学校代码： 10246
学　　号： 14110240008

博 士 学 位 论 文

（学术学位）

# 自然语言处理中的神经表示学习

# Neural Representation Learning for Natural Language Processing

院　　系： 计算机科学与技术学院

专　　业： 计算机应用与技术

姓　　名： 刘鹏飞

指 导 教 师： 黄萱菁 (教授) 邱锡鹏 (副教授)

完 成 日 期： 2019 年 8 月 30 日

# 摘　要

表示学习（representation learning），又称表征学习，是指将输入数据转化成适用于机器学习形式的过程。通常地，机器学习的性能依赖于对数据表示的选择，一个好的表示可以使得模型对输入数据进行更好的理解。近年来，神经网络的兴起，使得我们可以自动地对输入数据进行特征抽取。这极大推动了表示学习的发展，并给我们带来了进一步探究的可能性。

一般地，表示学习的研究可以按照不同角度进行划分：从学习方式上，可以分为有监督学习和无监督学习；从输入数据模态上，可以分为文本表示、图像表示以及语音表示；从共享独立性上，可以分为共享表示和私有表示。在自然语言处理中，使用深度学习技术（即深度神经网络）对文本进行表示学习已经成为一个很有价值的研究方向。本文工作围绕着以下问题展开：1）**对于不同粒度的文本（词语、句子、句对），如何设计合理的结构，使得模型可以学习到适合最终任务的表示？** 深度学习的到来使得自然语言处理中的研究工作由原来的*特征工程* (feature engineering) 过渡到了现在的*结构工程* (architecture engineering)，而对于文本的表示学习，首先要解决的最基本问题就是寻找合适的*归纳偏置* (inductive bias)，使得模型可以更好地对输入文本进行编码。而本文分别针对不同粒度的文本信号，进行相应的网络结构探索，希望找到更适合下游任务的结构偏置。2）**如何进行针对性的迁移学习？** 有针对性地进行迁移是指我们要对迁移的知识“按需分配”，这就要求我们学习的知识应该具备*可迁移性*，此外，我们还要对已有的知识进行*可理解分析*，从而可以*分离*我们真正需要的知识，最终实现知识的定向迁移。对于以上两个亟待解决的问题，本文通过两个方面，九个章节进行递进式探讨，其贡献总结如下：

一方面，对于不同粒度文本的表示学习，本文分别探索了最适合下游任务的归纳偏置，并且利用这些归纳偏置设计新的模型，这些模型在主流的数据集上都取得了当时最好的效果。

1. **词语**: 词语表示学习的研究是深度神经网络技术最先触及的领域。经典的基于神经网络的分布式语义表示学习方法可以将任意词映射到一个低维的向量空间，然而这种表示往往与上下文独立，无法处理一词多义的现象。针对这个问题，我们提出了融入“主题”信息的神经张量词语表示学习模型，该模型最大的特点是可以学习到与上下文相关的词表示，从而缓解一词多义现象带来的语义消歧问题。

2. **句子**：基于深度神经网络的句子表示学习是一个重要的研究任务。在句子建模的任务里，我们的研究围绕着三个问题展开：如何建模包含习语的句子？如何解决语义合成的多样性与函数单一性导致的网络表示能力不足的问题？如何动态学习句子的结构而不是预先指定？针对以上问题，我们分别提出了基于树结构的自适应语义合成网络、动态语义合成网络、和基于图的语境化网络。这些模型分别引入了不同的并且适用于当前任务的结构偏置。

3. **句对**：句对的表示学习在自然语言处理中有很广泛的应用场景，如语义匹配，自动问答等。解决这个任务的关键在于如何建模两个句子之间复杂的交互关系。这里我们提出了一种基于多维长短时记忆网络的学习框架，可以建立两个句子之间强交互关系。

另一方面，我们提出学习具有特殊性质的文本表示，这为我们实现针对性迁移做了铺垫。具体说来，我们通过利用对抗学习（adversarial learning）以及元学习（meta learning）方式，探索了如何学习具有可迁移性、可分离性，可理解性的文本表示。

1. **可迁移性**：深度学习技术不仅可以自动提取出有用的特征，它的另一个迷人之处在于可以对已经学习好的特征进行迁移学习。本文以循环神经网络为原型，提出了三种适用于文本序列可迁移性学习的框架。

2. **可分解性**：一个好的表示应该可以结构化，并且按照功能属性进行分离，这样我们才能更好地进行迁移使用。本文中，为了将不同任务之间共享和私

有的特征实现分离，我们将对抗学习的思想引入到多任务学习中，该模型可以实现对共享空间的净化，实现共享-私有特征的正交分离。

3. **可理解性**：很多时候，深度模型取得好结果是以牺牲我们对模型的理解能力为代价的。那么对于学习到的表示，如何对学习的知识进行可理解分析？本文通过动态建立图神经网络实现了一种可理解模型的学习。

**关键词**：深度学习；语义表示学习；自然语言处理；归纳偏置；知识迁移

# ABSTRACT

Representation learning refers to the transformation of input data into a form suitable for machine learning. Usually, the performance of machine learning depends on the choice of data representation. A good representation can make the model understand the input data better. In recent years, the rise of neural networks has enabled us to automatically extract features from input data, which has greatly promoted the development of representation learning and brought us various possibilities . Generally speaking, the research on representation learning can be divided into supervised and unsupervised learning; text representation, image representation and speech representation; shared representation and private representation from the perspective of sharing independence.

In Natural Language Processing, the use of deep learning technology (deep neural networks) for text representation has become a valuable research direction. In this thesis, our study makes a feasible step towards the answers of the following questions: 1) **For different granularities of texts (words, sentences, sentence pairs), how to design a reasonable inductive bias so that the model can learn the suitable representation for downstream tasks?** The arrival of deep learning leads to the shift from feature engineering to present architecture engineering. For text representation learning, the most fundamental problem is to find the appropriate structure bias, so as to better encode the input text signal. In this thesis, we explored the representation learning of different granularities of text spans, in order to find a more suitable structure bias for the specific task. 2) **How can we only transfer knowledge we need?** In order to achieve this goal, the knowledge learned should be *transferable* first, since what we really want to achieve is to transfer whatever knowledge we need. Towards this end, we also need to carry out *interpretable analysis* on knowledge, and then *disentangle* them to take out the part which we really need for the task we care about.

With the above two questions in mind, this thesis conducts a progressive discussion through two parts and nine chapters. On the one hand, to learn the representations of different granularities of texts, we have explored the most suitable inductive bias for different tasks and utilize them to re-design our models.

1. **Words**: The word representation learning is the first area touched by deep learning. Existing methods for learning word representation can map arbitrary words into a low-dimensional vector space. However, this representation is usually context-independent, which suffers from the problem of word polysemy. To address this problem, we present a model called neural tensor skip-gram network for context-dependent word representation learning, which could make full use of external "*topic*" information.

2. **Sentences**: Sentence representation learning based on deep neural networks is an important research task. A good sentence representation can benefit many downstream tasks. Here, our research focuses on three questions: how to learn the representation of a sentence consisting of idioms? how to model the richness of compositionality residing in phrases? How to learn sentence structure dynamically instead of specifying it beforehand? We answer the above questions by proposing three models: adaptive compositional networks (utilizing a semantic gate to automatically judge the compositionality), dynamic tree-structure neural networks (introducing a meta-network to generate parameters) and contextualized non-local neural networks (combining graph neural networks with Transformer).

3. **Sentence-pair**: Sentence-pair encoding is widely used in Natural Language Processing, such as semantic matching, question answering and so on. In this thesis, we approach the key step of this task by modeling the strong interaction between two sentences by coupled long short-term memory networks.

On the other hand, we propose to learn the text representations with specific prop-

erties: transferable, disentangled, and interpretable.

1. **Transferable**: Deep learning can not only allow us to automatically extract useful features but also has another fascinating property that the learned features can be transferred to other tasks. So how to transfer the text representation in Natural Language Processing is discussed in this thesis. Specifically, we take recurrent neural networks as the prototype, and design three frameworks for multi-task learning, in which shared features among different tasks are learned in diverse ways.

2. **Disentangled**: A good representation should be structured and disentangled based on some functional attributes so that we can transfer it better. In this thesis, we integrate adversarial learning with multi-task learning, which allows us to purify the shared feature space, avoiding the negative transferring problem.

3. **Interpretable**: Often, the good results of deep neural networks come at the expense of our understanding of models. So, how to make an interpretable analysis of the learning knowledge? This thesis proposes an interpretable multi-task learning method by dynamically constructing the graph neural networks.

**Key Words**: Deep learning; Semantic representation learning; Natural language processing; Inductive bias; Knowledge transfer

# 目　录

第 1 章　绪论 …… 1
1.1　研究背景和意义 …… 1
1.2　国内外研究现状 …… 2
1.2.1　不同粒度的文本表示学习 …… 2
1.2.2　特殊性质的文本表示学习 …… 4
1.3　本文研究内容结构 …… 5
第 2 章　神经表示学习概况 …… 8
2.1　发展历史 …… 8
2.2　基本研究问题 …… 9
2.3　下游任务 …… 10
2.4　基本模型 …… 10
2.4.1　卷积神经网络 …… 11
2.4.2　循环神经网络 …… 11
2.4.3　树结构神经网络 …… 13
2.4.4　图结构神经网络 …… 13
2.4.5　注意力网络 …… 15
2.5　数据集 …… 16

**第一部分　不同粒度的文本表示学习　19**

第 3 章　词表示学习 …… 20
3.1　引言 …… 20
3.1.1　上下文无关词表示 …… 20
3.1.2　上下文相关词表示 …… 20
3.2　研究动机 …… 21
3.3　基于神经网络的词表示学习 …… 22
3.4　基于 Skip-Gram 的神经张量网络 …… 23
3.4.1　张量分解 …… 24
3.4.2　相关模型与特例分析 …… 25
3.4.3　训练 …… 26
3.5　实验与分析 …… 27
3.5.1　最近邻实验 …… 27

3.5.2 上下文词相似性分析 · · · · · 28
3.5.3 文本分类 · · · · · 30
3.6 本章小结 · · · · · 31
第 4 章 语义合成问题探究：从词表示到句子表示学习 · · · · · 33
4.1 引言 · · · · · 33
4.2 句子中习语现象的建模 · · · · · 33
4.2.1 研究动机 · · · · · 33
4.2.2 习语的语言学解读 · · · · · 34
4.2.3 模型 · · · · · 36
4.2.4 iSent: 富含习语的情感分类数据集 · · · · · 39
4.2.5 实验 · · · · · 41
4.2.6 实验设置 · · · · · 41
4.2.7 对比模型 · · · · · 42
4.3 丰富语义组合关系的建模 · · · · · 46
4.3.1 研究动机 · · · · · 46
4.3.2 模型 · · · · · 47
4.3.3 基于递归神经网络的动态合成网络 · · · · · 48
4.3.4 基于 TreeLSTM 结构的动态合成网络 · · · · · 49
4.3.5 动态组合网络的应用 · · · · · 49
4.3.6 实验与分析 · · · · · 50
4.4 语义组合拓扑结构的动态学习 · · · · · 56
4.4.1 研究动机 · · · · · 56
4.4.2 模型 · · · · · 58
4.4.3 实验与分析 · · · · · 60
4.5 本章小结 · · · · · 66
第 5 章 句对表示学习 · · · · · 67
5.1 引言 · · · · · 67
5.2 耦合长短时记忆网络 · · · · · 68
5.2.1 松耦合长短时记忆网络 · · · · · 69
5.2.2 紧耦合长短时记忆网络 · · · · · 70
5.2.3 两个模型的对比分析 · · · · · 70
5.3 句子匹配的端到端架构 · · · · · 71
5.3.1 输入层 · · · · · 71
5.3.2 堆叠耦合-LSTMs 层 · · · · · 72

5.3.3 池化层 ····· 73
5.3.4 输出层 ····· 73
5.4 训练 ····· 73
5.5 实验与分析 ····· 74
5.5.1 超参数和训练 ····· 74
5.5.2 对比模型 ····· 75
5.5.3 实验一：文本蕴含识别 ····· 75
5.5.4 实验二：问答匹配 ····· 78
5.6 本章小结 ····· 79

**第二部分　特殊性质的表示学习　80**

第 6 章　可迁移性表示学习 ····· 81
6.1 引言 ····· 81
6.2 面向文本分类的循环神经网络 ····· 82
6.3 基于 RNN 的多任务学习共享模型 ····· 83
6.4 训练 ····· 85
6.5 实验与分析 ····· 86
6.5.1 超参数和训练 ····· 86
6.5.2 多任务训练的效果 ····· 87
6.5.3 与其他神经模型进行比较 ····· 88
6.5.4 案例分析 ····· 89
6.5.5 错误分析 ····· 91
6.6 本章小结 ····· 91
第 7 章　可分离性表示学习 ····· 92
7.1 引言 ····· 92
7.2 基于循环神经网络的文本分类 ····· 93
7.3 文本分类的多任务学习 ····· 94
7.3.1 两个用于句子建模的共享方案 ····· 95
7.3.2 任务特有输出层 ····· 95
7.4 对抗训练 ····· 96
7.4.1 对抗网络 ····· 96
7.4.2 多任务学习的任务对抗损失 ····· 97
7.4.3 最终的损失函数 ····· 98

7.5　实验与分析 ····· 98
7.5.1　数据集 ····· 98
7.5.2　多任务学习对比方法 ····· 99
7.5.3　超参数 ····· 99
7.5.4　性能评估 ····· 100
7.5.5　共享知识迁移 ····· 101
7.5.6　可视化分析 ····· 102
7.6　本章小结 ····· 105
第 8 章　可理解性表示学习 ····· 106
8.1　引言 ····· 106
8.2　用于多任务通信的消息传递框架 ····· 107
8.2.1　信息传递 ····· 109
8.2.2　任务相关的图层 ····· 111
8.3　实验与分析 ····· 112
8.3.1　超参数 ····· 112
8.3.2　文本分类 ····· 112
8.3.3　序列标记 ····· 115
8.4　讨论与定性分析 ····· 115
8.5　本章小结 ····· 117
第 9 章　总结与展望 ····· 119
9.1　本文的总结 ····· 119
9.2　对未来工作展望 ····· 120
参考文献 ····· 122
致谢 ····· 132
在读期间发表的学术论文与取得的研究成果 ····· 134

# 图 目 录

1.1 神经语义表示学习过程示意图 ····· 1

1.2 不同年份涌现出的方法在文本分类上的准确率变化趋势 ····· 2

1.3 论文组织结构 ····· 7

3.1 Skip-Gram, TWE-1 和 NTSG 结构图 ····· 21

3.2 神经张量网络可视化 ····· 24

3.3 NTSG-2 的二维主题词向量 ····· 32

4.1 自适应语义合成神经网络 ····· 35

4.2 习语感知组合网络的子模块结构 ····· 36

4.3 不同长度的评论数量的分布 ····· 41

4.4 不同模型在 iSent 数据集上的性能 ····· 43

4.5 不同结点处的情感类别的变化 ····· 44

4.6 树结构的神经网络 ····· 46

4.7 动态语义合成神经网络 ····· 48

4.8 DC-TreeNN 匹配网络 ····· 51

4.9 DC-RecNN 在不同向量 $z$ 下的准确率 ····· 53

4.10 DC-TreeNN 模型 $\mathbf{z}_{21}$ 和 $z_{27}$ 神经元的行为热力图 ····· 56

4.11 句子对应动态图的更新过程 ····· 57

4.12 (a) 句法依存树 (b-c) 不同模型学习到的依存图 ····· 64

5.1 四种典型的基于 LSTM 架构句对建模结构 ····· 69

5.2 用于句对编码的耦合 LSTM 的体系结构 ····· 72

5.3 两个可解释的神经元和一些词语对的可视化 ····· 77

6.1 用多任务学习建模文本的三种架构 ····· 84

6.2 不同时间步情感分数的变化 ····· 90

7.1 任务 A 和任务 B 的两种共享方式 ····· 93

7.2 两种用于多任务学习的架构 ····· 94

7.3 对抗共享-私有模型 ····· 96

7.4 使用预训练共享 LSTM 层的两种转移策略 ····· 102

7.5 (a) 不同时间步情感分数的变化；(b) 神经元的行为 ····· 104

8.1 多任务学习的不同拓扑结构 ····· 107

8.2 多任务交互的两种框架 ····· 109

8.3 相邻 “任务” 的查找 ····· 116

8.4 共享层在不同任务下捕获的学习模式 ····· 118

# 表 目 录

2.1 文本分类数据集的统计信息 ····· 17

2.2 序列标注数据集的统计 ····· 18

3.1 我们的模型和对比模型 Skip-Gram 的最近邻词 ····· 28

3.2 不同模型在 SCWS 数据集上的性能 ····· 29

3.3 多类别文本分类评估的结果 ····· 29

4.1 习语的主要属性和相应的挑战 ····· 35

4.2 形态和词汇方面的习语变体示例 ····· 40

4.3 iSent 数据集中习语和句子的统计数据 ····· 41

4.4 不同模型在四个主流数据集上的准确率 ····· 43

4.5 不同模型在 iSent 数据集上的准确率 ····· 43

4.6 不同类型的非组合短语示例 ····· 45

4.7 动态合成神经网络的超参数设置 ····· 52

4.8 不同模型在五个数据集上的准确率 ····· 53

4.9 不同模型在 SICK 数据上的准确率 ····· 53

4.10 多个可解释的神经元以及这些神经元捕捉到的词语/短语 ····· 55

4.11 不同模型在十个数据集上的性能 ····· 61

4.12 不同模型在 Chunking，NER，POS 三个任务上的性能 ····· 62

4.13 不同任务生成的多个可解释子结构 ····· 65

5.1 耦合 LSTM 模型的超参数设置 ····· 74

5.2 不同模型在 SNLI 语料库上的准确率 ····· 76

5.3 多个可解释的神经元以及具体实例 ····· 77

5.4 雅虎问答对数据集上的结果 ····· 78

6.1 统一层架构的实验结果 ····· 87

6.2 耦合层架构的实验结果 ····· 87

6.3 共享层架构的实验结果 ····· 88

6.4 已有方法与本文的基于共享模式方法的实验结果对比 ····· 89

7.1 16 个数据集的统计数据 ····· 100

7.2 我们的模型与典型基线模型在 16 个数据集上的错误率对比 ····· 101

7.3 我们的模型与标准多任务学习在 16 个数据集上的错误率对比 ····· 103

7.4 在 Movie 和 Baby 任务中不同模式下学习到的特征 ····· 104

8.1 我们的模型以及经典基线模型在 16 个数据集上的文本分类错误率 ··· 113

8.2 不同模型在 Chunking、NER 和与 POS 的 F1 值 ····· 115

8.3 由共享层学到的多个可解释的子结构 ····· 117

# 术 语 说 明

| 术语 | 缩写 | 说明 |
| --- | --- | --- |
| attention mechanism | – | 注意力机制 |
| bag-of-word | BOW | 词袋模型 |
| compositional function | – | 合成函数 |
| contextualized | – | 语境化的 |
| convolutional neural network | CNN | 卷积神经网络 |
| feature map | – | 特征映射 |
| filter | – | 滤波器 |
| graph-structure neural network | GraphNN | 图结构神经网络 |
| graph convolutional neural network | GCN | 图卷积神经网络 |
| inductive bias | – | 归纳偏置 |
| interaction | – | 交互 |
| locality bias | – | 局部偏置 |
| long short-term memory network | LSTM | 长短时记忆网络 |
| meta-network | – | 元网络 |
| multi-layer perceptron | MLP | 多层感知器 |
| multi-task learning | MTL | 多任务学习 |
| neural tensor skip-gram | NTSG | 神经张量跳词 |
| natural language processing | NLP | 自然语言处理 |
| one-hot vector | – | 独热向量 |
| peephole | – | 窥视孔 |
| pooling | – | 池化 |
| recurrent neural network | RNN | 循环神经网络 |
| self-attention mechanism | – | 自注意力机制 |
| skip-gram | – | 跳词 |
| structural bias | – | 结构偏置 |
| tree-structure LSTM | TreeLSTM | 树结构长短时记忆网络 |
| tree-structure neural network | TreeNN | 树结构神经网络 |
| transition function | – | 转移函数 |

# 第 1 章　绪　　论

## 1.1　研究背景和意义

深度学习的兴起给自然语言处理（NLP）的发展提供了许多重要的机会和挑战。其中，自然语言处理中的神经表示学习问题，即用深度神经网络将离散的文本信号转化为连续的低维向量的过程（如图1.1所示），已经成为基础而重要的研究课题。

尽管表示学习有着广泛的研究范畴，例如，有/无监督表示学习、单/多语言学习、单/多模态学习等。对自然语言处理应用而言，一个基础的问题是**如何选择合适结构的神经网络对不同粒度的文本信号进行表示学习**。该问题决定了许多下游任务性能的高低。例如，高质量的词表示可以使词性标注、命名实体识别等任务得到更好的初始训练状态，从而提高最终的性能。而句子表示则影响了文本分类、情感分析、意图分类等经典 NLP 任务的性能。由此可见，NLP 中许多任务最终可以转化为某种粒度文本表示学习的问题。

此外，当我们已经对词、短语、句子有了成型的研究策略之后，下一个要考虑的问题是：**如何学习到满足特殊性质的文本表示？**比如：可迁移性、可分离性以及可理解性。该研究的意义在于它拓展了我们学习到表示的使用边界。本质上，现有的不少工作（例如，多任务学习、跨语言学习和迁移学习等）可以看成某类特殊性质的表示学习问题。

从深度学习在自然语言处理的发展上来看，学习好不同粒度的文本表示是


**图 1.1　神经语义表示学习过程示意图**

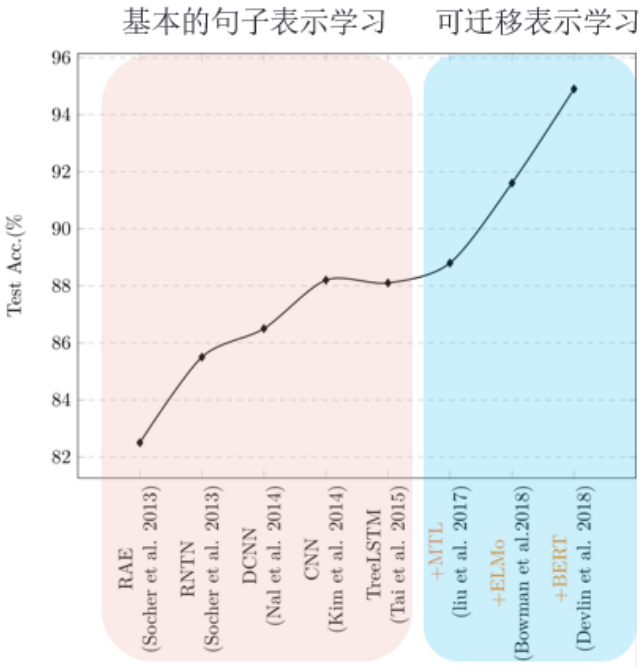



**图 1.2　不同年份涌现出的方法在文本分类上的准确率变化趋势**

学习满足特殊性质表示的基础。这两个研究问题都影响着下游任务的性能。这里我们做出了一个真实的案例统计，如图1.2, 纵坐标表示某个文本分类任务上的准确率，横坐标表示在不同年份涌现出的取得很好性能的方法。我们可以观测到：1）句子表示的方法多种多样，并且效果有所差异，合适的网络结构可以学习到更好的表示，从而获取更高的准确率。**这揭示了选择合适结构的神经网络对不同粒度的文本信号进行表示学习具有重要意义**；2）另一方面，当模型结构改变带来的提升遇到瓶颈时，通过引入可迁移的表示，可以进一步提升任务的性能，这反映了**学习满足特殊性质文本表示的意义**。

## 1.2　国内外研究现状

### 1.2.1.　不同粒度的文本表示学习

#### (1)　词表示的研究

基于神经网络的词表示研究最早可以追溯到神经语言模型[1]。该工作提供了一个可以将离散的文本信号连续化的训练框架。Minh 等人[2] 又对上述模型进行了训练速度上的优化。近些年，word2vec[3] 工作的出现，将词向量的发展推

向了一个高潮。然而，以上这些工作都基于这样一个假设：每一个词会被唯一地分配一个实值向量。该假设忽略了一词多义的语言现象。例如，“**苹果**”，既可以指一种水果，也可以表示公司的名字。最近，越来越多人关注该现象。Reisinger 等人[4] 提出使用多个稀疏且高维的向量表示词语的方法。Huang 等人[5] 通过引入全局上下文信息来扩展 Reisinger 等人的方法，并使用神经网络来学习多个稠密的低维词向量。这两种方法都依赖于对目标词的上下文进行聚类，而这种学习方式训练开销很大。Tian 等人[6] 从概率角度建模有多个义项的词语，并将其与高效的跳词（Skip-Gram）模型整合起来。Neelakantan 等人[7] 提出将多个跳词（Skip-Gram）联合起来进行语义消歧和词表示学习。这些模型中大多数是通过聚类方式生成不同义项的表示，但忽略词语之间的复杂关联以及它们的上下文信息。为了解决这个问题，Liu 等人[8] 引入了主题信息到词向量学习中，提出三个直观的模型，以增强词表示学习的能力。

**(2)　句子表示的研究**

句子表示研究的核心在于如何选择合适拓扑结构的神经网络去建模文本序列。这里，神经网络提供了一种通过参数化合成函数来建模不同词语之间依赖关系的有效方法。常见的合成函数包括循环神经网络，卷积神经网络[9]（Convolutional Neural Networks, CNN）和树结构神经网络 (Tree-structure Neural Networks, TreeNN)[10,11]，图神经网络 (Graph-structure Neural Networks, GraphNN)[12]。这些方法之间存在两个主要差异：一个是词语相互作用的范围（Scope）。另一个是它们所使用的合成函数形式。例如，长短时记忆网络能够显式地建模当前词语与先前词语之间的依赖关系，并且词语可以与同一窗口中的其它词语交互。句子建模的过程是一个复杂的语义合成过程，在这里有很多复杂的语言现象，比如“**习语现象**”，以前与习语相关的工作主要集中在它们的识别上，具体分为两种类型：习语类型分类[13,14] 和习语检测[15–19]。但是这些工作都没有把习语考虑在句子层面的理解上。还有其它的现象比如语义组合的多样性等。最近，有一些工作探索各种类型短语的组合性[20–23]，然而这些工作都没有考虑组合本身是多样的。

以上这些现象的存在给现有的神经网络模型带来了许多挑战，比如端到端

求导问题。为了能有效建模融入各种语言现象的神经网络模型，我们需要结合具体的任务寻求合适的结构偏置。

#### (3)　句对表示的研究

句对表示的研究核心在于如何建模两个句子之间的交互（interaction）关系。为了解决这个问题，直观的方法是计算两个句子的所有词语或短语之间的相似性。Socher 等人[24] 首先将这种方法应用在复述检测（Paragraph Identification）任务上。Wan 等人[25] 使用长短时记忆网络来增强两个句子之间的词语或短语的位置上下文交互。该方法通过预定义的相似性指标来捕获两个句子的相互作用。它们的不足之处在于难以给网络增加深度。Rock 等人[26] 使用两个有注意力机制的长短时记忆网络来捕获两个句子之间的交互。这种结构对于两个句子是不对称的，其中所获得的最终表示对两个句子的顺序敏感。

### 1.2.2.　特殊性质的文本表示学习

#### (1)　可迁移性表示

不同任务之间可以通过神经网络进行共享表示学习。基于神经网络的多任务学习已被证明在许多 NLP 问题中有效[27,28]。Liu 等人[28] 使用输入词的共享表示，并在一个框架内解决不同的 NLP 任务。该类模型的缺点在于共享的网络层过于简单（只共享一个查找表，其它查找表和网络层是和任务相关的），这样很容易使得任务之间产生负面的影响。由此，对于多个任务之间可迁移性表示学习的难点在于，如何构造合适的共享机制，让不同任务充分交换信息。

#### (2)　可分离性表示

大多数现有的关于多任务学习的工作[29,30] 试图仅仅基于是否应该共享某些组件的参数，将不同任务的特征划分为私有和共享空间。在这里，一个隐含的假设是共享空间里只包含共享的知识。然而，大部分情况下真实的训练结果无法保证两个空间特征的正交性。也就是说，共享空间也会包含大量的私有特征，这样的结果会降低表示的可迁移性。

**(3)　可理解性表示**

在过去的模型里，任务之间共享知识往往是不能理解的。为了朝着可理解表示学习的方向迈进一步，一个简单的思路就是先把表示结构化，即结构化地去建立任务之间的关系。在基于神经网络的模型爆发之前，已经有很多**非神经多任务学习**方法来建模任务之间的关系。例如，Bakker 等人[31] 学习使用贝叶斯方法对任务进行聚类。Kim 等人[32] 利用给定的树结构设计正则化器，而 Chen 等人[33] 学习在给定图上的结构化多任务问题。这些模型采用复杂的学习策略，并在不同任务之间引入先验信息，这些模型通常不适用于文本序列建模。这里存在的挑战在于如何结构化共享知识，使得模型学习到的知识具备可理解性。

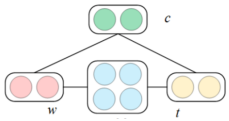
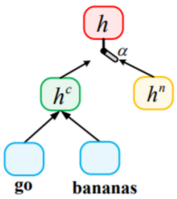
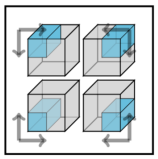
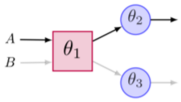
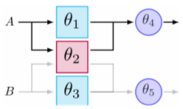
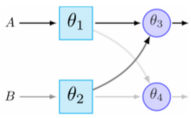
## 1.3　本文研究内容结构

本文的研究内容沿着文本表示学习的两个方面展开，从不同粒度的文本语义表示问题（词、句子、句对），到语义表示的可共享性、可分离性、可理解性的研究。全文共分为两个部分，九个章节，其中第一章是绪论；第二章是表示学习的基本概况；第三、四、五章为第一部分：探究不同粒度的文本表示学习存在的核心问题以及解决方案；第六、七、八章节为第二部分：探究了表示学习中的可迁移性、可分离性、以及可理解性；第九章为总结与展望。图1.3给出了本文的组织结构，其具体内容如下：

- 第一章是绪论，介绍了本文的研究背景意义，相关工作以及研究结构。
- 第二章对神经表示学习基本概况做了介绍，包括发展历史、涉及任务、基本模型等，为后续展开做了铺垫。
- 第三章研究了词语向量学习中的一词多义问题。过去工作对词向量学习过程中，大多忽略了一词多义的语言现象。为了解决这个问题，文本通过引入“主题”这个外部信息，去学习一种和上下文相关的词表示，它可以有效地解决一词多义的问题。
- 第四章介绍了在使用词进行句子表示学习时遇到的几个经典问题：1）如何

建模包含习语的句子？为了解决这个问题，我们提出了一种在树结构上对句子语义进行合成的方法，并且引入一个语义开关，可以自适应地对当前输入的短语进行判断（是否为习语），从而分发给不同的编码组件（习语编码器或字面理解编码器）。2）如何解决单一的语义合成函数参数和多变的语义合成场景之间的矛盾？我们借用元学习的思路，提出了一种可动态生成参数的语义合成网络。3）如何突破句子建模中先验结构的假设？现有模型进行句子表示学习，都是提前给句子结构进行了一个假设，比如，序列结构，树结构等等。这种假设限制了模型本身的表达能力，为此，我们提出了一种可以动态学习结构的网络模型。

- 第五章研究了句对建模中的核心问题，即如何建模句子之间的交互关系？过去模型都选择使用了一些简单的弱交互模型，本文首先将经典的 LSTM 模型拓展到多维，然后使用多维的耦合 LSTM 对句对进行一种强交互的编码。
- 第六章探究了如何使用神经网络去设计多个（相关）任务共同训练的框架。现有的工作很少有针对序列问题多任务学习框架。我们探讨三种不同的多任务学习框架，他们的区别在于任务之间的交互机制不同。我们通过选用 LSTM 作为基础模型，设计了“嵌入层共享机制”，“耦合共享机制”，“循环层共享机制” 三个模型，三个模型在表示的可迁移性学习上各有优劣。
- 第七章探究了多个任务共同训练时，如何构建一个共享空间，保证该共享空间只保存任务之间的共享信息，而不被任务特有的信息所污染。我们通过引入对抗训练来实现共享-私有空间的正交分离。
- 第八章探究了如何使得我们对学习到的共享特征更加可理解。我们把多任务学习问题转化成一个图（Graph）中任务之间的通信问题。并且图中不同结点（Node）之间的权重可以动态学习。这种数据驱动的方式可以增加我们对模型的可理解性。
- 第九章是全文总结以及未来展望。

| | | 提出模型<br>（解决的问题） | 归纳偏置 | 章节 |
|---|---|---|---|---|
| 不同粒度文本 | 词 |  <br>（Q1） | 引入外部主题信息 | 第3章 |
| | 句子 |  <br>（Q2） | 引入树型或者图型结构 | 第4章 |
| | 句对 |  <br>（Q3） | 引入多维交互机制 | 第5章 |
| 特殊性质 | 可迁移 |  <br>（Q4） | 不同任务信息可共享假设 | 第6章 |
| | 可分离 |  <br>（Q5） | 共享私有空间正交先验 | 第7章 |
| | 可理解 |  <br>（Q6） | 引入任务之间图结构 | 第8章 |

Q1：如何利用“主题”信息解决一词多义现象给词表示学习带来的困难？

Q2-1：如何利用深度神经网络处理习语这种特殊的语言现象？

Q2-2：如何用一套全局共享的网络参数去建模丰富的语义合成问题？

Q2-3：过去模型建模句子都是事先假设好一个结构，如何突破这个假设？

Q3：给定两个句子，如何建模他们之间的强交互关系？

Q4：如何用神经网络去设计多个相关任务共同学习的框架？

Q5：多个相关任务共同训练的时候，如何保证共享空间不被私有信息污染？

Q6：我们可以使得不同任务之间共享的信息具有可理解性吗？

**图 1.3　论文组织结构**

# 第 2 章　神经表示学习概况

神经表示学习，即通过使用神经网络将输入的离散信号转变成连续的实值向量的过程。特别地，对自然语言的处理，这里的输入信号特指文本信息。下面，我们先了解神经表示学习在自然语言处理领域的发展背景。

## 2.1　发展历史

### (1)　词语刻画从离散形式到连续形式

Bengio 等人[1] 提出神经语言模型。具体地，给定一个训练集合，其包含若干个词语组成的序列：$w_1, \cdots, w_T$ （$\mathbf{w}_t \in V$），这里 $V$ 是由训练集构建的词表。对于语言模型，我们的优化目标是学习一个好的模型使得出现序列的联合概率最大：$f(w_t, \cdots, w_{t-n+1}) = \hat{P}(w_t|w_t^{t-1})$。所谓的神经语言模型是指将上述函数 $f(\cdot)$ 通过神经网络参数化：

$$f(i, w_{t-1}, \cdots, w_{t-n+1}) = g(i, C(w_{t-1}), \cdots, C(w_{t-n+1}), \theta) \tag{2.1}$$

其中 $g(\cdot)$ 是神经网络，$C(i)$ 表示第 $i$ 个词语的特征。最后通过优化序列出现概率的最大似然可以学习到网络的参数：

$$L = \frac{1}{T}\sum_t f(w_t, w_{t-1}, \cdots, w_{t-n+1}) + R(\theta) \tag{2.2}$$

其中 $R(\theta)$ 是正则项。这篇文章的最大贡献在于做了以下开创性的研究：1）通过使用神经网络将离散的词转变成连续的实值向量。2）通过语言模型构造了一种无监督学习模式。3）分析了模型的训练瓶颈，为后续研究提供了重要启示。

### (2)　大规模语料上的词表示学习

Mikolov 等人[3,34] 提出了一种快速训练词表示的学习方式，即使在很大规模的数据集上，也可以无监督地学习出高质量的词向量。具体地，这个工作提出了负采样、层次化编码等策略，大大优化了词表示的学习过程。这个工作基本的思想是我们希望学习到一个好的词表示，以便可以用它成功地预测出它周边的词

是什么。形式化的优化目标可以简化如下：

$$L = \frac{1}{T}\sum_{t=1}^{T}\sum_{-c<j<c,j\neq 0} logp(w_{t+j}|w_t) \tag{2.3}$$

其中 $c$ 是目标词 $w_t$ 所在的上下文窗口大小。该模型大大推动了词向量的发展，因为它使得我们在大规模语料上训练模型变成了可能。正是这些词向量的使用，为后来深度神经网络在自然语言处理上的发展做了铺垫。

**(3) 简单语义合成函数**

尽管卷积神经网络和循环神经网络早在 90 年代就已经被提出，但把模型成功地使用到具体任务上的工作往往更容易被记住，并且加速这个领域的发展。Collobert 等人[27] 最早想到，神经网络不仅可以对词进行表示学习，还可以构造参数化的函数对更粗粒度的文本进行学习，而这样的函数被称为**合成函数**，这篇论文引入卷积神经网络对词语表示进行合成，从而得到短语和句子的表示。类似地，Sutskever 等人[35] 最早把循环神经网络成功应用到了翻译任务中，这也是神经机器翻译中标志性的事件之一。

**(4) 复杂交互机制**

Bahdanau 等人[36] 提出注意力机制（Attention mechanism），在很多场景下，当表示学习的对象很复杂时，简单的语义合成函数往往表示能力不足。注意力机制的提出，不仅提高了原有模型的表示能力，而且提高了模型工作机制的可理解性。

## 2.2 基本研究问题

尽管神经表示学习的研究范畴很广泛，但是在自然语言处理任务中，神经表示学习最基本的研究问题是对于不同粒度的文本进行学习研究。只有对词进行很好的建模之后，我们才能更好完成短语以及句子的建模。具体说来，自然语言处理中，神经语义表示学习最先要解决的是如何使用**合适的**模型学习词语、短语、句子、句对的表示。这里合适是指匹配于当前数据分布特点以及当前任务的归纳偏置（*inductive bias*）或结构偏置（*structural bias*）。

## 2.3　下游任务

虽然神经语义表示学习是个和任务无关（task-agnostic）的研究问题，但一个好的表示学习方法往往可以使得下游任务受益。下面给出表示学习在自然语言处理中涉及的常见任务。

**(1)　文本分类**

文本分类任务指给定一段文本，从预定义的类别中，预测出标签。具体说来我们将文本序列表示为 $X = \{x_1, x_2 \ldots, x_T\}$，输出为 $Y$。在文本分类中，$Y$ 是单个标签。而我们的目标是让神经网络来估计条件概率 $P(Y|X)$。文本分类任务常见的形式有：情感分析、垃圾邮件过滤、问题分类、意图分类等。

**(2)　序列标注**

在序列标注中，给定一段文本，我们需要对文本中每一个词都进行标签预测。这里输出标签 $Y = \{y_1, y_2, \ldots y_T\}$ 是一个序列。序列标注常见的任务形式有中文分词、词性标注、命名实体识别等。

**(3)　语义关系判断**

该任务旨在判断两个句子的语义关系。通常地，给定两段文本，$X^{(1)} = \{x_1, x_2 \ldots, x_T\}$ 和 $X^{(2)} = \{x_1, x_2 \ldots, x_T\}$，模型需要从指定的类别集合里预测出一个标签。常见的具体任务有：自然语言推理，问题-答案对匹配。

## 2.4　基本模型

针对2.2提出的基本研究问题，过去几年涌现了很多神经网络结构，它们可以对不同粒度的文本进行表示学习。为了更好地对这些模型进行理解，我们提出使用**局部偏置假设**（**locality bias**）去统一理解各种模型。在这里，局部偏置是一种归纳偏置，即局部依赖存在于相邻词之间。它首先在语音学中[37]，然后在自然语言处理中进行探索[38,39]。一般地，不同的模型有着不同的局部偏置假设。

### 2.4.1. 卷积神经网络

卷积神经网络 (Convolutional Neural Networks), 简称 CNN，由 LeCun 等人[40]提出。最初，CNN 用于计算机视觉，随后被验证了它对 NLP 也有效果，并且在语义分析[41]、查询检索[42]、和句子建模[9] 等方面取得了优异的成果。

卷积神经网络的局部偏置在于，事先指定一个窗口，并且窗口内部的词可以进行相互交互。形式化地，给定一个序列 $x = x_1, \cdots, x_t$，那么每个词 $x_i$ 通过卷积核 $\mathbf{w}$ 进行卷积操作映射成一个隐层向量。例如，特征 $h_i$ 可以通过词窗 $x_{i:i+d-1}$ 而生成：

$$\mathbf{h}_i = f(\mathbf{w} \cdot x_{i:i+d-1} + b), \tag{2.4}$$

这里 $\mathbf{w}$ 是卷积核参数，$d$ 表示窗口大小，$f$ 表示非线性函数，$b \in \mathbb{R}$ 表示偏置项。此过滤器 $\mathbf{w}$ 将应用于句子 $x$ 中的每个可能的词语窗口，从而生成特征映射（feature map）。

$$\mathbf{h} = [\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_{n-d+1}], \tag{2.5}$$

