清 华 大 学

综 合 论 文 训 练

题目：基于深度学习的翻译规则向量表示学习

系　　别：自动化

专　　业：自动化

姓　　名：朱懿

指导教师：刘洋副研究员

辅导教师：刘连臣副研究员

2015 年 7 月 12 日

关于学位论文使用授权的说明

本人完全了解清华大学有关保留、使用学位论文的规定，即：学校有权保留学位论文的复印件，允许该论文被查阅和借阅；学校可以公布该论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存该论文。

(涉密的学位论文在解密后应遵守此规定)

签　名：_____ 导师签名：_____ 日　期：_____

中文摘要

在机器翻译领域，双语短语对是短语翻译模型的核心，它的质量直接决定了译文的好坏。现今绝大多数衡量双语短语对的方法都是基于统计，以词频为基础的概率估计，但其深受数据稀疏问题的困扰，并不能准确估计其概率。本文从语义角度出发，在非监督式递归自动编码器的基础上，重现了文献 [16] 中的双语短语递归自动编码器，通过计算短语向量表示之间的距离来表示短语对的语义相似度。本文实现了短语表剪枝实验以验证其有效性。最后根据层次短语的特点，本文还提出了几种将双语短语递归编码器拓展到层次短语上的方法作为未来的研究工作。

关键词：短语翻译模型；深度学习；递归编码器；短语向量表示

ABSTRACT

In Machine Translation, bilingual phrase pairs are the core of the Phrase-Based Machine Translation, of which the quality is determinant for the translation text. Nowadays most methods used to evaluate the bilingual phrase pairs are the probability estimation based on the statistics of concurrent words in parallel corpus, which suffer the problem of data spareness that leads to inaccurate inference. This paper converted the standard Unsupervised Recursive Autoencoder into the Bilingual-Constrained Recursive Autoencoder(BRAE)[16] from the perspective of semantics by calculating the distance between the bilingual phrase embeddings to represent their similarity. This paper also realized the phrase table pruning to verify the feasibility of BRAE. At last the paper proposed several ideas to expand BRAE to the hierarchical phrase case as future work.

Keywords: Phrase-Based Machine Translation; Deep Learning; Recursive Autoencoder; Phrase Embeddings

目　录

第 1 章　引言 1
1.1　研究背景 1
1.2　文章结构 2
第 2 章　统计机器翻译 3
2.1　噪声信道模型与 IBM 模型 3
2.2　短语模型 4
2.3　层次短语模型 6
第 3 章　递归自动编码器 9
3.1　非监督式递归自动编码器 10
3.1.1　最小重建错误二叉树 12
3.1.2　参数训练 13
3.1.3　相关实验 17
3.2　双语短语递归自动编码器 21
3.2.1　双语语义错误 21
3.2.2　参数训练 23
3.2.3　相关实验 27
第 4 章　双语层次短语递归自动编码器 31
4.1　获取非终结符的向量表示 32
4.2　直接获取层次短语的向量表示 33
4.3　相关实验 33
第 5 章　结论 34
插图索引 35
表格索引 36
公式索引 37

参考文献 39
致　谢 41
声　明 42
附录 A　外文资料的调研阅读报告 43
A.1 References 46

第 1 章　引言

1.1　研究背景

作为自然语言处理（Natural Language Processing, NLP）的一个重要分支，机器翻译（Machine Translation, MT）一直被视为人类实现人工智能（Artificial Intelligence, AI）的最重要的环节之一。它不仅可以自动实现地球上任何两个人之间无障碍的交流，还包括密码破译，重现古老语言样等重要应用。

机器翻译最初被看作一个简单的任务，即将一种语言的单词逐个替换为另一种语言的单词，但随后它就被证明是一个非常困难的问题。因为语言的翻译不仅是单词的直接替换，更包括词组识别、词义消歧、句法分析甚至语义理解等诸多复杂的过程。根据 Bernard Vauquois 的语言翻译金字塔模型（图1.1），经典机器翻译方法大致可分成以下三类：

（1）直接翻译（Direct Translation）；

（2）转移翻译（Transfer-Based Translation）；

（3）跨语言翻译（Interlingua-Based Translation）。

顾名思义，直接翻译就是将源语言端（*s* 端）的单词或者词组直接翻译成目标语言端（*t* 端），在经过某些规则或语法的限制后使译文更加自然流畅；转移翻译则是首先分析 *s* 端语句，建立一个新的表示模型，之后通过某些规则将此模型转移为 *t* 端的模型，最后根据此模型反向生成 *t* 端的译文，最典型的就是建立 *s* 端句法分析树（Parse Tree），将此句法分析树转移为 *t* 端的句法分析树，再根据 *t* 端的句法分析树生成译文；跨语言翻译则是在分析 *s* 端语句的基础上，用更高一层的，与语言无关的表示模型来表示此语句，之后根据此表示直接生成 *t* 端的译文。

