号可能很难训练长序列，特别是简单的 RNN，因为消失梯度问题。 这也是一项普遍艰苦的学习任务，因为我们不告诉过程中哪些部分的输入要集中。


图 7：接受器 RNN 训练图。

在向量 y 中 n 总结整个文档。 然后，你 n 将与其他功能一起使用，以便选择要包含在摘要中的句子。

### 10.2.3 传送器

另一种选择是将 RNN 视为换能器，为它读取的每个输入产生输出。 这样建模，我们就可以根据一个真实的标签计算每个输出 YI 的局部损失信号 Liocai(5^I，YI)，然后展开序列的损失将是：L(Yi：n，Yi：n)= 或使用另一个组合而不是这样的总和
作为平均值或加权平均值（见图 8）。 这样一个换能器的一个例子是序列标记器，其中我们将 xi：n 作为句子 n 个单词的特征表示，而 yi 作为一个输入，用于预测基于单词 1：i 的单词 i 的标记分配。基于这种架构的 CCG 超级标签提供了最先进的 CCG 超级标签结果(Xu




等人，2015 年)。

转导设置的一个非常自然的用例是用于语言建模，其中单词 x 的序列 ₁₁ 用于预测(I+1)单词的分布。 基于 RNN 的语言模型比传统语言模型提供了更好的困惑(Mikolov 等人，2010 年；Sundermeyer，Schluter，&Ney，2012 年；Mikolov，2012 年；Jozefowicz，Vinyals，Schuster，Shazer，&Wu，2016 年)。

使用 RNNs 作为传感器可以让我们放松传统上在语言模型和 HMM 标记器中采用的马尔可夫假设，以及整个预测历史的条件。 在生成字符级 RNN 模型中，证明了对任意长历史进行条件处理的能力，其中文本是按字符生成的，每个字符都取决于以前的字符(Sutskever，



图 8：换能器 RNN 训练图。

Martens，&Hinton，2011)。 生成的文本显示了对 n-gram 语言模型未捕获的属性的敏感性，包括行长和嵌套括号平衡。 关于基于 RNN 的字符级语言模型的属性的良好演示和分析，请参阅 Karpathy、Johnson 和 Li（2015)的工作）。

### 10.2.4 编码器-解码器

最后，编码器场景的一个重要特例是编码器-解码器框架(Cho，van Merrienboer，Bahdanau，&Bengio，2014a；Sutskever 等人，2014 年)。 使用 RNN 将序列编码为向量表示 $y_n$然后，这个向量表示被用作另一个用作解码器的 RNN 的辅助输入。 例如，在机器翻译装置中，第一个 RNN 将源句编码为向量表示 yn，然后将该状态向量输入一个单独的（解码器）RNN，该 RNN 被训练以预测（使用类似换能器的语言建模目标）基于先前预测的单词以及 yn 的目标语言句子的单词。 监督只发生在解码器 RNN，但梯度一直传播到编码器 RNN（见图 9）。

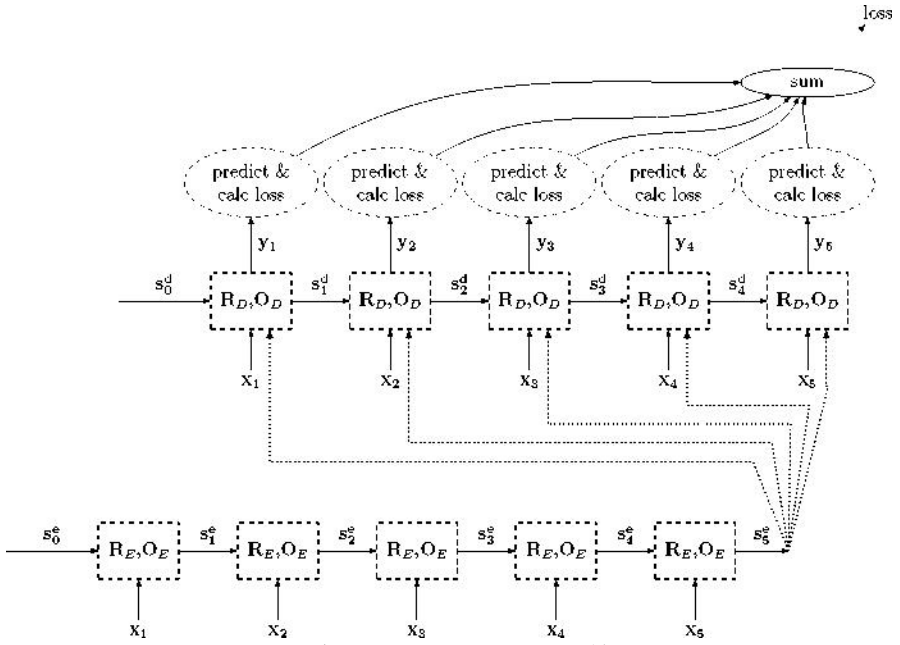



图 9：编码器-解码器 RNN 训练图。

这种方法被证明是令人惊讶的有效的机器翻译(Sutskever 等人，2014 年)使用 LSTMRNN。为了使这种技术发挥作用，Sutskever 等人。 发现反向输入源句是有效的，因此 Xn 对应于第一个

句子的单词。 这样，第二个 RNN 更容易建立源句的第一个单词与目标句的第一个单词之间的关系。 编解码框架的另一个用例是序列转导。 在这里，为了生成标签 TX，...，$t_n$首先使用编码器 RNN 对句子 xi 进行编码$_{*n}$ 变成固定大小的矢量。 然后将该向量作为另一个（换能器）RNN 的初始状态向量，与 xi：n 一起用于预测每个位置 i 处的标签 t。 这种方法被 Filippova、Alfonseca、Colmenares、Kaiser 和 Vinyals（2015）使用来通过删除来模拟句子压缩。

### 10.3 多层（堆叠）RNN

RNN 可以层层叠叠，形成网格(Hihi&Bengio，1996)。考虑 kRNN，RNNi，RNNk，其中 JT HRNN 有状态 s{：n，输出 y{：口 The 第一个 RNN 的输入是 xi：n，而 JT HRNN(j>2)的输入是 RNN 在其下面的输出，yl. ⁻ ‘•整个编队的输出是最后一个 RNN 的输出，yl*这样的分层结构通常被称为深度 RNN。 图 10 给出了三层 RNN 的视觉表示。

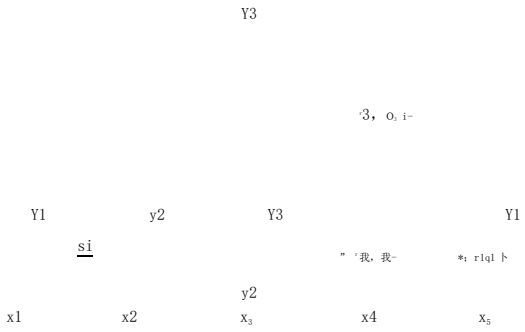



图 10：一个层（“深”）RNN 体系结构。

虽然理论上还不清楚深层结构所获得的额外力量是什么，但经验上观察到，深层 RNN 比较浅的 RNN 在某些任务上更好地工作。 特别是 Sutskever 等人。 （2014 年）报告说，一个 4 层的深度架构对于在编解码框架中实现良好的机器翻译性能至关重要。Irsoy 和 Cardie（2014）还报告了从 onelayer bi RNN 到具有多个层的体系结构的改进结果。 许多其他工作使用分层 RNN 体系结构报告结果，但没有显式地与 1 层 RNN 进行比较。

### 10.4 双向 RNN(biRNN)

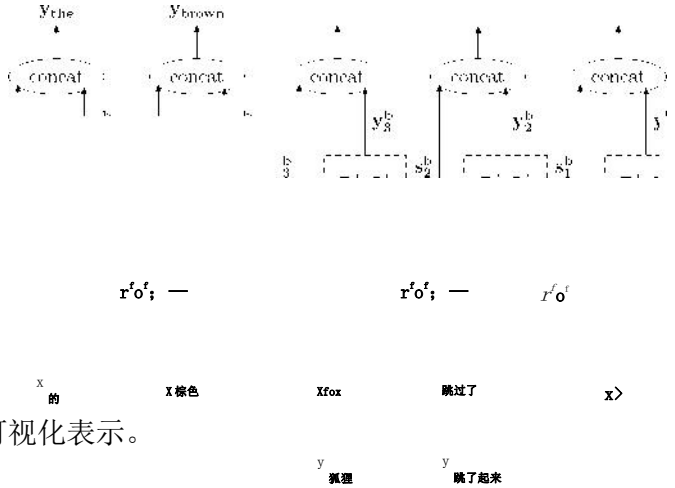
一个有用的 RNN 是一个双向 RNN(biRNN，也通常称为 biRNN)(Schuster&Paliwal，1997 年；Graves，2008 年)。[70]考虑在句子 xi、...、x 上进行序列标记的任务 $_n$。 一个 RNN 允许我们计算第一个单词的函数

70 当与特定的 RNN 体系结构(如 LSTM)一起使用时，该模型称为 biLSTM。

基于过去的习-习：我支持并包括它。 然，下面的词兮 也可能是有用的预测，这是显而易见的常见滑动窗口方法，其中焦点词是根据一个窗口的 k 个词周围的它分类。 就像 RNN 放松了马尔可夫假设，允许任意回顾过去一样，biRNN 放松了固定窗口大小的假设，允许任意遥远地看待过去和未来。