其中 $\mathbf{h} \in \mathbb{R}^{n-d+1}$。一般来说，获得特征映射后会通过最大池化操作（max-over-time pooling），将最大值 $\hat{\mathbf{h}} = max\{\mathbf{h}\}$ 作为该特定滤波器 $\mathbf{w}$ 对应的特征。这种池化操作能自然地处理可变句子长度。

以上描述了从一个滤波器中提取一个特征的过程。通常来说，为了获得多个特征，需要使用多个不同的滤波器（即具有不同窗口大小）。这些特征将传递到全连接层以及 Softmax 层，其输出的是标签上的概率分布。

常见的卷积神经网络有 Kalchbrenner 等人[9] 提出的动态池化卷积网络，Lai 等人[43] 提出的循环卷积神经网络，和 Chen 等人[44] 提出的多池化卷积神经网络。

### 2.4.2. 循环神经网络

循环神经网络（Recurrent Neural Networks) 简称 RNN，该类模型的局部偏置假设在于：假设句子是时序的序列，自左至右（或自右至左）依次编码。具体地，循环神经网络能够递归地将**转移函数**（transition function）应用于其输入序列的内部隐藏状态向量 $h_t$ 来处理任意长度的序列。在 $t$ 时刻隐藏状态的激活可看作

是当前输入 $\mathbf{x}_t$ 和前一个隐藏状态 $\mathbf{h}_{t-1}$ 的函数：

$$\mathbf{h}_t = \begin{cases} 0 & t = 0 \\ f(\mathbf{h}_{t-1}, \mathbf{x}_t) & \text{其它} \end{cases} \tag{2.6}$$

通常使用状态到状态转换函数 $f$ 进行元素非线性的组合，并且 $\mathbf{x}_t$ 与 $\mathbf{h}_{t-1}$ 两者都具有仿射变换。一般地，建模序列的简单策略是使用一个 RNN 将输入序列映射到固定大小的向量，然后将向量送到 softmax 层以进行分类或其他任务[45]。然而，具有这种形式的转换函数 RNN 的问题在于，在训练期间，梯度可以在长序列上指数地增长或衰减。这个问题使得 RNN 模型难以学习序列中的长距离依赖关系。

长短时记忆网络（LSTM）[46] 是一种循环神经网络[47]，专门解决了学习长期依赖的问题。LSTM 维护一个仅在必要时更新的存储器单元。虽然有许多 LSTM 变体，但在这里我们使用 Jozef 等人[48] 的 LSTM 架构，它类似于 Grave 等人[49] 的架构，但没有 peephole 连接。我们在每个时间步 $t$ 定义 LSTM 单元为 $\mathbb{R}^d$ 中的向量集合：输入门 $\mathbf{i}_t$，遗忘门 $\mathbf{f}_t$，输出门 $\mathbf{o}_t$，记忆单元 $\mathbf{c}_t$ 和隐藏状态 $\mathbf{h}_t$。$d$ 是 LSTM 单元的隐层大小。门控向量 $\mathbf{i}_t$，$\mathbf{f}_t$ 和 $\mathbf{o}_t$ 的元素取值范围在 $[0, 1]$ 中。

$$\begin{bmatrix} \tilde{\mathbf{c}}_t \\ \mathbf{o}_t \\ \mathbf{i}_t \\ \mathbf{f}_t \end{bmatrix} = \begin{bmatrix} \tanh \\ \sigma \\ \sigma \\ \sigma \end{bmatrix} T_{\mathbf{A},\mathbf{b}} \begin{bmatrix} \mathbf{x}_t \\ \mathbf{h}_{t-1} \end{bmatrix}, \tag{2.7}$$

$$\mathbf{c}_t = \tilde{\mathbf{c}}_t \odot \mathbf{i}_t + \mathbf{c}_{t-1} \odot \mathbf{f}_t, \tag{2.8}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t), \tag{2.9}$$

其中，$\mathbf{x}_t$ 是当前时间步的输入，$T_{\mathbf{Ab}}$ 是仿射变换，它取决于网络 $\mathbf{A}$ 和 $\mathbf{b}$ 的参数。$\sigma$ 表示逻辑 sigmoid 函数，$\odot$ 表示对位乘法。直观地，遗忘门控制擦除存储器单元的每个单元的程度，输入门控制每个单元的更新程度，并且输出门控制内部存储器状态的输出程度。

LSTM 的更新过程可以被重写为下面的精简模式:

$$(\mathbf{h}_t, \mathbf{c}_t) = \mathbf{LSTM}(\mathbf{h}_{t-1}, \mathbf{c}_{t-1}, \mathbf{x}_t) \tag{2.10}$$

这里，函数 $\mathbf{LSTM}(\cdot,\cdot,\cdot)$ 是公式 (2.7-2.9) 的缩写。

### 2.4.3. 树结构神经网络

树结构神经网络（Tree-structure Neural Networks），简称 TreeNN，它默认文本有一个树型的结构，然后在树的结构上对词语进行合成学习。形式上，对于任意一段文本，我们先通过外部工具（例如句法分析树）获取它的树（特别地，这里是二叉树）$T$，其中每个非叶子结点对应一个短语。我们将 $\mathbf{h}_j$ 和 $\mathbf{c}_j$ 称为每个结点 $j$ 的隐藏状态和内存单元。结点 $j$ 的转换方程如下:

$$\begin{bmatrix} \tilde{\mathbf{c}}_j \\ \mathbf{o}_j \\ \mathbf{i}_j \\ \mathbf{f}_j^l \\ \mathbf{f}_j^r \end{bmatrix} = \begin{bmatrix} \tanh \\ \sigma \\ \sigma \\ \sigma \\ \sigma \end{bmatrix} T_{\mathbf{W},\mathbf{b}} \begin{bmatrix} \mathbf{x}_j \\ \mathbf{h}_j^l \\ \mathbf{h}_j^r \end{bmatrix}, \tag{2.11}$$

$$\mathbf{c}_j = \tilde{\mathbf{c}}_j \odot \mathbf{i}_j + \mathbf{c}_j^l \odot \mathbf{f}_j^l + \mathbf{c}_j^r \odot \mathbf{f}_j^r, \tag{2.12}$$

$$\mathbf{h}_j = \mathbf{o}_j \odot \tanh(\mathbf{c}_j), \tag{2.13}$$

其中 $\mathbf{x_j}$ 表示输入向量，当且仅当它是叶子结点时才为非零。上标 $l$ 和 $r$ 分别代表左子结点和右子结点。$\sigma$ 表示逻辑 sigmoid 函数，$\odot$ 表示对位乘法。$T_{\mathbf{W},\mathbf{b}}$ 是仿射变换，它取决于网络 $\mathbf{W}$ 和 $\mathbf{b}$ 的参数。

### 2.4.4. 图结构神经网络

图结构网络 (Graph-structure Neural Networks)，简称 GNN，是对树结构的推广，这种网络结构允许建模各种复杂的语义关系。Kipf 等人[50] 通过学习结点表示提出了用于半监督图分类的图卷积神经网络（Graph Convolutional Neural

Network，GCN）；图卷积神经网络是用于编码图的卷积神经网络[40] 的变形，它是一种多层神经网络，能根据其邻居结点的属性合成结点的嵌入向量。形式化地，给定一个图 $G = (V, E)$，其中，$V(|V| = n)$ 和 $E$ 分别表示结点和边的集合。对于每个结点 $i$，假设每个结点 $i$ 都与它自身相连接，即 $(i, i) \in E$。可以用邻接矩阵（Adjacency Matrix）$\mathbf{A} \in \mathbb{R}^{n \times n}$ 来表示图 $G$ 中结点之间的连接关系，如果结点 $i$ 与结点 $j$ 之间有边相连，那么 $\mathbf{A}_{ij} = 1$。在一个具有 $L$ 层的 GCN 网络中，如果我们用 $\mathbf{h}_i^{(l-1)}$ 表示结点 $i$ 在第 $l$ 层的输入向量，$\mathbf{h}_i^{(l)}$ 表示结点 $i$ 在第 $l$ 层的输出向量，那么图卷积的操作过程可以写成：

$$\mathbf{h}_i^{(l)} = \sigma(\sum_{j=1}^{n} \mathbf{A}_{ij} \mathbf{W}^l \mathbf{h}_i^{(l-1)} + b^{(l)}), \tag{2.14}$$

其中，$\mathbf{W}^l$ 表示线性转换，$b^l$ 表示偏置项，$\sigma$ 表示非线性函数（例如，Sigmoid）。在每个图卷积周期中，每个结点从图中的相邻结点收集和汇总信息。

图注意网络（GAT) 是为了让图获得足够好的表现力，在将输入特征转换为更高级别的特征的过程中引入了自注意力机制（Self-attention Mechanism）。图中的注意力仅作用于相邻结点中，假设 $N_i$ 表示与结点 $i$ 相邻的结点集合，那么图注意网络可以定义为：

$$e_{ij} = atn(\mathbf{w}_a \hat{\mathbf{h}_i}, \mathbf{w}_a \hat{\mathbf{h}_j}), \tag{2.15}$$

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k \in N_i} exp(e_{ik})}, \tag{2.16}$$

其中，$e_{ij}$ 表示注意力系数（Attention Coefficients），$\mathbf{w_a}$ 表示共享的线性转换矩阵，$atn$ 表示共享的注意力机制。

Gilmer 等人[51] 提出了一种通用的神经信息传递算法来预测分子结构的性质。Velickovic 等人[12] 提出图注意网络（GAT）建模图形结构数据，如蛋白质和引文网络。虽然图已在上述领域得到充分研究，但如何为句子构建图表示仍然不太清楚。Marcheggiani 等人[52] 尝试从基于预定义结构的序列构建图，例如句法依存。这并不符合句子结构的复杂性，因为在特定任务中，句子中的依存结构可

能与给定的句子结构明显不同[53]。我们会在后面的章节中讨论如何合理地使用图神经网络建模句子的表示。

### 2.4.5. 注意力网络

注意力神经网络（Attention-based Neural Networks）按照功能属性可划分成两类：信息收集和语义合成注意力机制。

#### (1) 信息收集

Yang 等人[54] 和 Lin 等人[55] 利用可学习的查询向量来聚合每个词语的加权信息。给定一个文本序列：$w_1, \cdots, w_T$, 我们用 $\mathbf{h}_1, \cdots, \mathbf{h}_T$ 表示其对应的隐层表示。隐层表示可以通过以上神经网络合成函数获得：$\mathbf{h}_t = \textsc{Sent-encoder}(\mathbf{w}_t, \theta)$。收集功能的注意力机制的目的在于生成一组权重，使得可以聚合句子中每个词的隐层表示。

$$\tilde{\mathbf{h}} = \sum_t \alpha_t \mathbf{h}_t \tag{2.17}$$

其中 $\alpha$ 表示注意力大小，可以按照如下方式计算，

$$\alpha_t = \frac{s_t}{\sum_j s_j} \tag{2.18}$$

其中

$$s_t = \mathbf{v}^T tanh(\mathbf{W}[\mathbf{q}, \mathbf{h}_t]) \tag{2.19}$$

#### (2) 语义合成

注意力机制的引入可以用来学习词语之间的依存关系。例如, Yang 等人[54,55] 利用可学习的查询向量来聚合每个词语的加权信息。该机制可用于获得分类任务的句子级表示。Cheng 等人[56] 在处理每个词语时增强具有重读能力的神经网络。Vaswani 等人[53] 建议完全基于自注意力机制来建模词语之间的依赖关系，而不需要任何循环或卷积层。这里我们选取一个典型模型 Transformer 来介绍。

在 Transformer 里，每一个词语对应的 $l+1$ 层的隐状态向量是通过有权重地关注 $l$ 层隐状态量得到的。给定一个句子，我们令 $n$ 表示句子长度，$d$ 表示隐状态向量的大小。$\mathbf{H}^l \in \mathbb{R}^{n \times d}$ 表示第 $l$ 层句子中所有词语向量的集合。在 Transformer 里，$\mathcal{H}^l$ 可以被语境化（contextualized）得到 $\tilde{\mathcal{H}}^l$。

$$\tilde{\mathcal{H}}^l = \text{softmax}(\frac{\mathcal{Q}^l(\mathcal{K}^l)^T}{\sqrt{d}})\mathcal{V}^l \tag{2.20}$$

其中

$$\mathcal{Q} = \mathcal{H}\mathcal{W}^Q, \mathcal{K} = \mathcal{H}\mathcal{W}^K, \mathcal{V} = \mathcal{H}\mathcal{W}^V \tag{2.21}$$

这里, $\mathcal{W}^Q, \mathcal{W}^K, \mathcal{W}^V \in \mathbb{R}^{d\times d}$ 都是可学习的参数。

## 2.5　数据集

### (1)　文本分类

这里，我们将描述本文所使用的文本分类数据集，表2.1是详细统计信息。

- **SST-1** 在 Stanford Sentiment Treebank[①]中，电影评论有五类[57]（从正面到负面过渡的五个等级），属于句子级别的文本分类。
- **SST-2** 二分类的电影评论（正面，负面），同样来自 Stanford Sentiment Treebank，属于句子级别的文本分类。
- **SUBJ** 主观性数据集，其目标是将每个实例（片段）分类为主观或客观[58]，属于句子级别的文本分类。
- **MR** 二分类电影影评（正面，负面）。
- **IE** 具有丰富习语的情感分类数据集[59]。每个句子至少包含一个习语。
- **QC** TREC 问题集，它包含六种不同的问题类型（缩写、实体，描述，人类，地点，数字)[60]。
- **IMDB** IMDB 数据集[②]包含 100,000 个两类情感的电影评论。该数据集的一个关键方面是每个电影评论都有几个句子，属于文档级别的文本分类。
- **20NewsGroup** 新闻语料数据集。它包含来自 20 个类别约 20,000 个文档新闻组。

[①] http://nlp.stanford.edu/sentiment

[②] http://ai.stanford.edu/~amaas/data/sentiment/

**表 2.1　文本分类数据集的统计信息**

| 数据集 | 类型 | 训练集 | 校验集 | 测试集 | 类别数 | 平均长度 | 词表大小 |
|---|---|---|---|---|---|---|---|
| SST-1 | 句子 | 8544 | 1101 | 2210 | 5 | 19 | 18K |
| SST-2 | 句子 | 6920 | 872 | 1821 | 2 | 18 | 15K |
| MR | 句子 | 9596 | - | 1066 | 2 | 22 | 21K |
| SUBJ | 句子 | 9000 | - | 1000 | 2 | 21 | 21K |
| IE | 句子 | 2221 | - | 300 | 3 | 16 | 7.5K |
| QC | 句子 | 5452 | - | 500 | 6 | 10 | 9.4K |
| IMDB | 文档 | 25,000 | - | 25,000 | 2 | 294 | 392K |

### (2) 语义匹配

- **SNLI** 斯坦福自然语言推理数据集, 该语料库包含 57 万个句对, 每个句对都有一个语义关系（矛盾, 蕴含, 中性), 并且所有句子和标签都源自人工标注。为了更好理解这个数据集, 我们给出一个真实的例子。给定两段话, “`These girls are having a great time looking for seashells`”, “`The girls are happy`”, 模型需要正确判断出他们的关系是蕴含。

- **SICK** 该数据集全称 Sentences Involving Compositional Knowledge, 由 Marelli 等人[61] 提出。数据集包含 9927 个句对，其中训练，校验，测试集合分别包含 4500，500，4927 个句对。数据集中预定义的类别标签有三个，分别是: “蕴含”、“矛盾”和“中性”。

### (3) 序列标注

本文序列标注的实验使用到以下标准数据集：Penn Treebank（PTB）POS，CoNLL 2000 Chunking，CoNLL 2003 NER，详细统计如表2.2所示。

- **PTB** Penn Treebank 已经被广泛用于自然语言处理中的各个任务中。

- **CoNLL2000** 该数据集基于 CoNLL 2000 年的 Chunking 评测任务。

**表 2.2　序列标注数据集的统计**

| 数据集 | 任务 | 训练集 | 校验集 | 测试集 |
|---|---|---|---|---|
| CoNLL2000 | Chunking | 211,727 | - | 47,377 |
| CoNLL2003 | NER | 204,567 | 51,578 | 46,666 |
| PTB | POS | 912,344 | 131,768 | 129,654 |

- **CoNLL2003** 是基于路透社的 NER 基准数据集，它提供了训练、校验和测试集。

# 第一部分

# 不同粒度的文本表示学习

# 第 3 章　词表示学习

## 3.1　引言

基于神经网络的词表示学习，通常也称词的分布式表示学习（Distributed Representation Learning）。它是将离散的词语转化为连续实值向量的过程。这里，每个词向量包含了这个词有用的句法和语义等信息。词表示的学习有助于解决维度灾难并提高泛化能力，因为它们可以将具有相似语义和句法的词聚类到一起。因此，词表示学习被广泛用于许多自然语言处理（NLP）任务中，例如句法分析[62–64]，语义分析[65]，和形态学分析[2]。一般地，根据学习到的词表示是否和上下文相关，我们将其划分以下面两种类型。

### 3.1.1.　上下文无关词表示

这种词表示的特点是静态的。静态的意思是指，每个词被唯一地分配一个向量，这个向量不会随着上下文的改变而发生改变。最早关于这类词向量的相关研究可以追溯到 Bengio 等人的工作[1]：通过使用神经语言模型学习一个和上下文无关的低秩的词向量。因为这种模型训练开销十分昂贵，此后许多研究都集中在优化上，例如 C&W 词表示模型[27] 和分层对数线性（HLBL）词表示模型[64]。特别地，Mikolov 等人[3] 提出的 word2vec 模型，使用 CBOW 与 Skip-Gram 两种模式训练词向量。该方法可以非常高效地训练出一个静态的词向量表。这个工作对后来许多工作产生了很大的影响，因为 word2vec 得到的无监督词向量，可以作为一种现成（Off-the-shelf）的知识迁移到下游任务中，并使得下游任务受益。

### 3.1.2.　上下文相关词表示

上述这些方法使用相同的词向量来表示一个词语，这在某种程度上是不合理的，有时它会伤害模型的表达能力，因为很多词语都是多义的。例如，词语“*bank*” 可以表示 “金融机构（例如，银行）”，也可以表示 “河岸”。所以，同一个词语在不同的上下文应该有不同的词表示。

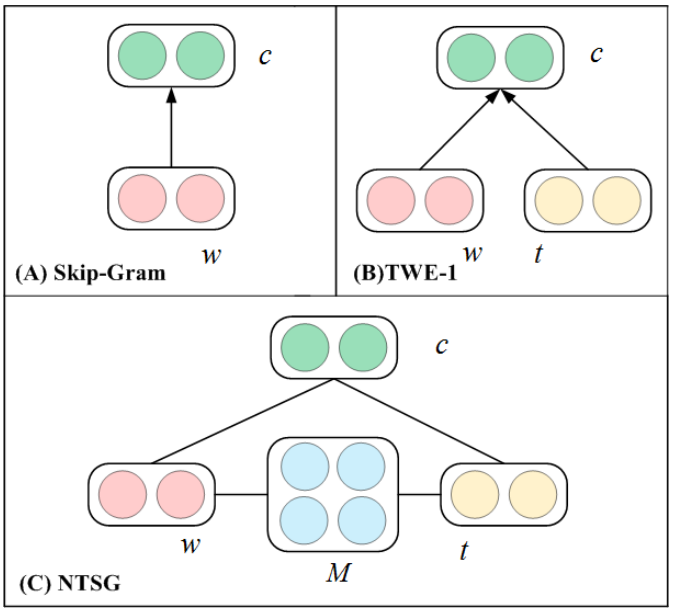



**图 3.1　Skip-Gram, TWE-1 和 NTSG 结构图**

为了解决这个问题，一些研究工作[4–7] 提出根据不同的上下文学习多原型（Multi-prototype）词向量。这些模型通过对每个词周围的上下文进行聚类以此实现一词多义的表示。但是，这些方法忽略了词语之间的相互关系以及它们的上下文（Context）。为了避免这种限制，Liu 等人[8] 提出了引入潜在主题模型[66]，该方法根据词语的上下文将全局聚类成不同的词主题（Topic）。他们提出三个基于主题的词向量学习模型来增强词表示的判别性。

最近，Peters 等人使用基于 LSTM 或者 Transformer[67,68] 的语言模型在大规模语料库上训练，从而获得与上下文相关的词表示。这些方法在文本分类、问答和序列标注等任务上获得了性能最好的结果。

## 3.2　研究动机

我们通过引入主题的概念，建立词、上下文、主题三者的关系，以此学习和上下文相关的词表示。具体地说，我们提出了一个神经张量跳词模型（Neural Tensor Skip-Gram, *NTSG*）来学习词语和主题的向量。该模型是 Skip-Gram 模型的扩展，它用张量层替换双线性层以捕获不同的词语和主题之间的更多交互。图3.1说明了 Skip-gram，TWE（Liu 等人提出的 Topical Word Embedding，*TWE*）和我们提

出的 NTSG 模型之间的差异，其中红色、黄色和绿色圆圈分别表示词语、主题和上下文的词向量。定性的实验分析显示了我们的模型和对比模型 Skip-Gram 相比，在邻近词发现上有明显的改进。并且，我们还做了上下文词语相似度判定和文本分类的任务，实验证明了我们的模型更能充分利用上下文信息学习符合语境的词向量。

我们的贡献如下：1) 引入神经张量网络层，来建立多个元组之间的关系。同时也证明 Skip-Gram 和 TWE 模型可以被认为是模型的特殊情况。2) 为了提高模型的效率，我们使用低秩张量因子分解方法将每个张量切片分解为两个低秩矩阵的乘积。

## 3.3　基于神经网络的词表示学习

虽然有很多方法可以从大量未标记数据学习词语向量表示, 这里我们只关注此类模型中最相关的方法。Bengio 等人[1] 最先提出用低维度向量表示每个词语并且在神经语言模型任务上进行训练。C&W 模型[27] 和分层对数线性（HLBL）模型[64] 通过对网络优化来加速训练过程。word2vec[3] 则利用 Skip-gram 模型使得词向量训练在海量数据上成为了可能。

Skip-Gram 是学习词向量的有效框架，旨在预测给定句子中目标词语的周围词语[34] 出现的概率。在 Skip-Gram 模型中，$\mathbf{w} \in \mathbb{R}^d$ 是词语在 $w \in \mathcal{V}$ 的向量表示形式，其中 $\mathcal{V}$ 是词汇表，$d$ 是词向量的维度。

给定一对词语 $(w, c)$，在目标词语 $w$ 的上下文中观察词语 $c$ 的概率由下式给出：

$$Pr(D = 1|w, c) = \frac{1}{1 + \exp(-\mathbf{w}^T\mathbf{c})} \tag{3.1}$$

其中，$\mathbf{w}$ 和 $\mathbf{c}$ 分别是 $w$ 和 $c$ 的词向量。

在 $w$ 的上下文中没有观察到词语 $c$ 的概率由下式给出:

$$Pr(D = 0|w, c) = 1 - \frac{1}{1 + \exp(-\mathbf{w}^T\mathbf{c})} \tag{3.2}$$

给定训练集 $\mathcal{D}$，通过最大化以下目标函数来学习词向量：

$$J(\theta) = \sum_{w,c \in \mathcal{D}} Pr(D = 1|w, c) + \sum_{w,c \in \mathcal{D}'} Pr(D = 0|w, c), \tag{3.3}$$

其中，设置 $\mathcal{D}'$ 是随机抽样的负样本（负样本是指通过采样的方式随机选取一个词，使得该词不兼容当前的语境）。

## 3.4　基于 Skip-Gram 的神经张量网络

为了增强词向量的表示能力，我们引入主题这个隐变量，并假设每个词语在不同主题下有不同的表示。例如，“苹果”这个词在“食品”这个主题下表示水果，而在“信息技术（IT）主题表示一个 IT 公司。

我们的目标是能够说明词语 $w$ 及其主题 $t$ 是否可以在 $c$ 上下文中很好地匹配。例如，$(w, t) = (apple, company)$ 匹配上下文 $c = iphone$，而 $(w, t) = (apple, fruit)$ 匹配上下文 $c = banana$。

在本文中，我们通过引入张量神经网络层[69] 来扩展 Skip-Gram 模型，以此捕获在不同上下文中，词语和主题之间相互作用。张量层的优点是它可以显式地建模多个数据的交互。具体说来，我们通过如下的能量公式来计算某个上下文词语 $c$ 中词语 $w$ 及其主题 $t$ 的可能性得分，

$$g(w, c, t) = \mathbf{u}^T f(\mathbf{w}^T \mathbf{M}_c^{[1:k]} \mathbf{t} + \mathbf{V}_c^T (\mathbf{w} \oplus \mathbf{t}) + \mathbf{b}_c), \tag{3.4}$$

其中，$\mathbf{w} \in R^d$,$\mathbf{t} \in R^d$ 分别是词语 $w$ 和主题 $t$ 的向量表示。$\oplus$ 是连接操作，并且 $\mathbf{w} \oplus \mathbf{t} = \begin{bmatrix} \mathbf{w} \\ \mathbf{t} \end{bmatrix}$; $\mathbf{M}_c^{[1:k]} \in \mathbb{R}^{d \times d \times k}$ 是一个张量，取两个向量 $\mathbf{w} \in \mathbb{R}^d$ 和 $\mathbf{t} \in \mathbb{R}^d$ 作为输入的双线性张量积，并生成一个 $k$ 维度短语向量 $\mathbf{z}$ 作为输出，

$$\mathbf{z} = \mathbf{w}^T \mathbf{M}_c^{[1:k]} \mathbf{t}, \tag{3.5}$$

其中，$\mathbf{z}$ 的每个分量由张量的一个片段 $i = 1, \cdots, k$ 计算：

$$\mathbf{z}_i = \mathbf{w} \mathbf{M}_c^{[i]} \mathbf{t}. \tag{3.6}$$

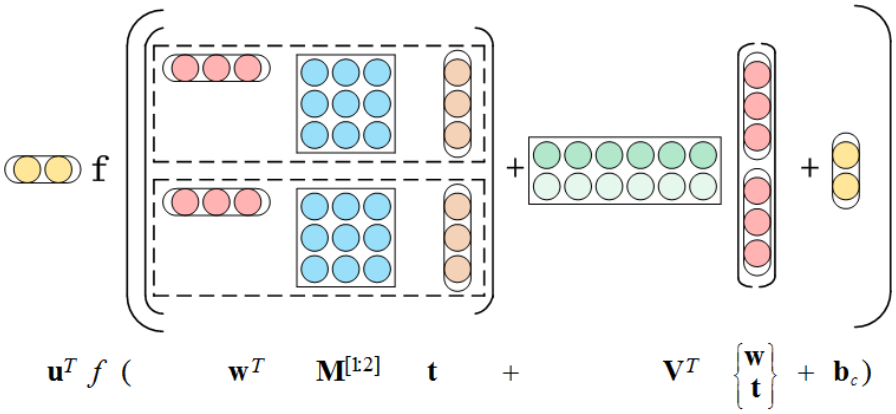



**图 3.2　神经张量网络可视化**

等式中的其他参数（3.4）是神经网络的标准形式：$\mathbf{u} \in \mathbb{R}^k$，$\mathbf{V}_c \in \mathbb{R}^{k\times 2d}$ 和 $\mathbf{b}_c \in \mathbb{R}^k$。$f$ 是标准非线性函数，设置为 $f(t) = \frac{1}{1+\exp -t}$，与 Skip-Gram 相同。

在等式中（3.4），张量 $\mathbf{M}_c^{[1:k]}$ 取决于上下文 $c$。然而，为每个上下文 $c$ 分配张量是不可行的，一种合理的处理方式是使用参数共享机制：我们对所有上下文使用相同的张量 $\mathbf{M}^{[1:k]}$。因此，我们改写了等式（3.4）如下:

$$g(w, c, t) = \mathbf{u}^T f(\mathbf{w}^T \mathbf{M}^{[1:k]} \mathbf{t} + \mathbf{V}_c^T (\mathbf{w} \oplus \mathbf{t}) + \mathbf{b}_c). \tag{3.7}$$

图3.2显示了该模型的可视化。主要优点是它同时建模词语、主题和语境之间的潜在关系。直观地，引入的张量可以把词语和主题之间的相互作用关系考虑进去。

### 3.4.1.　张量分解

尽管基于张量的模型可以捕捉丰富的交互信息，但是这种运算带来了很大的计算开销。不考虑矩阵优化算法，张量等式 (3.7) 中的操作复杂性是 $O(d^2k)$。而且，额外的张量操作可能会带来数百万的参数，从而使得模型遭受过度拟合的风险。为了解决这个问题，我们提出了一种张量分解方法，将每个张量切片转化为两个低秩矩阵的乘积。每个张量 $\mathbf{M}^{[i]} \in \mathbb{R}^{d\times d}$ 被分解为两个低秩的矩阵 $\mathbf{P}^{[i]} \in \mathbb{R}^{d\times r}$ 和 $\mathbf{Q}^{[i]} \in \mathbb{R}^{r\times d}$:

$$\mathbf{M}^{[i]} = \mathbf{P}^{[i]} \mathbf{Q}^{[i]}, 1 \leqslant i \leqslant k \tag{3.8}$$

其中，$r \ll d$ 是因子的数量。

$$g(w, c, t) = \mathbf{u}^T f(\mathbf{w}^T \mathbf{P}^{[1:k]} \mathbf{Q}^{[1:k]} \mathbf{t} + \mathbf{V}_c^T (\mathbf{w} \oplus \mathbf{t}) + \mathbf{b}_c), \tag{3.9}$$

现在张量运算的复杂性是 $O(rdk)$。只要 $r$ 足够小，就可以进行张量分解运算，这种方式将比未分解的快得多，并且自由参数也会少得多，可以防止模型过度拟合。

### 3.4.2. 相关模型与特例分析

在这一章节，我们将说明我们提出的 NTSG 词向量学习框架的通用性，很多已有的模型可以看成该框架下的一种特例。我们根据模型表达能力递增的顺序介绍几个相关模型。每个模型使用 $g$ 函数将分数分配给三元组，以估计在上下文 $c$ 中 $w$ 分配给主题 $t$ 的可能性。

#### (1) Skip-Gram

Skip-Gram 旨在给定滑动窗口中的词语预测上下文目标词。给定一对词语 $(w_i, c)$，我们将 $Pr(c|w_i)$ 表示为在目标词语 $w_i$ 的上下文中观察到词语 $c$ 的概率。

使用负采样方法，Skip-Gram 将概率 $Pr(c|w_i)$ 表示如下：

$$Pr(c|w_i) \approx Pr(D = 1|\mathbf{w}_i, \mathbf{c}) \tag{3.10}$$

$$= \frac{1}{1 + \exp(-\mathbf{w}_i^T \mathbf{c})} \tag{3.11}$$

$$= f(\mathbf{w}_i^T \mathbf{c}) \tag{3.12}$$

其中，$Pr(D = 1|\mathbf{w}_i, \mathbf{c})$ 是 $(w_i, c)$ 来自语料库真实共现的概率。

这个模型是我们设定的神经张量模型的一个特例：$f(t) = \frac{1}{1+\exp(-t)}$, $k = 1$, $\mathbf{M} = 0$, $\mathbf{b}_c = 0$ 并且 $\mathbf{V}_c = \mathbf{c}$。

#### (2) 主题相关的词表示模型

Liu 等人[8] 训练了类似的模型来学习主题词表示，展示在图3.1(B) 中，它使用目标词的主题 $t_i$ 来预测上下文词语。他们提出了三种具有不同词语和主题组

合的模型，这里我们只使用他们的第一个模型 TWE-1 用于比较，因为 TWE-1 取得了最佳结果。

$$Pr(c|w_i, t_i) \approx Pr(c|w_i)Pr(c|t_i) \tag{3.13}$$

$$\approx f\left((\mathbf{c} \oplus \mathbf{c})^T(\mathbf{w} \oplus \mathbf{t})\right). \tag{3.14}$$

根据公式（3.14），当 $k = 1$ 和 $\mathbf{M} = 0$，$\mathbf{b}_c = 0$ 和 $\mathbf{V}_c = (\mathbf{c} \oplus \mathbf{c})$ 的时候，我们可以看到 TWE-1 也是神经张量模型的一个特例。虽然这是对 Skip-Gram 的改进，但该模型的主要问题是向量 $\mathbf{w}$ 和 $\mathbf{t}$ 的参数不会相互交互，并且它们独立地映射到公共空间。

**(3) 我们的 NTSG 模型**

与 Skip-Gram 和 TWE 不同，我们的模型给出了一个更一般的框架来建模词语、主题和上下文三者的关系，如图3.1(C) 所示。Skip-Gram 和 TWE 可视为我们模型的特例。我们的模型以简单有效的方式结合了向量 $\mathbf{w}$ 和 $\mathbf{t}$ 的相互作用。

为了在不同的上下文中获得词语类型 $w$ 的不同表示，我们首先使用 LDA 获得其主题 $t$，并通过组合 $w$ 和 $t$ 的向量来获得上下文相关的表示。最简单的组合方式是连接词语及其主题向量 $\mathbf{w}^t = \mathbf{w} \oplus \mathbf{t}$。

### 3.4.3. 训练

我们使用对比最大边际标准[69,70] 来训练我们的模型。该方法的核心思想是：每个来自训练语料库的三元组 $\langle w, t, c\rangle$ 应该得到比随机替换后三元组更高的分数。这里我们令所有参数的集合为 $\Omega$，我们最小化以下目标：

$$J(\Omega) = \sum_{\langle w,t,c\rangle \in \mathcal{D}} \sum_{\langle w,\hat{t},\hat{c}\rangle \in \hat{\mathcal{D}}} \max\left(0, 1 - g(w,t,c) + g(w,\hat{t},\hat{c})\right) + \lambda\|\Omega\|_2^2, \tag{3.15}$$

其中，$\mathcal{D}$ 是来自训练语料库的三元组的集合，我们求得的正确三元组的得分高于采样本的三元组。对于每个正确的三元组，我们随机采样负样本。我们使用 $L_2$ 正则化所有参数，由超参数 $\lambda$ 进行加权。对于张量的第 $j$ 个切片，我们有以下导

数:

$$\frac{\partial g(w,c,t)}{\partial \mathbf{M}^{[j]}} = \mathbf{u}_j f'(\mathbf{z}_j)\mathbf{w}\mathbf{t}^T \tag{3.16}$$

其中，$\mathbf{z}_j = \mathbf{w}\mathbf{M}^{[j]}\mathbf{t} + \mathbf{V}_j^T(\mathbf{w} \oplus \mathbf{t})) + \mathbf{b}_j$,$\mathbf{V}_j$ 是矩阵 $\mathbf{V}$ 的 $j$ 行，我们将 $\mathbf{z}_j$ 定义为 $k$ 维隐藏张量层的第 $j$ 个元素。全部参数通过使用 SGD 进行优化，收敛到非凸目标函数的局部最优。

## 3.5　实验与分析

我们首先定性地介绍一些利用主题信息强化词表示学习的例子。然后定量地对两个任务进行评估：上下文词语相似度计算任务和文本分类任务。

在我们的实验中，我们在等式（3.7）中使用四个不同的张量 $\mathbf{M}$ 设置:

- NTSG-1: 我们设置 $k = 1$ 和 $M^{[1]}$ 是一个单位矩阵。
- NTSG-2: 我们设置 $k = 1$ 和 $M^{[1]}$ 是一个完整的矩阵。
- NTSG-3: 我们设置 $k = 2$，并且每个张量切片 $M^{[i]}$ 用两个 $r = 50$ 的低秩矩阵进行分解。
- NTSG-4: 我们设置 $k = 5$，并且每个张量切片 $M^{[i]}$ 用两个 $r = 50$ 的低秩矩阵进行分解。

### 3.5.1.　最近邻实验

表3.1定性地显示了一词多义学习的可视化结果。每一个块中的第一行是 Skip-Gram 的结果，余下的是 NTSG 的结果。对于每个词语，我们比较 Skip-Gram（第一行）和 NTSG-2 两个模型的最近邻情况。我们发现，由 Skip-Gram 返回的近邻词是多种义项的混合，暗示了 Skip-Gram 模型将多义词的多重意义转化为同一个词向量。相比之下，我们的模型可以通过主题信息成功区分词语的不同义项。

在图3.3中，我们展示了高维的可视化主题词表示。上方显示了四个不同主题中的主题词表示，下方显示了 *apple* 及其近邻词的两个主题表示。我们可以看到我们的模型可以有效地区分一个词的多种义项。

**表 3.1　我们的模型和对比模型 Skip-Gram 的最近邻词**

| 词语 | 相似的词语 | 词语 | 相似的词语 |
|---|---|---|---|
| bank | depositor, fdicinsured, river, idbi | left | right, pass, leftside, front |
| bank:1 | river, flood, road, hilltop | left:1 | leave, throw, put, go |
| bank:2 | finance, investment, stock, share | left:2 | right, back, front, forward |
| apple | blackberry, ipod, pear, macworld | orange | citrus, yellow, yelloworang, lemon |
| apple:1 | macintosh, iphone, inc, mirco | orange:1 | blue, maroon, brown, yellow |
| apple:2 | cherry, peach, berry, orange | orange:2 | pineapple, mango, grove, peach |
| run | wsvn, start, operate, pass | plant | nonflowering, factory, flowering, nonwoody |
| run:1 | walk, go, chase, move | plant:1 | factory, distillate, subdepot, refinery |
| run:2 | operate, running, driver, driven | plant:2 | warmseason, intercropped, seedling, highyield |

### 3.5.2. 上下文词相似性分析

我们在由 Huang 等人[5] 提出的 Stanford Contextual Word Similarities（SCWS）数据集上评估我们的词向量。SCWS 数据集中有 2003 个词语对，其中包括 1328 个名词 - 名词对，399 个动词 - 动词对，140 动词 - 名词，97 形容词 - 形容词，30 名词 - 形容词。

我们使用在线知识库 Wikipedia（2010 年 4 月版本[5]），学习这个任务的主题词表示。我们使用 Gibbs 采样 LDA[66,71] 来获取词语主题。给定一系列词语 $D = \{w_1, \ldots, w_M\}$，在 LDA 收敛后，每个词语 $w_i$ 将被标记为特定主题 $t_i$，形成词语 - 主题对 $(w_i, t_i)$，用于学习我们的模型。

为了进行公平比较，我们将部分参数设置为与[8] 相同。我们设置主题数 $T = 400$ 和迭代数 $I = 50$。在学习 Skip-Gram 和我们的模型时，我们将窗口大小设置为 5，并将词向量和主题向量的维度设置为 $K = 400$。

我们借鉴[4,8] 使用以下的相似度得分函数 $AvgSimC$ 和 $MaxSimC$。

对于每个词语 $w$ 及其上下文 $c$，我们将首先通过将 $c$ 视为文档来推断主题分布 $Pr(t|w, c)$。给定一对带有上下文的词语，即 $(w_i, c_i)$ 和 $(w_j, c_j)$，$AvgSimC$ 旨

**表 3.2　不同模型在 SCWS 数据集上的性能**

| 模型 | $\rho \times 100$ | |
|---|---|---|
| TFIDF | 26.3 | |
| Pruned TFIDF | 62.5 | |
| Skip-Gram | 65.7 | |
| C&W | 57.0 | |
| | **AvgSimC** | **MaxSimC** |
| **多原型模型** | | |
| Huang 等人[5] | 65.4 | 63.6 |
| Tian 等人[6] | 65.3 | 58.6 |
| Neelakantan 等人[7] | 69.3 | - |
| Liu 等人[8] | 68.1 | 67.3 |
| **我们的模型** | | |
| NTSG-1 | 68.2 | 67.3 |
| NTSG-2 | **69.5** | **67.9** |
| NTSG-3 | 68.5 | 67.2 |
| NTSG-4 | 67.1 | 65.7 |

**表 3.3　多类别文本分类评估的结果**

| 模型 | Acc. | Prec. | Rec. | F1 |
|---|---|---|---|---|
| NBOW | 79.7 | 79.5 | 79.0 | 79.0 |
| LDA | 72.2 | 70.8 | 70.7 | 70.0 |
| Skip-Gram | 75.4 | 75.1 | 74.3 | 74.2 |
| **多原型模型** | | | | |
| Liu 等人[8] | 81.5 | 81.2 | 80.6 | 80.6 |
| **我们的模型** | | | | |
| NTSG-1 | **82.6** | 82.5 | 81.9 | 81.2 |
| NTSG-2 | 82.5 | **83.7** | **82.8** | **82.4** |
| NTSG-3 | 81.9 | 83.0 | 81.7 | 81.1 |
| NTSG-4 | 79.8 | 80.7 | 78.8 | 78.8 |

在衡量不同主题下两个词语之间的平均相似度:

$$AvgSimC = \sum_{t,t' \in T} Pr(t|w_i, c_i) Pr(t'|w_j, c_j) S(\mathbf{w}'_i, \mathbf{w}'_j), \tag{3.17}$$