机器翻译虽然经过了长足的发展，但是其翻译效果距离达到令人满意还相差较远。现在大部分翻译系统还处在直接翻译的阶段，少部分系统探索了转移翻译模型或囊括了转移翻译的元素。本文则力求在比较强大的直接翻译系统：短语翻译系统和层次短语（Hierarchical Phrase-based Translation）翻译系统上增加更高层次的语义信息，并期望其能帮助层次短语翻译系统提高翻译质量。

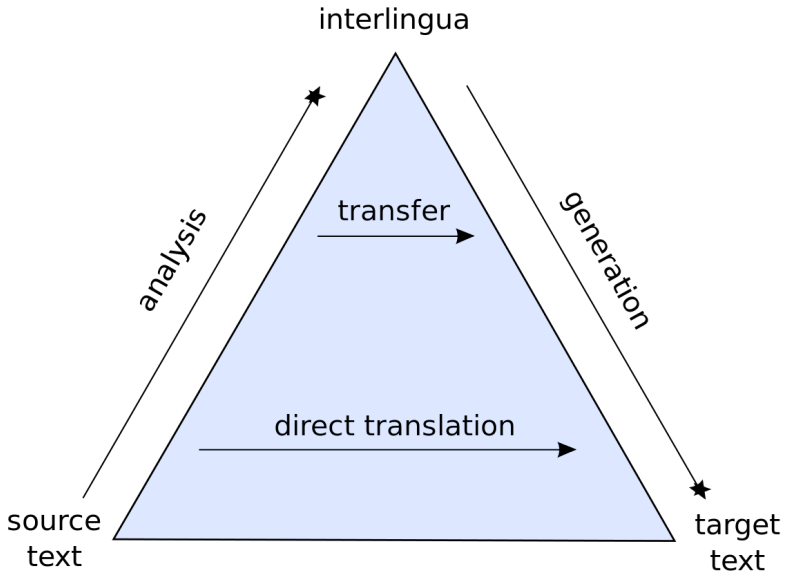

图 1.1　Bernard Vauquois 翻译金字塔模型

1.2　文章结构

本文结构大致分配如下：第 2 章介绍统计机器翻译的发展，并指出一些相关翻译系统的缺陷与语义信息对于翻译系统的必要性；第 3 章详细描述了用递归自动编码器表示语义信息的方法，从最简单的非监督式递归自动编码器到双语短语递归自动编码器，其结构生成，参数训练等均是本篇文章的重点；第 4 章在第 3 章的基础上，提出了几种简单直观的方法，将递归编码器应用于层次短语系统当中。第 5 章为结论与未来的工作。

第 2 章　统计机器翻译

在经典机器翻译的框架下，机器翻译系统的实现方法经历了 50 至 70 年代基于规则（Rule-Based）的翻译系统到 90 年代后由统计方法所主导的基于语料（Corpus-Based）的翻译系统，并分别经历了以单词，短语，层次短语为翻译核心的翻译系统的演变。

Brwon et al. (1990)[1] 利用大量平行语料[①]作为训练数据，提出了用统计学方法来研究机器翻译的思想，开启了机器翻译的统计时代。

2.1　噪声信道模型与 IBM 模型

IBM 公司的研究人员在文献 [1] 基础上提出了噪声信道模型（Noisy-Channel Model)[2]，奠定了 IBM 模型的基础。

根据噪声信道模型，给定一个源语言端 s 端的语句, 其目标语言端 t 端的译文概率可以用式 (2-1) 的贝叶斯公式表示。

$$p(t \mid s) = \frac{p(t)p(s \mid t)}{p} \tag{2-1}$$

故此时我们将翻译问题转化为了一个概率问题，最佳译文为在给定 s 端语句的情况下找到翻译概率最高的 t 端译文, 用式 (2-2) 表示。

$$\begin{aligned} t^* = \arg\max_{t \in T} p(t \mid s) &= \arg\max \frac{p(t)p(s \mid t)}{p(s)} \\ &= \arg\max p(t)p(s \mid t) \end{aligned} \tag{2-2}$$

其中 $p(t)$ 为 t 端的语言模型概率，$p(s \mid t)$ 为翻译概率。

IBM 模型将单词视为翻译单元，将机器翻译的翻译质量提升到了一个新的高度，是最成功的基于单词（Word-Based）的翻译系统。但由于在计算翻译概率

① 平行语料即以句子为单元的双语翻译集合，即一个 s 端的句子对应一个 t 端的句子，通常出现在官方语言为多语言的国家会议记录中，如加拿大的 Hansards，欧洲的 Europarl 等。