考虑输入序列 xi：。 双 RNN 的工作原理是保持两个单独的状态，SF 和 SB 为每个输入位置 I。*前向状态 SF 是基于 xi，X2，x；而后向状态 s* 是基于 Xn，Xn-i，...，x；。 前向和后向状态由两个不同的 RNN 产生。第一个 RNN(R O 输入序列 xi:n 原样,而第二个 RNN(R O )反向输入序列。 状态表示 s；然后由前向状态和后向状态组成。

位置 i 处的输出基于两个输出向量 y；=[yf；yb]=[0 的级联 (SF)；0 考虑到过去和未来。然后，向量 y；可以直接用于预测，或者作为输入的一部分输入到更复杂的网络中。 当两个 RNN 相互独立运行时，位置 I 的误差梯度将通过两个 RNN 向前和向后流动。图 11 给出了




biRNN 体系结构的可视化表示。

图 11：BiRNN 在句子“棕色狐狸跳

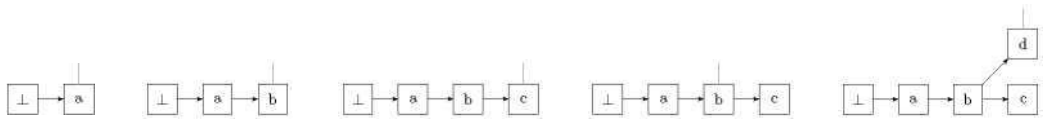
伊索和卡迪（2014）向NLP 社区介绍了使用 biRNN 进行序列标记的方法)。

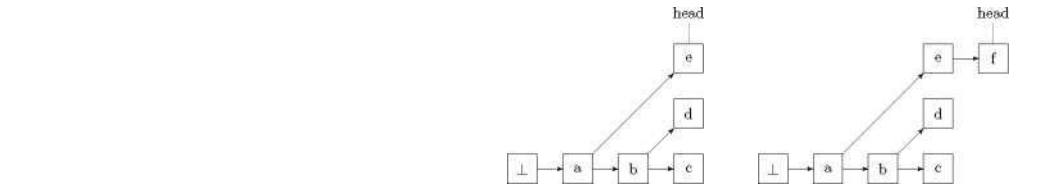
## 10.5 代表堆栈的 RNN

语言处理中的一些算法，包括基于转换的解析算法(Nivre，2008)，需要在堆栈上执行特征提取。 与其局限于查看堆栈的 k 个最顶层元素，RNN 框架可以用来提供整个堆栈的固定大小的向量编码。

主要的直觉是，堆栈本质上是一个序列，因此可以通过取栈元素并将它们按顺序输入到 RNN 来表示堆栈状态，从而对整个堆栈进行最终编码。 为了有效地完成这个计算(没有

每次堆栈更改时执行 O(N)堆栈编码操作)，RNN 状态与堆栈状态一起维护。 如果堆栈是只推送的，这将是微不足道的：每当新元素 x 被推入堆栈时，相应的向量 x 将与 RNN 状态 Si 一起使用，以获得新的状态 Si+i. 处理 POP 操作更具挑战性，但可以通过使用持久堆栈数据结构来解决(Okasaki，1999 年；Goldberg，赵，黄，2013 年)。 持久的或不可变的数据结构在修改时保持旧版本的完整。 持久堆栈构造将堆栈表示为指向链表头部的指针。 空栈是空列表。 推送操作在列表中附加一个元素，返回新头。 然后，POP 操作返回头的父列表，但保持原始列表完整。 从某人持有指向前一个头的指针的角度来看，堆栈没有改变。 后续的推送操作将为同一节点添加新的子节点。 在堆栈的整个生命周期中应用此过程会导致一个树，其中根是一个空栈，从节点到根的每个路径都表示一个中间堆栈状态。 图 12 提供了这样一个树的示例。 同样的过程可以应用于计算图的构造，创建一个具有树结构而不是链结构的 RNN。 然后，从给定节点反向传播错误将影响在创建节点时按顺序参与堆栈的所有元素。 图 13 显示了与图 12 中最后一个状态对应的堆栈-RNN 的计算图。 这种建模方法是由 Dyer 等人独立提出的。 （2015 年）和 Watanabe 和 Sumita（2015 年）用于基于过渡的




依赖分析。

头

图 12： 一种不可变的堆栈结构，用于操作顺序：推 a；推 b；推 c；POP；推 d；POP；POP；推 e；推 f。

流行音

推动 d

## 10.6 阅读文献的注释

不幸的是，通常情况下，从研究论文中阅读其描述来推断确切的模型形式可能是相当具有挑战性的。 模型的许多方面

(8) 推

推动 f

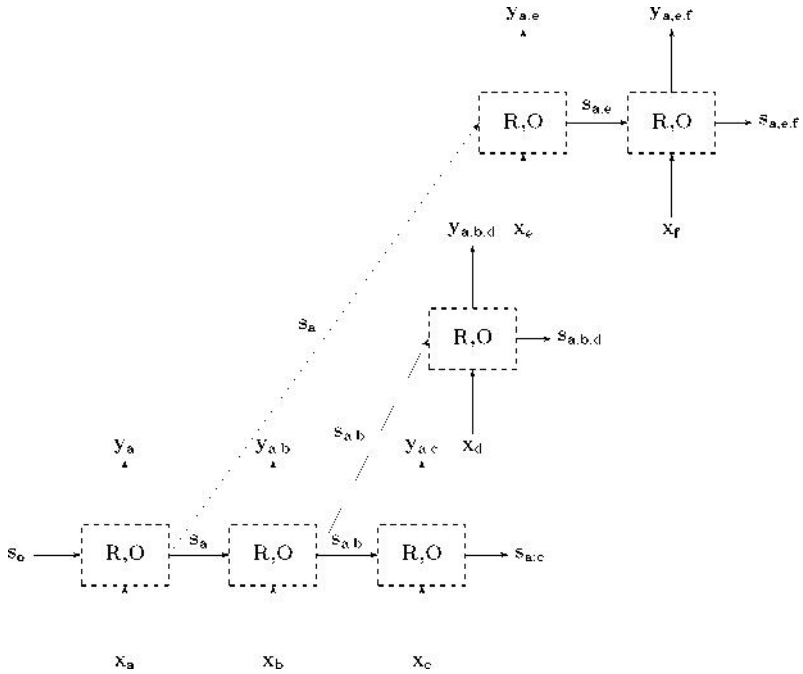



图 13：与图 12 中的最终状态对应的堆栈-RNN。

还没有标准化，不同的研究人员使用相同的术语来指稍微不同的东西。 为了列出几个例子，RNN 的输入可以是一个热向量(在这种情况下，嵌入矩阵是 RNN 的内部)，也可以是嵌入表示；输入序列可以用序列开始和/或序列结束符号填充，或者不填充；虽然 RNN 的输出通常被假定为一个向量，该向量预计将被馈送到附加层，然后是 Softmax 用于预测（如本教程中的介绍），一些论文假设 Softmax 是 RNN 本身的一部分；在多层 RNN 中，“状态向量”可以是最顶层的输出，也可以是来自所有层的输出的级联；当使用编解码器框架时，可以以各种不同的方式解释编码器的输出；等等。 此外，下一节中描述的 LSTM 体系结构有许多小变体，这些变体都是在公共名称 LSTM 下引用的。 其中一些选择在论文中是明确的，另一些需要仔细阅读，而另一些甚至没有提到，或者隐藏在模棱两可的数字或措辞背后。

作为读者，在阅读和解释模型描述时要意识到这些问题。 作为一个作者，也要意识到这些问题：要么用数学符号充分指定您的模型，要么参考一个完全指定模型的不同来源，如果有这样的来源。 如果在不了解详细信息的情况下使用软件包中的默认实现，请明确这一事实，并指定您使用的软件包。 无论如何，在描述你的模型时，不要仅仅依靠数字或自然语言文本，因为这些往往是模棱两可的。

## 11. 混凝土 RNN 结构

我们现在介绍上一节中讨论的抽象 RNN 体系结构的三个不同的实例化，提供了函数 R 和 O 的具体定义。 *这些是简单 RNN(SRNN)、长期短期存储器(LSTM)和门控递归单元(GRU)。*

### 11.1 简单 RNN

最简单的 RNN 公式，称为 Elman 网络或 Simple-RNN(S-RNN)，由 Elman（1990）提出，并由 Mikolov（2012)探索用于语言建模）。 S-RNN 的形式如下：

$$Si=RSRNN(si-i,\ xi)=g(xiW^{x}+Si\_iW^{S}+B)易=OSRNN(SD=SI \tag{38}$$

是的，是的 $_{i}g\quad r^{DS},\quad x_{i}\quad g\quad r^{DX},\ w^{x}\quad g,\quad w^{s}\quad G,\ b\ Gr^{DS}$

也就是说，位置 I 处的状态是输入在位置 I 和先前状态的线性组合，通过非线性激活(通常是 tanh 或 ReLU)。 位置 i 处的输出与该位置中的隐藏状态相同。[71]

尽管简单的 RNN 为序列标记(Xu 等人，2015 年)以及语言建模提供了强大的结果。 关于使用简单的 RNN 进行语言建模的全面讨论，请参阅 Mikolov（2012)的博士论文）。