其中，$\mathbf{w}'$ 是在 $t$ 主题下的词向量 $w$，通过连接词语和主题向量 $\mathbf{w}' = \mathbf{w} \oplus \mathbf{t}$ 获得; $S(\mathbf{w}'_i, \mathbf{w}'_j)$ 表示余弦相似度。

$MaxSimC$ 选择在上下文 $c$ 中使用 $w$ 推断的最可能主题 $t$ 的相应主题词向量 $\mathbf{w}'$ 作为上下文词向量。上下文词语相似度定义为:

$$MaxSimC = S(\mathbf{w}_i^t, \mathbf{w}_j^{t'}), \tag{3.18}$$

其中，$t = \arg\max_t Pr(t|w_i, c_i)$，$t' = \arg\max_t Pr(t|w_j, c_j)$。

在表3.2中我们展示了几种模型的评估结果。和前人工作相比，因为我们使用了相同的数据集和训练设置，所以我们直接报告他们的实验结果[5–8]。对于基

线模型 Skip-Gram，我们只使用词向量计算相似度。这里词的维度 K = 400。我们还对比了 C&W 模型（Collobert 等人[63]）提供的词向量。值得注意的是，这种模型学习到的词向量在使用的时候和上下文无关。

对于所有的基准模型和我们的模型，我们使用 AvgSimC 和 MaxSimC 作为评估的指标。表3.2显示 NTSG-2 模型优于其它的方法。使用 avgSimC 测量，此之前的任务最先进模型[7] 达到 69.3%，而 NTSG-2 在这个任务的最高分为 69.5%。我们发现，通过引入主题信息，模型可以更有效区分每个词的不同义项。此外，模型 NTSG-1、2 考虑了词语和主题之间相互交互，与对比模型相比，也可以获得更好的表现。对于我们提出的四种 NTSG 模型，我们发现 NTSG-2 优于其他模型。而对于模型 NTSG-3、4，我们发现张量分解的操作在加快训练速度的同时对性能也有些伤害。

### 3.5.3. 文本分类

我们还研究了模型对文本分类的有效性。我们使用主流的数据集 20News-Group，它包含来自 20 个类别约 20,000 个文档新闻组。

对于我们的模型，我们设置主题数量 $T = 80$ 来进行训练和测试，这个超参数与[8] 中的相同。然后我们在训练集上学习词向量和主题向量，它们的维度 $d = 400$。对于在文档 $q$ 中的每个词语及其主题，我们通过连接词语向量和主题向量来生成与上下文相关的词向量。此外，文档表示可以如下计算：$\mathbf{q} = \sum_{w \in q} Pr(w|q)\mathbf{w}^t$, 其中 $\mathbf{w}^t = \mathbf{w} \oplus \mathbf{t}$。t 和 $Pr(w|q)$ 可以用 TFIDF 加权。之后，我们把文档向量作为文档特征，并使用 Liblinear 工具包[72] 进行训练。

我们考虑以下基线（baselines）作为对比模型，即词袋模型 (BOW)，LDA，Skip-Gram 和 TWE。BOW 模型表示把每个文档的词语集合加权 (TFIDF) 平均。对于 TFIDF 方法，我们根据 TFIDF 得分选择最高的 50,000 个词语作为特征。LDA 将每个文档表示为其推断的主题分布。对于 Skip-Gram，我们通过简单地平均所有词向量构建文档的特征向量。

表3.3给出了在 20NewsGroup 文本分类的评估结果。我们可以看到 NTSG-1、

2、3 显著优于所有基线模型，特别是 NTSG-2 取得了最好的表现。对于 TWE 模型[8]，在学习过程中主题向量会影响相应的词向量，这会导致同一个主题下的词语学习到的表示差异很小，我们的模型在某种程度上解决了这个问题。类似上一个任务，NTSG-2 仍然是四个模型中表现最好的一个。

## 3.6　本章小结

我们提出了一种通用架构，即 Neural Tensor Skip-Gram 模型，来学习和上下文相关的词表示向量。该模型在两个主流任务上取得了超过基线模型的性能。更重要的是，词表示学习是更高层粒度文本（句子，句对）学习的基础，本章同时也揭示了“上下文相关”词向量学习的重要性，这也对后续几个工作做了一个很好的铺垫。

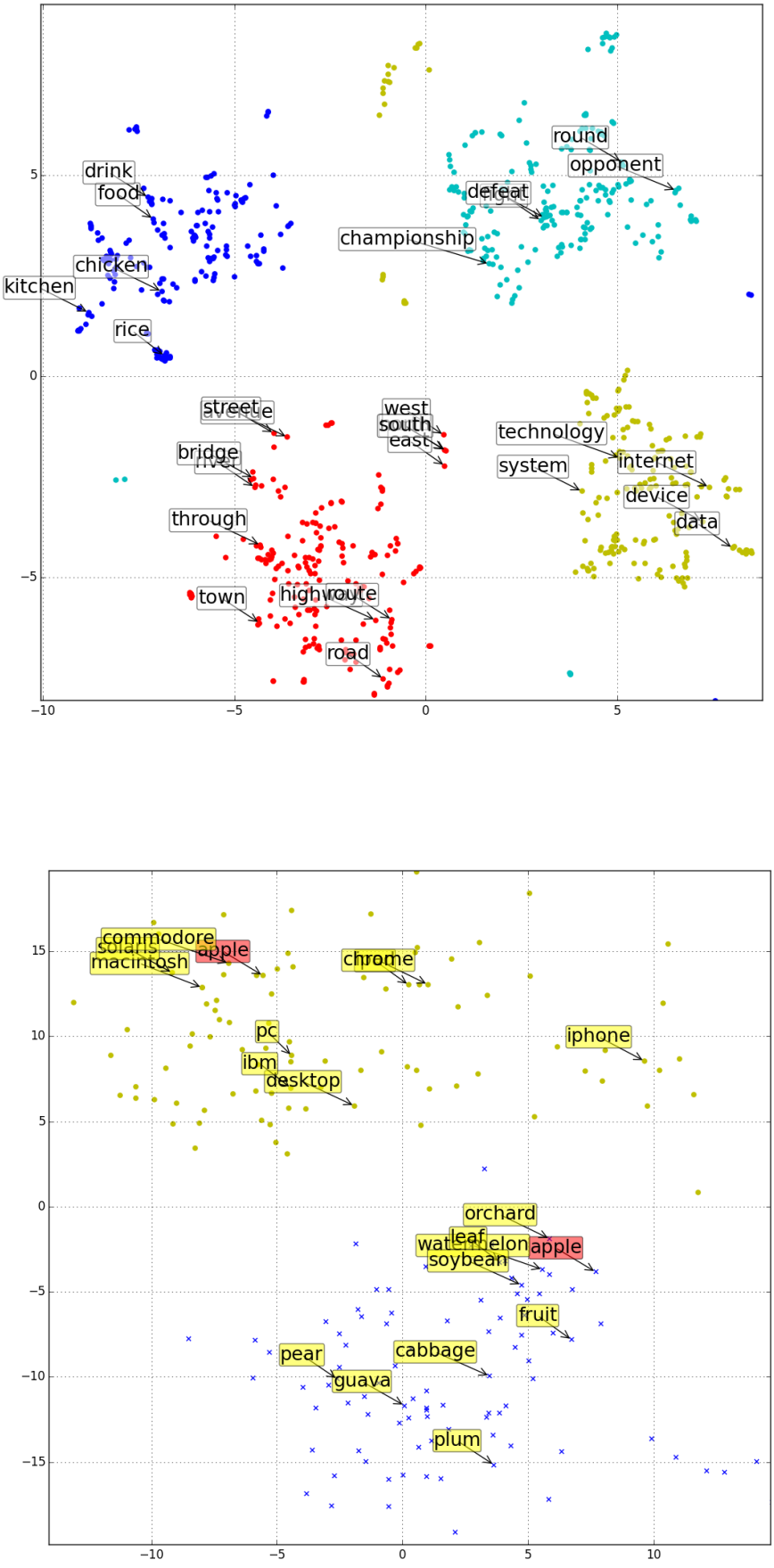





**图 3.3　NTSG-2 的二维主题词向量**

# 第 4 章　语义合成问题探究：从词表示到句子表示学习

## 4.1　引言

词表示的学习给更高层文本的表示学习提供了基础，例如，当我们实现了词语从离散化到连续化的转化，我们就可以通过引入语义合成函数（compositional fucntion）按照一定的拓扑结构（topological structure）去对句子中的词语进行合成，从而获取更高层的文本表示。而在这个语义合成的过程中，会遇到不同的挑战，在这一章中，我们将从三个方面介绍在语义合成过程中遇到挑战以及我们的应对策略。

## 4.2　句子中习语现象的建模

### 4.2.1.　研究动机

目前，神经网络模型已经在许多自然语言处理任务中取得了巨大成功。其中句子的表示学习成为一个基础而重要的任务。这其中的关键在于如何从其构成的词合成短语或句子。比如，如何通过“红色的”和“苹果”这两个词语的表示合成“红色的苹果”这个短语的表示。这种由较小粒度文本合成高层次文本信号的过程被定义为语义合成问题。通常地，直观的做法是使用一个共享的合成函数递归地对词向量进行合成，直至最终获得短语或者句子的表示。合成函数的形式也很多样，涉及多种神经网络，例如循环神经网络[46,73]、卷积神经网络[9,63]、以及递归神经网络[10,11,57]。

然而，这些方法在表示惯用语（习语）时表现出明显的缺陷，因为习语的含义往往不能由单个词的字面意思组成。如图4.1 是习语语义合成的例子，“go bananas”是个习语（该短语的字面意思为“去香蕉”，然而这个短语通常以“变得情感失控”这个语义出现），它的含义不能直接来自其包含的词语的文字组合。为了解决这个现象，以前的一些工作侧重于自动识别习语[15–19]。然而，以上这些工作没有解决这样一个问题：如何在理解句子含义的时候把习语的存在考虑

进去。受到**直译优先**[74] 心理语言学假设的启发（先将短语按照字面意思去理解，如果与其所处的语境冲突那么则将该短语视为习语），在本章中，我们提出了一种习语感知的分布式语义表示神经网络模型。这个模型建立在树结构的神经网络之上。具体地说，我们为句子中的每个短语引入一个神经习语检测器，以自适应地确定它们的组成：*直译方式*或*习语方式*。

对于可字面理解的短语，我们从其组成成分中计算其语义；而对于习语，我们设计了两种不同的方法来学习习语的语义表示，这两种方法基于两种不同的习语语言观点[75–77]。

为了评估我们的模型是否能够正确理解带有习语的句子，我们选择了情感分类任务作为评估实验，原因如下：

1）习语通常意味着对某事的情感立场，它们在评论中很常见[59]。2）情感分类结果的错误分析表明，大量错误是由习语引起的[78]。

这项工作的贡献总结如下：

- 我们增强了递归神经网络的能力，使其能够建模习语，并在学习句子表示时具备处理习语变体的现象的能力。
- 我们将习语理解融入传统的 NLP 任务，而不是将习语检测作为独立任务进行评估。
- 我们构建了一个新的数据集，涵盖了具有原始和变形形式的丰富习语。精细的定性和定量实验分析显示了我们模型的有效性。

### 4.2.2. 习语的语言学解读

最近，习语在语言学家、心理语言学家和词典编纂者中引起了广泛关注，因为它们在日常话语中普遍存在，并且在语言学文献中具有迷人的特性[79,80]。作为特殊的语言结构，习语有以下三个性质：

**不可见性** 习语总是将自己伪装成句子中的正常的短语，它使端到端训练变得困难。

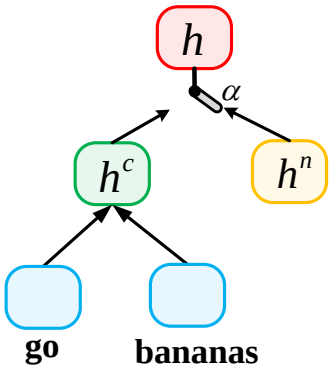



**图 4.1　自适应语义合成神经网络**

**表 4.1　习语的主要属性和相应的挑战**

| 性质 | 挑战 | 观点 1 | 观点 2 |
|---|---|---|---|
| 不可见性 | 端到端训练很困难 | | ✓ |
| 习语性 | 难以预测习语的意思 | × | ✓ |
| 灵活性 | 难以处理变体和泛化 | ✓ | × |

**习语性**　习语真正的意思不能由字面意思合成而来。

**灵活性**　虽然结构固定，但习语允许变化。一些组成习语的词可以被删除或用其他词代替。

表4.1介绍了习语的三个性质以及给分布式组合语义模型带来的挑战，其中✓ 表示存在对应性质所存在的挑战。为了解决这些挑战，现有思路有两种观点：第一个观点是不可合成的观点，将习语视为长词，其含义是任意规定的，不能从其成分中得到预测[75,76]。然而，相当多的习语在形态学和词汇方面都表现出一定程度的灵活性，因此这种方法处理差异很大而且不能泛化。第二种观点是可合成的观点，认为习语是语言的表达[77]，其含义由其组成部分的含义决定，一些组成规则可用于组合它们。这种完全可合成 (compositional) 的观点可以处理习语变体的现象，但它会受到习语性问题的影响，因为习语的含义是不透明的。

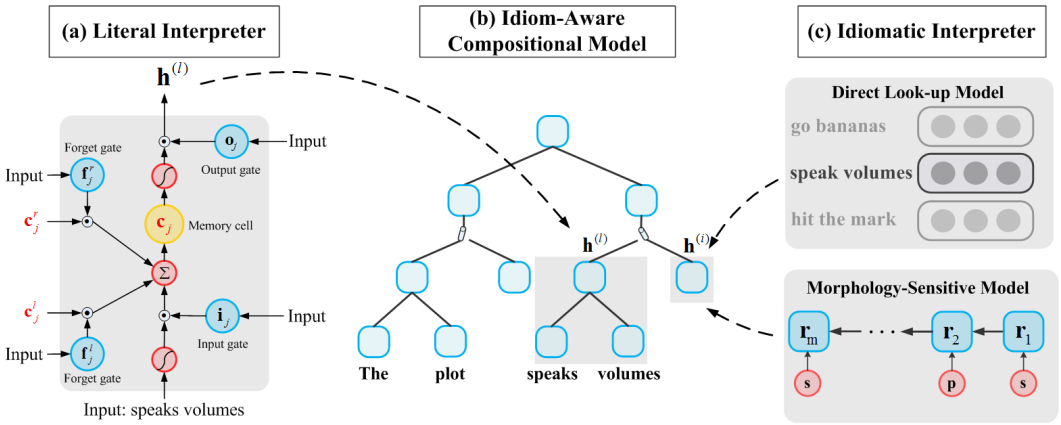



**图 4.2　习语感知组合网络的子模块结构**

### 4.2.3.　模型

我们提出了一种用于感知习语的分布式语义表示神经网络模型。具体来说，就不可见性而言，我们引入了一个习语检测器，以便在学习句子表示时自适应地区分每个短语是字面意义还是习语意义。对于可字面理解的短语，我们使用树结构长短时记忆网络（TreeLSTM）[10,11] 计算短语的语义表示。对于不可字面理解的短语（习语），我们设计了两种不同的方法来学习它的表达，这两种方法分别基于两种不同的语言学观点。

#### 1.　字面解释器

字面解释器本质上是一个组合语义模型，其中短语的语义由其组成词语的字面意义组成。几个现有的模型可以作为字面翻译。在本章中，我们采用 TreeLSTM[10]。图 4.2-(a) 给出了 TreeLSTM 框架结构的图示。

#### 2.　习语检测器

尽管 TreeLSTM 取得了成功，但仍然存在一个潜在弱点，即该模型默认短语或句子的含义可以从其成分的含义组成。以前的神经句子模型在学习习语的意义方面能力很差，更不用说对习语变体进行建模。

因此，我们引入了一个参数化的习语检测器，用于检测字面意思和习语意义之间的界限。具体来说，如果按照字面理解的语义与上下文是冲突的，那么

检测器应该检查是否应该采用习语的意义。这个**字面意思优先**理解模型是来自 Bobrow 等人[74] 提出的心理语言学假设。

由于忽略了上下文信息，TreeLSTM 遇到了歧义的问题。例如，短语 “in the bag” 在一个句子’‘The dictionary is in the bag” 中是字面意思，而在另一个句子 “The election is in the bag unless the voters find out about my past.” 中具有习语意义。为了解决这个问题，我们在对词语进行合成的时候给每个词引入了上下文信息。

**(1) 上下文表示**

形式化地，对于每个非叶结点 $i$ 及其对应的短语 $p_i$，我们将 $C$ 定义为包含短语 $p_i$ 周围词语的词语集合。然后可以获得上下文表示 $s_i$，如下所示：

$$\mathbf{s}_i = f(c_1, c_2, ..., c_k; \theta) \tag{4.1}$$

其中 $f$ 是一个具有可学习参数 $\theta$ 的函数，$c_i \in C$。这里，该函数有两种方式实现，利用 NBOW （Neural Bag-of-Words）或者 LSTM。

**(2) 识别器**

检测器输出标量 $\alpha$ 以确定短语的含义是字面的还是习语。对于短语 $i$（非叶结点 $i$）及其上下文信息 $\mathbf{s}_i$ 和字面意思 $\mathbf{h}_i^{(l)}$，我们使用 $\alpha_i$ 结合多层感知器来计算语义成分得分。

$$\alpha_i = \sigma(\mathbf{v}_s^T \tanh(\mathbf{W}_s[\mathbf{h}_i^{(l)}, \mathbf{s}_i])), \tag{4.2}$$

其中 $\mathbf{W}_s \in \mathbb{R}^{d \times 2d}$ 和 $\mathbf{v}_s \in \mathbb{R}^d$ 是可学习的参数。

### 3. 习语解释器

习语的存在对理解当前语言的模型是一项巨大的挑战。然而，在现实世界任务中，对习语的建模和研究却很少。根据习语理解的不同语言学观点（基于可合成，与基于不可合成的观点）[76,77]，我们提出了两种习语解释器来对它们进行建模。

**(1) 直接查找模型**

该模型灵感来自习语理解的直接访问理论[81]。在这个模型中，如果短语 $p$ 被检测为习语，那么它将被视为像键（Key）一样的长字，它们的含义可以直接从外部存储器 $M$ 中检索而得到，外部存储器 $M$ 存储了每个习语的信息。图4.2-(c)的顶部子图表示直接查找模型的框架图。形式上，获取习语的含义可以表示为：

$$\mathbf{h}^{(i)} = \mathbf{M}[k] \tag{4.3}$$

其中 $k$ 表示相应短语 $p$ 的索引。

**(2) 形态敏感模型**

由于大多数习语在形态学、词汇、语法方面都具有一定的灵活性，因此上述模型存在习语变体无法建模的问题。为了解决以上不足，受到习语可合成性[77]的观点以及最近基于字符的模型成功的启发[82–84]，我们提出使用 CharLSTM 直接编码习语，它不受其字面意义的影响，并且对不同的形态变化敏感。

形式化地，对于成分树中的每个非叶结点 $i$ 及其对应的短语 $p_i$，我们将 charLSTM 应用于短语 $p_i$，利用最终获得的隐藏状态 $\mathbf{r}_j$ 来表示短语的习语含义。如图4.2-(c) 中底部子图表示的是形态敏感模型的结构图。

$$\mathbf{r}_j = \textbf{Char-LSTM}(\mathbf{r}_{j-1}, \mathbf{c}_{j-1}, \mathbf{x}_j) \tag{4.4}$$

其中 $j = 1, 2 \cdots m$，$m$ 表示输入短语的长度。然后，我们可以得到习语的表示

$$\mathbf{h}^{(i)} = \mathbf{r}_m \tag{4.5}$$

在获得字面意义或习语意义后，我们可以计算短语 $p_i$ 的最终表示形式：

$$\mathbf{h}_i = \alpha_i \mathbf{h}_i^{(i)} + (1 - \alpha_i)\mathbf{h}_i^{(l)} \tag{4.6}$$

**4. 两种习语解释器的对比分析**

给定一个短语，两个解释器都可以生成相应的语义表示，而不受其字面含义的影响。它们区别在于查找模型采用完全非组合（Non-compositional）的观点，

即习语的含义可以直接从外部字典中访问。这种直接的检索机制更高效，并且可以利用预先训练好的外部字典引入外部先验知识。

相反地，形态敏感模型认为，习语意义仍然可以在习语空间中组成，这使得该模型能够更灵活地理解习语，并且该模型不需要使用额外的字典。

### 4.2.4. iSent: 富含习语的情感分类数据集

为了评估我们的模型，我们需要一个在很大程度上依赖习语理解的任务。在本章中，我们选择情感分类任务，原因如下：1）习语通常意味着对某事的情感立场，它们在评论中很常见[59]。2）情感分类结果的错误分析表明，大量错误是由习语引发的[78]。

在本节中，我们将首先简要介绍最常用的情感分类数据集，以此激发对新的基准数据集的需求。

#### 1. 用于情感分类的主流数据集

NLP 领域中最常用于情感分类的数据集有 SST-1、SST-2、MR、SUBJ，具体数据描述见表2.1。我们首先在这些数据集上评估我们的模型，并且和许多现有模型进行对比。

#### 2. 新数据集的需求

与以前的工作不同，我们希望将习语带入到一个具体的任务（情感分析）里去评价。然而，大多数现有的情感数据集都没有涵盖足够的习语或相关的语言现象。为了更好地评估习语理解任务模型，我们构建了一个富含习语的情感分类分类数据集 iSent，其中每个句子至少包含一个习语。

此外，考虑到大多数习语在形态学，词汇和语法方面具有一定的灵活性，我们通过引入不同类型的习语变体来丰富这个数据集，以便我们可以进一步评估该模型处理不同习语的能力。如表4.2所示，我们总结了两种类型的习语变体现象，这里 *Add.* 和 *Sub.* 分别表示习语通过增加词语或者删除词语后形成的习语变体。

**表 4.2　形态和词汇方面的习语变体示例**

| 变体 | | 例子 |
|---|---|---|
| 形态 | 动词 | go bananas （发疯）→ went bananas （发疯） |
| | 名词 | in a nutshell （简而言之）→ in nutshells （简而言之） |
| 词汇 | Add. | in the lurch （陷入困境）→ in the big lurch （陷入困境） |
| | Sub. | on the same page （达成共识）→ on different pages （没有达成共识） |

### 3. 数据集收集

我们抓取网站 `rottentomatoes.com` 以收集具有类别和包含习语的电影评论，并构造习语字典[85]。习语词典仅包含词汇变化而没有形态变化。为了解决这个问题，我们用动词（时态），名词（复数或单数）来手动标注每个习语的形态变体。

然后我们用习语过滤这些电影评论，确保每个句子至少涵盖一个习语。之后，我们获得了近 $15K$ 电影评论，其中包括 $1K$ 习语。为了进一步提高这些习语丰富的句子的质量，我们采取一些策略来过滤数据集，最后提取出 $13K$ 习语丰富的句子。

- 如果习语在所有评论中出现的频次低于 3，我们就会丢弃这个习语和相应的评论。
- 如果句子中的一些 “习语” 是电影名称，而不是表达惯用的习语含义，我们就将该评论过滤掉。

### 4. 统计信息

iSent 数据集最终包含 9752 个训练样本，1020 个校验样本和 2003 个测试样本。此外，校验和测试集涵盖了不同类型的习语变体，使我们能够测试模型泛化能力。表 4.3显示详细的统计信息，在这里，$O$、$M$、$L$ 分别表示习语的原形，习语的形态学变体，和习语的词汇变体。图4.3显示不同长度的评论数量的分布。

**表 4.3　iSent 数据集中习语和句子的统计数据**

| | 训练 | 校验 | | | 测试 | | |
|---|---|---|---|---|---|---|---|
| | | O | M | L | O | M | L |
| 习语 | 1247 | 124 | 21 | 40 | 124 | 21 | 40 |
| 句子 | 9752 | 720 | 200 | 100 | 1403 | 400 | 200 |

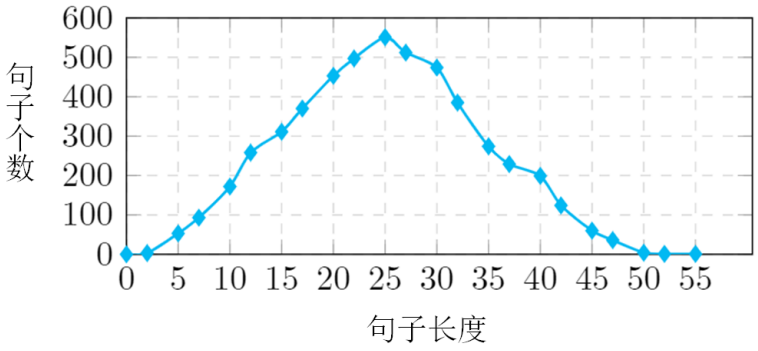



**图 4.3　不同长度的评论数量的分布**

### 4.2.5. 实验

我们首先在四个主流的情感数据集上评估提出的模型，并与其他模型比较实验结果。然后，我们使用新构造的数据集进行更详细的实验分析。

### 4.2.6. 实验设置

#### (1) 损失函数

给定一个句子及其标签，神经网络的输出是不同类别的概率。训练网络的参数以最小化预测和真实标签分布的交叉熵。为了最小化目标函数，我们使用带有 AdaGrad 对角变量的随机梯度下降法[86]。

#### (2) 初始化和超参数

在我们所有的实验中，所有模型的词语嵌入都是用 GloVe[87] 向量初始化的。其他参数服从范围为 $[-0.1, 0.1]$ 的均匀分布随机抽样初始化。

对于来自五个数据集的所有句子，我们用成分句法分析器[88] 来获得我们模型以及对比模型所需的树结构。

### 4.2.7. 对比模型

- CharLSTM: 基于字符级别的 LSTM。
- TLSTM: 树型 LSTM, 由[10] 提出。
- Cont-TLSTM: 上下文相关的树 LSTM[89]。
- iTLSTM-Lo: 我们提出的基于查表的模型。
- iTLSTM-Mo: 我们提出的形态学敏感的模型。

#### 1. 在主流数据集的结果

实验结果（准确率，即分类数据正确的比例）显示在表4.8中，我们可以看到 Cont-TLSTM 在所有四个任务上都优于 TLSTM，显示了上下文敏感特性的重要性。此外，iTLSTM-Lo 和 iTLSTM-Mo 都比 TLSTM 和 Cont-LSTM 获得更好的结果，这表明我们引入的模块（习语检测器和习语解释器）的有效性。此外，与 iTLSTM-Lo 相比，iTLSTM-Mo 表现更好，表明基于字符的习语解释器更强大。

虽然四个主流数据集中没有丰富的习语，但我们也可以观察到从我们的模型中获得的实质性改进。我们将这一成功归功于引入检测器在识别除习语之外的其他非组合搭配中的能力。我们稍后会讨论这个问题。

#### 2. 在 iSent 数据集上的结果

由于 iSent 是新引入的数据集，因此没有现成的对比模型。尽管如此，我们自己实现了几个基准方法，其准确率如表 4.5 所示，我们可以观察到：

- 与主流数据集的改进不同，所提出的模型在习语丰富的句子上显示出它们的优势，获得了更大的改进。
- 另外，iTLSTM-Lo 表现比 iTLSTM-Mo 略差，同时仍然超过基线模型，这也表明习语的形态学敏感（variation-sensitive）模型（iTLSTM-Mo）可以进一步提高性能。

**表 4.4　不同模型在四个主流数据集上的准确率**

| 模型 | MR | SST-1 | SST-2 | SUBJ |
|---|---|---|---|---|
| NBOW | 77.2 | 42.4 | 80.5 | 91.3 |
| RAE[90] | 77.7 | 43.2 | 82.4 | - |
| MV-RNN[65] | 79.0 | 44.4 | 82.9 | - |
| RNTN[57] | - | 45.7 | 85.4 | - |
| DCNN[9] | - | 48.5 | 86.8 | - |
| CNN-multichannel[91] | 81.5 | 47.4 | 88.1 | 93.2 |
| CharLSTM | 75.2 | 44.0 | 85.2 | 90.2 |
| TLSTM | 78.7 | 48.5 | 86.1 | 91.0 |
| Cont-TLSTM | 79.5 | 48.9 | 86.4 | 91.7 |
| iTLSTM-Lo | 81.6 | 49.9 | 87.7 | 93.2 |
| iTLSTM-Mo | **82.5** | **51.2** | **88.2** | **94.5** |

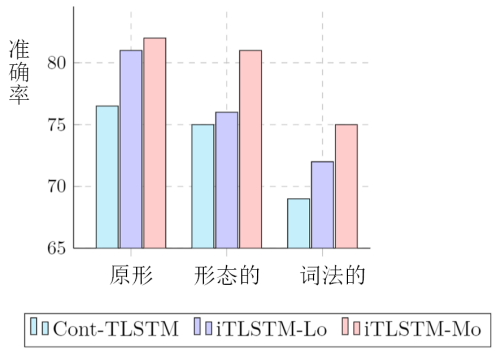



**图 4.4　不同模型在 iSent 数据集上的性能**

**表 4.5　不同模型在 iSent 数据集上的准确率**

| 模型 | 训练 | 校验 | 测试 |
|---|---|---|---|
| NBOW | 80.9 | 77.1 | 74.5 |
| LSTM | 87.5 | 76.9 | 75.0 |
| BiLSTM | 93.4 | 76.8 | 76.3 |
| CharLSTM | 92.4 | 75.1 | 74.4 |
| TLSTM | 88.2 | 75.3 | 74.9 |
| Cont-TLSTM | 90.8 | 76.2 | 75.5 |
| iTLSTM-Lo | 89.5 | 80.4 | 79.1 |
| iTLSTM-Mo | 92.7 | 82.2 | **81.0** |

### 3. 分析

在本节中，我们将根据表4.1中描述的习语的三个属性提供更详细的定量和定性分析：灵活性，不可见性和习语性。

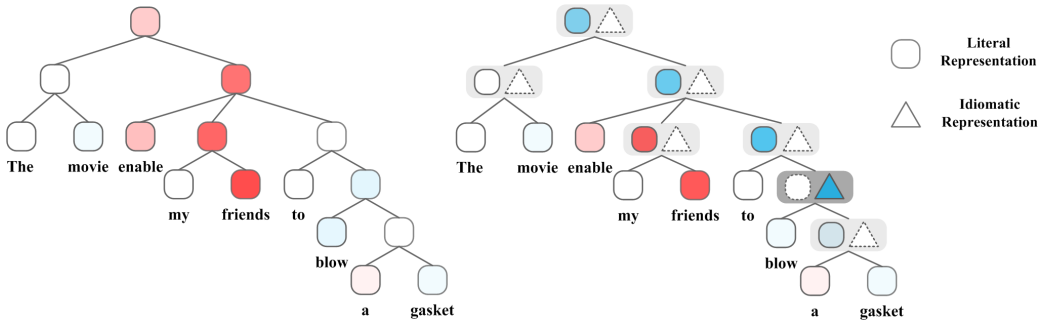



**图 4.5　不同结点处的情感类别的变化**

### (1)　灵活性

除了测试集的总体准确率之外，我们还列出了不同模型在测试集的不同部分所实现的性能：原始、形态变体和词汇变体，它们代表了不同类型的变化，其统计数据在表 4.3中呈现。我们可以从图4.4观测到，这两个习语感知模型在测试集的原始部分上比 Cont-TLSTM 获得了更好的性能，这表明在句子建模过程中理解习语的重要性。此外，iTLSTM + Mo 在测试集上优于其他两个模型，表明基于形态学的模型对于习语变体建模的有效性。

### (2)　不可见性和习语性

以上的实验结果表明了我们模型的有效性。为了了解引入的习语检测器如何有助于提高性能。我们分析了 iTLSTM-Mo 正确预测的所有 157 个样本发现，Cont-TLSTM 模型错误预测了 120 个，原因是因为忽略了习语的隐含意义。例如，如图4.5所示，我们随机抽样一个句子并分析其变化，预测树不同结点的情感类别，红色和蓝色分别代表正面和负面情感，其中较暗表示较高的置信度。虚线三角形或方框表示未被检测器选中。

句子“The movie enable my friends to blow a gasket （这部电影使我的朋友们大吃一惊）”有负面情感。由于忽略了习语“blow a gasket（勃然大怒）”所表达的信息，Cont-TLSTM 给出了错误的预测。相比之下，我们的模型（iTLSTM-Mo）正确地检测到这个习语（$\alpha = 0.97$），其含义在最终情感预测中起主要作用。

**表 4.6　不同类型的非组合短语示例**

| PN | VP | NP | AP |
|---|---|---|---|
| *3 D* | *thumb down* | *short cuts* | *at least* |
| *Holly Bolly* | *go on* | *deja vu* | *at once* |
| *star wars saga* | *rips off* | *black comedy* | *all in all* |
| *Barry Skolnick* | *fly over* | *femme fatale* | *not only* |
| *Apollo 13* | *fall apart* | *cuts corners* | *at times* |

### (3)　不可合成性短语的检测

除了习语之外，我们发现检测器还可以捕捉其他类型的非组合短语[①]。我们粗略地总结了所有五个校验集中检测器所捕捉的非组合短语，并将它们列在表4.6中。PN、VP、NP 和 AP 分别表示专有名词、名词短语、动词短语和副词短语。.

从表中我们可以看出，这些短语中的大多数都暗示了对某事物的情感立场：“thumbs down(拇指向下)”,对理解“动词短语”和“副词短语”等句子至关重要。例如，句子“More often than not, this mixed bag hit its mark(通常情况下,这种好坏参半的情况达到了目的)”具有正面的情感。Cont-TLSTM 更加注重“not”这个词而没有意识到它属于中性情感搭配“more often than not （more often than not）”。相比之下，我们的模型将这种搭配视为具有中性情感的整体，这对最终预测至关重要。

在本节中，我们基于两种语言学观点，在句子级语义表示的语境中对习语进行理解。为了将我们的模型应用到现实世界的任务中，我们构建了一个相当大的富含习语的情感分类数据集。我们精心地进行了实验设计和案例分析，验证了我们提出的模型的有效性。

[①]一个习语本身就是一个非组合短语。

## 4.3　丰富语义组合关系的建模

### 4.3.1.　研究动机

当我们对句子中所包含的词语进行合成时，现有模型存在的另一个不足之处在于，词语之间有着复杂的合成情况，然而却用了一套参数去建模，这会引起模型的欠拟合问题。例如，形容词和名词的合成与动词和名词的合成是不一样的，不同成分的语义合成情况往往随着上下文不同而发生改变。此外，许多语义合成现象需要更强大的合成机制[92]，如语义习语性。因此，普通的树结构神经网络无法很好地解决这个问题。如图4.6表示的是普通的树结构神经网络图，图中的 NP 和 VP 分别代表名词和动词短语，$\theta$ 表示复合函数的参数，虽然 NP 和 VP 是两种短语，但他们的参数是共享的。

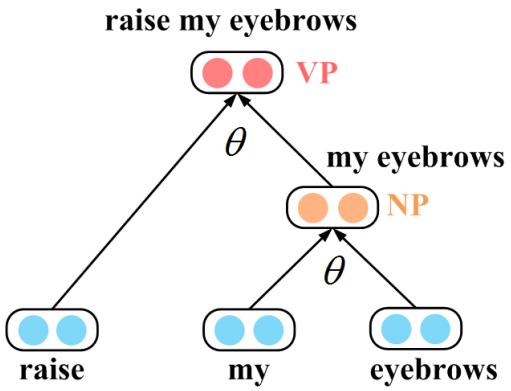



**图 4.6　树结构的神经网络**

为了解决这个问题，一些研究工作提出使用多套参数不共享的语义合成函数，并且，在不同的情况选取不同的合成函数。这样的方法确实能缓解一部分欠拟合问题，但是仍然存在以下缺点：1）多个合成函数往往引入大量的参数，从而容易导致模型的过拟合；2）不同函数使用的触发条件依赖于专家经验，而预定义的合成功能不能涵盖所有合成规则；

我们解决这个问题沿着这样的一个思路：不再使用共享且静态的参数，而是根据不同的上下文动态地生成参数。具体说来，我们提出了一个动态语义合成网络，它由两个子模块构成：元网络（Meta Network）和主网络 (Main Network)。其中元网络用来生成与上下文相关的参数给主网络，而主网络接受输入信号并

且将其转化为特征向量提供给终端任务。具体来说，我们构建了基于两种树结构神经网络模型：递归神经网络（Tree-RecNN）[65] 和树结构长短期记忆神经网络（Tree-LSTM）[10]。我们的工作受近期动态参数预测工作 [93–95] 的启发。元网络用于从不同的组成规则提取共享元知识，并动态生成依赖于上下文的合成函数。因此，我们模型的合成函数随位置、上下文和样本的不同而变化。然后，主网络将这些动态生成的参数应用于当前输入信息。元网络和动态网络都是可微分的，因此可以以端到端的方式训练整个网络。另外，为了降低整个网络的复杂性，我们以模仿低秩矩阵分解的方式定义动态权重矩阵。我们在多个文本分类任务上对提出的模型进行性能校验，实验结果显示了动态参数生成的有效性。

我们的贡献可以总结为：1）我们提供了一个关于如何将句子中的词语组合在一起，从而更好地构成句子表示的新视角。我们不是直接使用可学习的参数化合成函数，而是引入元神经网络，它可以生成一个组合网络，从而可以动态组合树结构上的成分。2）实验结果表明，在不增加模型参数的前提下，该结构具有“学习学习”（Learn to learn）的特性，具有较强的表达能力。3）我们给出了一个详细的定性分析，从语义和句法的角度直观地解释了我们的模型是如何工作的。

### 4.3.2. 模型

首先，不管是元网络还是主网络都是基于递归神经网络（Recursive Neural Networks）架构建立起来的，不同点在于主网络的参数是由元网络生成的。

在普通的树结构网络中，组成函数在树的所有结点间共享，但是，由于语义组成具有很大的差异性，从而导致了模型欠拟合。为了解决这一问题，我们提出了两种基于树结构的动态合成神经网络，它可以为不同类型的组合状态动态生成不同的参数。图4.7展示了动态语义合成的神经网络框架图，它包含两个组件:(1) 元网络；和 (2) 带有动态参数的主网络。

具体来说，我们提出了两个分别基于 RecNN 和 TreeLSTM 的元网络来生成与上下文相关的合成函数。

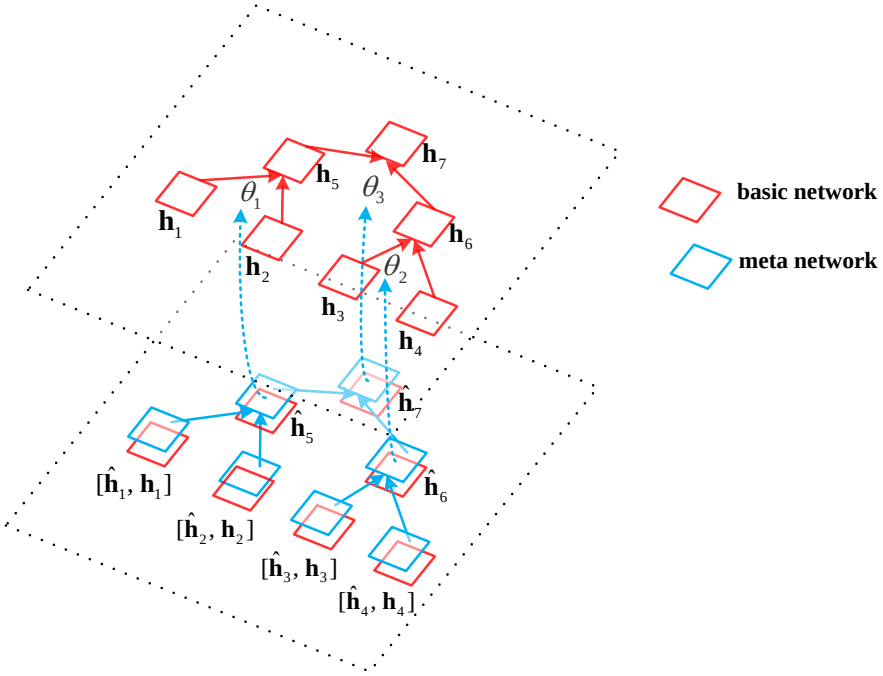



**图 4.7　动态语义合成神经网络**

### 4.3.3.　基于递归神经网络的动态合成网络

对于主网络，我们把公式 (2.13) 里的递归神经网络的参数 $\mathbf{W}$ 和 $\mathbf{b}$ 换成动态生成的参数 $\mathbf{W}(\mathbf{z}_j)$ 和 $\mathbf{b}(\mathbf{z}_j)$. 其中，这些动态的参数由元网络如下生成：

$$\hat{\mathbf{h}}_j = \tanh\left(\mathbf{W}_m \mathcal{H}_j + \mathbf{b}_m\right), \tag{4.7}$$

$$\mathbf{z}_j = \mathbf{W}_z \hat{\mathbf{h}}_j \tag{4.8}$$

其中 $\mathcal{H}_j = \mathbf{h}_j^l \oplus \mathbf{h}_j^r \oplus \hat{\mathbf{h}}_j^l \oplus \hat{\mathbf{h}}_j^r \in \mathbb{R}^{2m+2d}$, $\mathbf{W}_m \in \mathbb{R}^{m\times(2d+2m)}$ 并且 $\mathbf{b}_m \in \mathbb{R}^m$ 是元网络的参数; $\mathbf{W}_z \in \mathbb{R}^{z\times m}$ 是一个矩阵；$\oplus$ 表示拼接操作。