时 IBM 模型需要引入词语间的对齐（Alignment），而 IBM 模型设计的词对齐是一对多的，即一个 s 端语言的单词可以对应多个 t 端单词，反之则不行，这就造成 IBM 模型无法实现 s 端与 t 端单词间多对多对齐这样一个语言间的天然特性，因此从建模上来讲 IBM 模型存在固有的缺陷。虽然作为翻译系统 IBM 模型如今已经很少得到应用，但是其在词对齐过程中生成的最优对齐则成为了之后很多翻译系统的必要步骤。

2.2 短语模型

短语（Phrase-Based）[3] [4] 翻译系统将短语视为翻译单元，通过对句子中的单词进行多对多对齐（s 端与 t 端双向对齐）来抽取短语对，并根据短语对表以及从语料中提取出的特征训练特征的权重。翻译 s 端语句的过程同训练过程相似，只是此时模型的参数已经确定，对 s 端中语句的各短语进行翻译，调序，最终生成得分最高的译文或最高的 k 个译文。

短语模型利用对数线性（log-linear）模型 [5]，对于一个语句的 s 端，t 端通常存在许多可能的翻译译文 t。我们定义生成一个译文的若干个短语构成的集合为该译文的派生集合 D（Derivation）。因此，对于每一个派生集合 D 而言，我们定义以下函数，用来计算该派生集合所代表的译文分数。

$$\begin{aligned} P(D) &= \prod_i \phi_i^{\lambda_i}(D) \\ &= \exp(\lambda_i \phi_i(D)) \end{aligned} \tag{2-3}$$

其中 ϕ_i 是短语系统的第 i 个特征，λ_i 是该特征的权重。我们利用上式定义某种损失函数，利用大量平行语料优化此函数，学习 $\{\lambda_i\}_{i=1}^n$。该模型的特征既可以是某个短语的翻译概率，也可以是由某些启发式方法所指定的规则。因此对数线性模型的自由度非常大，扩展性很好。

在短语模型当中，抽取出的短语对的质量高低直接影响到模型的训练与解码效果。而在众多特征之间，有四个重要的特征衡量了抽取出的短语对的相似度或互异度，用以表示短语对之间翻译质量的高低，它们分别是式 (2-4)，式 (2-5)[3]。

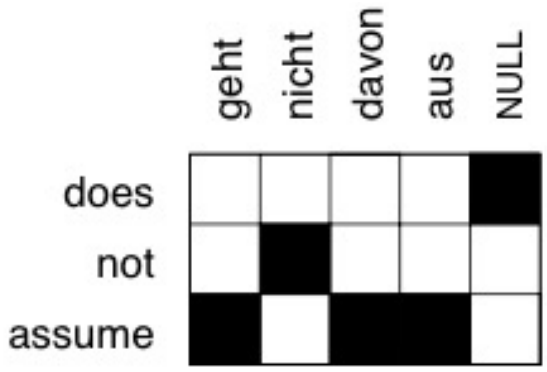

图 2.1　短语对齐

$$\phi(t \mid s) = \frac{count(s,t)}{\sum_{t_i} count(s,t_i)} \tag{2-4}$$

$$lex(t \mid s) = \max_{\alpha \in \mathcal{A}} \prod_{i=1}^{|t|} \frac{1}{|\{j \mid (i,j) \in \alpha\}|} \sum_{\forall |\{j|(i,j)\in\alpha\}|} w(t_i \mid s_i) \tag{2-5}$$

其中式 (2-4) 为短语翻译概率，式 (2-5) 为词汇化权重。上述两式为给定 s 端，计算 t 端的公式，如果反方向应用上述两式（即 t 端到 s 端），则一共就有 4 特征值。

我们以式 (2-4)，式 (2-5) 为代表分别解释上述四个特征。首先短语翻译概率较好理解，即（s，t）这个短语对在语料中同时出现的次数相比较 s 这个短语在语料中出现次数的频率，可以看到式 (2-4) 的分母中遍历了与 s 同时出现的短语 t_i。

我们试用图2.1为例计算式 (2-5)。由于对齐 α 已经给定，因此对齐集合 $\mathcal{A}$ 只有一个元素，否则应取 $\mathcal{A}$ 中词汇化权重最大的对齐。设德语为 s 端，英语为 t 端，我们可以用式 (2-6) 计算。

$$
\begin{aligned}
lex(t \mid s,a) = & w(does \mid Null)\times \\
& w(not \mid nicht)\times \\
& \frac{1}{3}\left(w(assume \mid geht) + w(assume \mid davon) + w(assume \mid aus)\right)
\end{aligned}
\tag{2-6}
$$