### 11.2 磅

由于消失梯度问题，S-RNN 很难有效地训练(Pascanu 等人，2012 年)。 在反向传播过程中，序列中稍后步骤中的错误信号（梯度）迅速减小，并且没有到达更早的输入信号，使得 S-RNN 很难捕获远程依赖。 长期短期记忆(LSTM)体系结构(Hochreiter&Schmidhuber，1997)是为了解决消失梯度问题而设计的。LSTM 背后的主要思想是作为状态表示的一部分引入“内存单元”（一个向量），它可以在一段时间内保持梯度。 对存储单元的访问由门控组件控制-模拟逻辑门的平滑数学函数。 在每个输入状态下，使用一个门来决定应该将多少新输入写入内存单元，以及应该忘记内存单元的当前内容。 具体来说，一个门 G[0,1]ⁿ 是范围内的值向量[0,1]它与另一个向量 vGR 按分量乘以ⁿ然后将结果添加到另一个向量中。 g 的值设计为接近 0 或 1，即。 通过使用乙状结肠函数。 在 v 中对应于 g 中的近一值的索引被允许通过，而对应于近零值的索引被阻塞。

在数学上，LSTM 体系结构被定义为：[72]

$$Sj=^{r}LSTM(Sj\_1,\ XJ)=[CJ;\ HJ]$$

$$Cj=Cj\text{-}i\ o\ f+g\ o\ i$$

$$HJ=Tanh(CJ)O$$

$$我=a(x_{j}w^{十一}+h_{j\text{-}1}w^{嗨})$$

71 一些作者将 I 位置的输出视为状态的一个更复杂的函数，例如。 线性变换，或 MLP。 在我们的演示中，输出的这种进一步转换不被认为是 RNN 的一部分，而是作为应用于 RNN 输出的单独计算。

72 这里介绍的 LSTM 体系结构有许多变体。 例如，忘记门不是 Hochreiter 和 Schmidhuber（1997）最初建议的一部分，但被证明是体系结构的重要组成部分。 其他变体包括窥视孔连接和门打字。 有关各种 LSTM 体系结构的概述和全面的经验比较，请参阅 Greff、Srivastava、Koutnik、Steunebrink 和 Schmidhuber（2015)的工作）。

$$f=a(x_j\mathbf{W}^{XF}+h_{j-1}\mathbf{W}^{HF}) \tag{39}$$

不=a(XJW$^{xo}$ +hj_iW$^{何}$g=tanh(xj W$^{xg}$ +hj_$\mathbf{W}^{HG}$)

YJ=Ol STM(S J)=HJ

Sj G R%, $x_i$ G，CJ，HJ，I，f，o，gG ，W$^{xo}$ g气w$^{何}$ g气

符号 O 用于表示组件级乘积。 时间 j 的状态由两个向量 Cj 和 hj 组成，其中 Cj 是内存分量，hj 是隐藏状态分量。 有三个门，i，f 和 o，控制输入，忘记和输出。 门值是基于通过乙状结肠激活函数的当前输入 Xj 和先前状态 hj_1 的线性组合来计算的。 更新候选 g 被计算为 x 的线性组合 $_j$ 和 $h_{j,1}$通过 Tanh 激活函数。 然后更新内存 Cj：忘记门控制要保存的前一个内存的多少(cj_1Of)，输入门控制要保存的拟议更新的多少(gOi)。 最后，根据内存 C 的内容确定 hj(也是输出 yj)的值 $_j$通过 tanh 非线性并由输出门控制。门控机制允许与内存部分 Cj 相关的梯度在非常长的时间范围内保持较高。

关于 LSTM 体系结构的进一步讨论，请参阅 AlexGraves（2008）的博士论文，以及 Olah(2015b)的在线帖子)。有关 LSTM 作为字符级语言模型时行为的分析，请参阅 Karpathy 等人的工作。 （2015）。

有关 LSTM(和 GRU)中门控机制背后的动机及其与解决递归神经网络中消失梯度问题的关系的进一步解释，请参阅 Cho（2015)详细课程说明中的第 4.2 和 4.3 节）。

LSTMS 是目前最成功的 RNN 体系结构类型，它们负责许多最先进的序列建模结果。LSTM-RNN 的主要竞争对手是 GRU，接下来将讨论。

11.2.1 实际考虑

当训练 LSTM 网络时，Jozefowicz 等人。 （2015 年）强烈建议始终初始化遗忘门的偏置项接近一个。 当将辍学应用于具有 LSTM 的 RNN 时，Zaremba 等人。 （2014 年）发现，只有非经常性联系，即非经常性联系，才适用辍学至关重要。 只应用于层之间，而不是序列位置之间。

### 11.3gru

LSTM 体系结构非常有效，但也相当复杂。 系统的复杂性使得它很难分析，而且计算上也很昂贵。门控递归单元(GRU)最近由 Cho 等人引入。(2014 年 b)作为 LSTM 的替代方案。Chung 等人随后证明了这一点。 （2014 年)在几个(非文本）数据集上执行与 LSTM 相当的性能。

与 LSTM 一样，GRU 也是基于门控机制的，但它的门要少得多，没有单独的内存组件。

$$\begin{aligned} SJ=R_{GRU}(S_{J-I},\ X_J)&=(1-z)O\ S_{J-1}+zO\ 帛\\ z&=a(xjW^{xz}+Sj\text{-}iW^{sz})\\ &=a(XJW^{XR}+Sj\text{-}iW^{Sr})\\ Sj&=\tanh(xj\ W^{xs}+(sj\_i)W^{sg}) \end{aligned} \tag{40}$$

YJ=° GRU(sj)=Sj

**Sj，但 G，Xi G，z，r G，W**$^{X}$。g，w$^{S}$。g，

一个门(R)用于控制对前一个状态 Sj-i 的访问，并计算建议的更新 Sj。 然后，根据先前状态 Sj-i 的插值和建议 Sj 确定更新的状态 Sj(也作为输出 yj)，其中插值的比例使用门 z 控制。[73]

GRU 在语言建模和机器翻译方面是有效的。 然而，GRU、LSTM 和可能的替代 RNN 体系结构之间仍存在争议，并对该主题进行了积极的研究。有关 GRU 和 LSTM 体系结构的经验探索，请参阅 Jozefowicz 等人的工作。 （2015）。

### 11.4 其他备选案文 S

LSTM 和 GRU 的门控架构有助于缓解简单 RNN 的消失梯度问题，并允许这些 RNN 捕获跨越长时间范围的依赖关系。 一些研究人员探索比 LSTM 和 GRU 更简单的体系结构，以获得类似的好处。

Mikolov 等人。 （2014 年）注意到矩阵乘法 Si_iW$^{s}$ 再加上简单 RNN 的更新规则 R 中的非线性 g，使得状态向量 Si 在每个时间步长上都发生了很大的变化，从而禁止它在长时间内记住信息。 他们建议将状态向量 Si 分成一个缓慢变化的分量 Ci（“上下文单元”）和一个快速变化的分量 Hi。[74] 慢变化分量 Ci 是根据输入和前一分量的线性插值更新的：Ci=(1-a)xiW$^{x1}$ +aci_i，其中 e（0，1）。 此更新允许 Ci 累积以前的输入。 快速变化的组件 hi 与简单的 RNN 更新规则类似，但也更改为考虑 Ci：hi=a(xiW$^{75x2}$ +hi_iW$^{h}$ +CiW$^{c}$)。 最后，输出 yi 是状态的缓慢和快速变化部分的级联：yi=[c]hi]。 Mikolov 等人。 证明了这种体系结构为语言建模任务中更复杂的 LSTM 提供了竞争的困惑。

米科洛夫等人的方法。 可以解释为约束矩阵 W 的块$^{s}$ 在对应于 Ci 的 S-RNN 中是恒等矩阵的乘法(参见 Mikolov 等人。 （2014 年）。 勒、Jaitly 和 Hinton（2015）提出了一种更简单的方法：将 S-RNN 的激活函数设置为 ReLU，并将偏差 b 初始化为零和矩阵 W$^{s}$ 作为识别矩阵。 这导致未经训练的 RNN 将先前的状态复制到当前状态，添加当前输入 xi 的效果并将负值设置为零。 在将这种初始偏差设置为状态复制之后，训练过程允许 W$^{s}$ 自由地改变。 Le 等人。 证明这种简单的修改使 S-RNN 可以与在几个任务上具有相同数量参数的 LSTM 相媲美，包括语言建模。

## 12. 建模树-递归神经网络

RNN 对于序列建模非常有用。 在语言处理中，使用树结构通常是自然的和可取的。 这些树可以是句法树、语篇树，甚至可以是代表句子各部分表达的情感的树(Socher 等人，2013 年)。我们可能希望基于特定的树节点预测值，基于根节点预测值，或者为完整的树或树的一部分

73 在 GRU 文献中，状态通常被称为 h。

74 我们偏离了 Mikolov 等人的符号。 （2014 年）并重用 LSTM 描述中使用的符号。

75 更新规则与 S-RNN 更新规则不同，也是通过将非线性固定为乙状结肠函数，而不是使用偏置项。然而，这些变化没有作为提案的核心加以讨论。

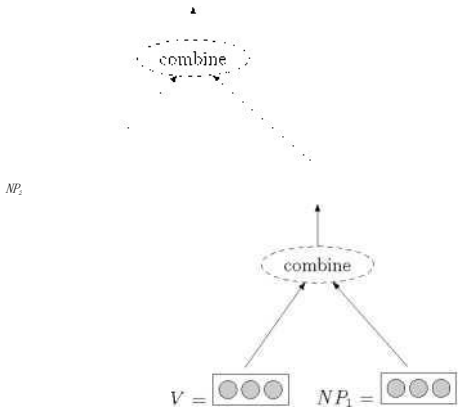
分配质量分数。 在其他情况下，我们可能不直接关心树结构，而是关心句子中跨度的原因。在这种情况下，树只是用作骨干结构，它有助于将序列的编码过程引导到一个固定大小的向量中。