为了减小上述计算的开销，我们定义了一个低秩因子表示权值的动态参数，它类似于奇异值分解。动态参数 $\mathbf{W}(\mathbf{z}_j)$ 和 $\mathbf{b}(\mathbf{z}_j)$ 可以被如下公式计算得出：

$$\mathbf{W}(\mathbf{z}_j) = \begin{bmatrix} P_l \mathbf{D}(\mathbf{z}_j) Q_l \\ P_r \mathbf{D}(\mathbf{z}_j) Q_r \end{bmatrix} \tag{4.9}$$

$$\mathbf{b}(\mathbf{z}_j) = \begin{bmatrix} B_l \mathbf{z}_j \\ B_r \mathbf{z}_j \end{bmatrix} \tag{4.10}$$

其中 $P \in \mathbb{R}^{d\times z}$, $Q \in \mathbb{R}^{z\times d}$, 并且 $\mathbf{D}(\mathbf{z}_t) \in \mathbb{R}^{z\times z}$ 是向量 $\mathbf{z}$ 对应的对角阵。

因此，基于 RecNN 的动态组合网络需要 $(6dz + mz)$ 参数，而普通 RecNN 需要 $(2d^2 + d)$ 参数。使用较小的 $z$ 和 $m$，基于 RecNN 的动态组合网络需要的参数比普通 RecNN 少。例如，如果我们设置 $d = 100$ 和 $z = m = 20$，我们的模型需要 $12,400$ 参数，而普通模型需要 $20,100$ 参数。

### 4.3.4. 基于 TreeLSTM 结构的动态合成网络

同样，我们还使用一个轻量级的元网络来通过动态参数 $\mathbf{W}(\mathbf{z}_j)$ 和 $\mathbf{b}(\mathbf{z}_j)$ 生成式 Eq.(2.11) 中的静态参数。基于 TreeLSTM 结构的动态合成网络中所使用的元网络是一个轻量级的 TreeLSTM。对于主网络，我们把公式2.11里的树结构的参数 $\mathbf{W}$ 和 $\mathbf{b}$ 换成动态生成的参数 $\mathbf{W}(\mathbf{z}_j)$ 和 $\mathbf{b}(\mathbf{z}_j)$。其中，这些动态的参数由元网络如下生成：

$$\mathbf{W}(\mathbf{z}_j) = \left[\mathbf{W}^g, \mathbf{W}^i, \mathbf{W}^{f^l}, \mathbf{W}^{f^r}, \mathbf{W}^o\right] \tag{4.11}$$

$$\mathbf{b}(\mathbf{z}_j) = \left[\mathbf{b}^g, \mathbf{b}^i, \mathbf{b}^{f^l}, \mathbf{b}^{f^r}, \mathbf{b}^o\right], \tag{4.12}$$

其中，$* \in \{c, o, i, f^l, f^r\}$,

$$\mathbf{W}^*(\mathbf{z}_j) = \begin{bmatrix} P_x^* \mathbf{D}(\mathbf{z}_t) Q_x^* \\ P_l^* \mathbf{D}(\mathbf{z}_t) Q_l^* \\ P_r^* \mathbf{D}(\mathbf{z}_t) Q_r^* \end{bmatrix}, \tag{4.13}$$

$$\mathbf{b}^*(\mathbf{z}_j) = \begin{bmatrix} B_x^* \mathbf{z}_t \\ B_l^* \mathbf{z}_t \\ B_r^* \mathbf{z}_t \end{bmatrix}, \tag{4.14}$$

其中，$P \in \mathbb{R}^{5d \times z}$, $Q \in \mathbb{R}^{z \times 3d}$, $B \in \mathbb{R}^{5d \times z}$, 和 $\mathbf{D}(\mathbf{z}_t) \in \mathbb{R}^{z \times z}$ 是 $\mathbf{z}$ 的对角矩阵。

使用较小的 $z$ 和 $m$，我们的动态 TreeLSTM 需要的参数与标准 TreeLSTM 相近。

### 4.3.5. 动态组合网络的应用

在本节中，我们展示了动态合成神经网络在两个经典的 NLP 任务中的应用。

#### 1. 文本分类

文本分类的过程是，给定一个句子 $x$，模型应该从预定义的标签集合 $y$ 预测标签 $\hat{y}$。根据上一节的描述，我们可以计算出短语在结点 $j$ 上的分布式表示 $\mathbf{h}_j$:

$$\mathbf{h}_j = \text{DC-TreeNN}(\mathbf{x}_j, \mathbf{h}_j^l, \mathbf{h}_j^r, \theta) \tag{4.15}$$

在这个递归过程之后，使用根结点上的隐藏状态 $\mathbf{h}_R$ 作为句子表示，然后用一个 Softmax 分类器预测类之间的概率分布。

$$\hat{\mathbf{y}} = \text{Softmax}(\mathbf{W}_t \mathbf{h}_R + \mathbf{b}_t) \tag{4.16}$$

其中 $\hat{\mathbf{y}}$ 是预测概率，$\mathbf{W}_t$ 和 $\mathbf{b}_t$ 是分类器的参数。

#### 2. 文本语义匹配

在许多自然语言处理 (NLP) 任务中，一个常见的问题是对一对文本的相关性建模。在本节中，我们将展示如何有效地使用动态合成神经网络来建模两个句子之间的语义关系。如图4.8所示，给定 $x_a$ 和 $x_b$ 两句话，每个句子 $\mathbf{h}_R$ 的表示可以由一个基本 TreeNN 来计算。

$$\mathbf{h}_R^{(a)} = \text{DC-TreeNN}(\mathbf{x}_R, \mathbf{h}_R^l, \mathbf{h}_R^r, \theta_a) \tag{4.17}$$

$$\mathbf{h}_R^{(b)} = \text{DC-TreeNN}(\mathbf{x}_R, \mathbf{h}_R^l, \mathbf{h}_R^r, \theta_b) \tag{4.18}$$

其中 $R$ 表示树的根结点。$\theta_a$ 和 $\theta_b$ 由共享的元 TreeNN 生成。然后，将每个句子的表示输入一个多层感知器中，得到一个统一的表示，最后进行关系分类。

这种主网络的参数和样本相关，而元网络的参数共享的模式可以确保，一方面我们可以动态地建模语义组合的多样性，另一方面我们可以捕获不同样例之间的一般规则。

### 4.3.6. 实验与分析

为了对我们的模型进行综合评价，我们在五个文本分类任务和一个语义匹配任务上进行实验评估。

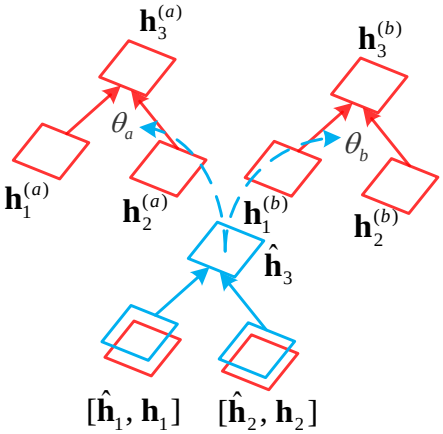



**图 4.8　DC-TreeNN 匹配网络**

### 1. 实验设置

#### (1) 损失函数

给定一个句子 (或一个句对) 及其标签，神经网络的输出是不同类别的概率。对网络参数进行训练，使预测的和真实的标签分布的交叉熵（Cross-Entropy）最小。为了最小化目标函数，我们使用 AdaGrad[86] 作为优化器。

#### (2) 参数定义

所有模型的词嵌入都是用 GloVe 向量[87] 初始化的。其他参数用范围为 $[-0.1, 0.1]$ 的均匀分布随机初始化。最后的超参数如下。初始学习率为 0.1。参数的正则化权重为 $1E-5$，其他参数的值的定义如表4.7所示。其中，$m$、$d$ 分别表示元 TreeNN 和基本 TreeNN 的隐藏状态。$e$ 和 $z$ 表示嵌入向量 $\mathbf{x}$ 和控制向量 $\mathbf{z}$ 的大小。我们提出的两个模型在所有数据集上的设置都是相同的，除了 SICK 数据集（在表4.7中，括号内和括号外的数字分别对应 DC-RecNN 和 DC-TreeLSTM 模型的设置）。

对于数据集中的所有句子，我们使用成分解析器[88] 加以解析，从而获取所需要的树结构表示。

### 2. 对比方法

- RecNN[65]: 具有标准合成函数的递归神经网络。

**表 4.7　动态合成神经网络的超参数设置**

| 超参数 | IE | MR | SST | SUBJ | QC | SICK |
|---|---|---|---|---|---|---|
| d | 100 | 100 | 100 | 100 | 100 | 50(30) |
| e | 100 | 100 | 100 | 100 | 100 | 50(50) |
| m=z | 40 | 30 | 30 | 20 | 40 | 40(20) |

- RNTN[57]: RNTN 是一个具有神经张量层的递归神经网络，它可以建模两种成分之间的强相互作用。

- MV-RecNN[65]: MV-RecNN 将解析树中的每个词语和较长的短语同时表示为一个向量和一个矩阵，从而对丰富的组合进行建模。

- TreeLSTM[10]: 树结构长短时记忆单元的神经网络。

### 3. 文本分类

我们在五个不同的文本分类数据集上评估我们的模型：SST-2, MR, QC, SUBJ, IE。关于这五个数据集的详细统计信息列在表2.1中。其评价指标为准确率。

#### (1) 实验结果

如表4.8所示，DC-TreeLSTM 始终比 RecNN、MV-RecNN、RNTN 和 TreeLSTM 有较好的结果，同时可以获得与 CNN 模型非常接近的结果。值得一提的是，与 CNN 相比，我们模型的参数要少得多。在我们的模型中参数的数量大约是 10K，而在 CNN 中参数的数量大约是 400K。与 RecNN 相比，DC-RecNN 具有更好的性能，说明了动态合成函数的有效性。此外，DC-RecNN 和 DC-TreeLSTM 都在 IE（一个涵盖了丰富的合成特性 (习语) 的数据集）数据集有了实质性的提高。在 IE 数据集上的成功可以归功于模型具有更强的表现能力，它可以建模更复杂的语义组合。

**表 4.8　不同模型在五个数据集上的准确率**

| 模型 | IE | MR | SST | SUBJ | QC |
|---|---|---|---|---|---|
| NBOW | 54.6 | 77.2 | 80.5 | 91.3 | 88.2 |
| DCNN | - | - | 86.8 | - | 93.0 |
| CNN-multichannel | - | 81.5 | **88.1** | 93.4 | 93.6 |
| RecNN | 52.0 | 76.4 | 82.4 | 90.6 | 88.8 |
| MV-RecNN | 54.8 | 76.8 | 82.9 | 90.9 | 89.2 |
| RNTN | 55.7 | 75.8 | 85.4 | 92.1 | 88.9 |
| TreeLSTM | 56.0 | 78.7 | 86.9 | 91.0 | 91.6 |
| DC-RecNN | 58.2 | 80.2 | 86.1 | 93.5 | 91.2 |
| DC-TreeLSTM | **60.2** | **81.7** | 87.8 | **93.7** | **93.8** |

**表 4.9　不同模型在 SICK 数据上的准确率**

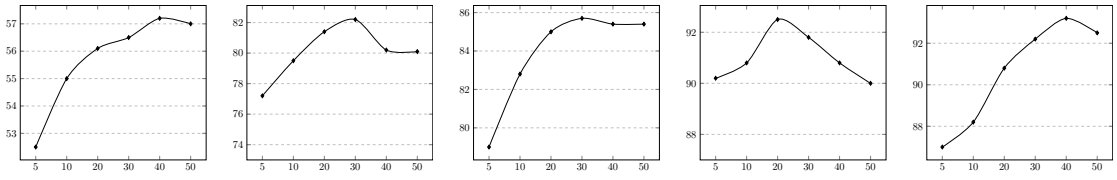
| 模型 | 隐状态 | 训练集 | 测试集 |
|---|---|---|---|
| NBOW | 30 | 96.6 | 73.4 |
| LSTM | 100 | 100.0 | 71.3 |
| RecNN | 30 | 95.4 | 74.9 |
| MV-RecNN | 50 | 95.9 | 75.5 |
| RNTN | 50 | 97.8 | 76.9 |
| TreeLSTM | 50 | 95.9 | 77.5 |
| DC-RecNN | 30 | 96.5 | 77.9 |
| DC-TreeLSTM | 50 | 98.5 | **80.2** |

### 4. 文本语义匹配

我们选择了由 Marelli 等人[61] 提出的（(Sentences Involving Compositional Knowledge （SICK)）的数据集，以验证文本语义匹配的效果。

#### (1) 结果

实验结果如表4.9所示，其中 NBOW、LSTM、RecNN 和 RNTN 的准确率引用了[96,97] 所报告的结果。为了公平比较，我们使用相同的设置来训练我们的模型以及对比模型。我们可以看到 DC-RecNN 和 DC-TreeLSTM 都优于对比模型。与 RecNN (TreeLSTM) 相比，DC-RecNN (DC-TreeLSTM) 的效果提高了 3%(2.7%)。我们认为这一突破主要归功于动态合成机制，它使我们的模型能够捕获各种语法模式 (我们将在稍后讨论)，因此能更准确地理解句子。



(a) IE　(b) MR　(c) SST　(d) SUBJ　(e) QC

**图 4.9　DC-RecNN 在不同向量 $z$ 下的准确率**

### 5. 定性分析

在我们的模型中，向量 **z** 控制网络参数的预测过程，其维度决定了模型参数的个数。接下来，我们将研究如何控制向量 **z** 来观测模型性能的变化。

#### (1) 维度的影响

图4.9表示的是 DC-RecNN 在不同维度 $[5, 10, \ldots, 50]$ 的控制向量 **z** 上的性能。我们有以下发现：

- 对于这 5 个数据集，即使向量 **z** 的大小减小到 5，该模型也可以获得相当好的性能。特别是，在数据集 QC 上，模型以一个非常小的元 Tree-RecNN 获得 87.0% 准确率 [②]，暗示小元网络可以用于生成一个更强大的合成函数来高效地对句子建模。

- 当处理具有很多标签的数据集时，更长的向量会带来更好的性能。例如，当 $z = 40$ 时，IE 和 QC 数据集的性能达到最大值，而对于其他三个数据集 MR、SST 和 SUBJ，模型分别在 $z$ 等于 30、30 和 20 时获得最佳性能。

#### (2) 神经元行为的可理解性

如前几节所述，我们知道合成函数是在树的子结点上进行更改的，由一个潜在的向量 $z$ 控制。为了直观地理解控制向量 $z$ 的工作原理，我们在每个结点上设计了一个实验来检验 **z** 的神经元的行为。我们将 $z_{jk}$ 称为结点 $j$ 处 $k$-神经元的激活值，其中 $j \in \{1, \ldots, N\}$，和 $k \in \{1, \ldots, z\}$。然后我们从实验中所使用的数据集中随机抽取校验集上的一些句子。通过可视化向量 $\mathbf{z}_j$ 并分析其最大激活值，我们可以发现当前神经元所关注的模式。

表4.10说明了多个可解释的神经元和一些能够激活这些神经元的代表性词语或短语。

我们可以观察到:

- 对于文本分类等简单任务，元网络将收集有用的语义信息到合成函数的生成过程中。这些组合之前的语义偏置是特定于任务的。例如，第 21 个神经

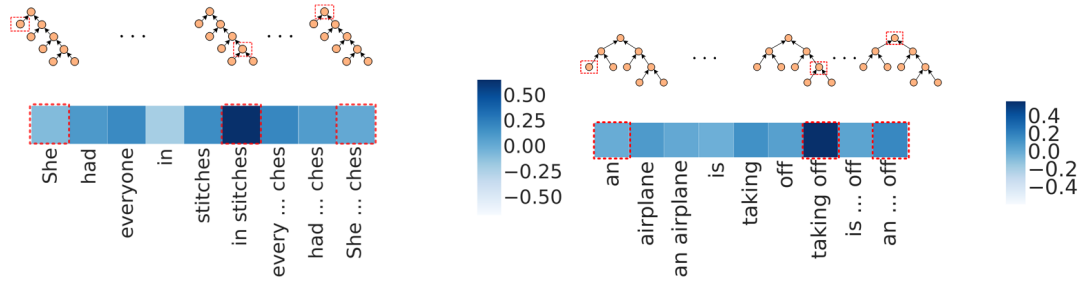
[②]在相同的参数设置下，RecNN 获得 74% 的准确性

**表 4.10　多个可解释的神经元以及这些神经元捕捉到的词语/短语**

| 类型 | | 神经元 | 例子 | 解释 |
|---|---|---|---|---|
| 语义 | 词汇 | 17 | fun, glad, terrific, wonderful | Words related to sentiment |
| | 短语 | 21 | pick holes, see red, in stitches | Phrases related to sentiment |
| 句法 | 名词短语 | 45 | blond boy, pink shirt, green grass | Containing modifiers related to color |
| | 动词短语 | 27 | waking up, take off, pulling up | Phrases constructed by light-verb |
| | | 11 | slicing a potato, playing guitar | Verb-object phrases |
| | 介词短语 | 13 | on a track, in rocky area, on a stage | Phrases related to places |

元对情感词汇更敏感，可以理解为一个哨兵，它告诉基本神经网络一个信息短语即将到来，在语义合成的过程中应该给予更多的关注。图4.10-(a) 给出了 DC-TreeNN 模型的 $\mathbf{z}_{21}$ 神经元行为的可视化。对于给定的句子 “`She had everyone in stitches`（她让每个人都笑了）”，神经元已经意识到这个习惯搭配 “`in stitch`” 是一个关键短语，它对于最终的情感预测至关重要。

- 对于更复杂的任务，如语义匹配，对句法结构的良好理解是至关重要的。在这种情况下，我们发现元网络可以捕获一些句法信息。例如，第 27 个神经元关注由轻动词（light-verb）构成的短语。如图4.10-(b) 所示，动词短语 “`taking off`” 被视为可执行合成操作的短语，这对于判断句子对 “`An airplane is taking off/A plane is landing`（一架飞机正在起飞/飞机正在着陆）” 之间的语义关系非常重要。




(a) 第 21 个神经元的行为　　(b) 第 27 个神经元的行为

**图 4.10　DC-TreeNN 模型 $\mathbf{z}_{21}$ 和 $z_{27}$ 神经元的行为热力图**

## 4.4　语义组合拓扑结构的动态学习

### 4.4.1.　研究动机

句子表示学习需要深入理解句子的复杂结构[98] 和词语的语境化表示[99]。最近，以变形器 (Transformer) 为典型的自注意力的机制取代了神经网络中的经典合成函数（CNN，RNN）。它在机器翻译[53] 和句法分析[100] 方面取得了目前最好的成果。Transformer 的成功可归因于其非局部结构偏置（Non-local bias），其中任何一对词之间的依存关系都可以被建模[38,101]。这个性质允许它**动态学习句子的句法和语义结构**[53]。尽管 Transformer 取得了成功，但缺乏局部偏置（相邻词语中存在的局部依赖性[38,39]）限制了其学习语境化表示的能力。

与 Transformer 相比，另一项工作旨在使用图神经网络 (Graph neural networks) 对序列进行建模。虽然图神经网络能够通过编码属性特征灵活地学习局部上下文信息[102]，但是目前还不太清楚如何有效地利用图神经网络进行序列学习，因为没有单一句子结构表示适合所有任务（句子的结构是和上下文相关的）。

一个常见的选择是使用句子中词语之间的句法依存关系来确定句子的图结构，这种情况下模型退化为具有树形结构的神经网络[10]。而受 Transformer[53] 的启发，我们的目标是通过学习依赖于任务的图表示来改进这个固定的结构，以便更好地捕获和任务相关的依赖关系。

在这里，我们从两个研究领域中汲取经验，并提出一个可以从其互补优势中受益的模型。一方面，Transformer 的成功启发我们探索动态的图结构。我们不是为句子定义硬编码的图结构，而是将非局部偏置纳入图神经网络，在不同任务

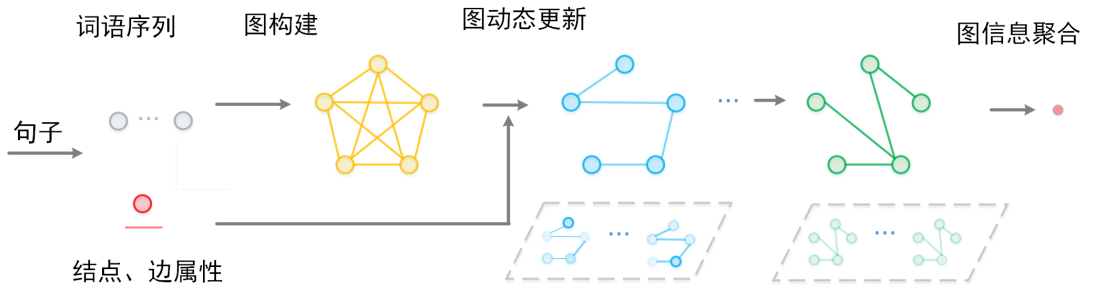



**图 4.11　句子对应动态图的更新过程**

上动态学习句子结构。另一方面，图神经网络框架本身的优点是通过编码结点或边属性（attributes）来提供丰富的局部信息，这可以弥补 Transformer 的不足。

因此，我们通过将自注意力 (self-attention) 扩展到图神经网络来提出一种语境化的非局部网络，它具有两种高度灵活表示的方式。首先，属性的表示（结点和边的特征编码）；第二，图本身的结构（动态学习任务相关的结构）。这两点使我们能够根据词语的语境化表示和句子的复杂结构更好地学习序列的表示。

我们在十个序列学习任务上进行了大量实验，包括文本分类、语义匹配、和序列标注。实验结果表明，我们提出的方法优于基线方法，并在所有任务中取得较好的结果。此外，我们能够发现词语之间依赖于任务的结构，并提供了良好的解释性。

我们总结了贡献如下:

1. 我们从模型局部偏置的角度分析了序列学习（句子结构和语境化词表示）的两个挑战，这鼓励我们找到 Transformer 和图神经网络之间的互补性。

2. 我们借鉴了 Transformer 和图神经网络的互补优势，提出了一个语境化的非局部神经网络 ($CN^3$)，其中句子结构可以快速地学习，句子中的词语可以灵活地进行语境化。

3. 我们以统一的方式对现有的序列学习方法进行了全面分析，并在许多自然语言理解任务上进行广泛的实验。实验结果表明，我们提出的模型在这些任务中取得了较好的结果，并为用户提供了更好的解释性。

### 4.4.2. 模型

为了更好地学习序列的表示，用词语[99] 和复杂结构[98] 的上下文表示，我们提出了语境化相关的非局部神经网络（CN $^3$），它将非局部偏置纳入图神经网络。图4.11是非局部神经网络动态更新句子结构的过程。CN $^3$ 不仅可以支持局部上下文信息的编码，还可以动态学习任务相关的结构。

语境化的非局部网络涉及两个步骤。第一步，从句子构造图，并从属性中引入语境化信息。第二步，动态更新特定任务的图结构。

#### 1. 图的构造

框架的第一部分是从句子构造有意义的图结构，以便编码不同类型的局部上下文信息。

##### (1) 结点

给定句子 $X = x_1, x_2 \cdots x_n$，我们将每个词语 $x_i$ 视为一个结点。对于每个结点 $i$，我们定义结点属性（$\mathcal{S}_{v_i} = v^1, \cdots, v^{|\mathcal{S}_v|}$）作为除了词语标识之外的额外信息。几个典型的结点属性，描述如下：

**位置信息** 句子中每个词语的位置。

**语境化信息** 结点的信息也可以通过短期 (short-term) 语境来丰富，短期语境化可以通过卷积神经网络或长短期记忆网络来获得。引入此信息意味着我们使用局部偏置扩充了模型：相邻单元倾向于提供有用的信息[56,102]。

**标签信息** 诸如 POS 标签的更多知识可以被编码到结点中。

##### (2) 边

对于词语对 $x_i, x_j$ 之间的每条边，我们定义边属性（$\mathcal{S}_{eij} = e_{ij}^1, \cdots, e_{ij}^{|\mathcal{S}_{eij}|}$）作为组合关系，例如词汇关系，从解析树获得的句法依存关系或共现关系。

##### (3) 语境化结点表示

每个词语和属性将首先通过查找表转换为实值向量，如果仅使用词语标识，则结点表示仅仅是词语表示。因此我们通过融合其属性来丰富词语的信息，从而

构造语境化的结点表示。

### 2. 动态图形结构更新

CN [3] 中的非局部偏置被实现为任意两个词语之间的成对交互，这使得可以动态地学习面向特定任务的句子结构。具体而言，该模型首先在低维向量中对每个结点进行编码。之后，不同结点通过消息相互通信，更新图结构和结点表示。在结点由低维向量表示之后，我们创建多个消息传递层 $l = 1, 2, \ldots, L$，用来更新图结构或结点表示。

#### (1) 图结构更新

我们通过促进不同结点之间的通信来更新图结构。具体来说，令 $\alpha_{ik}$ 表示结点 $i$ 和 $k$ 之间关系的强度。我们根据以下公式更新 $\alpha_{ik}$

$$s_{ik} = f(\mathbf{h}_k^l, \mathbf{h}_i^l, \mathbf{v}_i, \mathbf{v}_k, e_{ik}) \tag{4.19}$$

$$= \mathbf{u}^T \tanh(\mathcal{W}[\mathbf{h}_k^l, \mathbf{h}_i^l, \mathbf{v}_i, \mathbf{v}_k, e_{ik}]) \tag{4.20}$$

$$\alpha_{ik} = \mathrm{Softmax}(s_{ik}), \tag{4.21}$$

其中，$\mathbf{v}$ 表示结点属性。$e_{ik}$ 表示结点 $v_i$ 和 $v_k$ 之间的边属性，$\mathbf{h}_i^l$ 是图层 $l$ 中结点 $v_i$ 的表示。$\mathbf{u}$ 和 $\mathcal{W}$ 是可以通过反向传播学习的参数。$\mathbf{h}^0$ 表示词表示向量。

#### (2) 结点更新

一旦图结构被更新，对于每个结点 $k$，它首先聚合来自其邻居的信息。特别地，令 $\tilde{\mathbf{h}}_k^l$ 表示从第 $l$ 个图层中结点 $v_k$ 的邻居收集的信息，可以简单地计算为：

$$\tilde{\mathbf{h}}_k^l = \boldsymbol{\alpha_i} \mathcal{H}^l = \sum_i^m \alpha_{ik} \mathbf{h}_i^l \tag{4.22}$$

然后，我们根据当前结点表示 $\mathbf{h}_k^l$ 以及从其邻居 $\tilde{\mathbf{h}}_k^l$ 汇总的信息更新结点表示。在这里，我们选择基于门控机制的更新方法。

$$\mathbf{h}_k^{l+1} = \mathbf{g} \odot \tilde{\mathbf{h}}_k^l + (1 - \mathbf{g}) \odot \mathbf{h}_k^l \tag{4.23}$$

其中 $\odot$ 表示对位乘法，和

$$\mathbf{g} = \sigma(\mathcal{W} \mathbf{h}_k^l + \mathbf{b}), \tag{4.24}$$

$\sigma$ 表示 Sigmoid 函数。$\mathcal{W}$ 和 $\mathbf{b}$ 是可学习的参数。

#### 3. 应用层

经过多个图结构更新步骤 ($l = 1, 2, \ldots L$), 我们可以获得最后一层的结点级的表示 $L$：$\mathbf{h}_1^L, ..., \mathbf{h}_m^L$，用于不同的自然语言任务。在这里，我们将展示如何在结点级别使用这些结合表示来实现词性标注、文本组块识别、命名实体识别和图级别的文本分类、语义匹配任务。

##### (1) 结点级表示

此方案的直接应用是序列标记，其目的是将标记序列 $Y = \{y_1, y_2, \ldots, y_T\}$ 分配给文本序列 $X = \{x_1, x_2, \ldots, x_T\}$。每个时间步骤 $\mathbf{h}_i^L$ 的输出可以被视为前一个子序列的表示，然后将其输入到计算相应标签类别分数的 CRF 层。然后，在 CRF 层中，条件概率 $p(Y|X)$ 形式化为：

$$p(Y|X) = \mathrm{CRF}(\mathbf{H}_i^L, \theta^{(c)}) \tag{4.25}$$

CRF 代表 CRF 层，$\theta^{(c)}$ 是可学习的参数。$\mathbf{H}_i^L = \{\mathbf{h}_1^L, \cdots, \mathbf{h}_m^L\}$

##### (2) 图级表示

图级表示的计算,最简单的方法是取所有结点表示的平均值: $\mathbf{h}^L = \frac{1}{m}\sum_i \mathbf{h}_i^L$。然后，可以将表示 $\mathbf{h}^L$ 进一步馈送到 softmax 函数来产生用于文本分类或语义匹配任务的输出标签的概率分布。

### 4.4.3. 实验与分析

在本节中，我们在十项任务中评估了提出的 CN [3] 方法的有效性。我们将首先简要介绍任务和数据集。

#### 1. 任务与数据集

我们在五个文本分类任务，两个语义匹配任务和三个序列标记任务上评估我们的模型。

**表 4.11　不同模型在十个数据集上的性能**

| 模型 | 文本分类 | | | | | 语义匹配 | | 序列标记 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | QC | MR | SST | SUBJ | IMDB | SICK | SNLI | POS | Chunking | NER |
| NBOW | 88.2 | 77.2 | 80.5 | 91.3 | 87.5 | 73.4 | 75.1 | 96.38 | 90.51 | 87.91 |
| 不同结构偏置的神经网络 | | | | | | | | | | |
| CNN | 92.2 | 81.5 | **88.1** | 93.2 | 88.5 | 75.4 | 77.8 | 97.20 | 93.63 | 88.67 |
| RNN | 90.2 | 77.2 | 85.0 | 92.1 | 87.0 | 74.9 | 76.8 | 97.30 | 92.56 | 88.50 |
| LSTM | 91.3 | 77.7 | 85.8 | 92.5 | 88.0 | 76.3 | 77.6 | 97.55 | 94.46 | 90.10 |
| TreeLSTM | 92.8 | 78.7 | 88.0 | 93.3 | × | 77.5 | 78.3 | - | - | - |
| PDGraph | 93.2 | 80.2 | 86.8 | 92.5 | × | 78.8 | 78.9 | - | - | - |
| 基于注意力机制的模型 | | | | | | | | | | |
| SelfAtt1 | 93.2 | 80.3 | 86.4 | 92.9 | 90.3 | 78.4 | 77.4 | * | * | * |
| SelfAtt2 | 93.5 | 79.8 | 87.0 | 93.1 | 89.2 | 79.8 | 79.2 | - | - | - |
| SelfAtt3 | 90.1 | 77.4 | 83.6 | 92.2 | 88.5 | 76.3 | 76.9 | 96.42 | 91.12 | 87.61 |
| 与任务相关的图模型 | | | | | | | | | | |
| $CN^3{}_{LSTM}$ | 94.2 | 81.3 | 87.5 | 93.5 | 91.5 | 80.2 | 81.3 | 97.12 | 93.81 | 89.33 |
| $CN^3{}_{LSTM+POS}$ | 94.8 | 80.7 | 87.1 | 94.3 | 91.0 | 81.4 | 82.1 | - | - | - |
| $CN^3{}_{LSTM+char}$ | 94.6 | **82.5** | 88.0 | 94.0 | **92.5** | 81.5 | 82.7 | 97.65 | 94.82 | 90.51 |
| $CN^3{}^{Dep}_{LSTM}$ | **95.0** | 81.0 | 87.8 | **94.6** | * | **81.9** | **83.5** | - | - | - |
| $CN^3{}_{LSTM+char+Spell}$ | - | - | - | - | - | - | - | **97.78** | **95.13** | **91.10** |

- **文本分类**: QC（问题分类）；SST（斯坦福情感树库）；MR（二分类的电影评论[103]）；SUBJ（主客观二分类）；IMDB（长文电影评论）。

- **语义匹配**：在这个任务中，给定两个句子 A 和 B，确定这两个句子之间的语义关系。这里我们使用了 SICK[61] 和 SNLI[96] 两个典型的评测数据集。

- **序列标记**：我们分别选择 POS，Chunking 和 NER 作为 Penn Treebank，CoNLL 2000 和 CoNLL 2003 的评估任务。

我们使用斯坦福 NLP 工具包[104] 解析数据集中的句子，以获得依存关系和

**表 4.12　不同模型在 Chunking，NER，POS 三个任务上的性能**

| **模型** | Chunking | NER | POS |
|---|---|---|---|
| Collobert 等人[27] | 94.32 | 89.59 | 97.29 |
| Yang 等人[106] | **95.41** | 90.94 | 97.55 |
| Peters 等人[99] | - | **92.22** | - |
| Yasunaga 等人[107] | - | - | 97.58 |
| Ma 等人[108] | - | 91.21 | 97.55 |
| Ours | 95.13 | 91.10 | **97.78** |

词性标注信息。

### 2. 设置

我们使用随机梯度下降的变体，即 AdaDelta[105] 作为优化器。使用 GloVe 向量[87] 初始化所有词语嵌入的值，对于不在 GloVe 词表中的词，我们将其随机初始化到 $[-0.1, 0.1]$ 的分布中。对于每个任务，我们采用的超参数是通过网格搜索方法得到的并且在校验集上取得最好效果的。

### 3. 定量评估

#### (1) 动态结构学习评价

表4.11显示了 10 种不同任务的不同模型的性能。在表4.11中，“×” 表示相应的模型无法工作（因为句子太长而无法由解析器处理），“$*$” 表示相应的模型不能用于序列标注任务，“$-$” 表示相应的论文里没有该数据集上结果。CN $^{3}$ 的上标表示边属性，而下标表示结点属性。特别地，“$LSTM$” 表示每个结点的信息由 LSTM 进行增强，“ $Spell$” 表示词语的第一个字母是大写或小写，“$Dep$” 表示两个结点在语法依存关系树中具有边的信息。“$POS$” 表示 POS taging 标签，“$char$” 表示字符信息。

使用的基线模型有：**NBOW**: 求和图级表示的词语向量，并将位置嵌入作为结点级表示连接。**CNN**：使用[91] 进行图级表示，使用[63] 进行结点级表示。

**LSTM** 使用[10] 进行图形级表示，使用[109] 进行结点级表示。**TreeLSTM**：树结构 LSTM[10]。**PDGraph**：基于句法依存的预定义图神经网络。[52]。**SelfAtt1**：具有信息聚合功能的自注意力模型（见2.4.5小节）[55]。**SelfAtt2**：具有语义合成功能的自注意力机制（见2.4.5小节）。由[56] 提出。**SelfAtt3**：也称为 Transformer[53]。使用的评价指标是准确率。

我们有以下观察：

- 对于图级别的任务，CN $^{3}$ $_{LSTM}$ 始终优于具有不同结构偏置（序列、树和预定义图）和注意力机制的神经网络，这表明非局部偏置和句子结构偏置的有效性，以及动态学习的本质。特别是，与 PDGraph 相比，CN$^{3}$ $_{LSTM}$ 实现了更好的性能，并且不依赖于外部语法树，能够处理 IMDB 等更长的文本。与 LSTM 相比，CN $^{3}$ $_{LSTM}$ 的改进也表明局部和非局部偏置的功能是互补的。

- 对于结点级任务（序列标记），CN $^{3}$ $LSTM$ 与 CNN（利用多任务学习框架）和 LSTM（额外引入了许多外部特征: 地名词典特征和拼写功能）相比，取得了较好结果。

### (2) 属性评估

$CN^{3}$ 具有能够通过编码结点或边属性信息来对词语进行语境化的优点。这里我们使用上标和下标来引入结点或边属性。表4.11说明：

- 与 SelfAtt3（Transformer）相比，CN $^{3}$ $_{LSTM}$ 获得了实质性的改进，特别是在由大量具有复杂句型的句子构造的 SST 数据集上。对于那些结点级任务，CN $^{3}$ $_{LSTM}$ 超过了 SelfAtt3，我们将其成功归功于动态学习结构和编码短期上下文信息的能力。

- 考虑边属性可以增强性能。我们观察到不同属性的效果是不同的。具体来说，CN $^{3}$ $_{LSTM+char}$ 在 MR 和 IMDB 数据集上实现了最佳性能，而 CN $^{3}$ $^{Dep}_{LSTM}$ 在 QC，SUBJ，SICK 和 SNLI 数据集上的表现获得了最佳效果。原因

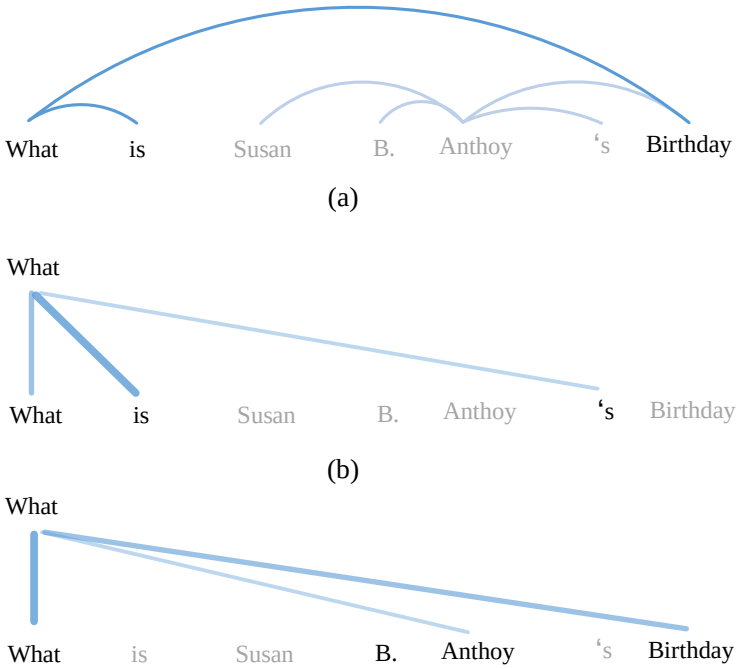



**图 4.12　(a) 句法依存树 (b-c) 不同模型学习到的依存图**

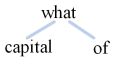
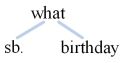
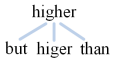
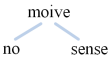
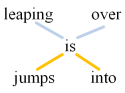
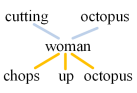
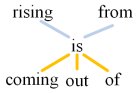
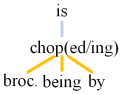
是 QC, SUBJ, SICK 和 SNLI 中的文本更加正式, 同时外部语言工具 (Parser 或 tagger) 也取得了更高的准确率。而对于 MR 和 IMDB，它们包含更多的非正式表达式，例如 “cooool，goood”，导致外部语言工具的失败，尽管它们可以通过字符识别模型来解决。

- 在使用 LSTM 模型引入相同的拼写功能时, 可以进一步改进 CN$^3$ $_{LSTM+char+Spell}$ 在标记任务中的性能。完整的比较，如表4.12所示。表4.12中的对比模型或引入多任务学习方法[27,106]，或利用更多无监督的知识[99,107]，或设计更多手工制作的特征[108]。而我们模型在数据或外部特征方面利用较少的知识，却实现了较好的结果。

#### 4. 定性分析

定性分析的目标是从以下两个方面更好地理解我们的模型：1）模型是否学习到了和任务相关的结构？2）这些属性如何影响结构的学习？我们设计了一系列实验来探索这些问题。

**表 4.13　不同任务生成的多个可解释子结构**

| | 可解释的子结构 | | | | 说明 |
|---|---|---|---|---|---|
| **语义** |  |  |  |  | 关键句型 |
| **句法** |  |  |  |  | 语义匹配任务的关键子图对 |

### (1)　任务依赖结构分析

由于所提出的模型可以通过计算边权重来明确地学习到每个词语对之间的关系，因此我们在不同任务（文本分类和语义匹配）中随机抽样几个示例并可视化它们的学习结构。

表4.13给出了多个可解释的子结构。我们观察到以下几点：

- 简单任务，如文本分类，通常组织句子中的词语，以便准确地表达某些语义信息。例如，对于问题分类任务，由疑问词 “what” 构造的信息模式很容易学习到。对于情感分类任务，词语 “movie” 很容易连接到情感词，如 “terrible”，“no sense”。

对于句法结构的理解更重要的任务，例如语义匹配，我们发现，我们的模型更有可能学习到句法信息。例如，对于句子对 “A man is rising from a swamp（一个人正从沼泽地里爬起来） /A man is coming out of the water（一个人从水里出来了）”，我们的模型得知 “is rising from” 和 “is coming out of” 分别是两个句子中的信息模式。这对于准确预测句子对的关系至关重要。