递归神经网络(RecNN)抽象(Pollack，1990)，由 RichardSocher 和他的同事在 NLP 中推广(Socher，Manning，&Ng，2010；Socher，Lin，Ng，&Manning，2011；Socher 等人，2013；Socher，2014)是 RNN 从序列到（二进制）树的推广。[76]

就像 RNN 将每个句子前缀编码为状态向量一样，RecNN 将每个树节点编码为 R 中的状态向量[d]。 然后，我们可以使用这些状态向量来预测相应节点的值，为每个节点分配质量值，或者作为植根于节点的跨的语义表示。

76 虽然用二进制解析树表示，但这些概念很容易转移到一般递归定义的数据结构中，主要的技术挑战是定义一个有效的形式 R，即组合函数。

递归神经网络背后的主要直觉是，每个子树被表示为 d 维向量，节点 p 与子 ci 和 C2 的表示是节点表示的函数：vec(p)=f(vec(ci)，vec(c2)，其中 f 是一个包含两个 d 维向量并返回单个 d 维向量的组合函数。 就像 RNN 状态 Si 被用来编码整个序列 xi：i 一样，与树节



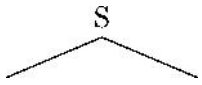

点 p 相关联的 RecN N 状态编码植根于 p 的整个子树。参见图 14 的说明。

图 14：递归神经网络的说明。 将 V 和 NPI 的表示相结合，形成 VP 的表示。 然后将 VP 和 NP2 的表示相结合，形成 S 的表示。

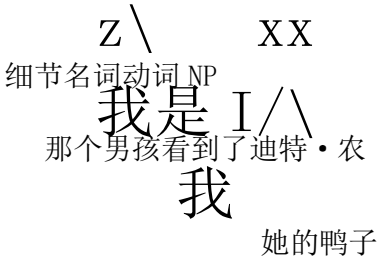
|0。。| vp=|啊

## 12.1 正式定义

考虑在 n 字句子上的二叉树 T。 作为提醒，一个有序的，未标记的树在字符串 xi，...，x 上。可以表示为一组唯一的三重奏(i，k，j)，s.t。 *我<k<j。* 每个这样的三重态表示一个跨越单词 Xi：j 的节点是跨越 Xi：k 和 Xk 的节点的父节点+i：j • 三重态(i，i)对应于树叶上的终端符号(单词 x ‘)。 从未标记的情况到标记的情况，我们可以将树表示为一组 6 个元组(A-B，C，I，k，j)，而 I，k 和 j 表示像以前一样的跨度，A，B 和 C 分别是跨越 x ‘：j，x ‘：k 和 xˆ+i：j 的节点标签。 在这里，叶节点有形式(A-A，A，I，I，I)，其中 A 是预终端符号。 我们把这种元组称为生产规则。 例如，考虑句子“男孩看到她的鸭子”的句法树”。




NP 副总裁




其对应的未标记和标记表示为：

| 无标签 | 有标签 | 相应的斯潘 |
|---|---|---|
| (1,1,1) | （细节，细节，细节，1，1， | xi：我的 |
| (2,2,2 | （名词、名词、名词、2、2） | 男孩 |
| (3,3,3 | （动词、动词、动词、3、3） | 锯子 |
| (4,4,4 | （细节、细节、细节、4、4） | 她 |
| (5,5,5 | （名词、名词、名词、5、5） | 鸭子 |
| (4,4,5 | (NP，Det，名词，4，4，5) | 她的鸭子 |
| (3,3,5 | (VP，Verb，NP，3，3，5) | 看到她的鸭子了 |
| (1,1,2 | (NP，Det，名词，1，1，2) | 那个男孩 |
| (1,2,5 | (s，np，vp，1，2，5) | 男孩看到了她的鸭子 |

通过简单地忽略每个生产规则中的元素(B、C、k)，可以唯一地将上面的生产规则集转换为集合树节点 q Aj(指示跨越 xi：j 的符号 A 的节点。 我们现在能够定义递归神经网络。

递归神经网络(RecN)是一个函数，它以 n 字句 xi、...、x 上的解析树作为输入 $_n$。 句子的每个单词都表示为 d 维向量 xi，树表示为生产规则的集合 T(A、B、C、I、j、k)。 用 qaj 表示 T 的节点。 作为输出，RecNN 返回一组相应的内部状态向量 sA，其中每个内部状态向量 sAe 表示相应的树节点 q Aj，并编码植根于该节点的整个结构。 与序列 RNN 一样，树状 RecNN 是使用函数 R 递归定义的，其中给定节点的内部向量被定义为其直接子节点的内部向量的函数。[77]形式上：

$$\text{RecNN}(x_i, \ldots, x_n) = \{ s_{\text{Aje}} \mid q^{a} e \ T \} \tag{41}$$

77 Le 和 Zuidema（2014）扩展了 RecNN 定义，使得每个节点除了其内部状态向量外，还有一个外部状态向量，表示植根于该节点的子树周围的整个结构。 它们的公式是基于经典的内外算法的递归计算，可以被认为是树 RECN 的双 RNN 对应。 详见勒和祖依德马的作品。

函数 R 通常采取简单的线性变换的形式，它可以也可以不跟随非线性激活函数 g：

$$r(a, b, c, \text{好吧}, {}^{s}k+i: j)=g(s\ Bk; s\ C+i: j]W) \tag{42}$$

这个 R 的公式忽略了树标签，使用相同的矩阵 $WeR^{2dxd}$ 所有的组合。 如果节点标签不存在，这可能是一个有用的公式(例如。 当树不表示具有明确定义的标签的句法结构时)或它们不可靠时。 然而，如果标签是可用的，通常有用的是将它们包含在组合函数中。 一种方法是引入标签嵌入 v(A)，将每个非终端符号映射到 $d_n$ 维向量，并将 R 改为包含组合函数中的嵌入符号：

$$r(a, \text{乙}, \text{丙}, \text{乙}, {}^{s}\text{丙}+i: j)=g({}^{s}Bk; s\ C+i: j; {}^{\text{五}(A)}; {}^{\text{五}(}b) \tag{43}$$

(这里，$W^{\text{二维}+\text{二维}\ XD}$)。 这种方法是由钱、田、黄、刘、朱和朱（2015)采取的）。 另一种方法，由于 Socher 等人。 （2013 年）是根据非终端解开权重，对每个 B、C 对符号使用不同的组合矩阵：[78]

$$R(A, B, C, s\text{好吧}, {}^{s}k+i: j)=g({}^{s}bk; {}^{s}k+i: j]w_{b}{}^{c}) \tag{44}$$

当非终端符号的数量（或可能的符号组合的数量）相对较少时，这个公式是有用的，就像短语结构解析树的情况通常是这样的。 桥本等人也使用了类似的模型。 （2013 年）在语义相关分类任务中编码子树。

### 12.2 扩展和变异

由于上述 R 的所有定义都受到简单 RNN 的梯度消失问题的影响，一些作者试图用长期短期记忆(LSTM)门控结构启发的函数来替换它，从而产生树状的 LSTM(Tai，Socher，&Manning，2015；朱，Sobhani，&Guo，2015b)。 最优树表示问题仍然是一个开放的研究问题，可能的组合函数 R 的广阔空间还有待探索。 关于树结构 RNN 的其他拟议变体包括递归矩阵向量模型(Socher、Huval、Manning 和 Ng，2012 年)和递归神经张量网络(Socher 等人，2013 年)。在第一个变体中，每个单词被表示为向量和矩阵的组合，其中向量像以前一样定义了单词的静态语义内容，而矩阵作为单词的学习“运算符”，允许比连接所隐含的加法和加权平均更微妙的语义组合，然后是线性变换函数。 在第二个变体中，词与向量像往常一样相关联，但组合函数通过基于张量而不是矩阵运算变得更有表现力。

### 12.3 训练递归神经网络

递归神经网络的训练过程遵循与训练其他形式网络相同的方法：定义损失，拼写计算图，使用反向传播计算梯度，以及使用 SGD 训练参数。[79]

关于损失函数，类似于序列 RNN，可以将损失与树的根、任何给定节点或一组节点相关联，在这种情况下，单个节点的损失被组合起来，通常是通过求和。 损失函数是基于标记

78 虽然没有在文献中探索，但一个微不足道的扩展将使变换矩阵也在 A 上。

79 在引入计算图抽象之前，计算 RecNN 中梯度的具体反向传播过程被称为结构反向传播(BPTS)算法(Goller&Kuchler，1996)。

的训练数据，将标签或其他数量与不同的树节点关联起来。

此外，可以将 RecNN 视为编码器，而与节点关联的内部向量则被视为植根于该节点的树的编码。 编码可能对结构的任意属性敏感。 然后将向量作为输入传递给另一个网络。

关于递归神经网络及其在自然语言任务中的应用的进一步讨论，请参阅 RichardSocher（2014)的博士论文）。

## 13. 结论

神经网络是强大的学习者，提供了从非线性分类到序列和树的非马尔可夫建模的机会。 我们希望这一论述有助于 NLP 研究人员将神经网络模型纳入他们的工作中，并利用他们的力量。

## 参考资料


Adel，H.，Vu，N.T.，&Schultz，T.（2013 年）。 将递归神经网络和因子语言模型结合起来进行代码转换语言建模。计算语言学协会第五十一届年会论文集（第 2 卷：短文），第页。 206-211，保加利亚索非亚。 计算语言学协会。

*Ando，R.，&Zhang，T.(2005a)。 一种高性能的文本分块半监督学习方法。 计算语言学协会第 43 届年会论文集(ACL '05)，pp。 1-9，安阿伯，密歇根州。 计算语言学协会。*

Ando，R.K.，&Zhang，T.(2005b)。 从多个任务和未标记数据中学习预测结构的框架。 *机器学习研究杂志，6，1817-1853。*

*Auli，M.，Galley，M.，Quirk，C.，&Zweig，G.（2013 年）。 递归神经网络的联合语言和翻译建模。 《2013 年自然语言处理经验方法会议记录》，第页。 1044-1054，西雅图，华盛顿，美国。 计算语言学协会。*

*Auli，M.，&Gao，J。 （2014）。 递归神经网络语言模型的译码器集成和期望 BLEU 训练。计算语言学协会第五十二届年会论文集（第 2 卷：短文），第页。 136-142，马里兰州巴尔的摩。 计算语言学协会。*

Ballesteros，M.，Dyer，C.，&Smith，N.A。 （2015）。 改进的基于过渡的解析，通过建模字符,而不是单词与 LSTM。在 2015 年自然语言处理经验方法会议记录,pp。349-359，葡萄牙里斯本。 计算语言学协会。

Ballesteros，M.，Goldberg，Y.，Dyer，C.，&Smith，N.A。 （2016）。 与探索训练改善贪婪的堆栈-LSTM 解析器。 *阿希夫：1603.03793[cs]。*

*Bansal，M.，Gimpel，K.，&Livescu，K.（2014 年）。 为依赖解析定制连续字表示。 计算语言学协会第五十二届年会论文集（第 2 卷：短文），第页。 809-815，巴尔的摩，马里兰州。 计算语言学协会。*

Baydin，A.G.，Pearlmutter，B。 A.，Radul，A. A.，Siskind，J.M.（2015 年）。 机器学习中的自动分化：一项调查。 *Ar Xiv：1502.05767[cs]。*

本吉奥，Y。 （2012）。 基于梯度的深层架构训练的实用建议。*Ar Xiv：1206.5533[cs]。*

Bengio，Y.，Ducharme，R.，Vincent，P.，&Janvin，C.（2003 年）。 神经概率语言模型。 *J.马赫。 学习。 第 3、1137-1155 号决议。*

Bengio，Y.，Goodfellow，I.J.，&Cour ville，A。 （2015）。 深入学习。 为麻省理工学院出版社准备的书。

*Bitvai，Z.，&Cohn，T.（2015 年）。 基于深度卷积神经网络的非线性文本回归。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 2 卷：短文），pp。 180-185，北京，中国。 计算语言学协会。*

博塔，J。 A.和布伦森，P.（2014 年）。 词汇表征和语言建模的构成形态。 第 31 届国际机器学习会议记录，北京，中国。 *最佳申请文件*。

Bottou，L.（2012 年）。 随机梯度下降技巧。 在神经网络：贸易的技巧，pp。 421-436. 斯普林格。

*Charniak，E.，&Johnson，M.（2005 年）。粗到精的 n-Best 解析和 Max Ent 判别 Reranking。计算语言学协会第 43 届年会论文集(ACL‘05)，pp。173-180，安阿伯，密歇根州。计算语言学协会。*

*陈，D.，&Manning，C.（2014 年）。一种使用神经网络的快速、准确的依赖解析器。《2014 年自然语言处理经验方法会议记录》，第页。 740-750，卡塔尔多哈。 计算语言学协会。*

*陈，Y，徐，L，刘，K，曾，D，赵，J。 （2015）。 动态多工具卷积神经网络的事件提取。计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。 167176，北京，中国。 计算语言学协会。*

Cho，K.（2015 年）。 具有分布式表示的自然语言理解。ar Xiv：1511.07916[cs，stat]。

*Cho，K.，van Merrienboer，B.，Bahdanau，D.，&Bengio，Y。 (2014a)。 神经机器翻译的性质：编码器-解码器方法。 在 SSST8 的会议记录中，第八次统计翻译中的语法、语义和结构研讨会，pp。 103-111，卡塔尔多哈。 计算语言学协会。*

Cho，K.，van Merrienboer，B.，Gulcehre，C.，Bahdanau，D.，Bougares，F.，Schwenk，H.，&Bengio，Y。 (2014 年 b)。 使用 RNN 编码器-解码器学习短语表示用于统计机器翻译。 《2014 年自然语言处理经验方法会议记录》，第页。 1724-1734，卡塔尔多哈。 计算语言学协会。

Grupala，G.（2014 年）。 用编辑脚本和递归神经嵌入规范推文。 计算语言学协会第五十二届年会论文集（第 2 卷：短文），第页。 680-686，巴尔的摩，马里兰州。 计算语言学协会。

Chung，J.，Gulcehre，C.，Cho，K.，&Bengio，Y。 （2014）。 门控递归神经网络序列建模的经验评价。 *Ar Xiv：“12.3555[cs]。*

Collins，M.（2002 年）。隐马尔可夫模型的判别训练方法：感知器算法的理论与实验。2002 年自然语言处理经验方法会议记录，第页。 1-8. 计算语言学协会。

Collins，M.，&Koo，T.（2005 年）。 自然语言分析中的辨别力。 *计算语言学，31（1），25-70。*

Collobert，R.，&Weston，J。 （2008）。 自然语言处理的统一架构：具有多任务学习的深度神经网络。 在第 25 届机器学习国际会议记录，pp。 160-167. ACM。

Collobert，R.，Weston，J.，Bottou，L.，Karlen，M.，Kavukcuoglu，K.，&Kuksa，P.（2011 年）。 自然语言处理（几乎）从头开始。 *机器学习研究杂志，12,2493-2537。*

克莱默，K，&辛格，Y。 （2002）。 基于多类核的向量机的算法实现。*机器学习研究杂志，2，265-292。*

Creutz，M.，&Lagus，K.（2007 年）。 睡眠分割和形态学学习的无监督模型。 *ACM Trans。语言朗。 过程，4（1），3：1-3：34。*

Cybenko，G.（1989 年）。 乙状结肠函数的叠加近似。 控制、信号和系统数学，2（4），303-314。

Dahl，G.，Sainath，T.，&Hinton，G.（2013 年）。 改进 LVCSR 的深层神经网络，使用校正的线性单元和辍学。在 2013 年 IEEE 声学、语音和信号处理国际会议(ICASSP)，pp。8609-8613.

Dauphin，Y.N.，Pascanu，R.，Gulcehre，C.，Cho，K.，Ganguli，S.，&Bengio，Y。（2014）。高维非凸优化中鞍点问题的识别与攻击。在 Ghahramani，Z.，Welling，M.，Cortes，C.，Lawrence，N.D.，&Weinberger，K.Q。 (ED。 )，《神经信息处理系统的进展》27 页。 2933-2941. Curran Associates，Inc。

*de Gispert，A.，Iglesias，G.，&Byrne，B。 （2015）。 利用神经网络快速、准确地进行 SMT 预排序。 在计算语言学协会北美分会 2015 年会议记录：人类语言技术，pp。1012-1017，丹佛，科罗拉多州。 计算语言学协会。*

*Do、T.、Arti、T.和其他人（2010 年）。 神经条件随机场。 国际人工智能和统计会议，pp。 177-184.*

*董，L，魏，F，谭，C，唐，D，周，M，徐，K.（2014 年）。 自适应递归神经网络对目标相关的 Twitter 情感分类。计算语言学协会第五十二届年会论文集（第 2 卷：短文），第页。 49-54，巴尔的摩，马里兰州。 计算语言学协会。*

*董，L，魏，F，周，M，徐，K.（2015）。 多列卷积神经网络在自由基上的问题解答。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录(第 1 卷:长论文)，pp。 260269，北京，中国。 计算语言学协会。*

多斯桑托斯，C.，&Gatti，M.（2014 年）。 深度卷积神经网络用于短文本情感分析。 在 2014 年 COLING 会议记录中，第 25 届计算语言学国际会议：技术论文，pp。 69-78，爱尔兰都柏林。 都柏林城市大学和计算语言学协会。

*多斯桑托斯，C，翔，B，&周，B。 （2015）。 利用卷积神经网络进行关系分类。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。 626-634，北京，中国。 计算语言学协会。*

多斯桑托斯，C.，&Zadrozny，B。 （2014）。 学习部分语音标记的字符级表示。 《第 31 届国际机器学习会议记录》，第页。 1818-1826.