### (2)　属性分析

如上所述，属性（词性或依存关系）将影响不同结点之间关系的学习过程。为了获得更好的直观理解，我们从数据集（QC）中随机选取样本，并比较 $\mathrm{CN}^{3}_{LSTM}$ 与 $\mathrm{CN}^{3}\,{}^{Dep}_{LSTM}$ 所学习到的词语之间的依赖性。

如图4.12-(b,c) 所示，给定句子，$\mathrm{CN}^{3}\,{}^{Dep}_{LSTM}$ 对此类型做出正确的预测，但是

CN $^{3}$ $_{LSTM}$ 预测失败。我们注意到以下几点。

1）借助边属性，可以在图形结构中构建更多有用的模式。例如，“What”强烈地连接到“What”和“birthday”，因此它捕获了潜在的句子模式“What...birthday”，这是预测这个问题类型的关键。

2）不同词语之间的关系是与任务相关的，而不是与预定义的句法依存关系完全一致。虽然我们的模型得到暗示，依存解析器中的“What”和“is”之间存在很强的联系，如图4.12(a) 所示，但 CN $^{3}$ $^{Dep}_{LSTM}$ 认为它对当前判断没有帮助，因此没有利用该结构信息。

## 4.5　本章小结

在章节中，我们研究了语义合成中的三个问题：1）如何建模包含习语的句子？我们基于两种语言学观点，在句子级语义表示的语境中对习语进行理解。为了将我们的模型应用到现实世界的任务中，我们构建了一个相当大的富含习语的情感分类数据集，精心地进行了实验设计和案例分析，以评估我们提出的模型的有效性。

2）如何解决合成的多样性与函数单一性导致的表示能力不足的问题？我们提出了一种元神经网络，它可以生成一个合成网络，在树结构上动态地合成成分语义。合成函数的参数因位置和样本的不同而不同，并且它允许对输入进行更复杂的操作。为了评估我们的模型，我们选择了两个经典自然语言处理任务，共涉及 6 个数据集。定性和定量实验结果验证了我们所提出来的模型的有效性。

3）如何动态学习句子的结构而不是预先指定？我们首先从模型的局部偏置的角度分析序列学习的两个挑战，这促使我们研究 Transformer 和图神经网络的互补性质。然后，我们利用它们的互补优势来提出一个情境化的非局部神经网络。实验结果表明，学习任务依赖的句子结构和语境化词表示对许多 NLP 任务至关重要。

# 第 5 章　句对表示学习

## 5.1　引言

词语和句子的表示学习在许多自然语言处理任务中得到了广泛的使用，例如文本分类[9]、自动问答和机器翻译[35] 等等。在这些任务中，有一类问题是对句对的相关性/相似性建模，这也称为文本语义匹配。最近，基于深度学习的模型广泛应用于文本语义匹配，并取得了一些重大进展[25,110,111]。根据两个句子之间相互作用的程度，现有的模型可以分为三类。

**(1)　弱交互模型**

一些早期的工作侧重于句子级别的交互，例如 ARC-I[110]，CNTN[111] 等等。这些模型首先分别用神经网络的一些基本（NBOW）或高级（RNN，CNN）部件编码两个序列得到单个句子的表示，然后使用这两个句子的向量计算匹配分数。在这种模型中，两个句子在到达最后阶段之前没有相互作用。

**(2)　半交互模型**

一些改进的方法侧重于利用多粒度表示进行交互学习（词语，短语和句子级别），例如 MultiGranCNN[112] 和 Multi-Perspective CNN[113]。另一种模型使用注意力机制通过依赖于另一个句子的表示来获得一个句子的表示，例如 ABCNN[114]，Attention LSTM[26,115]。这些模型可以缓解弱交互问题，但仍不足以建模词语和短语级别的上下文交互。

**(3)　强交互模型**

这些模型直接在两个句子之间建立交互空间，并在不同位置建模交互，如 ARC-II[110] 和 MV-LSTM[25]。这种强交互使模型能够捕获两个句子的语义容量之间的差异。

在本章中，我们提出了一种新的深度神经网络架构来建模两个句子的强相互作用。与使用两个分离的 LSTM 建模两个句子不同，我们利用两个相互依赖的 LSTM（称为耦合 LSTM）实现在任何交互的时刻两个句子的信息都可以相互作

用。耦合 LSTM 在每一步的输出取决于两个句子。具体而言，我们提出了两种相互依赖的耦合 LSTM 方式：松耦合模型（LC-LSTM）和紧耦合模型（TC-LSTM）。类似于单句的双向 LSTM[116,117]，在耦合 LSTM 中可以使用四个方向。为了利用耦合 LSTM 的四个方向的所有信息，我们将它们聚合并采用动态汇集策略来自动选择最有信息量的信号。最后，我们将它们送入全连接层以计算匹配得分。

本章的贡献可以总结如下：

1. 与使用相似度矩阵的体系结构不同，我们提出的体系结构直接建模两个句子的强相互作用，可以捕获两个句子的局部语义相关性。我们的架构还可以通过多个堆叠耦合 LSTM 层捕获多粒度交互。

2. 与之前的文本匹配工作相比，我们对两个大规模数据集进行了广泛的实验验证。大规模的数据集使我们能够训练一个非常深的神经网络。实验结果表明，我们提出的架构比之前最先进方法更有效。

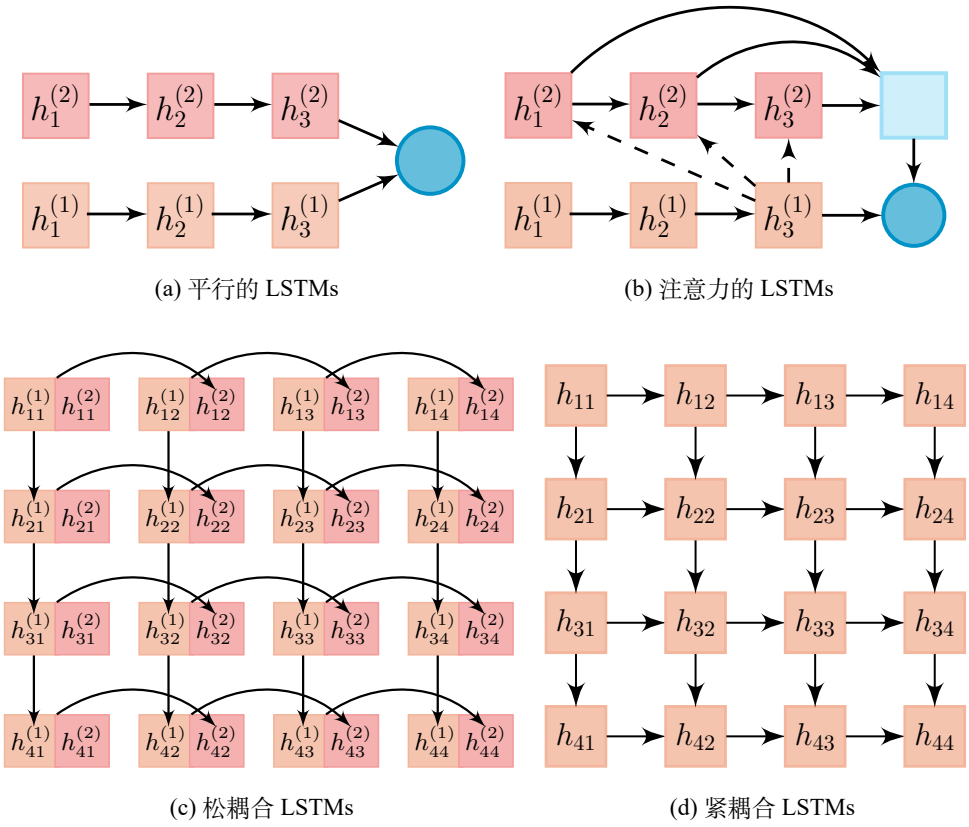
## 5.2　耦合长短时记忆网络

为了处理两个句子，一种简单的方法是用两个单独的 LSTM 进行建模。然而，这种方法难以建模两个句子的局部交互。一种改进的方法是引入注意力机制，它已被用于许多任务，例如机器翻译[36] 和自动问答[115]。

受计算机视觉社区中多维递归神经网络[118] 和网格 LSTM[119] 的启发，我们提出了两种模型来捕获两个并行 LSTM 之间的相互依赖关系，称为**耦合长短时记忆网络**（C-LSTMs）。

为了方便介绍我们的模型，我们首先给出一些定义。给定两个序列 $X = x_1, x_2, \cdots, x_n$ 和 $Y = y_1, y_2, \cdots, y_m$，我们让 $\mathbf{x}_i \in \mathbf{R}^d$ 表示词语 $x_i$ 的向量表示。标准的 LSTM 具有一个时间维度。在处理句子时，LSTM 将该位置视为时间步。在句子 $x_{1:n}$ 的 $i$ 位置，输出 $\mathbf{h}_i$ 反映了子序列 $x_{0:i} = x_0, \cdots, x_i$ 的含义。

为了尽可能早地建模两个句子的交互，我们定义 $\mathbf{h}_{i,j}$ 来表示子序列 $x_{0:i}$ 和 $y_{0:j}$ 的交互。图5.1-(c) 和5.1-(d) 说明了我们提出的两个模型。为了和弱相互并



(a) 平行的 LSTMs　(b) 注意力的 LSTMs

(c) 松耦合 LSTMs　(d) 紧耦合 LSTMs

**图 5.1　四种典型的基于 LSTM 架构句对建模结构**

行 LSTM 进行比较，我们还给出了图5.1-(a) 和5.1-(b) 中的并行 LSTM 和注意力 LSTM。

### 5.2.1. 松耦合长短时记忆网络

为了建模两个句子的局部上下文相互作用，我们使两个 LSTM 在不同位置相互依赖。受网格 LSTM[119] 和逐字注意力 LSTMs[26] 的启发，我们提出了两个相互依赖的 LSTM 松耦合模型。

更具体地说，我们将 $\mathbf{h}_{ij}^{(1)}$ 称为第一个 LSTM 中子序列 $x_{0:i}$ 的编码，它受第二个 LSTM 输出影响。同时，$\mathbf{h}_{i,j}^{(2)}$ 是第二个 LSTM 中子序列 $y_{0:j}$ 的编码，它受第一个 LSTM 输出的影响。$\mathbf{h}_{i,j}^{(1)}$ 和 $\mathbf{h}_{i,j}^{(2)}$ 可以通过以下方式计算：

$$\mathbf{h}_{i,j}^{(1)} = \mathbf{LSTM}^1(\mathbf{H}_{i-1}^{(1)}, \mathbf{c}_{i-1,j}^{(1)}, \mathbf{x}_i), \tag{5.1}$$

$$\mathbf{h}_{i,j}^{(2)} = \mathbf{LSTM}^2(\mathbf{H}_{j-1}^{(2)}, \mathbf{c}_{i,j-1}^{(2)}, \mathbf{y}_j), \tag{5.2}$$

其中，$\mathbf{H}_{i-1}^{(1)}$ 和 $\mathbf{H}_{j-1}^{(2)}$ 可以通过以下方式获得：

$$\mathbf{H}_{i-1}^{(1)} = [\mathbf{h}_{i-1,j}^{(1)}, \mathbf{h}_{i-1,j}^{(2)}], \tag{5.3}$$

$$\mathbf{H}_{j-1}^{(2)} = [\mathbf{h}_{i,j-1}^{(1)}, \mathbf{h}_{i,j-1}^{(2)}]. \tag{5.4}$$

### 5.2.2. 紧耦合长短时记忆网络

LC-LSTM 的隐状态是两个相互依赖的 LSTM 的隐状态的组合，其存储单元是分开的。受到多维 LSTM[120] 设计的启发，我们进一步将两个 LSTM 的隐状态和存储单元放在一起。我们假设 $\mathbf{h}_{i,j}$ 直接建模子序列 $x_{0:i}$ 和 $y_{0:j}$ 的交互，这取决于之前的两个交互 $\mathbf{h}_{i-1,j}$ 和 $\mathbf{h}_{i,j-1}$，其中 $i, j$ 是句子 $X$ 和 $Y$ 中的位置。

我们如下定义紧耦合的长短时记忆单元。

$$\begin{bmatrix} \tilde{\mathbf{c}}_{i,j} \\ \mathbf{o}_{i,j} \\ \mathbf{i}_{i,j} \\ \mathbf{f}_{i,j}^{1} \\ \mathbf{f}_{i,j}^{2} \end{bmatrix} = \begin{bmatrix} \tanh \\ \sigma \\ \sigma \\ \sigma \\ \sigma \end{bmatrix} T_{\mathbf{A},\mathbf{b}} \begin{bmatrix} \mathbf{x}_i \\ \mathbf{y}_j \\ \mathbf{h}_{i,j-1} \\ \mathbf{h}_{i-1,j} \end{bmatrix}, \tag{5.5}$$

$$\mathbf{c}_{i,j} = \tilde{\mathbf{c}}_{i,j} \odot \mathbf{i}_{i,j} + [\mathbf{c}_{i,j-1}, \mathbf{c}_{i-1,j}]^{\mathrm{T}} \begin{bmatrix} \mathbf{f}_{i,j}^{1} \\ \mathbf{f}_{i,j}^{2} \end{bmatrix} \tag{5.6}$$

$$\mathbf{h}_{i,j} = \mathbf{o}_t \odot \tanh(\mathbf{c}_{i,j}) \tag{5.7}$$

其中，$T_{\mathbf{A},\mathbf{b}}$ 是仿射变换，它取决于网络 $\mathbf{A}$ 和 $\mathbf{b}$ 的参数。与随时间定义的标准 LSTM 相比，紧耦合 LSTM 的每个存储器单元 $\mathbf{c}_{ij}$ 具有两个依赖状态：$\mathbf{c}_{i,j-1}$ 和 $\mathbf{c}_{i-1,j}$，以及两个对应的遗忘门 $\mathbf{f}_{i,j}^{1}$ 和 $\mathbf{f}_{i,j}^{2}$。

### 5.2.3. 两个模型的对比分析

我们提出的两个耦合 LSTM 都可以表示为：

$$(\mathbf{h}_{i,j}, \mathbf{c}_{i,j}) = \textbf{C-LSTMs}(\mathbf{h}_{i-1,j}, \mathbf{h}_{i,j-1}, \mathbf{c}_{i-1,j}, \mathbf{c}_{i,j-1}, \mathbf{x}_i, \mathbf{y}_j), \tag{5.8}$$

其中 **C-LSTMs** 可以是 **TC-LSTMs** 或者 **LC-LSTMs**。

输入由耦合 LSTM 步骤 $(i, j)$ 中的两种类型的信息组成：时间信号 $\mathbf{h}_{i-1j}$，$\mathbf{h}_{ij-1}$，$\mathbf{c}_{i-1j}$，$\mathbf{c}_{ij-1}$ 和深度信号 $\mathbf{x}_i$，$\mathbf{y}_j$。TC-LSTM 和 LC-LSTM 之间的差异来自时间和深度信号的信息的依赖性。

**(1) 时间维度之间的相互作用**

TC-LSTM 通过合并内部记忆单元来计算位置 $(i, j)$ 处的交互 $\mathbf{c}_{i-1,j}$ 和 $\mathbf{c}_{i,j-1}$ 在行和列维度上和隐状态 $\mathbf{h}_{i-1,j}$ 和 $\mathbf{h}_{i,j-1}$ 。与 TC-LSTM 相比，LC-LSTM 首先并行使用两个标准 LSTM，分别沿行和列维度产生隐状态 $\mathbf{h}_{i,j}^1$ 和 $\mathbf{h}_{i,j}^2$，然后在下一步合并在一起。

**(2) 深度维度之间的相互作用**

在 TC-LSTM 中，较高层的每个隐状态 $\mathbf{h}_{i,j}$ 接收到来自较低层的信息融合 $\mathbf{x}_i$ 和 $\mathbf{y}_j$。但是，在 LC-LSTM 中，信息 $\mathbf{x}_i$ 和 $\mathbf{y}_j$ 分别由较高层的两个相应 LSTM 接受。

#### 1. LC-LSTMs 和词注意力 LSTMs 对比

注意力 LSTM 的主要思想是句子 X 的表示是基于句子 X 和 Y 中的词语之间的对齐程度动态获得的，这是非对称单向编码。LC-LSTM 则考虑到具有对称编码方式的两个序列之间的相互作用，以获得每个步骤的每个隐状态。

## 5.3 句子匹配的端到端架构

在本节中，我们提出了一个用于匹配两个句子的端到端深度架构，如图5.2所示。

### 5.3.1. 输入层

为了用神经模型对句子进行建模，我们首先需要将词语的独热编码（one-hot）转换为分布式表示。两个序列 $X = x_1, x_2, \cdots, x_n$ 和 $Y = y_1, y_2, \cdots, y_m$ 的所有词语将被映射到低维向量表示，被视为网络的输入。

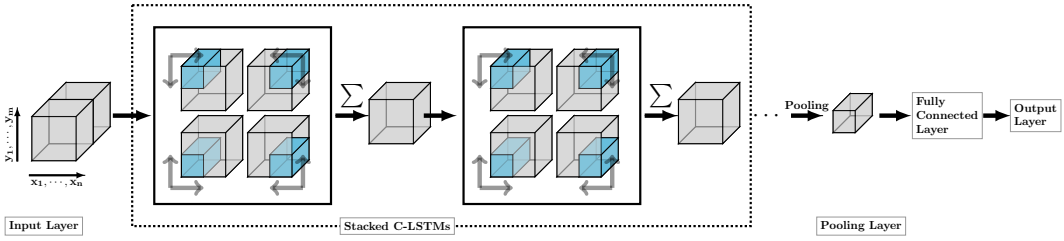



**图 5.2　用于句对编码的耦合 LSTM 的体系结构**

### 5.3.2. 堆叠耦合-LSTMs 层

在嵌入层之后，我们使用耦合 LSTM 来捕获两个句子之间的强相互作用。其基本块由五层组成。我们首先使用四个定向耦合 LSTM 来建模不同信息流的本地交互，然后我们通过聚合层对这些 LSTM 的输出求和。为了提高耦合 LSTM 的学习能力，我们将基本块叠加在一起。

#### 1. 四向耦合 LSTM 层

与双向 LSTM 类似，耦合 LSTM 中有四个方向。

$$(\mathbf{h}^1_{i,j}, \mathbf{c}^1_{i,j}) = \textbf{C-LSTMs}(\mathbf{h}_{i-1,j}, \mathbf{h}_{i,j-1}, \mathbf{c}_{i-1,j}, \mathbf{c}_{i,j-1}, \mathbf{x}_i, \mathbf{y}_j),$$

$$(\mathbf{h}^2_{i,j}, \mathbf{c}^2_{i,j}) = \textbf{C-LSTMs}(\mathbf{h}_{i-1,j}, \mathbf{h}_{i,j+1}, \mathbf{c}_{i-1,j}, \mathbf{c}_{i,j+1}, \mathbf{x}_i, \mathbf{y}_j),$$

$$(\mathbf{h}^3_{i,j}, \mathbf{c}^3_{i,j}) = \textbf{C-LSTMs}(\mathbf{h}_{i+1,j}, \mathbf{h}_{i,j+1}, \mathbf{c}_{i+1,j}, \mathbf{c}_{i,j+1}, \mathbf{x}_i, \mathbf{y}_j),$$

$$(\mathbf{h}^4_{i,j}, \mathbf{c}^4_{i,j}) = \textbf{C-LSTMs}(\mathbf{h}_{i+1,j}, \mathbf{h}_{i,j-1}, \mathbf{c}_{i+1,j}, \mathbf{c}_{i,j-1}, \mathbf{x}_i, \mathbf{y}_j).$$

#### 2. 聚合层

聚合层将四个方向耦合 LSTM 的输出转化成一个向量。

$$\hat{\mathbf{h}}_{i,j} = \sum_{d=1}^{4} \mathbf{h}^d_{i,j}, \tag{5.9}$$

其中 $\mathbf{h}_{i,j}$ 的上标 $d$ 表示不同的方向。

#### 3. 堆叠 C-LSTM 块

为了增加学习网络多粒度交互的能力，我们堆叠了四个 C-LSTM 层和一个聚合层以形成深层架构。

### 5.3.3. 池化层

堆叠耦合 LSTMs 层的输出是张量 $\mathbf{H} \in \mathbf{R}^{n \times m \times d}$，其中 $n$ 和 $m$ 是句子的长度，$d$ 是隐藏神经元的数量。我们应用动态池化来自动提取每个切片 $\mathbf{H}_i \in \mathbf{R}^{n \times m}$ 中的 $\mathbf{R}^{p \times q}$ 降采样矩阵，其过程类似于[24]。

具体地说，对于每个切片矩阵 $\mathbf{H}_i$，我们将 $\mathbf{H}_i$ 的行和列分区为 $p \times q$ 相等的网格。这些网格不重叠。然后我们选择每个网格中的最大值。由于每个切片 $\mathbf{H}_i$ 由神经元在不同位置的隐状态组成，因此可以将合并操作视为由神经元捕获的最具信息量的交互。

### 5.3.4. 输出层

输出层取决于任务的类型，我们根据不同任务选择相应的输出层形式。NLP 中有两种常见的文本匹配任务：一个是排序任务（Ranking），例如社区问答；另一个是分类任务，例如文本蕴含。

1. 对于排序任务，其输出是衡量匹配度的分数，通过在最后一个全连接层之后的线性变换获得。
2. 对于分类任务，其输出是不同类别上的概率，其由最后一个全连接层之后的 Softmax 函数计算。

## 5.4 训练

我们提出的架构可以处理不同的句子匹配任务。损失函数因不同的任务而异。

**(1) 最大边界损失函数**

给定正样本句对 $(X, Y)$ 及其对应的负样本对 $(X, \hat{Y})$，匹配分数 $s(X, Y)$ 应大于 $s(X, \hat{Y})$。

对于此任务，我们使用对比最大边界损失函数[69,70] 来训练我们的匹配任务模型。

**表 5.1　耦合 LSTM 模型的超参数设置**

| | 问答匹配（MQA） | 文本蕴含识别 (RTE) |
|---|---|---|
| 嵌入维度 | 100 | 100 |
| 隐状态维度 | 50 | 50 |
| 学习率 | 0.05 | 0.005 |
| 正则化项 | $5E{-}5$ | $1E{-}5$ |
| 池化大小 $(p,q)$ | (2,1) | (1,1) |

形式化地，损失函数被定义为

$$L(X,Y,\hat{Y}) = max(0, 1 - s(X,Y) + s(X,\hat{Y})). \tag{5.10}$$

其中 $s(X,Y)$ 是预测出的分数。

**(2)　交叉熵损失函数**

给出句对 $(X,Y)$ 及其标签 $l$，神经网络的输出 $\hat{l}$ 是不同类的概率。训练网络的参数以最小化预测和真实标签分布的交叉熵。

$$L(X,Y;\boldsymbol{l},\hat{\boldsymbol{l}}) = -\sum_{j=1}^{C} \boldsymbol{l}_j \log(\hat{\boldsymbol{l}}_j), \tag{5.11}$$

其中 $\boldsymbol{l}$ 是正确的标签，它用一个 one-hot 向量表示; $\hat{\boldsymbol{l}}$ 是标签的预测概率; $C$ 是类别编号。

## 5.5　实验与分析

在本节中，我们研究了所提出的模型在两个不同的文本匹配任务上的性能表现：分类任务（识别文本蕴含）和问答匹配任务。

### 5.5.1.　超参数和训练

我们使用 100 维的 GloVe 向量做为预训练的初始化词向量[87]，其他参数从 $[-0.1, 0.1]$ 随机初始化。最终的超参数设置为表5.1。

### 5.5.2. 对比模型

- 神经词袋模型 (NBOW): 将句子中每个词对应的词向量加权求和，然后传递给 MLP 层。
- 单个 LSTM: 用一个 LSTM 去编码两个句子[26]。
- 平行的 LSTM：两个序列分别由两个 LSTM 编码，然后将它们连接起来并送到 MLP。
- 注意力 LSTM: 弱交互的基于注意力的 LSTM[26]。
- 逐词注意力 LSTMs: 任何两个词之间通过 LSTM 进行交互[26]。

### 5.5.3. 实验一：文本蕴含识别

文本蕴含识别是确定两个句子之间的语义关系的任务。我们使用斯坦福自然语言推理语料库（SNLI)[96] 作为评测数据集。该语料库包含 57 万个句对，并且所有句子和标签都源自人工标注。SNLI 比所有其他现有的 RTE 语料库大两个数量级。因此，大规模的 SNLI 允许我们训练强大的神经网络。

#### 1. 实验结果

表 5.2显示 SNLI 的评估结果。该表的第三列给出了不带词嵌入的模型参数数量。

我们提出的两个 C-LSTM 模型具有四个堆叠块，优于所有对比模型，这表明我们更瘦更深的网络确实有效。

此外，我们可以看到 LC-LSTM 和 TC-LSTM 都受益于多方向层，而后者比前者效果更好。我们将这两种模型之间的差异归因于它们从深度维度控制信息流的不同机制。

与注意力 LSTM 相比，我们的两个模型使用更少的参数（接近 $1/5$）获得了与它们相当的结果。通过堆叠 C-LSTM，它们的性能得到显著改善，并且四个堆叠的 TC-LSTM 在该数据集上达到了 $85.1\%$ 的准确度。

此外，我们可以看到 TC-LSTM 在此任务上实现了比 LC-LSTM 更好的效果，说明其对成对的词语和短语进行了细粒度推理。

**表 5.2　不同模型在 SNLI 语料库上的准确率**

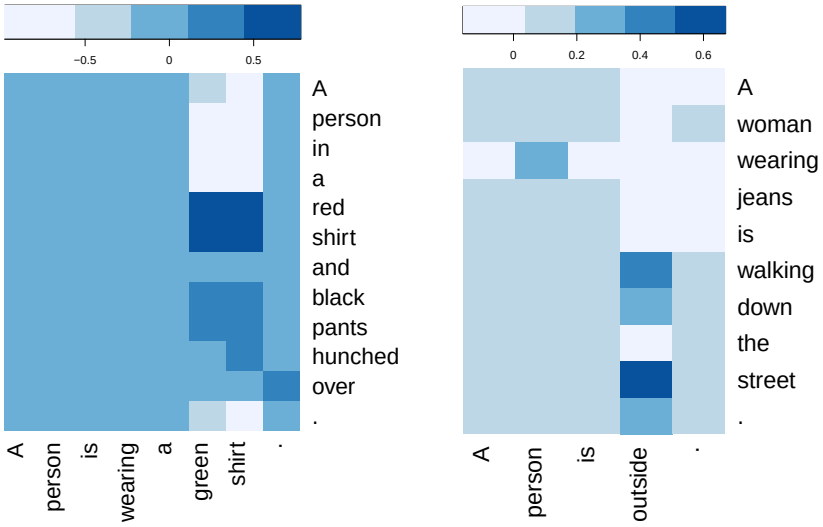
| **模型** | $k$ | $\|\theta\|_M$ | 训练 | 测试 |
|---|---|---|---|---|
| 神经词袋模型 | 100 | 80K | 77.9 | 75.1 |
| 单个 LSTM[26] | 100 | 111K | 83.7 | 80.9 |
| 平行 LSTM[96] | 100 | 221K | 84.8 | 77.6 |
| 注意力 LSTM[26] | 100 | 252K | 83.2 | 82.3 |
| 逐词注意力 LSTM[26] | 100 | 252K | 83.7 | 83.5 |
| LC-LSTM (单方向) | 50 | 45K | 80.8 | 80.5 |
| LC-LSTM | 50 | 45K | 81.5 | 80.9 |
| 四向 LC-LSTM | 50 | 135K | 85.0 | 84.3 |
| TC-LSTM (单方向) | 50 | 77.5K | 81.4 | 80.1 |
| TC-LSTM | 50 | 77.5K | 82.2 | 81.6 |
| 四向 TC-LSTM | 50 | 190K | 86.7 | **85.1** |

### 2. 了解 C-LSTM 中神经元的行为

为了直观地了解 C-LSTM 如何解决这个问题，我们在使用 TC-LSTM 评估测试集时，观测最后一个聚合层中神经元的行为。我们发现某些单元（cell）扮演着特定的角色。

具体地说，假设 $h_{i,j,k}$ 表示在 $(i,j)$ 的位置第 $k$ 个激活神经元，其中 $i \in \{1,\ldots,n\}$，和 $j \in \{1,\ldots,m\}$。通过可视化隐状态 $\mathbf{h}_{i,j,k}$ 并分析最大激活值，我们可以发现存在多个可解释的神经元。例如，当隐藏神经元的激活值 $h_{i,j,k}$ 最大时，意味着句对中 $(i,j)$ 位置的模式对当前句对关系的判定是很重要的。

图 5.3说明了这种现象。在图5.3-(a) 中，神经元显示其监视局部上下文相互作用的能力，颜色越深值越大。




(a) 第三个神经元　　(b) 第十七个神经元

**图 5.3　两个可解释的神经元和一些词语对的可视化**

词对“(red, green)”的激活值应远远高于其他部分。这说明两个句子的基本事实是矛盾的。一个有趣的事情是，在一个句子中，有两个词描述颜色“A person in a red shirt and black pants hunched over.（一个穿红衬衫和黑裤子的人弓着背。）”。我们的模型忽略了无用的词语“black”，这表明这个神经元通过上下文理解选择性地捕获模式，而不仅仅是词语级别的交互。在图5.3-(b) 中，另一个神经元表明它可以捕获局部的上下文交互，例如“walking down the street, outside（走在街上，外面）”。这些模式可以通过汇集层轻松捕获，并为最终预测提供强有力的支持。

表 5.3说明了多个可解释的神经元和一些可以激活这些神经元的代表性词语或短语对。

**表 5.3　多个可解释的神经元以及具体实例**

| 神经元的索引 | 词或者短语对 |
|---|---|
| 第三 | (in a pool, swimming), (near a fountain, next to the ocean), (street, outside) |
| 第九 | (doing a skateboard, skateboarding), (sidewalk with, inside), (standing, seated) |
| 第十七 | (blue jacket, blue jacket), (wearing black, wearing white) |
| 第二十五 | (a man, two other men), (a man, two girls), (an old woman, two people) |

**表 5.4　雅虎问答对数据集上的结果**

| **模型** | 词嵌入维度 | P@1(5) | P@1(10) |
|---|---|---|---|
| 随机分类 | - | 20.0 | 10.0 |
| 神经词袋模型 | 50 | 63.9 | 47.6 |
| 单个 LSTM | 50 | 68.2 | 53.9 |
| 平行 LSTM | 50 | 66.9 | 52.1 |
| 注意力 LSTM | 50 | 73.5 | 62.0 |
| LC-LSTM (单方向) | 50 | 75.4 | 63.0 |
| LC-LSTM | 50 | 76.1 | 64.1 |
| 三向 LC-LSTM | 50 | **78.5** | **66.2** |
| TC-LSTM (单方向) | 50 | 74.3 | 62.4 |
| TC-LSTM | 50 | 74.9 | 62.9 |
| 三向 TC-LSTM | 50 | 77.0 | 65.3 |

### 5.5.4. 实验二：问答匹配

问答匹配（MQA）是语义匹配的典型任务。给定一个问题，我们需要从一些候选答案中选择正确的答案。

在本章中，我们使用从 Yahoo! 收集的数据集。我们能使用 Yahoo! 提供的 getByCategory，产生 $963,072$ 个问题和相应的最佳答案。然后，我们选择问题和答案长度均在 $[4, 30]$ 区间内的问答对，从而获得 $220,000$ 个问答对。对于负例样本的构建，我们首先使用每个问题的最佳答案作为查询，从整个答案集中检索最好的 $1,000$ 个结果，其中排在第 4 或第 9 的答案将被随机选择作为负例样本。

整个数据集分为训练、校验和测试数据，比例为 $20:1:1$。此外，我们提供两个测试设置：分别从 5 个和 10 个候选者中选择最佳答案。

#### 1. 实验结果

MQA 的结果显示在表5.4中，采用的评估指标是 $P@N$（即返回第 $N$ 个结果的精确度）。对于我们的模型，由于堆叠块超过三层无法对此任务进行显著提升，

我们只使用三个堆叠的 C-LSTM。

通过分析表5.4中的问答配对的评估结果，我们可以看到强相互作用模型（注意力 LSTM，和 C-LSTM）始终优于弱相互作用模型（NBOW，和平行 LSTM），这表明建模两个句子强相互作用的重要性。

我们提出的两个 C-LSTM 超越了对比模型的方法，其原因在于 C-LSTM 增加了多方向层和多个堆叠块，充分利用多级抽象来直接提升性能。此外，LC-LSTM 优于 TC-LSTM。原因可能是 MQA 是一个相对简单的任务，与 RTE 任务相比，它需要较少的推理能力。此外，LC-LSTM 的参数小于 TC-LSTM，这确保了前者可以避免在相对较小的语料库上遭受过度拟合。

## 5.6　本章小结

在本文中，我们提出了一种端到端的深层体系结构来捕获句对的强交互信息。在两个大型文本匹配任务的实验证明了我们提出的模型的有效性及其对基线模型的优越性。此外，通过可视化分析，我们发现提出的模型中多个可解释神经元可以捕获词语或短语的上下文交互。

在未来的工作中，我们希望将门控策略纳入所提出的模型的深度维度，如 Highway 网络[121] 或残差网络[122]，以增强深度和其他维度之间的相互作用，从而使得训练更深的神经网络变成可能。

# 第二部分

# 特殊性质的表示学习

# 第 6 章　可迁移性表示学习

在第一部分，我们介绍了如何对不同粒度文本进行表示学习，然而，当我们对文本有了基本的学习方法时，接下来要考虑的是如何使得所学习的表示具备某种特定性质，其意义在于可以让我们更加灵活去使用这些表示。以下三个章节，我们分别对表示学习的可迁移性、可分离性和可理解性进行研究探讨。

## 6.1　引言

词语的分布式表示已广泛用于许多自然语言处理（NLP）任务中。在取得这一成功之后，学习连续词语的分布式表示（例如短语、句子、段落和文档）引起了极大的兴趣[9,57,123,124]。这些模型的主要作用是将可变长度的句子或文档表示为固定长度的向量。

基于深度神经网络（DNN）的方法由于参数数量众多，通常需要大规模的语料库，很难用有限数据训练出泛化性好的网络。但是，为某些 NLP 任务构建大规模资源的成本非常昂贵。为了解决这个问题，这些模型通常涉及无监督的预训练阶段。最终模型用梯度优化方法进行微调。最近的研究表明，在无监督预训练词表示的帮助下，几个 NLP 任务的准确性得到了显著提高[63]。大多数预训练方法都基于无监督目标[3,62,63]。这种无监督的预训练对于提高最终性能是有效的，但它不能直接优化所需的任务。

多任务学习通过利用任务之间的相关性来并行学习多个任务。在多任务学习成功的推动下，一些基于神经网络的 NLP 模型利用多任务学习来共同学习多项任务，目的是互惠互利。这些模型的基本多任务架构是共享一些较低层来确定共同特征。共享层之后对接的是和任务相关的网络层。在本文中，我们提出了三种共享信息的模型。第一个模型仅为所有任务使用一个共享层。第二个模型为不同的任务分配一个特定层，但每个层可以从其他层读取信息。第三个模型不仅为每个任务分配一个特定层，还为所有任务构建共享层。此外，我们引入了一个门

控机制，使模型能够有选择地利用共享信息。整个网络协同训练所有这些任务。

四个文本分类任务的实验结果表明，与单任务训练相比，多个相关任务的联合学习可以获取明显的收益。

我们的贡献如下：

- 首先，我们为 RNN 提出了三种多任务架构。尽管多任务学习的想法并不新颖，但我们的工作是将 RNN 集成到多任务学习框架中，该框架学习将任意文本映射到具有任务特定层和共享层的语义向量表示。
- 其次，在几个文本分类任务中，我们的多任务模型优于大多数当时最先进的对比模型。

## 6.2　面向文本分类的循环神经网络

神经模型的主要作用是将可变长度文本表示为固定长度的向量。这些模型通常包括将词语、子词单元或 n-gram 映射到向量表示的投影层（通常预先使用无监督方法进行训练），然后将它们与神经网络的不同体系结构相结合。

常见神经网络文本模型，包括词袋模型（NBOW）、循环神经网络（RNN）[73]、递归神经网络（TreeRNN）[57,65] 和卷积神经网络（CNN）[9,63]。这些模型将文本序列中的词语嵌入作为输入，并用固定长度的向量表示来概括其含义。

其中，循环神经网络（RNN）是 NLP 问题中最常用的架构之一，因为它们的重复结构非常适合处理可变长度文本。

在单个特定任务中，一个简单的策略是使用 RNN 将输入序列映射到固定大小的向量，然后将向量传递给 Softmax 层以进行分类或其他任务。

给定文本序列 $x = \{x_1, x_2, \cdots, x_T\}$，我们首先使用查找层来获取每个词语 $x_i$ 的向量表示（嵌入）$\mathbf{x}_i$。最后一时刻的输出 $\mathbf{h}_T$ 可以被视为整个序列的表示，其后接全连接层以及 Softmax 非线性层，用于预测每个类别上的概率分布。

我们训练网络的参数以最小化预测分布和真实分布的交叉熵。

$$L(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_{i=1}^{N}\sum_{j=1}^{C} \mathbf{y}_i^j \log(\hat{\mathbf{y}}_i^j), \tag{6.1}$$

其中，$\mathbf{y}_i^j$ 是真实标签，$\hat{\mathbf{y}}_i^j$ 是预测概率 $N$ 表示训练样本的数量，$C$ 表示类别的总数量。

## 6.3　基于 RNN 的多任务学习共享模型

现有的神经网络方法大多数基于单任务的监督训练[9,57,63]。这些方法经常受到有限训练数据的影响。为了解决这个问题，这些模型通常涉及无监督的预训练阶段。这种无监督的预训练对于提高最终性能是有效的，但它并没有直接优化目标任务。

在多任务学习取得成功的推动下[125]，我们提出了三个多任务模型来同时利用其他相关任务的监督数据。深度神经模型非常适合多任务学习，因为从某个任务中学习的特征可能对其他任务有用。图6.1给出了我们提出的模型的图示。

**(1)　模型-I: 统一层架构**

除了私有的嵌入层之外，不同任务共享相同的 LSTM 层和嵌入层。

对于任务 $m$，它的输入 $\hat{\mathbf{x}}_t^{(m)}$ 包含两部分:

$$\hat{\mathbf{x}}_t^{(m)} = \mathbf{x}_t^{(m)} \oplus \mathbf{x}_t^{(s)}, \tag{6.2}$$

其中，$\mathbf{x}_t^m$，$\mathbf{x}_t^s$ 分别表示任务特定和共享的词嵌入，$\oplus$ 表示连接操作。

LSTM 层在所有任务共享的。任务 $m$ 的最终序列表示是 LSTM 在最后一个时间步 $T$ 的输出。

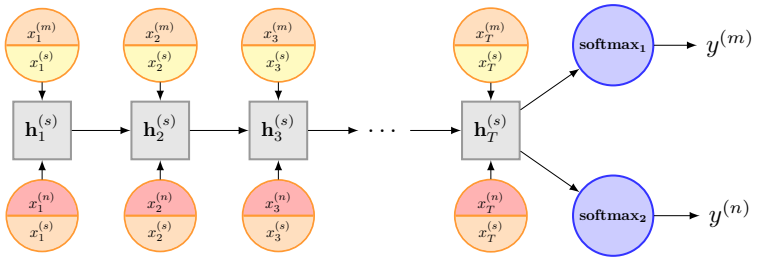
$$\mathbf{h}_T^{(m)} = LSTM(\hat{\mathbf{x}}^{(m)}). \tag{6.3}$$

**(2)　模型-II: 耦合层架构**

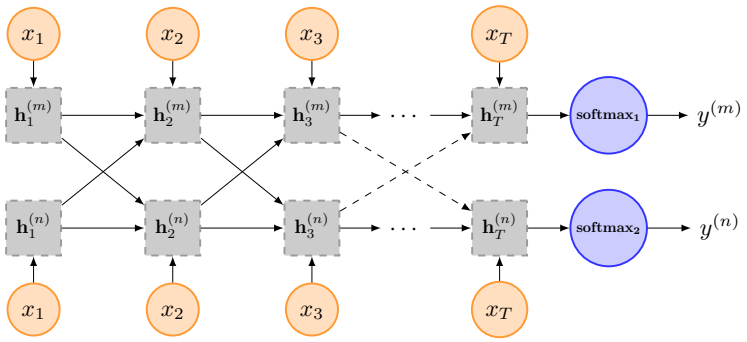
我们为每个任务分配一个 LSTM 层，它可以使用另一个任务的 LSTM 层的信息。

给定一对任务 $(m, n)$，每个任务具有自己特定的 LSTM。我们用 $\mathbf{h}_t^{(m)}$ 和 $\mathbf{h}_t^{(n)}$ 表示两个耦合 LSTM 层在时间步 $t$ 的输出。