Duchi，J.，Hazan，E.，&Singer，Y。 （2011）。 在线学习和随机优化的自适应次梯度方法。 *机器学习研究杂志，12，21212159。*

*Duh，K.，Neubig，G.，Sudoh，K.，&Tsukada，H.（2013 年）。 神经语言模型的适应数据选择：机器翻译实验。 计算语言学协会第五十一届年会论文集（第 2 卷：短文），第页。 678-683，保加利亚索非亚。 计算语言学协会。*

*Durrett，G.&Klein，D.（2015 年）。 神经 CRF 解析。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。 302312，北京，中国。 计算语言学协会。*

*Dyer，C.，Ballesteros，M.，Ling，W.，Matthews，A.，&Smith，N.A。 （2015）。 基于过渡的依赖解析与堆栈长期短期内存。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。 334-343，中国北京。 计算语言学协会。*

Elman，J.L.（1990 年）。 寻找时间结构。 认知科学，14（2），179-211。

*Faruqui，M.，&Dyer，C.（2014 年）。 利用多语言相关性改进向量空间词表示。 计算语言学协会欧洲分会第十四届会议论文集，第页。 462-471，瑞典哥德堡。 计算语言学协会。*

*Filippova，K.，Alfonseca，E.，Colmenares，C.A.，Kaiser，L.，&Vinyals，O。 （2015）。用 LSTM 删除句子压缩。 在 2015 年自然语言处理经验方法会议记录，pp。 360-368，葡萄牙里斯本。 计算语言学协会。*

Forcada，M.L.，&Neco，R.P.（1997 年）。 递归异质联想记忆的翻译。 生物和人工计算：从神经科学到技术，pp。 453462.斯普林格。

*Gao，J.，Pantel，P.，Gamon，M.，He，X.，&Deng，L.（2014 年）。 用深度神经网络建模有趣。 《2014 年自然语言处理经验方法会议记录》，第页。 2-13，卡塔尔多哈。 计算语言学协会。*

Gimenez，J.，&Marquez，L.（2004 年）。 SVMTool：一种基于支持向量机的通用 POS 标签生成器。 在第四次 LREC 会议记录，里斯本，葡萄牙。

Glorot，X.，&Bengio，Y。 （2010）。 理解训练深度前馈神经网络的难点。 在国际人工智能和统计会议上，pp。 249-256.

*Glorot，X.，Bordes，A.，&Bengio，Y。 （2011）。 深度稀疏整流神经网络。 国际人工智能和统计会议，pp。 315-323.*

*Goldberg，Y.，&Elhadad，M.（2010 年）。 一种简单的非定向解析算法。 《人类语言技术：计算语言学协会北美分会 2010 年年会》，第页。 742-750，加州洛杉矶。 计算语言学协会。*

Goldberg，Y.，&Levy，O。 （2014）。 解释：推导 Mikolov 等人。 负采样词嵌入方法。

*阿希夫：1402.3722[cs，统计]。*

Goldberg，Y.，&Nivre，J。 （2013）。 训练具有非定论结构的解析器。 *计算语言学协会的交易，1（0），403414。*

*Goldberg，Y.，Zhao，K.，&Huang，L.（2013 年）。 波束搜索增量解析器的有效实现。 计算语言学协会第五十一届年会论文集（第 2 卷：短文），第页。 628-633，保加利亚索非亚。 计算语言学协会。*

Goller，C.，&Kuchler，A。 （1996）。 通过结构反向传播学习任务依赖分布表示。 在检察院。 ICNN-96，pp。 347-352. IEEE。

*Gouws，S.，Bengio，Y.，&Corrado，G.（2015 年）。 Bi1 BOWA：快速双语分布式表示，没有文字对齐。 在第 32 届国际机器学习会议记录，pp。 748-756.*

*格雷夫斯，A。 （2008）。 用递归神经网络监督序列标记。 博士学位。 D.论文，门辰科技大学。*

Greff，K.，Srivastava，R.K.，Koutnik，J.，Steunebrink，B.R.，&Schmidhuber，J。（2015）。 LSTM：搜索空间奥德赛。 *Ar Xiv：1503.04069[cs]。*

Hal Daume III，Langford，J.，&Marcu，D.（2009 年）。 基于搜索的结构化预测。 *机器学习杂志(MLJ)。*

哈里斯，Z。 （1954）。 分配结构。 单词，10（23），146-162。

Hashimoto，K.，Miwa，M.，Tsuruoka，Y.，&Chikayama，T.（2013 年）。 语义关系分类递归神经网络的简单定制。《2013 年自然语言处理经验方法会议记录》，第页。 1372-1376，西雅图，华盛顿，美国。 计算语言学协会。

何，K，张，X，任，S，孙，J。 （2015）。 深入到整流器：超越人级性能的图像网络分类。 *Ar Xiv：1502.01852[cs]。*

Henderson，M.，Thomson，B.，&Young，S.（2013 年）。 对话状态跟踪挑战的深度神经网络方法。 在 SIGDIAL2013 会议记录中，pp。 467-471，Metz，法国。 计算语言学协会。

Hermann，K.M.，&Blunsom，P.（2013 年）。 句法在构成语义向量空间模型中的作用。 计算语言学协会第 51 届年会论文集（第一卷：长论文），pp。 894-904，保加利亚索非亚。 计算语言学协会。

Hermann，K.M.，&Blunsom，P.（2014 年）。 组合分布式语义的多语言模型。 计算语言学协会第 52 届年会论文集（第一卷：长论文），pp。 58-68，巴尔的摩，马里兰州。 计算语言学协会。

Hihi，S.E.，&Bengio，Y。 （1996）。 用于长期依赖的层次递归神经网络。 在图雷茨基，D.S.，Mozer，M.C.，&Hasselmo，M.E。 (ED。 )，《神经信息处理系统的进展》，第 8 页。 493-499. 麻省理工学院出版社。

Hill，F.，Cho，K.，Jean，S.，Devin，C.，&Bengio，Y。 （2014）。 用神经机器翻译嵌

入单词相似性。 *阿希夫：1412.6448[cs]。*

Hinton，G.E.，Srivastava，N.，Krizhevsky，A.，Sutskever，I.，&Salakhutdinov，R.R.（2012 年）。通过防止特征检测器的共适应来改进神经网络。*阿希夫：1207.0580[cs]。*

Hochreiter，S.，&Schmidhuber，J。（1997）。长的短期记忆。*神经计算，9（8），1735-1780。*

Hornik，K.，Stinchcom be，M.，&White，H.（1989 年）。 多层前馈网络是一种通用的逼近器。 *神经网络，2（5），359-366。*

Ioffe，S.，&Szegedy，C.（2015 年）。 批量标准化：通过减少内部协方差移位来加速深度网络训练。 *阿希夫：1502.03167[cs]。*

*Irsoy，O.，&Cardie，C.（2014 年）。 深度递归神经网络的意见挖掘。 《2014 年自然语言处理经验方法会议记录》，第页。 720-728，卡塔尔多哈。 计算语言学协会。*

*Iyyer，M.，Boyd-Graber，J.，Claudino，L.，Socher，R.，&Daume III，H.(2014a)。 一种针对段落进行事实性问题解答的神经网络。 《2014 年自然语言处理经验方法会议记录》，第页。 633-644，卡塔尔多哈。 计算语言学协会。*

*Iyyer，M.，Enns，P.，Boyd-Graber，J.，&Resnik，P.(2014b)。 利用递归神经网络进行政治意识形态检测。 计算语言学协会第 52 届年会论文集（第一卷：长论文），pp。 1113-1122，马里兰州巴尔的摩。 计算语言学协会。*

*Iyyer，M.，Manjunatha，V.，Boyd-Graber，J.，&Daume III，H.（2015）。 文本分类的深层无序成分竞争句法方法。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。 1681-1691，北京，中国。 计算语言学协会。*

*Johnson，R.，&Zhang，T.（2015 年）。 利用卷积神经网络有效地利用词序进行文本分类。在计算语言学协会北美分会 2015 年会议记录：人类语言技术，pp。 103-112，丹佛，科罗拉多州。 计算语言学协会。*

Jozefowicz，R.，Vinyals，O.，Schuster，M.，Shazeer，N.，&Wu，Y。 （2016）。 探索语言建模的极限。 *阿希夫：1602.02410[cs]。*

*Jozefowicz，R.，Zaremba，W.，&Sutskever，I。 （2015）。 递归网络架构的经验探索。在第 32 届国际机器学习会议记录(ICML-15)，pp。 2342-2350.*

Kalchbrenner，N.，Grefenstette，E.，&Blunsom，P.（2014 年）。 模拟句子的卷积神经网络。 计算语言学协会第 52 届年会论文集（第一卷：长论文），pp。 655-665，马里兰州巴尔的摩。 计算语言学协会。

Karpathy，A.，Johnson，J.，&Li，F.-F.（2015 年）。 可视化和理解递归网络。 *Ar Xiv：1506.02078[cs]。*

*金，Y。 （2014）。 用于句子分类的卷积神经网络。 《2014 年自然语言处理经验方法会议记录》，第页。 1746-1751，卡塔尔多哈。 计算语言学协会。*

Kim，Y.，Jernite，Y.，Sontag，D.，&Rush，A.M.（2015 年）。 字符-软件神经语言模型。 *ar Xiv：1508.06615[cs，stat]。*

金马，D.，&Ba，J。 （2014）。 亚当：随机优化方法。 *阿希夫：1412.6980[cs]。*

Krizhevsky，A.，Sutskever，I.，&Hinton，G.E.（2012 年）。 深度卷积神经网络图像网分类。 在 Pereira，F.，Burges，C.J.C.，Bottou，L.，&Weinberger，K.Q。 (ED。 )，《神经信息处理系统的进展》，第 25 页。 1097-1105. Curran Associates，Inc。

Kudo，T.，&Matsumoto，Y。 （2003）。 基于核的文本分析的快速方法。 第 41 届计算语言学协会年会论文集-第 1 卷，ACL '03 页。 24-31，Stroudsburg，PA，USA。 计算语言学协会。

Lafferty，J.，Mc Callum，A.，&Pereira，F.C.（2001 年）。 条件随机字段：分割和标记序列数据的概率模型。 在 ICML 诉讼中。

Le，P.，&Zuidema，W.（2014 年）。 依赖解析的内外递归神经网络模型。 《2014 年自然语言处理经验方法会议记录》，第页。 729-739，卡塔尔多哈。 计算语言学协会。

*Le，P.，&Zuidema，W.（2015 年）。 森林卷积网络：具有神经图和无二值化的组合分布语义。在 2015 年自然语言处理经验方法会议记录，pp。 1155-1164，葡萄牙里斯本。计算语言学协会。*

Le，Q.V.，Jaitly，N.，&Hinton，G.E.（2015）。 一种简单的初始化递归线性单元网络的方法。 *Ar Xiv：1504.00941[cs]。*

Le Cun，Y.，&Bengio，Y。 （1995）。 图像、语音和时间序列的卷积网络。 在 Arbib，M.A。 (艾德。 )，《大脑理论与神经网络手册。 麻省理工学院出版社。

Le Cun，Y.，Bottou，L.，Orr，G.，&Muller，K.(1998a)。 有效的后援。 在奥尔，G，&K，M。 (ED。 )，神经网络：交易的技巧。 斯普林格。

Le Cun，Y.，Bottou，L.，Bengio，Y.，&Haffner，P.(1998b)。 基于梯度的模式识别学习。 *IEEE 会议记录，86（11），2278-2324。*

Le Cun，Y.，Chopra，S.，Hadsell，R.，Ranzato，M.，&Huang，F.（2006 年）。 关于基于能量的学习的教程。 *预测结构化数据，1，0。*

Le Cun，Y.，&Huang，F.（2005 年）。 能量模型判别训练的损失函数。 在 AISTATS 会议记录中。 卫星。

Lee，G.，Flowers，M.，&Dyer，M.G.（1992 年）。 学习概念知识的分布式表示及其在基于脚本的故事处理中的应用。 在 Connectionist 自然语言处理，pp。 215-247. 斯普林格。

*Levy，O.，&Goldberg，Y。 (2014a)。 基于依赖的 Word 嵌入。 计算语言学协会第五十二届年会论文集（第 2 卷：短文），第页。 302-308，巴尔的摩，马里兰州。 计算语言学协会。*

Levy，O.，&Goldberg，Y。 (2014 年 b)。 神经词嵌入作为隐式矩阵分解。 在 Ghahramani，Z.，Welling，M.，Cortes，C.，Lawrence，N.D.，&Weinberger，K.Q。 (ED。 )，《神经信息处理系统的进展》27 页。 2177-2185. Curran Associates，Inc。

Levy，O.，Goldberg，Y.，&Dagan，I。 （2015）。 改进分布相似性与从 Word 嵌入中吸取的经验教训。 *计算语言学协会的交易，3（0），211-225。*

Lewis，M.，&Steedman，M.（2014 年）。 改进的 CCG 解析与半监督监督监督。*计算语言学协会的交易，2（0），327-338。*

*李，J，李，R，&Hovy，E.（2014 年）。 话语解析的递归深层模型。 《2014 年自然语言处理经验方法会议记录》，第页。 2061-2069 年，卡塔尔多哈。 计算语言学协会。*

*Ling，W.，Dyer，C.，Black，A.W.，&Trancoso，I。 (2015a)。 单词 2Vec 对语法问题的简单适应。 在计算语言学协会北美分会 2015 年会议记录：人类语言技术，pp。 1299-1304，丹佛，科罗拉多州。 计算语言学协会。*

Ling，W.，Dyer，C.，Black，A.W.，Trancoso，I.，Fermandez，R.，Amir，S.，Marujo，L.，&Luis，T.(2015b)。形式发现函数：开放词汇词汇表示的组合字符模型。在 2015 年自然语言处理经验方法会议记录，pp。 1520-1530，葡萄牙里斯本。 计算语言学协会。

*刘，Y，魏，F，李，S，冀，H，周，M，王，H.（2015）。 关系分类的依赖神经网络。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 2 卷：短文），pp。 285-290，北京，中国。 计算语言学协会。*

Luong，M.-T.，Le，Q.V.，Sutskever，I.，Vinyals，O.，&Kaiser，L.（2015）。 多任务序列到序列学习。 *ar Xiv：1511.06114[cs，stat]。*

马，J，张，Y，朱，J。 （2014）。 标记网络：用神经网络构建一个健壮的 Web 标记。计算语言学协会第 52 届年会论文集（第一卷：长论文），pp。 144-154，巴尔的摩，马里兰州。 计算语言学协会。

*马，M，黄，L，周，B，和翔，B。 （2015）。 基于依赖的卷积神经网络用于句子嵌入。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 2 卷：短文），pp。 174-179，中国北京。 计算语言学协会。*

Mc Callum，A.，Freitag，D.，&Pereira，F.C.（2000 年）。 信息提取和分割的最大熵马尔可夫模型。。 在 ICML 中，第一卷。 17 页。 591-598.

Mikolov，T.，Chen，K.，Corrado，G.，&Dean，J。 （2013）。 向量空间中字表示的有效估计。 *阿希夫：1301.3781[cs]。*

Mikolov，T.，Joulin，A.，Chopra，S.，Mathieu，M.，&Ranzato，M.（2014 年）。 在递归神经网络中学习更长的记忆。 *Ar Xiv：1412.7753[cs]。*

*Mikolov，T.，Karafiat，M.，Burget，L.，Cernocky，J.，&Khudanpur，S.（2010）。基于递归神经网络的语言模型。。 在 IN TERSPEECH2010，第 11 届国际语音通信协会年会，Makuhari，千叶，日本，2010 年 9 月 26 日至 30 日，pp。 1045-1048.*

*Mikolov，T.，Kombrink，S.，Lukas Burget，Cernocky，J.H.，&Khudanpur，S.（2011 年）。递归神经网络语言模型的扩展。 声学，语音和信号处理(ICASSP)，2011 年 IEEE 国际会议，pp。 5528-5531. IEEE。*

Mikolov，T.，Sutskever，I.，Chen，K.，Corrado，G.S.，&Dean，J。 （2013）。 词和短语的分布表示及其组成。Burges，C.J.C.，Bottou，L.，Welling，M.，Ghahramani，Z.，&Weinberger，K.Q。(ED。)，《神经信息处理系统的进展》，第 26 页。3111-3119. Curran Associates，Inc。

Mikolov，T.（2012 年）。 基于神经网络的统计语言模型。 博士学位。 D.论文，博士。 D.论文，布尔诺理工大学。

Mnih，A.，&Kavukcuoglu，K.（2013 年）。学习单词嵌入有效的噪声估计。Burges，C.J.C.，Bottou，L.，Welling，M.，Ghahramani，Z.，&Weinberger，K.Q。 (ED。 )，《神经信息处理系统的进展》，第 26 页。 2265-2273. Curran Associates，Inc。

*Mrksic，N.，O Seaghdha，D.，Thomson，B.，Gasic，M.，Su，P.-H.，Vandyke，D.，Wen，T.-H.，&Young，S.（2015）。 利用递归神经网络进行多域对话状态跟踪。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 2 卷：短文），pp。 794-799，北京，中国。 计算语言学协会。*

Neidinger，R.（2010 年）。 自动微分和 MATLAB 面向对象编程导论。SIAM Review，52（3），545-563。

内斯特洛夫，Y。 （1983）。 收敛速度 O(1/k2)的凸规划问题的求解方法)。 在苏联数学 Doklady，第一卷。 27，pp。 372-376.

*内斯特洛夫，Y。 （2004）。 关于凸优化的介绍性讲座。 Kluwer 学术出版社。*

*Nguyen，T.H.，&Grishman，R.（2015 年）。 卷积神经网络的事件检测与领域适应。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 2 卷：短文），pp。 365-371，北京，中国。 计算语言学协会。*

尼夫，J。 （2008）。 确定性增量依赖解析算法。 计算语言学，34（4），513-553。

*Okasaki，C.（1999 年）。 纯粹的功能数据结构。* 剑桥大学出版社，英国剑桥；纽约。

Olah ， C.(2015a) 。 计 算 图 上 的 微 积 分 ： 反 向 传 播 。 从 http://colah.github.io/posts/2015-08-Backprop/。检索

Olah ， C.(2015b) 。 了 解 LSTM 网 络 。 从 http://colah. 检 索 github.io/posts/2015-08-Understanding-LSTM/。

Pascanu，R.，Mikolov，T.，&Bengio，Y。（2012）。论递归神经网络训练的难点。*Ar Xiv：1211.5063[cs]。*

*裴，W，葛，T.，&Chang，B。 （2015）。 基于图的依赖解析的有效神经网络模型。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。 313-322，北京，中国。 计算语言学协会。*

彭，J，博，L，徐，J。（2009）。条件神经场。Bengio，Y.，Schuurmans，D.，Lafferty，J.D.，Williams，C.K.I.，&Culotta，A。 (ED。 )，《神经信息处理系统的进展》，第 22 页。 1419-1427. Curran Associates，Inc。

*彭宁顿，J.，Socher，R.，&Manning，C.（2014 年）。手套：全球向量的文字表示。《2014 年自然语言处理经验方法会议记录》，第页。 1532-1543，卡塔尔多哈。 计算语言学协会。*

波拉克，J。 b. （1990）。 递归分布式表示。 人工智能，46,77-105。

Polyak，B.T.（1964 年）。 一些加快迭代方法收敛的方法。 苏联计算数学和数学物理，4（5），1-17。

*钱，问，田，B，黄，M，刘，Y，朱，X，&朱，X。 （2015）。 学习递归神经网络中的标签嵌入和特定于标签的组合函数。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。 1365-1374，北京，中国。 计算语言学协会。*