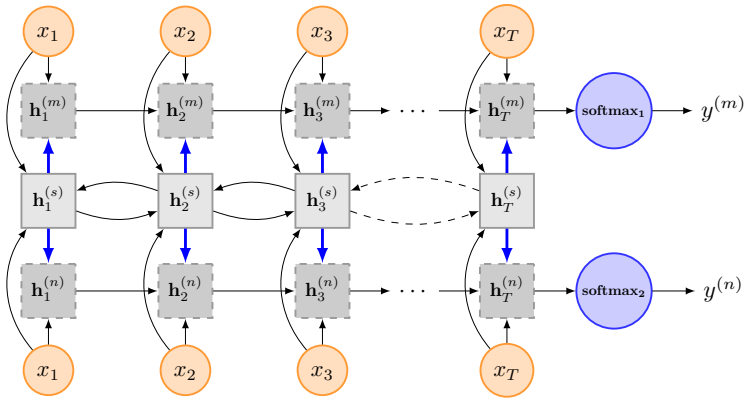
为了更好地控制信息从一个任务流向另一个任务，我们使用全局门控单元，该模块赋予模型决定它应该接受多少信息的能力。我们重新定义等式 (2.7) 里的




(a) Model-I: 统一层架构




(b) Model-II: 耦合层架构




(c) Model-III: 共享层架构

**图 6.1　用多任务学习建模文本的三种架构**

记忆单元 $\hat{c}_t$，并且第 $m$ 个任务 LSTM 的新记忆单元由下等式计算：

$$\tilde{\mathbf{c}}_t^{(m)} = \tanh\left(\mathbf{W}_c^{(m)}\mathbf{x}_t + \sum_{i\in\{m,n\}} \mathbf{g}^{(i\to m)} U_c^{(i\to m)} \mathbf{h}_{t-1}^{(i)}\right) \tag{6.4}$$

其中，$\mathbf{g}^{(i\to m)} = \sigma(\mathbf{W}_g^{(m)}\mathbf{x}_t + g^{(i)}\mathbf{h}_{t-1}^{(i)})$，其他的设置与标准的 LSTM 一致。

该模型可用于联合学习任意两项任务。对于任务 $m$ 和 $n$，我们可以获得两个特定任务的表示 $\mathbf{h}_T^m$ 和 $\mathbf{h}_T^n$。

**(3) 模型-III：共享层架构**

模型-III 为每个任务分配一个单独的 LSTM 层，但引入了双向 LSTM 层来捕获所有任务的共享信息。

我们标记 LSTM 在时刻 $t$ 前向和后向的输出分别为 $\overrightarrow{\mathbf{h}}_t^{(s)}$，$\overleftarrow{\mathbf{h}}_t^{(s)}$。共享层的输出为 $\mathbf{h}_t^{(s)} = \overrightarrow{\mathbf{h}}_t^{(s)} \oplus \overleftarrow{\mathbf{h}}_t^{(s)}$。

为了增强任务特定层和共享层之间的交互，我们使用门控机制赋予任务特定层中的神经元能够接受或拒绝共享层中神经元传递的信息。

与模型-II 不同，我们计算 LSTM 的新状态如下：

$$\tilde{\mathbf{c}}_t^{(m)} = \tanh\left(\mathbf{W}_c^{(m)}\mathbf{x}_t + \mathbf{g}^{(m)}c^{(m)}\mathbf{h}_{t-1}^{(m)} + \mathbf{g}^{(s\to m)}c^{(s)}\mathbf{h}_t^{(s)}\right), \tag{6.5}$$

其中，$\mathbf{g}^{(m)} = \sigma(\mathbf{W}_g^{(m)}\mathbf{x}_t + g^{(m)}\mathbf{h}_{t-1}^{(m)})$，且 $\mathbf{g}^{(s\to m)} = \sigma(\mathbf{W}_g^{(m)}\mathbf{x}_t + g^{(s\to m)}\mathbf{h}_t^{(s)})$。

## 6.4 训练

由上述所有多任务架构输出的特定于任务的表示最终被传递给不同的输出层，输出层也是特定于任务的。

$$\hat{\mathbf{y}}^{(m)} = \mathrm{Softmax}(\mathbf{W}^{(m)}\mathbf{h}^{(m)} + \mathbf{b}^{(m)}), \tag{6.6}$$

其中，$\hat{\mathbf{y}}^{(m)}$ 是任务 $m$ 的预测概率，$\mathbf{W}^{(m)}$ 是需要学习的权重，$\mathbf{b}^{(m)}$ 是一个偏置。

我们的全局代价函数是联合所有损失函数的线性组合。

$$\phi = \sum_{m=1}^{M} \lambda_m L(\hat{y}^{(m)}, y^{(m)}) \tag{6.7}$$

其中，$\lambda_m$ 分别是每个任务 $m$ 的权重。

值得注意的是，用于训练每项任务的标记数据可以来自完全不同的数据集。参照[27] 的工作，训练过程是以随机的方式在任务之间循环迭代，具体训练过程步骤如下：

1. 随机选择一个任务；

2. 从这个任务中选择一个随机训练样本；

3. 通过采用关于这个样本的梯度来更新参数；

4. 返回步骤 1。

**(1) 微调**

对于模型-I 和模型-III，所有任务都有一个共享层。因此，在联合学习阶段之后，我们可以使用微调策略来进一步优化每项任务的性能。

**(2) 使用神经语言模型预先训练共享层**

对于模型-III，可以通过无监督的预训练阶段来初始化共享层。这里，对于模型-III 中的共享 LSTM 层，我们通过语言模型[126] 对其进行初始化，该模型在所有四个任务的数据集上进行训练。

## 6.5 实验与分析

我们在文本分类数据集上评估了本章提出的三个模型的效果，并与其他模型进行比较。实验所使用的文本分类数据集是 SST-1、SST-2、SUBJ、和 IMDB。有关四个数据集的详细统计信息如表2.1所示。

### 6.5.1. 超参数和训练

使用反向传播训练网络，并使用 Adagrad 更新规则进行基于梯度的优化[86]。在所有实验中，我们使用的预训练词语嵌入是基于维基百科语料库（10 亿词语）训练的 word2vec[3]。词汇量大约为 500,000。在训练期间对词嵌入进行了微调，以提高性能[63]。其他参数采用值的范围在 [-0.1,0.1] 中的均匀分布，进行随机初始化。最终，我们选择在校验集上实现最佳性能的超参数进行评估模型性能。对于没有校验集的数据集，我们使用 10 折交叉验证（CV）。

最终的超参数定义如下：特定任务和共享层的嵌入大小为 64，对于模型-I，每个词语有两个词嵌入，它们的大小均为 64。LSTM 的隐藏层大小为 50，初始学习率为 0.1。参数的正则化权重为 $10^{-5}$。

### 6.5.2. 多任务训练的效果

在实验中，单任务分类使用的是由 Graves 等人提出的 LSTM[49]。表 6.1-6.3显示四个数据集的分类准确度。每个表的第二行（“单任务”）表示每个单独任务的标准 LSTM 的结果。

**表 6.1　统一层架构的实验结果**

| 模型 | SST-1 | SST-2 | SUBJ | IMDB | Avg$\Delta$ |
|---|---|---|---|---|---|
| 单任务 | 45.9 | 85.8 | 91.6 | 88.5 | - |
| 联合训练 | 46.5 | 86.7 | 92.0 | 89.9 | +0.8 |
| + 精调 | **48.5** | **87.1** | **93.4** | **90.8** | +2.0 |

**表 6.2　耦合层架构的实验结果**

| 模型 | SST-1 | SST-2 | SUBJ | IMDB | $Avg\Delta$ |
|---|---|---|---|---|---|
| 单任务 | 45.9 | 85.8 | 91.6 | 88.5 | - |
| SST1-SST2 | **48.9** | **87.4** | - | - | +2.3 |
| SST1-SUBJ | 46.3 | - | 92.2 | - | +0.5 |
| SST1-IMDB | 46.9 | - | - | 89.5 | +1.0 |
| SST2-SUBJ | - | 86.5 | 92.5 | - | +0.8 |
| SST2-IMDB | - | 86.8 | - | **89.8** | +1.2 |
| SUBJ-IMDB | - | - | **92.7** | 89.3 | +0.9 |

#### (1) 统一层架构

对于统一层架构，我们同时在四个数据集上训练模型。实验结果如表6.1所示，联合训练表示 LSTM 层在所有任务中共享，四个数据集的平均性能提高了 0.8%。随着进一步的微调，平均分类准确率提高了 2.0%。

#### (2) 耦合层架构

对于耦合层架构，信息在任意两个任务共享，因此，四个数据集有六种组合。我们在不同的数据集对上训练六个模型。实验结果如表6.2所示，我们可以发现，

**表 6.3　共享层架构的实验结果**

| 模型 | SST-1 | SST-2 | SUBJ | IMDB | Avg△ |
|---|---|---|---|---|---|
| 单任务 | 45.9 | 85.8 | 91.6 | 88.5 | - |
| 联合训练 | 47.1 | 87.0 | 92.5 | 90.7 | +1.4 |
| + 语言模型 | 47.9 | 86.8 | 93.6 | 91.0 | +1.9 |
| + 精调 | **49.6** | **87.9** | **94.1** | **91.3** | +2.8 |

成对联合学习也提高了效果。任务越相关，改进越明显。由于 SST-1 和 SST-2 来自同一语料库，因此它们的改进比其他组合更明显。SST-1 和 SST-2 同时学习，平均改善率为 2.3%。

(3)　**共享层架构**

共享层架构比统一层架构更通用。除了所有任务的共享层，每个任务都有自己的任务特定层。如表6.3所示，我们可以看到四个数据集的性能平均改善为 1.4%，比统一层架构更好。我们还研究了对共享 LSTM 层进行无监督预训练的策略。通过语言模型预训练，平均性能提高了 0.5%。此外，进一步的微调可以显著提高性能 0.9%。

回顾一下，我们提出的所有模型都优于单任务学习基线。共享层架构提供最佳性能。此外，与标准的 LSTM 相比，我们提出的三种模型在收敛速度更快时不会产生太多的额外计算代价。在我们的实验中，最复杂的共享层架构模型的计算代价是标准的 LSTM 的 2.5 倍。

### 6.5.3.　与其他神经模型进行比较

我们与下面的模型进行了比较：

- **NBOW** 该模型对词语向量求和，然后应用非线性进行变换，最后是 Softmax 分类层。

- **MV-RNN** 具有解析树的矩阵向量递归神经网络[65]。

- **RNTN** 具有基于张量的特征函数和解析树的递归神经张量网络[57]。
- **DCNN** 动态 k-max 池化的卷积神经网络[9]。
- **PV** 在段落向量之上的逻辑回归[123]。
- **Tree-LSTM** LSTM 到树状结构网络拓扑的推广。[10]

表6.4显示了共享层架构与对比模型相比的分类准确率，这表明我们的模型对于当时最先进模型具有竞争力。

虽然 Tree-LSTM 在 SST-1 上优于我们的模型，但它需要一个外部解析器来获得句子拓扑结构。值得注意的是，我们的模型与其他基于 RNN 的模型兼容。例如，我们可以轻松地将 LSTM 扩展到 Tree-LSTM 模型。

**表 6.4　已有方法与本文的基于共享模式方法的实验结果对比**

| 模型 | SST-1 | SST-2 | SUBJ | IMDB |
|---|---|---|---|---|
| NBOW | 42.4 | 80.5 | 91.3 | 83.62 |
| MV-RNN | 44.4 | 82.9 | - | - |
| RNTN | 45.7 | 85.4 | - | - |
| DCNN | 48.5 | 86.8 | - | - |
| PV | 44.6 | 82.7 | 90.5 | **91.7** |
| Tree-LSTM | **50.6** | 86.9 | - | - |
| Multi-Task | 49.6 | **87.9** | **94.1** | 91.3 |

### 6.5.4. 案例分析

为了直观地了解我们使用单个 LSTM 或共享层 LSTM 预测文本分类时发生的情况，我们设计了一个实验来分析每个时间步的单个 LSTM 和共享层 LSTM 的输出。我们从 SST-2 测试数据集中抽取两个句子，并且将在不同的时间步骤中预测的情感分数的变化显示在图6.2中。Y 轴表示情感分数，而 X 轴表示按时间顺序的输入。红色水平线给出正面和负面情感之间的边界。6.2-(c,d) 可视化了全

局门控的激活值 ($g^{(s)}$)，以便更多了解共享结构如何影响特定任务。这里的全局门控激活值 $g^s$ 控制从一个共享 LSTM 层流向任务特定层的信号。为了解神经元的行为，我们按时间绘制全局门的演化激活值 $g^s$ 并根据它们在最后一步的激活值对神经元进行排序。

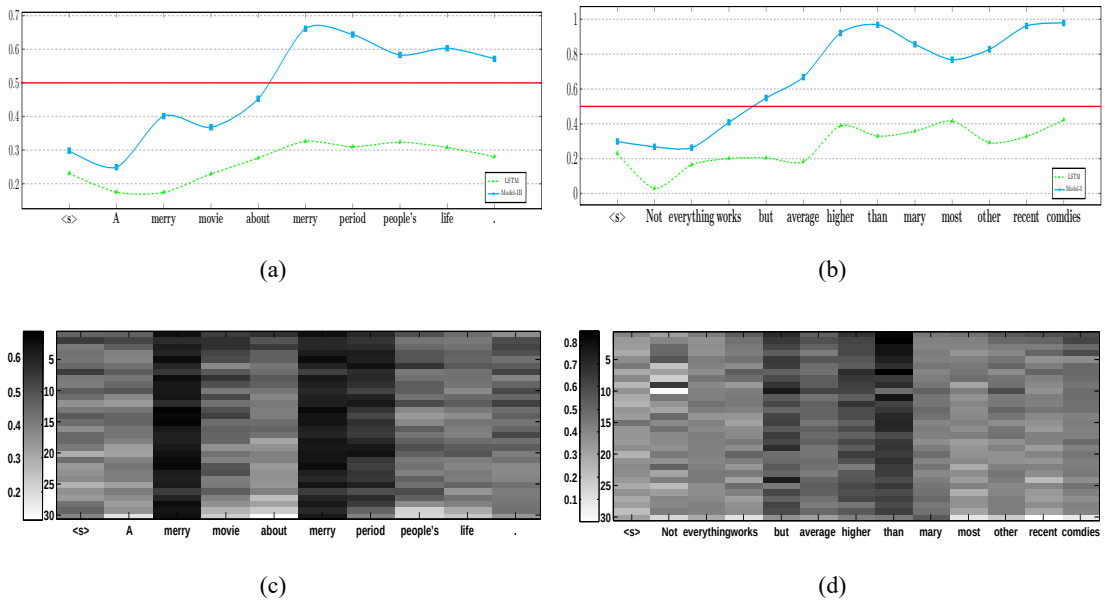



**图 6.2　不同时间步情感分数的变化**

对于具有正面情感的句子 “A merry movie about merry period people's life.”，标准 LSTM 给出了错误的预测，因为标准的 LSTM 是单任务的，无法从其他任务中获得外部信息。如图6.2- (c) 所示，当我们将输入视为 “merry” 时，我们可以清楚地看到神经元被广泛激活，这表明任务特定层需要从共享层得到 “merry” 这个词的大量信息，这最终使模型给出了正确的预测。

另一个案例是正面的，并且有复杂的语义构成。如图6.2-(b,d) 所示，简单的 LSTM 无法捕获结构 “but ... higher than ”，而我们的模型对其敏感，这表明共享层不仅可以丰富某些词语的含义，还可以学到一些具体任务的结构信息。

### 6.5.5. 错误分析

我们分析了所提出的共享层模型在 SST-2 数据集产生的错误情况。大多数错误案例可以概括为两类。

**(1) 复杂的句子结构**

一些涉及复杂结构的句子无法正确处理，例如双重否定和虚拟句子 “`it never fails to engage us.`”。要解决这些问题，需要进行一些架构改进，例如基于树的 LSTM[10]。

**(2) 需要推理的句子**

如果只考虑字面意思，一些句子的情感可能会被误导。例如，句子“`I tried to read the time on my watch.`”表达对电影的消极态度，可以通过基于常识的推理来正确理解。

## 6.6 本章小结

在本文中，我们提出了三种基于 RNN 的架构，用于建模多任务学习的文本序列。它们之间的差异是在几个任务之间共享信息的机制。实验结果表明，我们的模型可以通过探索共同特征来提高相关任务的性能。

# 第 7 章　可分离性表示学习

## 7.1　引言

大多数现有的关于多任务学习的工作[29,30] 试图仅仅基于是否应该共享某些组件的参数，将不同任务的特征划分为私有和共享空间。如图7.1-(a) 所示，两个黑圈之间的重叠表示共享空间。蓝色三角形和方框表示和任务相关的特征，而红色圆圈表示可以共享的特征。共享-私有模型是指在多任务学习中构建两个特征空间：一个用于存储任务特有的特征，另一个用于捕获任务之间共享特征。此框架的主要缺点是共享特征空间可能包含一些不必要的任务特有的特征，而一些共享特征也可能混入任务特有特征空间中。

例如，以下两个句子分别来自电影评论和婴儿产品评论。

*The **infantile** cart is simple and easy to use.*

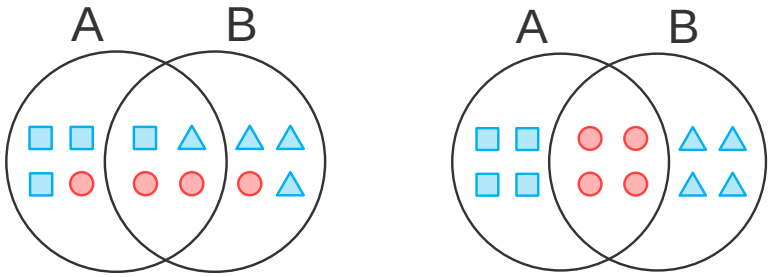
*This kind of humour is **infantile** and boring.*

词语“infantile”在电影任务中表示负面情感，而在“Baby”任务中它是中性的。但是，一般的共享-私有模型可能将任务特有的词语“infantile”放在共享空间中，从而为其他任务留下潜在的危害。此外，共享空间的容量也可能被一些不必要的特征所浪费。

为了解决这个问题，我们提出了一个基于对抗训练的多任务学习框架，其中共享和私有特征空间通过引入正交性约束而满足正交性。具体来说，我们设计了一个通用的共享-私有学习框架来处理文本序列。为了防止共享和私有的特征空间相互干扰，我们引入了两种策略：对抗训训练（Adversarial training）和正交性约束（Orthogonal constraint）。对抗训练用于确保共享特征空间仅包含在任务之间可共享的特征，而正交性约束用于消除私有和共享空间的冗余特征。

本章的贡献可归纳如下。

1. 提出的模型以更精确的方式划分任务特有和共享空间，而不是粗略地共享参数。




(a) 共享私有模型　　(b) 基于对抗的共享私有模型

**图 7.1　任务 A 和任务 B 的两种共享方式**

2. 我们将原始的二元对抗训练扩展到多类，这不仅可以使多个任务得到联合训练，而且允许使用未标记的数据。

3. 我们可以将多个任务之间的共享知识压缩成现成的神经层，这使得我们可以轻松地将其转移到新任务中。

## 7.2　基于循环神经网络的文本分类

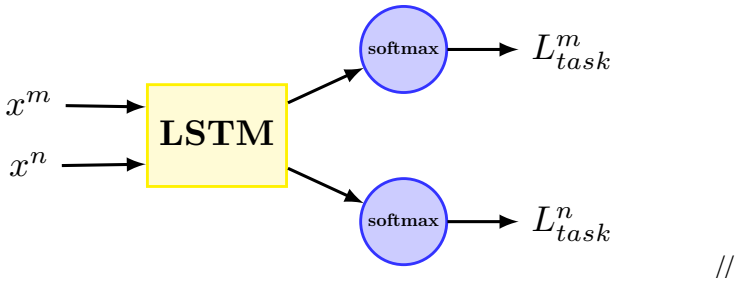
神经句子建模模型，涉及循环神经网络[35,73,124]，卷积神经网络[9,63] 和递归神经网络[57]。这里我们采用具有长短期记忆（LSTM）的递归神经网络，因为它们在各种自然语言处理任务中取得了优越的性能[55,127]。

长短期记忆网络（LSTM）[46] 是一种循环神经网络（RNN）[47]，专门解决学习长期依赖性的问题。虽然有许多 LSTM 变体，但在这里我们使用 Jozefowicz 等人的工作[48] 所使用的 LSTM 架构，它类似于[49] 的架构，但没有窥视孔（peephole）连接。

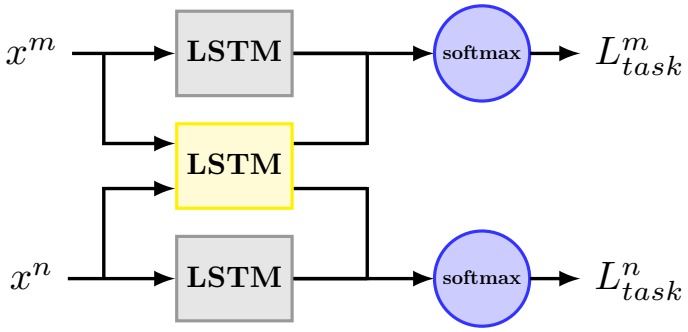
给定文本序列 $x = \{x_1, x_2, \cdots, x_T\}$，我们首先使用查找层来获取每个词语 $x_i$ 的向量表示（嵌入）$\mathbf{x}_i$。最后一个时间步 $\mathbf{h}_T$ 的输出可以被视为整个序列的表示，其具有全连接层，其后是 Softmax 非线性层，用来预测类别上的概率分布。

$$\hat{\mathbf{y}} = \text{Softmax}(\mathbf{W}\mathbf{h}_T + \mathbf{b}) \tag{7.1}$$

$\hat{\mathbf{y}}$ 是预测概率，$\mathbf{W}$ 是需要学习的权重，$\mathbf{b}$ 是偏置项。




(a) 完全共享模型 (FS-MTL)




(b) 共享私有模型 (SP-MTL)

**图 7.2　两种用于多任务学习的架构**

给定具有 $N$ 个训练样本 $(x_i, y_i)$ 的语料库，我们通过最小化分布的交叉熵来训练网络参数。

$$L(\hat{y}, y) = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_i^j \log(\hat{y}_i^j), \tag{7.2}$$

$y_i^j$ 是真实标签，$\hat{y}_i^j$ 是预测概率，$C$ 是类别数量。

## 7.3　文本分类的多任务学习

多任务学习的目标是利用不同任务之间的相关性来联合学习任务。为此，我们对本文中使用的符号进行了一些解释。在形式上，我们将 $D_k$ 称为具有 $N_k$ 个样本用于任务 $k$ 的数据集：

$$D_k = \{(x_i^k, y_i^k)\}_{i=1}^{N_k} \tag{7.3}$$

$x_i^k$ 和 $y_i^k$ 表示任务 $k$ 的句子和相应标签。

### 7.3.1. 两个用于句子建模的共享方案

多任务学习的关键因素是特征空间中的共享方案。在基于神经网络的模型中，特征可以被视为神经元的状态。对于句子分类，特征是句子末尾的 LSTM 的隐状态。不同多任务学习框架，共享方案是不同的。在这里，我们首先介绍两种具有多任务学习的共享方案：完全共享方案和共享私有方案。

**(1) 完全共享模型（FS-MTL）**

在完全共享模型中，我们使用单个共享 LSTM 层来提取所有任务的特征。例如，给定两个任务 $m$ 和 $n$，它认为任务 $m$ 的特征可以完全由任务 $n$ 共享，反之亦然。此模型忽略了某些特征与任务相关的事实。图7.2b-(a) 说明了完全共享的模型。黄色和灰色框分别代表共享和私有 LSTM 层。

**(2) 共享-私有模型 (SP-MTL)**

如图7.2b-(b) 所示，共享-私有模型为每个任务引入了两个特征空间：一个用于存储与任务特有的特征，另一个用于捕获任务无关特征。因此，我们可以看到每个任务都分配了一个私有 LSTM 层和共享 LSTM 层。对于任务 $k$，我们可以计算任意句子的共享表示 $\mathbf{s}_t^k$ 和任务特有的表示 $\mathbf{h}_t^k$，如下所示：

$$\mathbf{s}_t^k = \mathbf{LSTM}(x_t, \mathbf{s}_{t-1}^k, \theta_s), \tag{7.4}$$

$$\mathbf{h}_t^k = \mathbf{LSTM}(x_t, \mathbf{h}_{t-1}^m, \theta_k) \tag{7.5}$$

最后的特征是特有空间的表示和共享表示的拼接。

### 7.3.2. 任务特有输出层

对于任务 $k$ 中的句子，由深度多任务架构输出的特征 $\mathbf{h}^k$，最终被传递到相应的任务特定的 Softmax 层来用于分类或其他任务。我们通过最小化所有任务上的预测分布和真实分布的交叉熵来训练网络的参数。损失 $L_{task}$ 可以计算为：

$$L_{Task} = \sum_{k=1}^{K} \alpha_k L(\hat{y}^{(k)}, y^{(k)}) \tag{7.6}$$

其中，$\alpha_k$ 表示任务 $k$ 的权重。$L(\hat{y}, y)$ 定义为公式7.2。

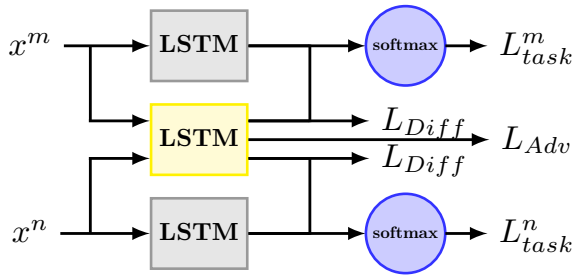



**图 7.3　对抗共享-私有模型**

## 7.4　对抗训练

虽然共享私有模型将特征空间分隔为共享空间和私有空间，但无法保证可共享特征不存在于私有特征空间中，反之亦然。因此，在共享-私有模型中可能忽略一些有用的可共享特征，并且共享特征空间也易受某些特定于任务的信息的污染。

因此，一个简单的原则可以应用于多任务学习，即一个好的共享特征空间应该包含更多的公共信息，而不包含任务特定的信息。为了解决这个问题，我们将对抗训练引入到多任务学习框架中，如图7.3 （ASP-MTL）所示。

### 7.4.1.　对抗网络

最近，对抗网络出现并首次用于生成模型[128]。目标是学习与实际数据分布 $P_{data}x$ 匹配的生成分布 $p_Gx$ ，具体而言，对抗网络学习生成网络 G 和判别模型 D，其中 G 从生成器分布 $p_Gx$ 生成样本。并且 D 学会确定样本是否来自 $p_Gx$ 或 $P_{data}x$。这种最小 - 最大博弈可以通过以下风险进行优化:

$$\phi = \min_G \max_D \Big( E_{x\sim P_{data}}[\log D(x)] + E_{z\sim p(z)}[\log(1 - D(G(z)))] \Big) \tag{7.7}$$

虽然最初提出用于生成随机样本，但是对抗网络可以用作测量分布之间等效性的通用工具[129]。[130] 将对抗损失与两个分布之间的 $H$-divergence 联系起来，并成功将对抗网络应用到了无监督领域迁移。在领域迁移理论[131–133] 的推动下，可迁移特征应该具备这样一个特征：让一个领域判定器无法区分。

### 7.4.2. 多任务学习的任务对抗损失

受到对抗网络的启发[128]，我们提出了一种用于多任务学习的对抗共享-私有模型，其中共享的递归神经层面向可学习的多层感知器工作，防止任务判别器对其任务类型进行准确预测。这种对抗训练鼓励更加纯粹的共享空间，并确保共享特征不受特定任务特征的污染。

#### (1) 任务判别器

判别器用于将句子的共享表示映射到概率分布，判别输入句子来自哪个任务。

$$D(\mathbf{s}_T^k, \theta_D) = \text{Softmax}(\mathbf{b} + \mathbf{U}\mathbf{s}_T^k) \tag{7.8}$$

其中，$\mathbf{U} \in \mathbb{R}^{d \times d}$ 是可学习参数，$\mathbf{b} \in \mathbb{R}^d$ 是偏置。

#### (2) 对抗损失

与大多数现有的多任务学习算法不同，我们添加额外的任务对抗损失 $L_{Adv}$ 以防止特定任务的特征进入共享空间。任务对抗损失用于训练模型以产生共享特征，使得分类器不能基于这些特征可靠地预测任务。对抗网络的原始损失只能用于二元分类的情况。为了克服这个问题，我们将其扩展为多类形式，这样我们的模型就可以与多个任务一起训练:

$$L_{Adv} = \min_{\theta_s} \left( \lambda \max_{\theta_D} (\sum_{k=1}^{K} \sum_{i=1}^{N_k} d_i^k \log[D(E(\mathbf{x}^k))]) \right) \tag{7.9}$$

其中，$d_i^k$ 表示当前任务类型的真实标签。这里，存在最小-最大优化，并且基本思想是，在给定句子条件下，共享 LSTM 生成句子的表示以误导任务判别器。同时，判别器尽力对任务类型进行正确的分类。在训练阶段之后，共享特征提取器和任务判别器达到两者都无法改善的状态，并且判别器无法区分当前的句子属于哪一个任务。

#### (3) 半监督学习多任务学习

我们注意到 $L_{Adv}$ 仅需要输入句子 $x$ 并且不需要相应的标签 $y$，这使得我们可以将我们的模型与半监督学习相结合。最后，在这个半监督的多任务学习框架中，我们的模型不仅可以利用相关任务的数据，而且可以使用丰富的无标签语料

库。

#### (4)　正交性约束

我们注意到上述模型存在潜在的缺点。也就是说，任务不变特征可以出现在共享空间和私有空间中。

在最近关于共享-私有潜在空间分析的工作[133–135] 的启发下，我们引入了正交性约束，以缓解特征空间冗余的问题，且鼓励共享和私有提取器关注输入文本不同层面的信息。在探索了许多可选方法之后，我们发现下面的损失是最优的，它被用于 Bousmalis 等人的工作中[133]，并获得较好的效果。损失函数的定义如下：

$$L_{\text{diff}} = \sum_{k=1}^{K} \left\| \mathbf{S}^{k\top} \mathbf{H}^{k} \right\|_F^2 , \tag{7.10}$$

其中，$|\cdot|_F^2$ 是 Frobenius 的平方范数。$\mathbf{S}^k$ 和 $\mathbf{H}^k$ 是两个矩阵，它们的行是输入句子的共享特征提取器 $E_s(,;\theta_s)$ 和特定任务特征提取器 $E_k(,;\theta_k)$ 的输出。

### 7.4.3.　最终的损失函数

我们模型的最终损失函数可以写成:

$$L = L_{Task} + \lambda L_{Adv} + \gamma L_{Diff} \tag{7.11}$$

其中，$\lambda$ 和 $\gamma$ 是超参数。我们通过反向传播训练网络，并且通过使用梯度反转层[136] 可以实现这种极小极大的优化。

## 7.5　实验与分析

### 7.5.1.　数据集

为了进行广泛的评估，我们从几个常用的评论语料库中收集了 16 个不同的数据集。

前 14 个数据集是产品评论，其中包含来自不同领域的亚马逊产品评论，例如书籍、DVD、电子产品等。分类目标是将产品评论分为正面或负面。这些数据

集是根据[137] 提供的原始数据收集的。具体而言，我们从未处理的原始数据[①]中提取句子和相应的标签。这些句子的唯一预处理操作是使用斯坦福标注器进行 *tokenize* 处理[②]。

剩下的两个数据集是关于电影评论的。IMDB 数据集包含二分类电影评论[138]。该数据集的一个关键地方是每个电影评论都有几个句子，包含正面和负面两种情感。MR 数据集还包括来自烂番茄网站的电影评论，有两个类[③][103]。

每个任务中的所有数据集都被随机分为 70% 作为训练集，20% 作为校验集和 10% 作为测试集。表7.1中列出了有关所有数据集的详细统计信息，第 2-5 列分别表示训练、校验、测试和未标记集中的样本数。最后两列表示相应数据集的平均长度和词汇量。

### 7.5.2. 多任务学习对比方法

现有的工作提出的多任务框架是多种多样的，而并非所有框架都可以应用于我们关注的任务。然而，我们为多任务学习选择了两个最相关的神经模型，并将它们作为对比方法实现。

- MT-CNN: 这个模型由 Collobert 等人[27] 提出，其中查找表是部分共享的，而其他和任务相关的。
- MT-DNN: 该模型由 Liu 等人[28] 提出，具有词袋输入和多层感知器，其中共享隐藏层。

### 7.5.3. 超参数

所有模型的词嵌入使用 200 维 GloVe 向量[87] 进行初始化。其他参数通过服从范围为 $[-0.1, 0.1]$ 的均匀分布随机初始化。mini-batch size 设置为 16。

对于每个任务，我们采用网格搜索的方法，在校验集上实现最佳性能的参数。最后，我们选择学习率为 0.01，$\lambda$ 为 0.05 和 $\gamma$ 为 0.01。

[①] [137] 还提供了两个额外的处理过的数据集，其格式为 Bag-of-Words，不适用于基于神经的模型

[②] http://nlp.stanford.edu/software/tokenizer.shtml

[③] https://www.cs.cornell.edu/people/pabo/movie-review-data/.

**表 7.1　16 个数据集的统计数据**

| 数据集 | 训练集 | 校验集 | 测试集 | 无标记样本 | 平均长度 | 词表大小 |
|---|---|---|---|---|---|---|
| Books | 1400 | 200 | 400 | 2000 | 159 | 62K |
| Elec. | 1398 | 200 | 400 | 2000 | 101 | 30K |
| DVD | 1400 | 200 | 400 | 2000 | 173 | 69K |
| Kitchen | 1400 | 200 | 400 | 2000 | 89 | 28K |
| Apparel | 1400 | 200 | 400 | 2000 | 57 | 21K |
| Camera | 1397 | 200 | 400 | 2000 | 130 | 26K |
| Health | 1400 | 200 | 400 | 2000 | 81 | 26K |
| Music | 1400 | 200 | 400 | 2000 | 136 | 60K |
| Toys | 1400 | 200 | 400 | 2000 | 90 | 28K |
| Video | 1400 | 200 | 400 | 2000 | 156 | 57K |
| Baby | 1300 | 200 | 400 | 2000 | 104 | 26K |
| Mag. | 1370 | 200 | 400 | 2000 | 117 | 30K |
| Soft. | 1315 | 200 | 400 | 475 | 129 | 26K |
| Sports | 1400 | 200 | 400 | 2000 | 94 | 30K |
| IMDB | 1400 | 200 | 400 | 2000 | 269 | 44K |
| MR | 1400 | 200 | 400 | 2000 | 21 | 12K |

### 7.5.4. 性能评估

表7.2显示了 16 个文本分类任务的错误率。“单任务” 列显示了标准 LSTM，双向 LSTM（BiLSTM），堆叠 LSTM （sLSTM）以及前三种模型的平均错误率。“多任务” 列显示了相应多任务模型的结果。从该表中可以看出，在多任务学习的帮助下，大多数任务的性能可以大幅度提高，其中我们的模型实现了最低的错误率。更具体地说，与 SP-MTL 相比，ASP-MTL 效果平均提高了 4.1%，超过了 SP-MTL 1.0%，这表明了对抗学习的重要性。值得注意的是，对于 FS-MTL，某些任务的性能会降低，因为该模型将所有私有和共享信息放入统一空间。

**表 7.2　我们的模型与典型基线模型在 16 个数据集上的错误率对比**

| 任务 | 单任务 | | | | 多任务 | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | LSTM | BiLSTM | sLSTM | Avg. | MT-DNN | MT-CNN | FS-MTL | SP-MTL | ASP-MTL |
| Books | 20.5 | 19.0 | 18.0 | 19.2 | $17.8_{(-1.4)}$ | $15.5_{(-3.7)}$ | $17.5_{(-1.7)}$ | $18.8_{(-0.4)}$ | $16.0_{(-3.2)}$ |
| Electronics | 19.5 | 21.5 | 23.3 | 21.4 | $18.3_{(-3.1)}$ | $16.8_{(-4.6)}$ | $14.3_{(-7.1)}$ | $15.3_{(-6.1)}$ | $13.2_{(-8.2)}$ |
| DVD | 18.3 | 19.5 | 22.0 | 19.9 | $15.8_{(-4.1)}$ | $16.0_{(-3.9)}$ | $16.5_{(-3.4)}$ | $16.0_{(-3.9)}$ | $14.5_{(-5.4)}$ |
| Kitchen | 22.0 | 18.8 | 19.5 | 20.1 | $19.3_{(-0.8)}$ | $16.8_{(-3.3)}$ | $14.0_{(-6.1)}$ | $14.8_{(-5.3)}$ | $13.8_{(-6.3)}$ |
| Apparel | 16.8 | 14.0 | 16.3 | 15.7 | $15.0_{(-0.7)}$ | $16.3_{(+0.6)}$ | $15.5_{(-0.2)}$ | $13.5_{(-2.2)}$ | $13.0_{(-2.7)}$ |
| Camera | 14.8 | 14.0 | 15.0 | 14.6 | $13.8_{(-0.8)}$ | $14.0_{(-0.6)}$ | $13.5_{(-1.1)}$ | $12.0_{(-2.6)}$ | $10.8_{(-3.8)}$ |
| Health | 15.5 | 21.3 | 16.5 | 17.8 | $14.3_{(-3.5)}$ | $12.8_{(-5.0)}$ | $12.0_{(-5.8)}$ | $12.8_{(-5.0)}$ | $11.8_{(-6.0)}$ |
| Music | 23.3 | 22.8 | 23.0 | 23.0 | $15.3_{(-7.7)}$ | $16.3_{(-6.7)}$ | $18.8_{(-4.2)}$ | $17.0_{(-6.0)}$ | $17.5_{(-5.5)}$ |
| Toys | 16.8 | 15.3 | 16.8 | 16.3 | $12.3_{(-4.0)}$ | $10.8_{(-5.5)}$ | $15.5_{(-0.8)}$ | $14.8_{(-1.5)}$ | $12.0_{(-4.3)}$ |
| Video | 18.5 | 16.3 | 16.3 | 17.0 | $15.0_{(-2.0)}$ | $18.5_{(+1.5)}$ | $16.3_{(-0.7)}$ | $16.8_{(-0.2)}$ | $15.5_{(-1.5)}$ |
| Baby | 15.3 | 16.5 | 15.8 | 15.9 | $12.0_{(-3.9)}$ | $12.3_{(-3.6)}$ | $12.0_{(-3.9)}$ | $13.3_{(-2.6)}$ | $11.8_{(-4.1)}$ |
| Magazines | 10.8 | 8.5 | 12.3 | 10.5 | $10.5_{(+0.0)}$ | $12.3_{(+1.8)}$ | $7.5_{(-3.0)}$ | $8.0_{(-2.5)}$ | $7.8_{(-2.7)}$ |
| Software | 15.3 | 14.3 | 14.5 | 14.7 | $14.3_{(-0.4)}$ | $13.5_{(-1.2)}$ | $13.8_{(-0.9)}$ | $13.0_{(-1.7)}$ | $12.8_{(-1.9)}$ |
| Sports | 18.3 | 16.0 | 17.5 | 17.3 | $16.8_{(-0.5)}$ | $16.0_{(-1.3)}$ | $14.5_{(-2.8)}$ | $12.8_{(-4.5)}$ | $14.3_{(-3.0)}$ |
| IMDB | 18.3 | 15.0 | 18.5 | 17.3 | $16.8_{(-0.5)}$ | $13.8_{(-3.5)}$ | $17.5_{(+0.2)}$ | $15.3_{(-2.0)}$ | $14.5_{(-2.8)}$ |
| MR | 27.3 | 25.3 | 28.0 | 26.9 | $24.5_{(-2.4)}$ | $25.5_{(-1.4)}$ | $25.3_{(-1.6)}$ | $24.0_{(-2.9)}$ | $23.3_{(-3.6)}$ |
| AVG | 18.2 | 17.4 | 18.3 | 18.0 | $15.7_{(-2.2)}$ | $15.5_{(-2.5)}$ | $15.3_{(-2.7)}$ | $14.9_{(-3.1)}$ | $13.9_{(-4.1)}$ |

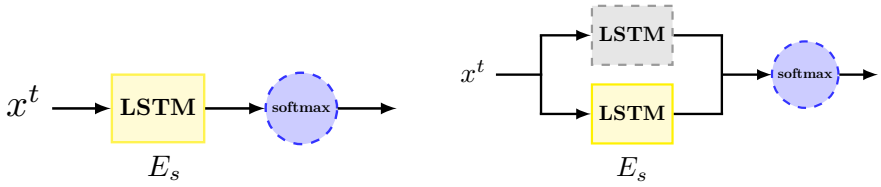
### 7.5.5. 共享知识迁移

在对抗学习的帮助下，共享特征提取器 $E_s$ 可以生成更纯粹、任务无关的表示，可以将其视为现成的知识，然后用于未见过的新任务。

为了测试我们学习的共享特征提取器的可迁移性，我们还设计了一个实验，我们轮流选择 15 个任务来训练我们的模型 $M_S$ 进行多任务学习，然后将学习的共享层转移到第二个网络 $M_T$，用于剩余的一个任务。传输层的参数**保持冻结**，并且随机初始化网络 $M_T$ 的其余参数。