荣，X。 （2014）。 Word2vec 参数学习解释。 ar Xiv：1411.2738[cs]。

鲁默哈特，D.E.，Hinton，G.E.，&Williams，R.J。 （1986）。 通过反向传播错误来学习表示。 自然，323（6088），533-536。

Schuster，M.，&Paliwal，K.K.（1997 年）。 双向递归神经网络。 *IEEE 信号处理事务，45（11），2673-2681。*

Schwenk，H.，Dchotte，D.，&Gauvain，J.-L.（2006 年）。 统计机器翻译的连续空间语言模型。 在主要会议海报会议的会议记录中，pp。 723-730. 计算语言学协会。

Shawe-Taylor，J.，&Cristianini，N.（2004 年）。 *模式分析的核方法。* 剑桥大学出版社。

*史密斯，纽约。 （2011）。 语言结构预测。*人类语言技术综合讲座。 摩根和克莱普尔。

*Socher，R.（2014 年）。 自然语言处理和计算机视觉的递归深度学习。 博士学位。 D.论文，斯坦福大学。*

*Socher，R.，Bauer，J.，Manning，C.D.，&Ng，A.Y。 （2013）。 用组合向量语法进行解析。 计算语言学协会第 51 届年会论文集（第一卷：长论文），pp。 455-465，保加利亚索非亚。 计算语言学协会。*

*Socher，R.，Huval，B.，Manning，C.D.，&Ng，A.Y。 （2012）。 通过递归矩阵-向量空间的语义构成。 《2012 年自然语言处理和计算自然语言学习经验方法联席会议论文集》，第页。 1201-1211，韩国济州岛。 计算语言学协会。*

*索彻，R，林，C。 C.-Y.，Ng，A.Y.，&Manning，D.（2011 年）。 用递归神经网络解析自然场景和自然语言。 在 Getoor，L.，&Scheffer，T。 (ED。 )，第 28 届国际机器学习会议记录，ICML2011，Bellevue，Washington，USA，June28-July2，2011，pp。129-136. 巫女。*

Socher，R.，Manning，C.，&Ng，A。 （2010）。 利用递归神经网络学习连续短语表示和句法分析。 在{NIPS}2010 年深度学习和无监督特征学习研讨会论文集，pp。 1-9.

Socher，R.，Perelygin，A.，Wu，J.，Chuang，J.，Manning，C.D.，Ng，A.，&Potts，C.

（2013 年）。 情感树库语义组合的递归深层模型。《2013 年自然语言处理经验方法会议记录》，第页。 1631-1642，西雅图，华盛顿，美国。 计算语言学协会。

*Sogaard，A.，&Goldberg，Y。 （2016）。 深度多任务学习，低层次任务监督在较低层次。计算语言学协会第 54 届年会论文集（第 2 卷：短文），pp。 231-235. 计算语言学协会。*

*Sordoni，A.，Galley，M.，Auli，M.，Brockett，C.，Ji，Y.，Mitchell，M.，Nie，J.-Y.，Gao，J.，&Dolan，B。 （2015）。 一种神经网络方法，用于上下文敏感的会话响应生成。 在计算语言学协会北美分会 2015 年会议记录：人类语言技术，pp。 196-205 年，科罗拉多州丹佛市。 计算语言学协会。*

*Sundermeyer，M.，Alkhouli，T.，Wuebker，J.，&Ney，H.（2014 年）。 双向递归神经网络的翻译建模。 《2014 年自然语言处理经验方法会议记录》，第页。 14-25，卡塔尔多哈。 计算语言学协会。*

Sundermeyer，M.，Schluter，R.，&Ney，H.（2012 年）。 语言建模的 LSTM 神经网络。。在 INTERSPEECH。

Sutskever，I.，Martens，J.，Dahl，G.，&Hinton，G.（2013 年）。 论初始化和动量在深度学习中的重要性。在第 30 届机器学习国际会议记录(ICML-13)，pp。1139-1147.

*Sutskever，I.，Martens，J.，&Hinton，G.E.（2011 年）。用递归神经网络生成文本。在第 28 届国际机器学习会议记录(ICML-11)，pp。 1017-1024.*

Sutskever，I.，Vinyals，O.，&Le，Q.V.V.（2014 年）。 用神经网络进行序列学习。 在 Ghahramani，Z.，Welling，M.，Cortes，C.，Lawrence，N.D.，&Weinberger，K.Q。(ED。)，《神经信息处理系统的进展》27 页。 3104-3112. Curran Associates，Inc。

Tai，K.S.，Socher，R.，&Manning，C.D.（2015 年）。 从树结构的长期短期记忆网络中改进语义表示。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。 1556-1566，北京，中国。 计算语言学协会。

*Tamura，A.，Watanabe，T.，&Sumita，E.（2014 年）。 词对齐模型的递归神经网络。 计算语言学协会第 52 届年会论文集（第一卷：长论文），pp。 1470-1480，马里兰州巴尔的摩。 计算语言学协会。*

Telgarsky，M.（2016）。 深度在神经网络中的好处。 Ar Xiv：1602.04485[cs，stat]。

T.Tieleman&Hinton，G.（2012 年）。 第 6.5-Rms Prop：将梯度除以其最近大小的运行平均值。 机器学习的神经网络。

Van de Cruys，T.（2014 年）。 选择偏好获取的神经网络方法。 《2014 年自然语言处理经验方法会议记录》，第页。 26-35，卡塔尔多哈。 计算语言学协会。

*Vaswani，A.，Zhao，Y.，Fossum，V.，&Chiang，D.（2013 年）。 用大尺度神经语言模型进行解码可以改善翻译。 《2013 年自然语言处理经验方法会议记录》，第页。 1387-1392，西雅图，华盛顿，美国。 计算语言学协会。*

Wager，S.，Wang，S.，&Liang，P.S.（2013 年）。 辍学培训作为适应性正规化。 Burges，C.J.C.，Bottou，L.，Welling，M.，Ghahramani，Z.，&Weinberger，K.Q。 (ED。 )，《神经信息处理系统的进展》，第 26 页。 351-359. Curran Associates，Inc。

Wang，M.，&Manning，C.D.（2013 年）。 序列标注中非线性深层结构的影响。。 在 IJCNLP，pp。 1285-1291.

*王，P，徐，J，徐，B，刘，C，张，H，王，F，郝，H.(2015a)。 短文本分类的语义聚类和卷积神经网络。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 2 卷：短文），pp。 352-357，北京，中国。 计算语言学协会。*

*王，X，刘，Y，孙，C，王，B，&王，X。 (2015 年 b)。 通过计算具有长期短期内存的 Word 嵌入来预测推特的极性。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。 1343-1353，中国北京。 计算语言学协会。*

*Watanabe，T.，&Sumita，E.（2015）。 基于过渡的神经成分解析。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。1169-1179，北京，中国。 计算语言学协会。*

*Weiss，D.，Alberti，C.，Collins，M.，&Petrov，S.（2015）。 神经网络转换解析的结构化训练。计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。 323-333，北京，中国。 计算语言学协会。*

韦博斯，P.J。 （1990）。 时间的反向传播：它做什么和如何做。。 IEEE 会议记录，78（10），1550-1560。

Weston，J.，Bordes，A.，Yakhnenko，O.，&Usunier，N.（2013 年）。 用嵌入关系提取模型连接语言和知识库。 《2013 年自然语言处理经验方法会议记录》，第页。 1366-1371，西雅图，华盛顿，美国。 计算语言学协会。

*徐，W，Auli，M，&Clark，S.（2015）。 用递归神经网络进行 CCG 超标记。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录(第 2 卷:短文),pp。250-255，北京，中国。 计算语言学协会。*

尹，W.，&Schutze，H.（2015）。 用于解析识别的卷积神经网络。 在计算语言学协会北美分会 2015 年会议记录：人类语言技术，pp。 901-911，丹佛，科罗拉多州。 计算语言学协会。

Zaremba，W.，Sutskever，I.，&Vinyals，O。 （2014）。 递归神经网络正则化。 *ar Xiv：1409.2329[cs]*。

Zeiler，M.D.（2012 年）。ADADELTA：一种自适应学习速率方法。阿希夫：1212.5701[cs]。

*曾，D，刘，K，赖，S，周，G，赵，J。 （2014）。 通过卷积深度神经网络进行关系分类。在 2014 年 COLING 会议记录中,第 25 届计算语言学国际会议:技术论文,pp。2335-2344，爱尔兰都柏林。 都柏林城市大学和计算语言学协会。*

*张，Y，&Weiss，D.（2016）。 堆栈传播：改进语法的表示学习。 计算语言学协会第 54 届年会论文集（第 1 卷：长论文），pp。 1557-1566. 计算语言学协会。*

*周，H，张，Y，黄，S，陈，J。 （2015）。 基于过渡关联解析的神经概率结构预测模型。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。 1213-1222，北京，中国。 计算语言学协会。*

朱，C，邱，X，陈，X，黄，X。 (2015a)。 递归卷积神经网络依赖型排序器模型。 计算语言学协会第 53 届年会和第 7 届国际自然语言处理联合会议记录（第 1 卷：长论文），pp。 1159-1168，北京，中国。 计算语言学协会。

朱，X，Sobhani，P.，&郭，H.(2015b)。树结构上的长期短期记忆。*Ar Xiv：1503.04881[cs]*。

58. https://github.com/clab/cnn