我们调研了两种迁移的策略：如图7.4所示，单通道（SC）模型由一个来自 $M_S$ 的共享特征提取器 $E_s$ 组成，然后提取的表示将被传递到输出层。相比之下，

双通道（BC）模型引入了额外的 LSTM 层来编码更多任务特有的信息。为了评估我们引入的对抗训练框架的有效性，我们还与典型的多任务学习方法进行了比较。




(a) 单通道　　(b) 双通道

**图 7.4　使用预训练共享 LSTM 层的两种转移策略**

#### (1)　结果与分析

如表7.3所示，我们可以看到，与 SP-MTL 相比，ASP-MTL 的共享层实现了更好的性能。此外，对于两种转移策略，双通道模型表现更好。原因是双通道模型中引入的任务特定层可以存储一些私有特征。总的来说，实验结果表明我们可以使用对抗多任务学习将现有知识保存到共享的循环层中，这对于新任务非常有用。

### 7.5.6.　可视化分析

为了直观地了解引入的正交性约束如何与普通的共享-私有模型相比较，我们设计了一个实验来检查私有层和共享层中神经元的行为。更具体地说，我们将 $h_{tj}$ 称为在时间步 $t$ 的 $j$-神经元的激活值，其中 $t \in \{1 \dots n\}$ 和 $j \in \{1 \dots d\}$。通过可视化隐状态 $\mathbf{h}_j$ 并分析最大激活值，我们可以找到当前神经元关注的模式类型。

图7.5-(a) 说明了这种现象，Y 轴表示情感分数，而 X 轴表示按时间顺序的输入。较深的灰色水平线给出正面和负面情感之间的边界，我们从 Baby 任务的校验集中随机抽样一个句子，并分析在不同时间步预测的情感分数的变化。此外，为了更深入地了解共享层中的神经元针对不同的输入的变化，我们可视化两个典型神经元的激活值。对于正面句子，“`Five stars, my baby can`

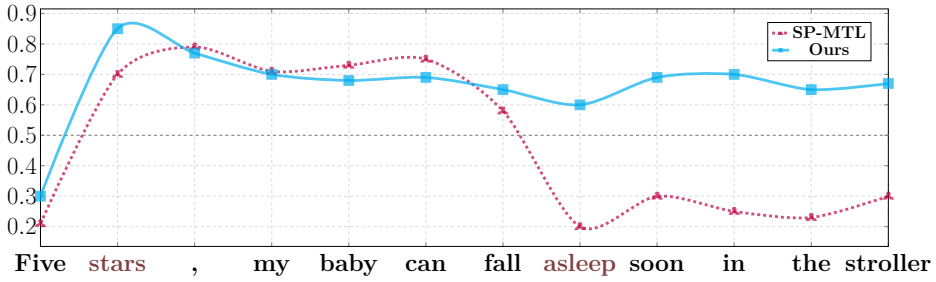
**表 7.3　我们的模型与标准多任务学习在 16 个数据集上的错误率对比**

| 源任务 | 单任务 | | | | 迁移模型 | | | |
|---|---|---|---|---|---|---|---|---|
| | LSTM | BiLSTM | sLSTM | Avg. | SP-MTL-SC | SP-MTL-BC | ASP-MTL-SC | ASP-MTL-BC |
| $\phi$ (Books) | 20.5 | 19.0 | 18.0 | 19.2 | $17.8_{(-1.4)}$ | $16.3_{(-2.9)}$ | $16.8_{(-2.4)}$ | $16.3_{(-2.9)}$ |
| $\phi$ (Electronics) | 19.5 | 21.5 | 23.3 | 21.4 | $15.3_{(-6.1)}$ | $14.8_{(-6.6)}$ | $17.8_{(-3.6)}$ | $16.8_{(-4.6)}$ |
| $\phi$ (DVD) | 18.3 | 19.5 | 22.0 | 19.9 | $14.8_{(-5.1)}$ | $15.5_{(-4.4)}$ | $14.5_{(-5.4)}$ | $14.3_{(-5.6)}$ |
| $\phi$ (Kitchen) | 22.0 | 18.8 | 19.5 | 20.1 | $15.0_{(-5.1)}$ | $16.3_{(-3.8)}$ | $16.3_{(-3.8)}$ | $15.0_{(-5.1)}$ |
| $\phi$ (Apparel) | 16.8 | 14.0 | 16.3 | 15.7 | $14.8_{(-0.9)}$ | $12.0_{(-3.7)}$ | $12.5_{(-3.2)}$ | $13.8_{(-1.9)}$ |
| $\phi$ (Camera) | 14.8 | 14.0 | 15.0 | 14.6 | $13.3_{(-1.3)}$ | $12.5_{(-2.1)}$ | $11.8_{(-2.8)}$ | $10.3_{(-4.3)}$ |
| $\phi$ (Health) | 15.5 | 21.3 | 16.5 | 17.8 | $14.5_{(-3.3)}$ | $14.3_{(-3.5)}$ | $12.3_{(-5.5)}$ | $13.5_{(-4.3)}$ |
| $\phi$ (Music) | 23.3 | 22.8 | 23.0 | 23.0 | $20.0_{(-3.0)}$ | $17.8_{(-5.2)}$ | $17.5_{(-5.5)}$ | $18.3_{(-4.7)}$ |
| $\phi$ (Toys) | 16.8 | 15.3 | 16.8 | 16.3 | $13.8_{(-2.5)}$ | $12.5_{(-3.8)}$ | $13.0_{(-3.3)}$ | $11.8_{(-4.5)}$ |
| $\phi$ (Video) | 18.5 | 16.3 | 16.3 | 17.0 | $14.3_{(-2.7)}$ | $15.0_{(-2.0)}$ | $14.8_{(-2.2)}$ | $14.8_{(-2.2)}$ |
| $\phi$ (Baby) | 15.3 | 16.5 | 15.8 | 15.9 | $16.5_{(+0.6)}$ | $16.8_{(+0.9)}$ | $13.5_{(-2.4)}$ | $12.0_{(-3.9)}$ |
| $\phi$ (Magazines) | 10.8 | 8.5 | 12.3 | 10.5 | $10.5_{(+0.0)}$ | $10.3_{(-0.2)}$ | $8.8_{(-1.7)}$ | $9.5_{(-1.0)}$ |
| $\phi$ (Software) | 15.3 | 14.3 | 14.5 | 14.7 | $13.0_{(-1.7)}$ | $12.8_{(-1.9)}$ | $14.5_{(-0.2)}$ | $11.8_{(-2.9)}$ |
| $\phi$ (Sports) | 18.3 | 16.0 | 17.5 | 17.3 | $16.3_{(-1.0)}$ | $16.3_{(-1.0)}$ | $13.3_{(-4.0)}$ | $13.5_{(-3.8)}$ |
| $\phi$ (IMDB) | 18.3 | 15.0 | 18.5 | 17.3 | $12.8_{(-4.5)}$ | $12.8_{(-4.5)}$ | $12.5_{(-4.8)}$ | $13.3_{(-4.0)}$ |
| $\phi$ (MR) | 27.3 | 25.3 | 28.0 | 26.9 | $26.0_{(-0.9)}$ | $26.5_{(-0.4)}$ | $24.8_{(-2.1)}$ | $23.5_{(-3.4)}$ |
| AVG | 18.2 | 17.4 | 18.3 | 18.0 | $15.6_{(-2.4)}$ | $15.2_{(-2.8)}$ | $14.7_{(-3.3)}$ | $14.3_{(-3.7)}$ |

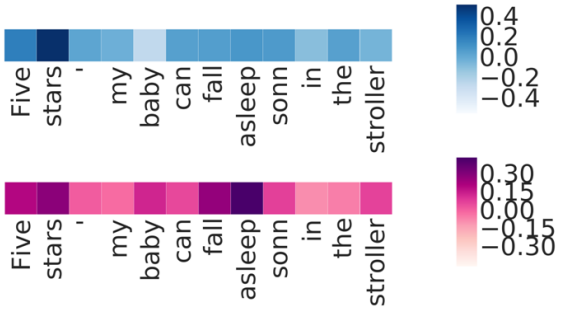
`fall asleep soon in the stroller`”，两个模型都捕获了更有信息量的特征 “`Five stars`”[④]。然而，由于对 “`asleep`” 这个词的误解，SP-MTL 做出了错误的预测。

相反地，我们的模型 ASP-MTL 做出了正确的预测，其原因可以从图7.5-(b)的激活值中推断出来，其中紫色热图描述了来自 SP-MTL 共享层的神经元 $\mathbf{h}^s_{18}$ 的行为，蓝色热图表示我们模型的共享层神经元 $\mathbf{h}^s_{21}$ 的行为。我们发现，SP-MTL 的共享层非常敏感，以至于包含了其他任务相关的特征会影响模型的预测，例如”`asleep`”，它误导了最终的预测。这表明引入对抗学习可以防止共享层被任务特有的特征污染。

[④]对于这种情况，由于忽略了 “`Five stars`” 这个特征，典型 LSTM 也给出了错误的答案。




(a) 两种模型预测情感评分




(b) 神经元行为 $\mathbf{h}^s_{18}$ and $\mathbf{h}^s_{21}$

**图 7.5　(a) 不同时间步情感分数的变化；(b) 神经元的行为**

**表 7.4　在 Movie 和 Baby 任务中不同模式下学习到的特征**

| 模型 | 共享层 | 任务-Movie | 任务-Baby |
|---|---|---|---|
| SP-MTL | good, great | good, great | love, bad |
| | bad, love | well-directed | cute, safety |
| | simple, cut | pointless, cut | mild, broken |
| | slow, cheap | cheap, infantile | simple |
| | infantile | | |
| ASP-MTL | good, great | well-directed | cute, safety |
| | love, bad | pointless, cut | mild, broken |
| | poor | cheap, infantile | simple |

表7.4给出了来自共享层和任务特定层的神经元捕获的一些典型模式，我们观察到：1）对于 SP-MTL，如果某些模式由任务特定层捕获，则它们可能被放置在共享空间中。显然，假设我们有许多共同训练的任务，共享层将承受很大的压力，它必须牺牲大量的精力来捕捉它们实际上不需要的特征。此外，一些典型

的任务无关的特征也会进入任务特定层。2）对于 ASP-MTL，我们发现共享层和任务特定层捕获的特征具有少量交集，这使得这两种层可以有效地工作。

## 7.6　本章小结

在本文中，我们提出了一种基于对抗训练的多任务学习框架，它能使任务特有和任务无关的特征进行非冗余的学习，因此捕获不同任务的共享-私有分离的特征。我们将模型应用于 16 种不同的文本分类任务，证明了我们方法的有效性。我们还进行了广泛的定性分析，并解释该方法性能提升的机理。

# 第 8 章　可理解性表示学习

## 8.1　引言

神经多任务学习模型已在许多 NLP 任务中取得了领先的结果，包括词性（POS）标注[106,139]，句法解析[140,141]，文本分类[142,143] 和机器翻译[144,145]。

对于基于神经网络的多任务学习，现有工作通常通过简单地在一些预定义的任务结构上共享参数来学习任务相关性。**扁平结构**[27] 假设所有任务共享一个隐藏空间，而 **层次结构**[139,146] 指定任务之间信息流方向的顺序。图8.1-(a，b) 显示了两种典型预定义的拓扑结构。每个蓝色圆圈表示任务（A，B，C，D），而红色框表示额外引入的虚拟结点，它可以用来存储共享信息并促进任务之间的通信强度。线条的粗细表明了任务之间关系的强弱程度。有向边表示信息流的方向。例如，在图8.1-(b) 中，任务 C 从任务 A 接收信息，反之亦然。

上述方法有两个主要缺点。首先，预定义静态的结构对任务之间交互是一种很强的假设，它限制了模型利用共享信息的能力。例如，图8.1-(a) 中的结构不允许模型显式地学习任务之间的相关性，它限制了模型对数据固有结构[147] 的利用。通常地，任务之间的相关性程度本身并不是静态的，可能会发生变化，具体取决于手头的数据样本。另一个缺点是，研究人员和系统开发人员通常无法解释这些多任务学习模型，这意味着除了参数本身之外，我们对共享的模式类型知之甚少。以前的非神经网络模型[31–33] 已经证明了利用多任务学习来学习任务间关系的重要性。然而，在神经网络环境中进行深入分析的工作却很少。

以上分析激发了以下研究问题：1）我们如何显式地建模不同任务之间的复杂关系？2）我们可以设计学习可解释共享结构的模型吗？

为了解决这些问题，我们提出通过**图结构**对文本表示学习任务之间的关系进行建模，其中每个任务都被视为一个结点。我们从消息传递的概念中获得启发[50,51,148,149]，设计了两种任务通信的方法，其中消息可以直接（利用完全图，图8.1-(c)）或间接（利用 星型图，图8.1- (d)）在任意两个结点之间传递。

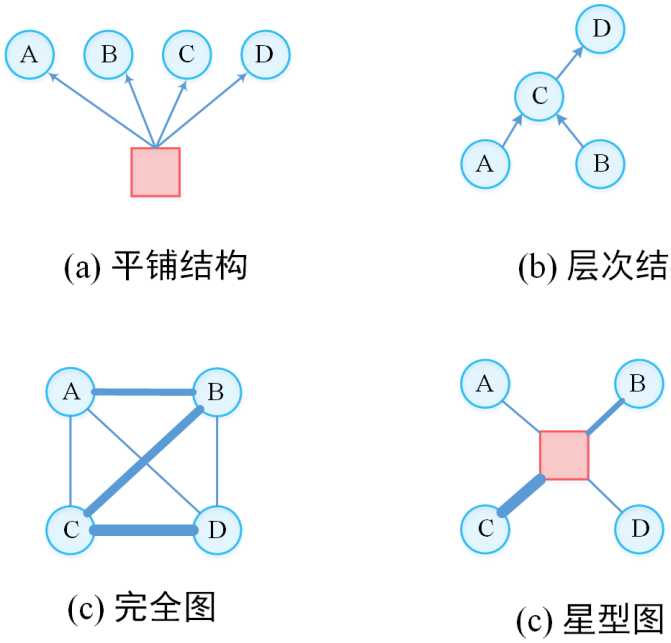



**图 8.1　多任务学习的不同拓扑结构**

我们在文本分类和序列标注任务上评估我们所提出的模型。此外，我们在多任务设置和迁移学习设置中进行实验，验证了我们的模型所学习的共享知识对新任务的有效性。实验结果表明，我们的方法不仅能够有效地降低错误率，还提供了共享知识的良好解释。

本文的贡献可归纳如下：

1. 我们探讨了建模多个任务之间关系建模的问题，并将其转化为通过图神经网络进行消息传递的问题。
2. 我们所提出的方法允许多个任务之间动态通信，而不是遵循预定义的结构。
3. 与传统的黑盒学习模型不同，本文朝着学习**可迁移性**和**可解释性**表示迈出了一步，这使我们能够对共享的知识有更多的理解。

## 8.2　用于多任务通信的消息传递框架

我们提出使用带有消息传递的图神经网络来处理多任务序列学习的问题，并在文本分类和序列标注任务上进行评估。我们将文本序列表示为 $X = \{x_1, \ldots, x_T\}$，输出为 $Y$。在文本分类中，$Y$ 是单个标签；而在序列标记中，$Y = \{y_1, y_2, \ldots, y_T\}$

是一个序列。

假设有 $K$ 个相关任务，对于任务 $k$，我们将 $D_k$ 表示为任务 $k$ 的数据集，$N_k$ 表示为任务 $k$ 的样本数量。

$$D_k = \{(X_i^{(k)}, Y_i^{(k)})\}_{i=1}^{N_k}, \tag{8.1}$$

$X_i^{(k)}$ 和 $Y_i^{(k)}$ 分别表示任务 $k$ 的文本序列和相应的标签序列。

通常，在将多任务学习与序列学习相结合时，应该对两种交互进行建模：一种是句子中不同词语之间的相互作用，另一种是不同任务之间的交互。

对于第一种类型的交互（**句子中词语的交互**），可以通过应用合成函数来获得句子的表示。定义合成函数的典型选择包括递归神经网络[46]，卷积神经网络[9] 和树结构神经网络[10]。在本文中，我们采用 LSTM 架构来学习句子中的依赖关系，因为它们在许多 NLP 任务中表现出色[56]。我们将 $\mathbf{h}_t$ 称为 $t$ 时刻词语 $w_t$ 的隐状态。然后，$\mathbf{h}_t$ 可以计算为

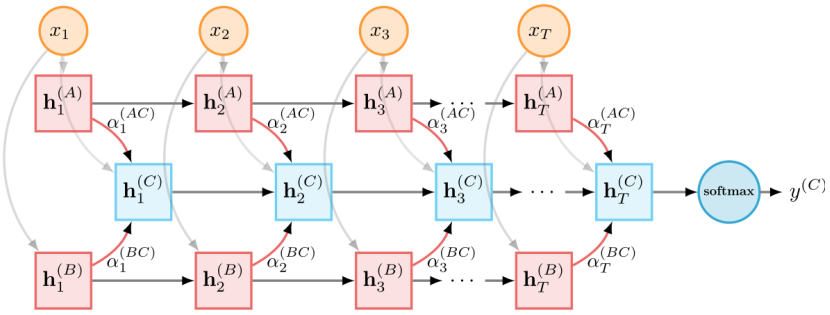
$$\mathbf{h}_t = \mathbf{LSTM}(\mathbf{x}_t, \mathbf{h}_{t-1}, \theta). \tag{8.2}$$

这里，$\theta$ 代表 LSTM 的所有参数。

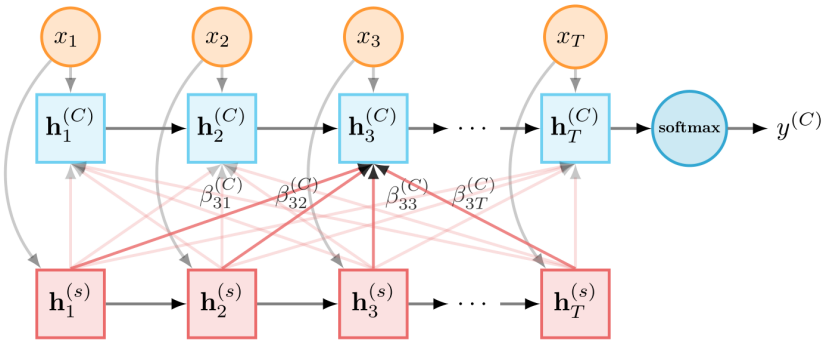
对于第二种类型的交互（**跨越不同任务的交互**），我们提出将任务及其交互概念化为图，并利用消息传递机制来进行通信。我们的框架受到消息传递的启发，它在现代计算机软件[148] 和编程语言[149] 中无处不在。这个框架的一般概念是我们提供一个图网络，允许不同的任务相互合作和互动。下面，我们描述了图构建过程的概念化，然后描述了用于任务间通信的消息传递机制。

一个图 $G$ 可以定义为有序对 $G = (V, E)$，其中 $V$ 是一组结点 $\{V_1, \ldots, V_m\}$，$E$ 是一组边。在这项工作中，我们使用有向图来建模任务之间的通信流，因此边被定义为两个结点的有序关系 $(V_i, V_j)$，$i \neq j$。

在我们的模型中，我们将每个任务表示为一个结点。此外，我们允许引入虚拟结点。这些虚拟结点不对应于任务。相反，它们的目的是促进不同任务之间的通信。直观地，虚拟结点就像是邮箱，存储来自其他结点的消息并根据需要将消息分发出去。




(a) 完全图的多任务学习框架




(b) 星型图的多任务学习框架

**图 8.2　多任务交互的两种框架**

任务和虚拟结点通过加权边连接，加权边表示不同结点之间的通信。用于多任务学习的扁平和分层体系结构可被视为具有固定边连接的图形。我们的模型动态地学习每个边的权重，这允许模型调整消息的强度。

### 8.2.1.　信息传递

在我们的图结构中，我们使用有向边来建模任务之间的通信。换句话说，结点通过边发送和接收消息来相互通信。给定一个带有来自任务 $k$ 的词语 $w_1^k...w_T^k$ 的句子，我们使用 $\mathbf{r}_t^k$ 来表示在时刻 $t$ 任务 $k$ 的词语可以获得的**聚合消息**，我们使用 $\mathbf{h}_t^k$ 来表示任务相关词语 $w_t^k$ 的隐状态。

下面，我们提出了两种用于消息传递的基本通信体系结构：完全图（*Complete-graph*）和星型图（*Star-graph*）。它们根据是否允许任务之间直接通信，或者通信是否依赖于一个中间虚拟结点来区分。

**1. 完全图 (CG): 用于多任务学习的直接通信。** 在该模型中，每个结点可以直接向（或从）任何其他结点发送（或接收）消息。具体来说，如图8.2-(a) 所示，

我们首先为每个任务分配一个与任务相关的 LSTM 层。任务 $k$ 中的每个句子都可以传递给任务相关的所有其他 LSTM 以获得相应的表示形式 $\mathbf{h}_t^{(i)}$，$i = 1...K$，$i \neq k$ 然后，这些消息将汇总为：

$$\mathbf{r}_t^{(k)} = \sum_{i=1...K s.t. i \neq k} \alpha_t^{(i \to k)} \mathbf{h}_t^{(i)} \tag{8.3}$$

这里，$\alpha_t^{(ki)}$ 是一个标量，它控制两个任务 $k$ 和 $i$ 之间的相关性，并且可以动态计算为：

$$\mathbf{s}_t^{(i \to k)} = f(\mathbf{x}_t^{(k)}, \mathbf{h}_{t-1}^{(k)}, \mathbf{h}_t^{(i)}) \tag{8.4}$$

$$= \mathbf{u}^{(s)} \tanh(\mathbf{W}^{(s)}[\mathbf{x}_t, \mathbf{h}_{t-1}^{(k)}, \mathbf{h}_t^{(i)}]) \tag{8.5}$$

$\mathbf{u}^{(s)}$ 和 $\mathbf{W}^{(s)}$ 是可学习的参数。相关性得分将归一化为概率分布:

$$\boldsymbol{\alpha}_t = \mathrm{Softmax}(\mathbf{s}_t) \tag{8.6}$$

**2. 星型图 (SG): 是用于多任务学习的间接通信**。以上的的 CG-MTL 模型的潜在局限在于其计算成本，因为成对交互的数量随着任务的数量呈二次方增长。受传统消息传递范例[150] 中使用的邮箱构思的启发，我们在图中引入了一个额外的虚拟结点来解决这个问题。在此设置中，消息不会直接从一个结点发送到另一个结点，而是由虚拟结点桥接。直观地，虚拟结点在所有任务中存储共享消息，不同的任务可以将消息放入这个全局空间，然后其他任务可以从同一空间中为自己取出有用的消息。图8.2-(b) 展示了星型图多任务学习框架中一个任务从邮箱收集信息 (共享层) 的过程。详细地说，我们引入了一个额外的 LSTM 作为虚拟结点，其参数在任务之间共享。给定来自任务 $k$ 的句子，可以通过以下操作将其信息写入共享 LSTM:

$$\mathbf{h}_t^{(s)} = \mathbf{LSTM}(x_t^{(k)}, \mathbf{h}_{t-1}^{(s)}, \theta^{(s)}), \tag{8.7}$$

其中，$\theta^s$ 表示在所有任务中共享的参数。

然后，可以从共享 LSTM 读取 $t$ 时刻的聚合消息

$$\mathbf{r}_t^{(k)} = \sum_{i=1}^{T} \beta_{t \to i}^{(k)} \mathbf{h}_i^{(s)}, \tag{8.8}$$

其中，$T$ 表示句子的长度，$\beta^k_{t\to i}$ 用于从共享 LSTM 中检索有用的消息，它的计算方式类似于等式8.5和等式8.6。

一旦定义了结点之间消息传递和图，下一个问题是如何使用当前输入信息 $\mathbf{x}_t$ 和汇总的消息 $\mathbf{r}^{(k)}_t$ 给结点 $k$ 更新任务相关的表示 $\mathbf{h}^{(k)}_t$ 。我们使用一个门控单元，允许模型决定应该将多少汇总消息用于目标任务，从而避免不必要的信息冗余。$\mathbf{h}^{(k)}_t$ 可以这样计算:

$$\mathbf{h}^{(k)}_t = \mathbf{LSTM}^{\dagger}(\mathbf{x}_t, \mathbf{h}^{(k)}_{t-1}, \mathbf{r}^{(k)}_t, \theta^{(k)}, \theta^{(s)}). \tag{8.9}$$

函数 $\mathbf{LSTM}^{\dagger}$ 与等式8.2相同，只不过我们在用以下等式替换等式8.2内部函数的记忆更新步骤。

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t + \mathbf{g}_t \odot (\mathbf{W}^{(s)}_f \mathbf{r}_t)), \tag{8.10}$$

其中，$\mathbf{W}^{(s)}_f$ 是参数矩阵，$\mathbf{g}_t$ 是选择聚合消息的融合门。$\mathbf{g}_t$ 的计算方法如下:

$$\mathbf{g}_t = \sigma(\mathbf{W}^{(s)}_r \mathbf{r}^{(k)}_t + \mathbf{W}^{(s)}_c \mathbf{c}_t), \tag{8.11}$$

其中，$\mathbf{W}^{(s)}_r$ 和 $\mathbf{W}^{(s)}_c$ 是参数矩阵。

#### (1)　完全图 (CG) 与星型图 (SG) 的对比

CG-MTL 的优点是可以计算出从源任务中的词语到目标任务中的词语的关联强度。但是，如果任务数量太大，则 CG-MTL 的计算效率不高。SG-MTL 的优点是学习的共享结构是可解释的，并且它学习的知识可用于未见过的任务。

总之，CG-MTL 可用于以下场景：1）任务数量不是太大，2）需要明确分析不同任务之间的相关性。相比之下，SG-MTL 可用于以下场景：1）任务数量很大，2）需要将共享知识转移到新任务上，3）需要分析模型学习的共享模式类型。

### 8.2.2.　任务相关的图层

给定一个来自任务 $k$ 的句子 $X$，其标签为 $Y$（注意 $Y$ 是分类标签或序列标签）及其由以上两个通信方法产生的特征向量 $\mathbf{h}^k$，我们可以通过使用不同的任务特定层使我们的模型适应于不同的任务。我们将任务特定层称为输出层。对于

文本分类任务，常用的输出层是 Softmax 层，而对于序列标记任务，它可以是条件随机场（CRF）层。最后，输出概率 $P(Y|X)$ 可以计算为:

$$P(Y|X) = \text{OUTPUT-LAYER}(X, \mathbf{h}_T^{(k)}, \theta^{(k)}) \tag{8.12}$$

然后，我们可以最大化上述概率来优化每个任务的参数:

$$\mathcal{L}_k(X, Y, \theta^{(k)}, \theta^{(s)}) = -P(Y|X). \tag{8.13}$$

一旦定义了特定任务的损失函数，我们就可以计算出多任务模型的总体损失，如下所示:

$$\mathcal{L} = \sum_{k=1}^{K} \lambda_k \mathcal{L}_k(X, Y, \theta^{(k)}, \theta^{(s)}) \tag{8.14}$$

其中，$\lambda_k$ 是任务 $k$ 的权重。整个网络的参数在所有数据集上进行训练。

## 8.3　实验与分析

在本节中，我们描述了超参数设置，并在两种类型的多任务学习数据集上展示了我们提出的两个模型的性能，首先是文本分类，然后是序列标记。每个数据集包含几个相关任务。

### 8.3.1.　超参数

所有模型的词嵌入使用 200 维 GloVe 向量[87] 进行初始化。其他参数随机初始化为服从 $[-0.1, 0.1]$ 范围的均匀分布。

对于每个任务，我们通过网格搜索隐状态数目 $[100, 200, 300]$ 和 $l_2$ 正则化 $[0.0, 5E-5, 1E]$ 的组合超参数。另外，对于文本分类任务，我们为每个任务设置相等的 $\lambda$；对于序列标记任务，我们使用网格搜索 $\lambda$，范围为 $[1, 0.8, 0.5]$。我们选择隐状态的数目为 200 和 $l_2$ 为 0.0。我们的优化器采用随机梯度下降[105]。

### 8.3.2.　文本分类

为了研究多任务学习的有效性，我们在 16 种常用的文本分类任务[143] 上评估所提出来的模型。每个子任务旨在预测给定句子的正确情感标签（正面或负

**表 8.1　我们的模型以及经典基线模型在 16 个数据集上的文本分类错误率**

| 任务 | 单任务 | 多任务 | | | | | 迁移 | |
|---|---|---|---|---|---|---|---|---|
| | Avg. | MT-CNN | FS-MTL | SP-MTL | CG-MTL* | SG-MTL* | SP-MTL | SG-MTL* |
| Books | 19.2 | $15.5_{(-3.7)}$ | $17.5_{(-1.7)}$ | $16.0_{(-3.2)}$ | $13.3_{(-5.9)}$ | $13.8_{(-5.4)}$ | $16.3_{(-2.9)}$ | $14.5_{(-4.7)}$ |
| Electronics | 21.4 | $16.8_{(-4.6)}$ | $14.3_{(-7.1)}$ | $13.2_{(-8.2)}$ | $11.5_{(-9.9)}$ | $11.5_{(-9.9)}$ | $16.8_{(-4.6)}$ | $13.8_{(-7.6)}$ |
| DVD | 19.9 | $16.0_{(-3.9)}$ | $16.5_{(-3.4)}$ | $14.5_{(-5.4)}$ | $13.5_{(-6.4)}$ | $12.0_{(-7.9)}$ | $14.3_{(-5.6)}$ | $14.0_{(-5.9)}$ |
| Kitchen | 20.1 | $16.8_{(-3.3)}$ | $14.0_{(-6.1)}$ | $13.8_{(-6.3)}$ | $12.3_{(-7.8)}$ | $11.8_{(-8.3)}$ | $15.0_{(-5.1)}$ | $12.8_{(-7.3)}$ |
| Apparel | 15.7 | $16.3_{(+0.6)}$ | $15.5_{(-0.2)}$ | $13.0_{(-2.7)}$ | $13.0_{(-2.7)}$ | $12.5_{(-3.2)}$ | $13.8_{(-1.9)}$ | $13.5_{(-2.2)}$ |
| Camera | 14.6 | $14.0_{(-0.6)}$ | $13.5_{(-1.1)}$ | $10.8_{(-3.8)}$ | $10.5_{(-4.1)}$ | $11.0_{(-3.6)}$ | $10.3_{(-4.3)}$ | $11.0_{(-3.6)}$ |
| Health | 17.8 | $12.8_{(-5.0)}$ | $12.0_{(-5.8)}$ | $11.8_{(-6.0)}$ | $10.5_{(-7.3)}$ | $10.5_{(-7.3)}$ | $13.5_{(-4.3)}$ | $11.0_{(-6.8)}$ |
| Music | 23.0 | $16.3_{(-6.7)}$ | $18.8_{(-4.2)}$ | $17.5_{(-5.5)}$ | $14.8_{(-8.2)}$ | $14.3_{(-8.7)}$ | $18.3_{(-4.7)}$ | $14.8_{(-8.2)}$ |
| Toys | 16.3 | $10.8_{(-5.5)}$ | $15.5_{(-0.8)}$ | $12.0_{(-4.3)}$ | $11.0_{(-5.3)}$ | $10.8_{(-5.5)}$ | $11.8_{(-4.5)}$ | $10.8_{(-5.5)}$ |
| Video | 17.0 | $18.5_{(+1.5)}$ | $16.3_{(-0.7)}$ | $15.5_{(-1.5)}$ | $13.0_{(-4.0)}$ | $14.0_{(-3.0)}$ | $14.8_{(-2.2)}$ | $13.5_{(-3.5)}$ |
| Baby | 15.9 | $12.3_{(-3.6)}$ | $12.0_{(-3.9)}$ | $11.8_{(-4.1)}$ | $10.8_{(-5.1)}$ | $11.3_{(-4.6)}$ | $12.0_{(-3.9)}$ | $11.5_{(-4.4)}$ |
| Magazines | 10.5 | $12.3_{(+1.8)}$ | $7.5_{(-3.0)}$ | $7.8_{(-2.7)}$ | $8.0_{(-2.5)}$ | $7.8_{(-2.7)}$ | $9.5_{(-1.0)}$ | $8.8_{(-1.7)}$ |
| Software | 14.7 | $13.5_{(-1.2)}$ | $13.8_{(-0.9)}$ | $12.8_{(-1.9)}$ | $10.3_{(-4.4)}$ | $12.8_{(-1.9)}$ | $11.8_{(-2.9)}$ | $11.0_{(-3.7)}$ |
| Sports | 17.3 | $16.0_{(-1.3)}$ | $14.5_{(-2.8)}$ | $14.3_{(-3.0)}$ | $12.3_{(-5.0)}$ | $13.3_{(-4.0)}$ | $13.5_{(-3.8)}$ | $12.8_{(-4.5)}$ |
| IMDB | 17.3 | $13.8_{(-3.5)}$ | $17.5_{(+0.2)}$ | $14.5_{(-2.8)}$ | $13.0_{(-4.3)}$ | $13.5_{(-3.8)}$ | $13.3_{(-4.0)}$ | $13.3_{(-4.0)}$ |
| MR | 26.9 | $25.5_{(-1.4)}$ | $25.3_{(-1.6)}$ | $23.3_{(-3.6)}$ | $21.5_{(-5.4)}$ | $22.0_{(-4.9)}$ | $23.5_{(-3.4)}$ | $22.8_{(-4.1)}$ |
| AVG | 18.0 | $15.5_{(-2.5)}$ | $15.3_{(-2.7)}$ | $13.9_{(-4.1)}$ | $\mathbf{12.5}_{(-5.5)}$ | $12.7_{(-5.3)}$ | $14.3_{(-3.7)}$ | $13.1_{(-4.9)}$ |

面）。每个任务中的所有数据集都划分为训练集、校验集和测试集，数据量分别是 1400、200、和 400 个样本。评价指标利用错误率，即错误分类样本所占的比例。

我们选择几个相关且具有代表性的模型作为基线模型。

- MT-CNN：该模型由 Collobert 等人[27] 提出，带有卷积层，其中查找表部分共享，而其他层是任务特定层。
- FS-MTL：完全共享的多任务学习框架。不同的任务完全共享神经网络层（LSTM）。
- SP-MTL：具有对抗性学习的共享-私有多任务学习框架[143]（即第七章的模

型)。不同的任务不仅具有共享信息的公共层，而且具有自己的私有层。

**多任务学习的结果：** 表8.1为我们的模型在多个数据集上的文本分类错误率，表中括号里的数字表示平均错误率。实验结果表明，我们提出的模型远远优于所有单任务基线模型，这里我们展示平均误差的原因如下：1）更容易显示多任务学习模型相比于单任务模型的性能增益。2）BiLSTM 和堆叠 LSTM 也是 SG-MTL 的必要基线模型，因为 SG-MTL 中共享和私有层的组合类似于双层 LSTM。

表8.1展示了在单任务、多任务和迁移学习设置下 16 个不同任务的文本分类错误率。通常，我们可以看到几乎所有任务都受益于多任务学习，这大大提高了性能。具体来说，CG-MTL 实现了最佳性能，超过 SP-MTL 1.4%，这表明显式通信更容易共享信息。尽管 SG-MTL 的提升不如 CG-MTL 那么大，但 SG-MTL 的训练效率很高。此外，SG-MTL 和 SP-MTL 之间的比较说明了选择性共享模式的有效性。此外，我们可以结合 SP-MTL 中引入的对抗性训练机制来进一步改进我们的模型。

**可迁移性上的评测：** 我们展示所提的方法在迁移学习方面的潜力，因为我们期望所提的模型架构所学到的共享知识可以用于新任务。特别地，SG-MTL 模型中的虚拟结点可以在多任务学习之后将共享信息压缩到公共空间中，这允许我们将该知识转移到新任务。为了测试 SG-MTL 学习的共享知识的可转移性，我们按照监督预训练范式设计了一个实验。具体来说，我们采用了 16 倍的 “leave-one-task-out” 范式：即我们轮流选择 15 个任务在多任务学习的框架下训练模型，然后将学习的共享层迁移到第二个网络，用于测试剩余的任务 $k$。传输层的参数保持冻结，新网络的其余参数随机初始化。表8.1在 “迁移’ 列中展示了相应的结果，其中每行中的任务被视为目标任务。我们观察到，我们的模型在单一任务设置上的错误率（13.1 对 18.0）方面获得了 4.9% 的平均提高，超过 SP-MTL 平均 1.2%（13.1 对 14.3）。这种改进表明，SP-MTL 没有用选择性机制读取共享信息，并忽略任务之间的关系，而我们所提的 SG-MTL 模型使用选择性机制的检索方法能有效地从共享空间中查找相关信息。

**表 8.2　不同模型在 Chunking、NER 和与 POS 的 F1 值**

| 模型 | CoNLL2000 | CoNLL2003 | PTB |
|---|---|---|---|
| LSTM + CRF | 94.46 | 90.10 | 97.55 |
| MT-CNN | 94.32 | 89.59 | 97.29 |
| FS-MTL | 95.41 | 90.94 | 97.55 |
| SP-MTL* | 95.27 | 90.90 | 97.49 |
| CG-MTL | 95.49 | 91.25 | 97.61 |
| SG-MTL | **95.61** | **91.47** | **97.69** |

### 8.3.3. 序列标记

在本节中，我们将介绍模型在序列标记任务上的结果。我们按照[106] 的设置进行了实验。我们在实验中使用以下基准数据集：Penn Treebank（PTB）POS，CoNLL 2000 Chunking，CoNLL 2003 NER。

**结果与分析：**表8.2展示了模型在序列标记任务上的性能。其中，LSTM+ CRF 由 Huang 等人[109] 提出；MT-CNN 由 Collobert 等人[27] 提出；FS-MTL 由 Yang 等人[106] 提出。CG-MTL 和 SG-MTL 明显优于三个强大的多任务学习基线模型。具体而言，在 CoNLL2003 数据集上，与最佳对比模型 FS-MTL 相比，SG-MTL 的 F1 得分获得了 0.53% 的提升，表明我们的模型能够通过建模不同任务之间的关系来利用共享信息。与最佳基线模型 FS-MTL 相比，我们的模型在 CoNLL2000 和 PTB 数据集上的 F1 值也略有提高。

## 8.4　讨论与定性分析

为了获得所提的模型如何在任务之间传递消息的运行机制和详细解释，我们设计了一系列针对以下方面的实验：

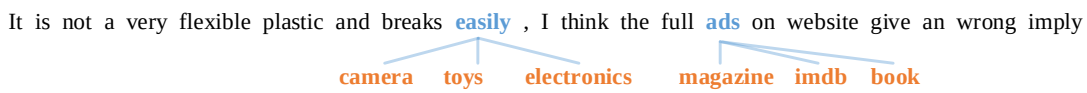
1. CG-MTL 可以学习不同任务之间的关系吗?

2. SG-MTL 中的共享层是否可以学习可解释的结构？这些共享模式是否与语

言结构类似？是否可以迁移到其他任务中？

### (1)　显性关系学习

为了回答第一个问题，我们在等式8.3中可视化 CG-MTL 的权重 $\alpha_t$。由于每个任务都可以从 CG-MTL 中任意任务接收消息，因此 $\alpha_t$ 会在时刻 $t$ 直接指示其他任务与当前任务的相关性。如图 8.3 所示，我们分析了我们的模型 CG-MTL 从随机抽样句子中学到的关系。我们发现任务之间的关系不能用静态得分建模。相反，它取决于具体的样本和背景。图8.3-(a) 是一个来自 KITCHEN 任务的例句，其中，“easy” 和 “ads” 在不同外部任务的环境下表达不一样的情感。例如，在 CAMERA 和 TOYS 任务中，“break easy” 通常用于表达负面情感，而 “ads” 这个词经常出现在 MAGAZINE 任务中表达负面情感。图8.3-(b) 在 “quality” 和 “name-brand” 上也有类似的情况。




(a) Kitchen Task


(b) Video Task

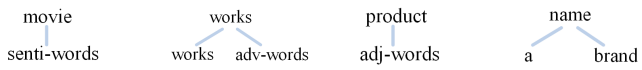
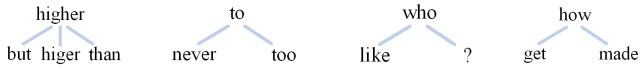
**图 8.3　相邻 “任务” 的查找**

### (2)　可解释的结构学习

为了回答第二个问题，我们在 SG-MTL 模型中可视化等式8.8中权重 $\beta$。由于不同的任务可以从 SG-MTL 中的共享层读取信息，因此可视化 $\beta$ 允许我们分析共享哪种类型的句子结构。具体来说，每个词语 $wt^{(k)}$ 都可以接收共享消息：$w_1^{(s)}, \cdots, w_T^{(s)}$，且消息量由 $\beta$ 值控制。为了说明 SG-MTL 中共享层学习的可解释结构，我们从不同任务中随机抽取几个示例并可视化它们的共享结构。三个随机抽样的情况如图8.4 所述。

我们在 SG-MTL 中查看 $\beta$ 值，并观察到以下情况：

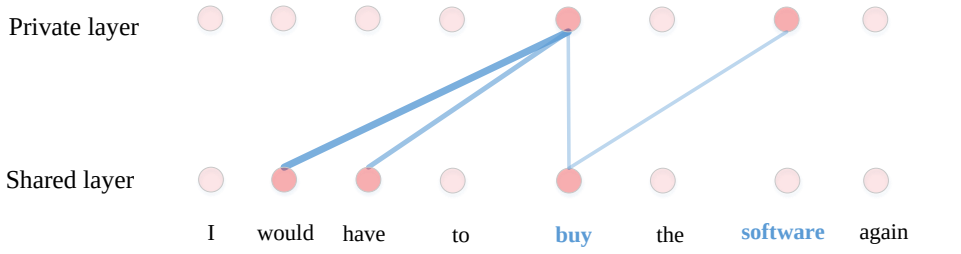
**表 8.3　由共享层学到的多个可解释的子结构**

| | 可解释的子结构 | 解释描述 |
|---|---|---|
| **短距离** |  | Interpretable phrases to be shared across tasks |
| **长距离** |  | Interpretable sentence patterns, they usually determine the sentence meanings. |

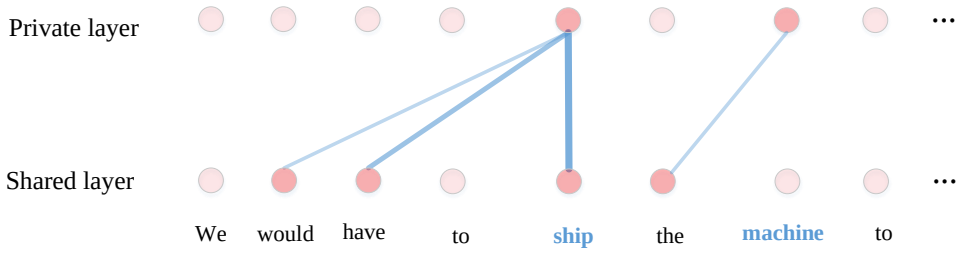
- 所提出的模型不仅可以利用不同任务的共享信息，还可以告诉我们共享了哪些类型的特征。如表 8.3所示，模型可以捕获不同词语之间的短期和长期依赖关系。例如，“movie” 这个词很容易与情感词联系起来，比如“boring,exciting”,而“products”更有可能和“stable,great,fantastic’搭配。
- 如图8.4-(a) 和 (b) 所示，共享层已从任务 KITCHEN 训练集中学习了一个信息量的句型“would have to...”。此模式对于另一个任务 SOFTWARE 的情感预测很有用。

## 8.5　本章小结

我们将多任务之间关系学习这个任务转化成图神经网络消息传递的问题。我们所提的方法显式地建模了不同任务之间的关系，并在多任务和迁移学习设置中表现了更好的结果，而且这些共享模式具备可解释性。从本文的实验中，我们认为学习可解释的共享结构是一个很有前景的方向，这对知识迁移也非常有用。




(a) Shared patterns in “Software” task




(b) Shared patterns in “Kitchen” task

**图 8.4　共享层在不同任务下捕获的学习模式**

# 第 9 章　总结与展望

## 9.1　本文的总结

基于深度神经网络的分布式语义表示学习已成为目前自然语言处理中必不可缺少的组成成分。

- **词语表示：**词语学习日益成熟，一词多义的研究也有若干有效的解决方法。本文介绍了一种通用架构，即基于神经张量网络的 Skip-Gram 模型，来学习和上下文相关的词表示向量。该模型在两个主流任务上取得了超过基线模型的性能。此外，本文通过可视化的方式展示了基于神经张量网络学习到的词语表示在分离多义词不同义项的优越性。

- **句子表示：**本文关注了三个研究问题：如何建模包含习语的句子？如何解决合成的多样性与函数单一性导致的表示能力不足的问题？如何动态学习句子的结构而不是预先指定？围绕这个三个问题，我们探索了不同类型的结构偏置：基于树结构的自适应语义合成网络和动态语义合成网络、基于图的语境化网络。我们先后在文本分类、语义匹配、序列标注等任务对以上提出的模型进行验证，实验结果显示了这些偏置的引入可以使得网络表示能力更强。

- **句对：**针对句对的建模任务，我们提出了一种端到端的深层体系结构来捕获句子对的强交互信息。我们在两个大规模数据集上对模型进行了验证，实验结果不但显示了我们模型相对基线模型的优越性，我们还发现了该模型中某些神经元的行为是可被解释的，这增强了我们模型的解释性。

- **可迁移性：**我们通过多任务学习架构去实现表示的可迁移性学习。具体地，我们提出了三种基于循环神经为网络的架构，用于建模多任务学习的文本序列。它们之间的差异是在几个任务之间共享信息的机制。实验结果表明，我们的模型可以通过探索共同特征来提高相关任务的性能。

- **可分离性：** 我们提出了一种基于对抗训练的多任务学习框架，它可以捕获不同任务的共享-私有分离的特征。我们将模型应用于 16 种不同的文本分类任务，证明了我们方法的有效性。我们还进行了广泛的定性分析，并解释该方法性能提升的机理。

- **可理解性：** 我们将多任务之间关系学习这个任务转化成图神经网络消息传递的问题。我们所提的方法显式地建模了不同任务之间的关系，并在多任务和迁移学习设置中表现了更好的结果，而且这些共享模式具备可解释性。

此外，关于语义表示的可共享性、可迁移性、可理解性的研究已经成为目前非常火热的研究课题，本文首先以循环神经网络为原型，探究了三种适合序列建模的可迁移性表示学习框架；然后为了将不同任务之间共享和私有的特征实现分离，将对抗学习的思想引入到多任务学习中；最后通过利用图结构的神经网络，提出了一种可理解的可迁移知识学习框架。

## 9.2　对未来工作展望

- 对于词表示的学习，上下文语境化的词向量会成为今后的研究重点，尤其是如何在大规模的无监督语料上训练出上下文相关的词表示，并且为下游任务提供高效的初始化服务。

- 对于句子表示学习，无监督的句子表示学习将成为今后热点，如何设计无监督学习的损失函数是个难点，一个潜在的突破点在于和语言模型的训练相结合。

- 对于句对的建模，目前句对的建模主要是在问答的任务背景下被研究，今后，会有越来越多的任务被开发，并且所处理句子的长度可能会大大增加，如何设计出能够处理长文档的模型会成为新的挑战。

- 深度网络给我们提供最大的机会就是实现知识的可存储、可理解、可分离、可迁移。而至今为止，我们只是在初步阶段，简单实现不同任务之间的知

识迁移，和简单的理解和分离。如何使得我们可以更加深入地理解知识，并且进行定制化地迁移将成为今后自然语言处理研究重要目标。

- 现有的迁移学习中，迁移的往往都是已经学习好的参数，但是另一个思路是如何先通过对网络的可理解性分析，然后直接迁移经验（网络运行的内部机制）。

# 参 考 文 献

[1] BENGIO Y, DUCHARME R, VINCENT P, et al. A neural probabilistic language model[J]. Journal of machine learning research, 2003, 3(Feb): 1137–1155.

[2] LUONG M T, SOCHER R, MANNING C. Better word representations with recursive neural networks for morphology[J]. CoNLL-2013, 2013, 104.

[3] MIKOLOV T, CHEN K, CORRADO G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.

[4] REISINGER J, MOONEY R J. Multi-prototype vector-space models of word meaning[C]// Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. 2010: 109–117.

[5] HUANG E, SOCHER R, MANNING C, et al. Improving word representations via global context and multiple word prototypes[C]//Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2012.

[6] TIAN F, DAI H, BIAN J, et al. A probabilistic model for learning multi-prototype word embeddings[C]//Proceedings of COLING. 2014: 151–160.

[7] NEELAKANTAN A, SHANKAR J, PASSOS A, et al. Efficient non-parametric estimation of multiple embeddings per word in vector space[C]//Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014.

[8] LIU Y, LIU Z, CHUA T S, et al. Topical word embeddings[C]//AAAI. 2015.

[9] KALCHBRENNER N, GREFENSTETTE E, BLUNSOM P. A convolutional neural network for modelling sentences[C]//Proceedings of ACL. 2014.

[10] TAI K S, SOCHER R, MANNING C D. Improved semantic representations from tree-structured long short-term memory networks[C/OL]//Proceedings of the 53rd ACL. 2015: 1556–1566. http://www.aclweb.org/anthology/P15-1150.

[11] ZHU X D, SOBHANI P, GUO H. Long short-term memory over recursive structures.[C]// ICML. 2015: 1604–1612.

[12] VELICKOVIC P, CUCURULL G, CASANOVA A, et al. Graph attention networks[J]. arXiv preprint arXiv:1710.10903, 2017.

[13] GEDIGIAN M, BRYANT J, NARAYANAN S, et al. Catching metaphors[C]//Proceedings of the Third Workshop on Scalable Natural Language Understanding. [S.l.]: Association for Computational Linguistics, 2006: 41–48.

[14] SHUTOVA E, SUN L, KORHONEN A. Metaphor identification using verb and noun clustering[C]//Proceedings of the 23rd International Conference on Computational Linguistics.

[S.l.]: Association for Computational Linguistics, 2010: 1002–1010.

[15] KATZ G, GIESBRECHT E. Automatic identification of non-compositional multi-word expressions using latent semantic analysis[C]//Proceedings of the Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties. [S.l.]: Association for Computational Linguistics, 2006: 12–19.

[16] LI L, SPORLEDER C. Classifier combination for contextual idiom detection without labelled data[C]//Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1. [S.l.]: Association for Computational Linguistics, 2009: 315–323.

[17] FAZLY A, COOK P, STEVENSON S. Unsupervised type and token identification of idiomatic expressions[J]. Computational Linguistics, 2009, 35(1): 61–103.

[18] PENG J, FELDMAN A, VYLOMOVA E. Classifying idiomatic and literal expressions using topic models and intensity of emotions.[C]//EMNLP. 2014: 2019–2027.

[19] SALTON G D, ROSS R J, KELLEHER J D. Idiom token classification using sentential distributed semantics[C]//Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. 2016: 194–204.

[20] KARTSAKLIS D, SADRZADEH M, PULMAN S. A unified sentence space for categorical distributional-compositional semantics: Theory and experiments[C]//In Proceedings of COLING: Posters. [S.l.]: Citeseer, 2012.

[21] MURAOKA M, SHIMAOKA S, YAMAMOTO K, et al. Finding the best model among representative compositional models[C]//Proceedings of the 28th Pacific Asia Conference on Language, Information, and Computation (PACLIC 2014). 2014: 65–74.

[22] HERMANN K M. Distributed representations for compositional semantics[J]. arXiv preprint arXiv:1411.3146, 2014.

[23] HASHIMOTO K, TSURUOKA Y. Adaptive joint learning of compositional and non-compositional phrase embeddings[J]. arXiv preprint arXiv:1603.06067, 2016.

[24] SOCHER R, HUANG E H, PENNIN J, et al. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection[C]//Advances in Neural Information Processing Systems. 2011.

[25] WAN S, LAN Y, GUO J, et al. A deep architecture for semantic matching with multiple positional sentence representations[C]//AAAI. 2016.

[26] ROCKTÄSCHEL T, GREFENSTETTE E, HERMANN K M, et al. Reasoning about entailment with neural attention[J]. arXiv preprint arXiv:1509.06664, 2015.

[27] COLLOBERT R, WESTON J. A unified architecture for natural language processing: Deep neural networks with multitask learning[C]//Proceedings of the 25th international conference

on Machine learning. [S.l.]: ACM, 2008: 160–167.

[28] LIU X, GAO J, HE X, et al. Representation learning using multi-task deep neural networks for semantic classification and information retrieval[C]//NAACL. 2015.

[29] LIU P, QIU X, HUANG X. Recurrent neural network for text classification with multi-task learning[C]//Proceedings of International Joint Conference on Artificial Intelligence. 2016.

[30] LIU P, QIU X, HUANG X. Deep multi-task learning with shared memory[C]//Proceedings of EMNLP. 2016.

[31] BAKKER B, HESKES T. Task clustering and gating for bayesian multitask learning[J]. JMLR, 2003, 4(May): 83–99.

[32] KIM S, XING E P. Tree-guided group lasso for multi-task regression with structured sparsity [Z]. 2010.

[33] CHEN X, KIM S, LIN Q, et al. Graph-structured multi-task regression and an efficient optimization method for general fused lasso[J]. stat, 2010, 1050: 20.

[34] MIKOLOV T, SUTSKEVER I, CHEN K, et al. Distributed representations of words and phrases and their compositionality[C]//Advances in neural information processing systems. 2013: 3111–3119.

[35] SUTSKEVER I, VINYALS O, LE Q V. Sequence to sequence learning with neural networks [C]//Advances in NIPS. 2014: 3104–3112.

[36] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate[J]. ArXiv e-prints, 2014.

[37] VAN DER LELY H. Domain-specific cognitive systems: Insight from grammatical specific language impairment.[J]. Trends in Cognitive Sciences, 2005, 9(2): 53–59.

[38] BAERMAN M. The oxford handbook of inflection[M]. [S.l.]: Oxford Handbooks in Linguistic, 2015.

[39] FUTRELL R L J. Memory and locality in natural language[D]. [S.l.]: Massachusetts Institute of Technology, 2017.

[40] LECUN Y, BENGIO Y. Convolutional networks for images, speech, and time series[J]. The handbook of brain theory and neural networks, 1995, 3361(10).

[41] YIH W T, HE X, MEEK C. Semantic parsing for single-relation question answering[C/OL]// Proceedings of ACL. 2014. http://www.aclweb.org/anthology/P14-2105.

[42] SHEN Y, HE X, GAO J, et al. Learning semantic representations using convolutional neural networks for web search[C]//Proceedings of the 23rd International Conference on World Wide Web. [S.l.]: ACM, 2014: 373–374.

[43] LAI S, XU L, LIU K, et al. Recurrent convolutional neural networks for text classification [C]//Twenty-ninth AAAI conference on artificial intelligence. 2015.

[44] CHEN Y, XU L, LIU K, et al. Event extraction via dynamic multi-pooling convolutional neural networks[C]//Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers): volume 1. 2015: 167–176.

[45] CHO K, VAN MERRIENBOER B, GULCEHRE C, et al. Learning phrase representations using rnn encoder-decoder for statistical machine translation[C]//Proceedings of EMNLP. 2014.

[46] HOCHREITER S, SCHMIDHUBER J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735–1780.

[47] ELMAN J L. Finding structure in time[J]. Cognitive science, 1990, 14(2): 179–211.

[48] JOZEFOWICZ R, ZAREMBA W, SUTSKEVER I. An empirical exploration of recurrent network architectures[C]//Proceedings of The 32nd International Conference on Machine Learning. 2015.

[49] GRAVES A. Generating sequences with recurrent neural networks[J]. arXiv preprint arXiv:1308.0850, 2013.

[50] KIPF T N, WELLING M. Semi-supervised classification with graph convolutional networks [J]. arXiv preprint arXiv:1609.02907, 2016.

[51] GILMER J, SCHOENHOLZ S S, RILEY P F, et al. Neural message passing for quantum chemistry[C]//ICML. 2017: 1263–1272.

[52] MARCHEGGIANI D, TITOV I. Encoding sentences with graph convolutional networks for semantic role labeling[J]. arXiv preprint arXiv:1703.04826, 2017.

[53] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[C]//Advances in Neural Information Processing Systems. 2017: 6000–6010.

[54] YANG Z, YANG D, DYER C, et al. Hierarchical attention networks for document classification[C]//Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2016: 1480–1489.

[55] LIN Z, FENG M, SANTOS C N D, et al. A structured self-attentive sentence embedding[J]. arXiv preprint arXiv:1703.03130, 2017.

[56] CHENG J, DONG L, LAPATA M. Long short-term memory-networks for machine reading [C]//Proceedings of the 2016 Conference on EMNLP. 2016: 551–561.

[57] SOCHER R, PERELYGIN A, WU J Y, et al. Recursive deep models for semantic compositionality over a sentiment treebank[C]//Proceedings of EMNLP. 2013.

[58] PANG B, LEE L. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts[C]//Proceedings of ACL. 2004.

[59] WILLIAMS L, BANNISTER C, ARRIBAS-AYLLON M, et al. The role of idioms in senti-

ment analysis[J]. Expert Systems with Applications, 2015, 42(21): 7375–7385.

[60] LI X, ROTH D. Learning question classifiers[C]//Proceedings of the 19th International Conference on Computational Linguistics. 2002: 556–562.

[61] MARELLI M, BENTIVOGLI L, BARONI M, et al. Semeval-2014 task 1: Evaluation of compositional distributional[J]. SemEval-2014, 2014.

[62] TURIAN J, RATINOV L, BENGIO Y. Word representations: a simple and general method for semi-supervised learning[C]//Proceedings of ACL. 2010.

[63] COLLOBERT R, WESTON J, BOTTOU L, et al. Natural language processing (almost) from scratch[J]. The JMLR, 2011, 12: 2493–2537.

[64] MNIH A, HINTON G. Three new graphical models for statistical language modelling[C]// Proceedings of ICML. 2007.

[65] SOCHER R, HUVAL B, MANNING C D, et al. Semantic compositionality through recursive matrix-vector spaces[C]//Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. [S.l.]: Association for Computational Linguistics, 2012: 1201–1211.

[66] BLEI D M, NG A Y, JORDAN M I. Latent dirichlet allocation[J/OL]. J. Mach. Learn. Res., 2003, 3: 993–1022. http://dx.doi.org/10.1162/jmlr.2003.3.4-5.993.

[67] PETERS M E, NEUMANN M, IYYER M, et al. Deep contextualized word representations [C]//Proc. of NAACL. 2018.

[68] DEVLIN J, CHANG M, LEE K, et al. BERT: pre-training of deep bidirectional transformers for language understanding[J/OL]. CoRR, 2018, abs/1810.04805. http://arxiv.org/abs/1810.04805.

[69] SOCHER R, CHEN D, MANNING C D, et al. Reasoning with neural tensor networks for knowledge base completion[C]//Advances in NIPS. 2013: 926–934.

[70] BORDES A, USUNIER N, GARCIA-DURAN A, et al. Translating embeddings for modeling multi-relational data[C]//NIPS. 2013.

[71] GRIFFITHS T L, STEYVERS M. Finding scientific topics[J]. Proceedings of the National Academy of Sciences, 2004, 101(suppl 1): 5228–5235.

[72] FAN R E, CHANG K W, HSIEH C J, et al. Liblinear: A library for large linear classification [J]. The Journal of Machine Learning Research, 2008, 9: 1871–1874.

[73] CHUNG J, GULCEHRE C, CHO K, et al. Empirical evaluation of gated recurrent neural networks on sequence modeling[J]. arXiv preprint arXiv:1412.3555, 2014.

[74] BOBROW S A, BELL S M. On catching on to idiomatic expressions[J]. Memory & Cognition, 1973, 1(3): 343–346.

[75] KATZ J J. Semantic interpretation of idioms and sentences containing them[M]. [S.l.]: Re-

search Laboratory of Electronics, Massachusetts Institute of Technology, 1963.

[76] FRASER B. Idioms within a transformational grammar[J]. Foundations of language, 1970: 22–42.

[77] NUNBERG G, SAG I A, WASOW T. Idioms[J]. Language, 1994: 491–538.

[78] BALAHUR A, STEINBERGER R, KABADJOV M, et al. Sentiment analysis in the news [J]. arXiv preprint arXiv:1309.6202, 2013.

[79] VILLAVICENCIO A, BOND F, KORHONEN A, et al. Introduction to the special issue on multiword expressions: Having a crack at a hard nut[Z]. [S.l.]: Academic Press, 2005.

[80] SALTON G, ROSS R, KELLEHER J. An empirical study of the impact of idioms on phrase based statistical machine translation of english to brazilian-portuguese[Z]. [S.l.]: Dublin Institute of Technology, 2014.

[81] GLUCKSBERG S. Idiom meanings and allusional content[J]. Idioms: Processing, structure, and interpretation, 1993: 3–26.

[82] KIM Y, JERNITE Y, SONTAG D, et al. Character-aware neural language models[C]// Thirtieth AAAI Conference on Artificial Intelligence. 2016.

[83] LAMPLE G, BALLESTEROS M, SUBRAMANIAN S, et al. Neural architectures for named entity recognition[C]//Proceedings of NAACL-HLT. 2016: 260–270.

[84] CHUNG J, CHO K, BENGIO Y. A character-level decoder without explicit segmentation for neural machine translation[J]. arXiv preprint arXiv:1603.06147, 2016.

[85] FLAVELL L, FLAVELL R. Dictionary of idioms and their origins[M]. [S.l.]: Kyle Cathie Limited, 2006.

[86] DUCHI J, HAZAN E, SINGER Y. Adaptive subgradient methods for online learning and stochastic optimization[J]. The JMLR, 2011, 12: 2121–2159.

[87] PENNINGTON J, SOCHER R, MANNING C. Glove: Global vectors for word representation[C]//Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014: 1532–1543.

[88] KLEIN D, MANNING C D. Accurate unlexicalized parsing[C]//Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1. 2003: 423–430.

[89] BOWMAN S R, GAUTHIER J, RASTOGI A, et al. A fast unified model for parsing and sentence understanding[J]. arXiv preprint arXiv:1603.06021, 2016.

[90] SOCHER R, PENNINGTON J, HUANG E H, et al. Semi-supervised recursive autoencoders for predicting sentiment distributions[C]//Proceedings of the conference on empirical methods in natural language processing. [S.l.]: Association for Computational Linguistics, 2011: 151–161.

[91] KIM Y. Convolutional neural networks for sentence classification[J]. arXiv preprint

arXiv:1408.5882, 2014.

[92] PYLKKÄNEN L, MCELREE B. The syntax-semantics interface: On-line composition of sentence meaning[J]. Handbook of psycholinguistics, 2006, 2: 537–577.

[93] DE BRABANDERE B, JIA X, TUYTELAARS T, et al. Dynamic filter networks[C]//NIPS. 2016.

[94] BERTINETTO L, HENRIQUES J F, VALMADRE J, et al. Learning feed-forward one-shot learners[C]//Advances in NIPS. 2016: 523–531.

[95] HA D, DAI A, LE Q V. Hypernetworks[J]. arXiv preprint arXiv:1609.09106, 2016.

[96] BOWMAN S R, ANGELI G, POTTS C, et al. A large annotated corpus for learning natural language inference[C/OL]//Proceedings of the 2015 Conference on EMNLP. 2015. http://aclweb.org/anthology/D15-1075.

[97] BOWMAN S R, POTTS C, MANNING C D. Recursive neural networks can learn logical semantics[J]. arXiv preprint arXiv:1406.1827, 2014.

[98] BIBER D, CONRAD S, REPPEN R. Corpus linguistics: Investigating language structure and use[M]. [S.l.]: Cambridge University Press, 1998.

[99] PETERS M, NEUMANN M, IYYER M, et al. Deep contextualized word representations [C]//Proceedings of the 2018 Conference of NAACL: volume 1. 2018: 2227–2237.

[100] KITAEV N, KLEIN D. Constituency parsing with a self-attentive encoder[J]. arXiv preprint arXiv:1805.01052, 2018.

[101] WANG X, GIRSHICK R, GUPTA A, et al. Non-local neural networks[J]. arXiv preprint arXiv:1711.07971, 2017.

[102] BATTAGLIA P W, HAMRICK A, RAPOSO, et al. Relational inductive biases, deep learning, and graph networks[J]. arXiv preprint arXiv:1806.01261, 2018.

[103] PANG B, LEE L. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales[C]//Proceedings of the ACL. 2005: 115–124.

[104] MANNING C, SURDEANU M, BAUER J, et al. The stanford corenlp natural language processing toolkit[C]//Proceedings of 52nd ACL. 2014.

[105] ZEILER M D. Adadelta: an adaptive learning rate method[J]. arXiv preprint arXiv:1212.5701, 2012.

[106] YANG Z, SALAKHUTDINOV R, COHEN W. Multi-task cross-lingual sequence tagging from scratch[J]. arXiv preprint arXiv:1603.06270, 2016.

[107] YASUNAGA M, KASAI J, RADEV D. Robust multilingual part-of-speech tagging via adversarial training[C]//Proceedings of the NAACL. 2018.

[108] MA X, HOVY E. End-to-end sequence labeling via bi-directional lstm-cnns-crf[C]// Proceedings of the 54th Annual Meeting of ACL: volume 1. 2016: 1064–1074.

[109] HUANG Z, XU W, YU K. Bidirectional lstm-crf models for sequence tagging[J]. arXiv preprint arXiv:1508.01991, 2015.

[110] HU B, LU Z, LI H, et al. Convolutional neural network architectures for matching natural language sentences[C]//Advances in Neural Information Processing Systems. 2014.

[111] QIU X, HUANG X. Convolutional neural tensor network architecture for community-based question answering[C/OL]//Proceedings of International Joint Conference on Artificial Intelligence. 2015. http://ijcai.org/papers15/Papers/IJCAI15-188.pdf.

[112] YIN W, SCHÜTZE H. Convolutional neural network for paraphrase identification[C]// Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2015: 901–911.

[113] HE H, GIMPEL K, LIN J. Multi-perspective sentence similarity modeling with convolutional neural networks[C]//Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. 2015: 1576–1586.

[114] YIN W, SCHÜTZE H, XIANG B, et al. Abcnn: Attention-based convolutional neural network for modeling sentence pairs[J]. arXiv preprint arXiv:1512.05193, 2015.

[115] HERMANN K M, KOCISKY T, GREFENSTETTE E, et al. Teaching machines to read and comprehend[C]//Advances in Neural Information Processing Systems. 2015: 1684–1692.

[116] SCHUSTER M, PALIWAL K K. Bidirectional recurrent neural networks[J]. Signal Processing, IEEE Transactions on, 1997, 45(11): 2673–2681.

[117] GRAVES A, SCHMIDHUBER J. Framewise phoneme classification with bidirectional lstm and other neural network architectures[J]. Neural Networks, 2005, 18(5): 602–610.

[118] GRAVES A, FERNÁNDEZ S, SCHMIDHUBER J. Multi-dimensional recurrent neural networks[M]//Artificial Neural Networks–ICANN 2007. [S.l.]: Springer, 2007: 549–558.

[119] KALCHBRENNER N, DANIHELKA I, GRAVES A. Grid long short-term memory[J]. arXiv preprint arXiv:1507.01526, 2015.

[120] BYEON W, BREUEL T M, RAUE F, et al. Scene labeling with lstm recurrent neural networks [C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 3547–3555.

[121] SRIVASTAVA R K, GREFF K, SCHMIDHUBER J. Highway networks[J]. arXiv preprint arXiv:1505.00387, 2015.

[122] HE K, ZHANG X, REN S, et al. Deep residual learning for image recognition[C]// Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770–778.

[123] LE Q, MIKOLOV T. Distributed representations of sentences and documents[C]// International Conference on Machine Learning. 2014: 1188–1196.

[124] LIU P, QIU X, CHEN X, et al. Multi-timescale long short-term memory neural network for modelling sentences and documents[C/OL]//Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2015. http://www.aclweb.org/anthology/D/D15/D15-1280.pdf.

[125] CARUANA R. Multitask learning[J]. Machine learning, 1997, 28(1): 41–75.

[126] BENGIO Y, LAMBLIN P, POPOVICI D, et al. Greedy layer-wise training of deep networks [J]. Advances in neural information processing systems, 2007, 19: 153.

[127] LIU P, QIU X, CHEN J, et al. Deep fusion LSTMs for text semantic matching[C]// Proceedings of ACL. 2016.

[128] GOODFELLOW I, POUGET-ABADIE J, MIRZA M, et al. Generative adversarial nets[C]// Advances in Neural Information Processing Systems. 2014: 2672–2680.

[129] TAIGMAN Y, POLYAK A, WOLF L. Unsupervised cross-domain image generation[J]. arXiv preprint arXiv:1611.02200, 2016.

[130] AJAKAN H, GERMAIN P, LAROCHELLE H, et al. Domain-adversarial neural networks [J]. arXiv preprint arXiv:1412.4446, 2014.

[131] BEN-DAVID S, BLITZER J, CRAMMER K, et al. A theory of learning from different domains[J]. Machine learning, 2010, 79(1-2): 151–175.

[132] BEN-DAVID S, BLITZER J, CRAMMER K, et al. Analysis of representations for domain adaptation[J]. Advances in neural information processing systems, 2007, 19: 137.

[133] BOUSMALIS K, TRIGEORGIS G, SILBERMAN N, et al. Domain separation networks [C]//Advances in Neural Information Processing Systems. 2016: 343–351.

[134] JIA Y, SALZMANN M, DARRELL T. Factorized latent spaces with structured sparsity[C]// Advances in Neural Information Processing Systems. 2010: 982–990.

[135] SALZMANN M, EK C H, URTASUN R, et al. Factorized orthogonal latent spaces.[C]// AISTATS. 2010: 701–708.

[136] GANIN Y, LEMPITSKY V. Unsupervised domain adaptation by backpropagation[C]// Proceedings of the 32nd International Conference on Machine Learning (ICML-15). 2015: 1180–1189.

[137] BLITZER J, DREDZE M, PEREIRA F, et al. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification[C]//ACL: volume 7. 2007: 440–447.

[138] MAAS A L, DALY R E, PHAM P T, et al. Learning word vectors for sentiment analysis[C]// Proceedings of the ACL. 2011: 142–150.

[139] SØGAARD A, GOLDBERG Y. Deep multi-task learning with low level tasks supervised at lower layers[C]//Proceedings of ACL. 2016.

[140] PENG H, THOMSON S, SMITH N A. Deep multitask learning for semantic dependency parsing[C]//Proceedings of the 55th ACL: volume 1. 2017: 2037–2048.

[141] GUO J, CHE W, WANG H, et al. Exploiting multi-typed treebanks for parsing with deep multi-task learning[J]. arXiv preprint arXiv:1606.01161, 2016.

[142] LIU P, QIU X, HUANG X. Recurrent neural network for text classification with multi-task learning[C]//Proceedings of IJCAI. 2016.

[143] LIU P, QIU X, HUANG X. Adversarial multi-task learning for text classification[C]// Proceedings of the 55th Annual Meeting of ACL: volume 1. 2017: 1–10.

[144] LUONG M T, LE Q V, SUTSKEVER I, et al. Multi-task sequence to sequence learning[J]. arXiv preprint arXiv:1511.06114, 2015.

[145] FIRAT O, CHO K, BENGIO Y. Multi-way, multilingual neural machine translation with a shared attention mechanism[C]//Proceedings of NAACL-HLT. 2016: 866–875.

[146] HASHIMOTO K, TSURUOKA Y, SOCHER R, et al. A joint many-task model: Growing a neural network for multiple nlp tasks[C]//Proceedings of the 2017 Conference on EMNLP. 2017: 1923–1933.

[147] LI Y, TIAN X. Graph-based multi-task learning[C]//Communication Technology (ICCT), 2015 IEEE 16th International Conference on. [S.l.]: IEEE, 2015: 730–733.

[148] BERENDSEN H J, VAN DER SPOEL D, VAN DRUNEN R. Gromacs: a message-passing parallel molecular dynamics implementation[J]. Computer Physics Communications, 1995, 91(1-3): 43–56.

[149] SERLET B, BOYNTON L, TEVANIAN A. Method for providing automatic and dynamic translation into operation system message passing using proxy objects[Z]. [S.l.]: Google Patents, 1996.

[150] NETZER R H, MILLER B P. Optimal tracing and replay for debugging message-passing parallel programs[J]. The Journal of Supercomputing, 1995, 8(4): 371–388.

# 致　谢

过去五年的博士生活对我来说是珍贵的经历，也是难忘的回忆。我要感谢我的两位导师。刚进实验室那会，我记得自己还是一个 latex 都用不熟的自然语言处理爱好者。幸运地，我遇到了邱老师和黄老师。邱老师为以后致力在学术上发展的我树立了典范，他给予我的远不止学术上种种能力的提升；生活中，他会告诉我如何克服遇到的困境，如何选择人生的分支。无论是哪方面，我已不能向他要求更多。

黄老师对我充满着鼓励，宽容。她总是在投稿前帮我耐心的修改论文，鼓励并推荐我去参评荣誉奖项。特别地，我深深感动于博士论文完成过程中得到的帮助，上百页的论文，她几乎逐字阅读并写下了详细的修改建议。

我要感谢吴立德老师，他在讨论班上所授的机器学习课程是让我得以快速成长的宝贵资源。感谢张奇老师和周雅倩老师，感谢你们在学术上、生活上的帮助。感谢陶晓鹏老师，他帮助我补充很多传统自然语言处理的背景知识，以及美国的文化等，我的很多工作灵感是源于这些分享的。感谢 Jackie Chi Kit Cheung 老师，他总是对我提出的想法给出正面的鼓励并在写作上传教了很多宝贵的经验。感谢 Yoshua Bengio 老师，给我提供了去 Mila 学习的宝贵机会。感谢 Jian Tang 老师，虽然合作短暂但受益匪浅。感谢 Neubig Graham 老师，他很善于鼓励学生，并支持我去做一些自己想做的事情。感谢在博后申请期间您给予的鼓励和帮助，期待与您进一步合作。感谢刘铁江老师、厉家鼎老师和付雁老师在我申请各个奖项给予的帮助。

读博前期，有幸去百度 NLP 部门学习，认识了很多领域专家，这无疑对我知识的储备、阅历的提升起到了巨大的作用。感谢百度 NLP 部门的前辈们：刘占一、Dianhai Yu，吴先超，牛罡，忻州，董大祥等。

在我博士期间，我遇到了很多优秀的合作者，感谢他们的帮助。吴世宇：我第一篇学术论文的合作者；陈新驰：我们相互学习，相互督促，感谢一路有你；陈济凡：在句对建模工作上我们有很多讨论，他还是我初学王者荣耀的“导师”。

郭琦鹏：相互启发，进行了很多深刻的讨论。付杰：亦师亦友，感谢一路帮我培养许多优秀的科研习惯。董悦：帮我快速融入了 CompLing，并认识了很多新成员。张赛峥：值得信赖的合作伙伴，那一个月的创业时光对我弥足珍贵。

博士期间，我也有幸和一些聪明的年轻小朋友合作，感谢你们的信任，愿意和我一起做些事情：陈俊坤、常帅晨、王少敬、李林阳、王丹青、钟鸣、林泽辉、黄璐瑶、郑逸宁、陈怡然、胡雅茹、孙天祥。

感谢我的室友王文山，五年相处，兄弟般情谊；感谢谭伟敏，从你身上学习到很多优秀习惯（从青铜到王者，我们一起奋战的日子是难得且欢乐的时光）；同样，致敬所有一起读博的 14 级的那些兄弟们：田璐超、戴大伟，吴昊、刘汶健、张季博宁、朱海平等人。

感谢 Prof. Hippo，16 年被评为最受喜爱讲者的荣誉虽小，但对我产生的影响巨大，感谢你的帮助；

特别要提的是，很感谢黄萱菁老师和傅金兰同学在论文修改过程中给予的巨大帮助，让这篇文稿的质量不断得到提高。

感谢百忙之中，出席答辩委员会的何晓丰教授、王晓玲教授、赵海教授、路红教授。

感谢百度公司、IBM 公司、微软公司和腾讯科技的奖学金支持。

感谢复旦大学张江校区图书馆，安静舒适的学习环境，让我得以培养很多优秀的学习习惯；感谢宿舍园区海艺影院，便捷的娱乐方式调剂了我紧张的学术生活；感谢华佗路爱米思油商铺，蔡伦路炒面商摊，让我对张江的美食回味无穷。最后，感谢我的家人们，你们的爱，无以回报。

# 在读期间发表的学术论文与取得的研究成果

## 已发表论文

[1] **Pengfei Liu**, Xipeng Qiu, Xuanjing Huang, Learning Context-Sensitive Word Embeddings with Neural Tensor Skip-Gram Model, Proceedings of International Joint Conference on Artificial Intelligence (IJCAI 2015) (**CCF A**)

[2] **Pengfei Liu**, Xipeng Qiu, Xuanjing Huang, Recurrent Neural Network for Text Classification with Multi-Task Learning, Proceedings of International Joint Conference on Artificial Intelligence (IJCAI2016) (**CCF A**)

[3] **Pengfei Liu**, Xipeng Qiu, and Xuanjing Huang, Adversarial Multi-task Learning for Text Classification, The annual meeting of the Association for Computational Linguistics (ACL 2017) (**CCF A**)

[4] **Pengfei Liu**, Xipeng Qiu, Jifan Chen, and Xuanjing Huang, Deep Fusion LSTMs for Text Semantic Matching, The annual meeting of the Association for Computational Linguistics (ACL2016) (**CCF A**)

[5] **Pengfei Liu**, Xipeng Qiu, Xuanjing Huang, Adaptive Semantic Compositionality for Sentence Modelling , Proceedings of International Joint Conference on Artificial Intelligence, (IJCAI 2017) (**CCF A**)

[6] **Pengfei Liu**, Xipeng Qiu, Xuanjing Huang Dynamic Compositional Neural Networks over Tree Structure, Proceedings of International Joint Conference on Artificial Intelligence (IJCAI 2017) (**CCF A**)

[7] **Pengfei Liu**, Shuaicheng Chang, Jian Tang, Jackie Chi Kit Cheung, Contextualized non-local neural networks for sequence learning, Proceedings of the 33th AAAI Conference on Artificial Intelligence (AAAI 2019) (**CCF A**)

[8] **Pengfei Liu**, Jie Fu*, Yue Dong*, Xipeng Qiu, Jackie Chi Kit Cheung, Learning multi-task communication with message passing, Proceedings of the 33th AAAI Conference on Artificial Intelligence (AAAI 2019) (**CCF A**)

[9] Ming Zhong*, **Pengfei Liu***, Danqing Wang, Xipeng Qiu, and Xuanjing Huang , Searching for Effective Neural Extractive Summarization: What Works and What's Next, The annual meeting of the Association for Computational Linguistics (ACL 2019) (**CCF A**)

[10] **Pengfei Liu**, Xipeng Qiu and Xuanjing Huang, Modelling Interaction of Sentence Pair with Coupled-LSTMs , Conference on Empirical Methods in Natural Language Processing (EMNLP2017) (**CCF B**)

[11] **Pengfei Liu**, kaiyu Qian, Xipeng Qiu and Xuanjing Huang, Idiom-Aware Compositional Distributed Semantics, Conference on Empirical Methods in Natural Language Processing (EMNLP 2017) (**CCF B**)

[12] **Pengfei Liu**, Xipeng Qiu, Xinchi Chen, Shiyu Wu, Xuanjing Huang, Multi-Timescale Long Short-Term Memory Neural Network for Modelling Sentences and Documents, Conference on Empirical Methods in Natural Language Processing (EMNLP 2015) (**CCF B**)

[13] **Pengfei Liu**, Xipeng Qiu and Xuanjing Huang, Deep Multi-Task Learning with Shared Memory, Conference on Empirical Methods in Natural Language Processing (EMNLP2016) (**CCF B**)

[14] Qipeng Guo, Xipeng Qiu, **Pengfei Liu**, Yunfan Shao, Xiangyang Xue, Zheng Zhang, Star Transformer, The North American Chapter of the Association for Computational Linguistics (NAACL 2019) (**CCF C**)

[15] Dayiheng Liu, Jie Fu, **Pengfei Liu** and Jiancheng Lv, TIGS: An Inference Algorithm for Text Infilling with Gradient Search, The annual meeting of the Association for Computational Lin-

guistics (ACL 2019) (**CCF A**)

[16] Junkun Chen, Xipeng Qiu, **Pengfei Liu**, Xuanjing Huang, Meta Multi-Task Learning for Sequence Modeling, Proceedings of the 32th AAAI Conference on Artificial Intelligence (AAAI 2018) (**CCF A**)

[17] Jifan Chen, Qi Zhang, **Pengfei Liu**, Xipeng Qiu and Xuanjing Huang, Implicit Discourse Relation Detection via a Deep Architecture with Gated Relevance Network, The annual meeting of the Association for Computational Linguistics (ACL2016) (**CCF A**)

[18] Jifan Chen, Qi Zhang, **Pengfei Liu**, Xuanjing Huang, Discourse Relations Detection via a Mixed Generative-Discriminative Framework, Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI2016) (**CCF A**)

[19] Xinchi Chen, Xipeng Qiu, Chenxi Zhu, **Pengfei Liu**, Xuanjing Huang, Long Short-Term Memory Neural Networks for Chinese Word Segmentation, Conference on Empirical Methods in Natural Language Processing (EMNLP 2015) (**CCF B**)

[20] Ming Zhong*, Danqing Wang*, **Pengfei Liu***, Xipeng Qiu, and Xuanjing Huang , A Closer Look at Data Bias in Neural Extractive Summarization Models, Conference on Empirical Methods in Natural Language Processing (EMNLP2019) Workshop on New Frontiers in Summarization