相比较于IBM模型，短语模型有着更好的翻译效果，但Koehn et al. (2003)[3]发现长度超过3的短语对BLEU值（Papineni et al. 2002)[6]①的提升帮助非常之小，由于短语模型搜索候选译文的空间是随着短语数目呈指数增长的，故长短语在调序时效果往往不好[7]，可见短语系统忽略了语句一个很重要的特征：无限迭代性。我们可以借用语句分析（Parsing）中的例子来分析。

例2.1：

$$NP \rightarrow NP \quad VP$$

一个名词短语可以从语法上分解为另一个名词短语与动词短语，而箭头右端的名词短语又可以继续重复该过程，这样的递归特性可以一直持续下去，直到实际的单词为止。同理，对于短语来说，长短语仍然可以依照类似的过程构成许多短语。

2.3 层次短语模型

基于2.2节中所论述的短语模型的缺点，Chiang（2007)[7]提出了层次短语模型（Phrase-Based)，将短语模型的扁平化翻译结构增加一级层次结构，一个翻译语句由基础短语表和层次短语表由下至上分两级生成。

例2.2：下面这个例子展示了利用短语系统所生成的译文，其中中文为*s*端，英文为*t*端。

[澳洲][是]$_1$[与北韩]$_2$[有][邦交][的少数国家之一][。]

① BLUE（Bilingual Evaluation Understudy）是一种自动评价翻译质量的方法，其结果与人类的评价结果拟合度非常高，是现在最主流的自动评测方法。BLEU值越高说明翻译效果越好。

[Australia][has][dipl. rels.][with North Korea]$_2$[is]$_1$[one of the few countries][.]
Australia is one of the few countries that have diplomatic relations with North Korea.

下标表示了两个语言语句中相互对应的部分。可以看到，译文可以准确将”与北韩有邦交”翻译为“has diplomatic relations with North Korea”，但却无法将“的少数国家之一”在英文中的翻译“one of the few countries”调整为正确的顺序，这正是由于此短语与其他短语的组合过长，在训练集中很难找到非常相近的短语样本而造成的。

而如果在普通短语的基础上引入层次短语，那么我们就可以抽取以下常见层次短语对:

$$[\text{与 } X_1 \text{ 有 } X_2, \text{have } X_2 \text{ with } X_1]$$
$$[X_1 \text{ 的 } X_2, \text{the } X_2 \text{ that } X_1]$$
$$[X_1 \text{ 之一}, \text{one of } X_1]$$

其中 X^* 为子短语的占字符，它可以扩展为普通短语。利用这样的结构，我们可以在训练集中抽取的短语与层次短语表中由下至上从短语到层次短语派生出源语言端的句子，同时利用短语表中的翻译信息与调序模型生成相应译文。

例 2.3：我们用层次短语系统来生成之前例子中的译文。

[澳洲][是][[[与 [北韩]$_1$ 有 [邦交]$_2$] 的 [少数国家]$_3$] 之一][。]
[Australia][is][one of[the[few countries]$_3$that[have[dipl. rels.]$_2$with [N. Korea]$_1$]]][.]

如果应用上述所列的 3 个层次短语，就可以较好地解决在短语系统中所产生的上述问题。

层次短语模型沿用了短语模型中式 (2-3) 的对数线性模型与式 (2-4)，式 (2-5) 所表示的 4 个重要特征。这 4 个特征刻画了短语对的质量，而它们估计的精准度将很大程度上影响候选译文的产生，继而影响翻译质量。由于式 (2-4) 利用统计语料中词频的方法来估计概率，因此在层次短语模型的训练中仍然会出现与短语模型相同的数据稀疏问题。假设在训练中抽取了一个短语对，它的 s 端，t 端都各只出现了一次，则短语概率特征 $\phi(t \mid s)$，$\phi(t \mid s)$ 均为 1，这就会造成该短语的翻译概率不准确。在实际应用的解码过程中，一旦出现该短语，它就只能被翻译成相应的目标语言短语，而不能是其他实际情况中可能出现的目标语言短语。事实上我们可以把语言言集合看作一个维度很高的空间，句子，短语数据稀疏的现象也就非常常见。

在之前的例子中我们可以看到式 (2-5) 反映了一个短语对翻译质量的高低，是由短语中各对齐单词对的翻译概率所决定的。而层次短语中包含不确定的子短语 X^*，现有的层次短语模型会跳过 X^*，只计算确定单词的词汇化权重。如果计算[与 X_1 有 X_2, have X_2 with X_1] 的词汇化权重, 我们会得到

$$lex(t \mid s, a) = w(have \mid 有) \times w(with \mid 与)$$

显然由于没有计算 X^*，这种方法便无法准确估计层次短语的词汇化权重。

综上，上述 4 个特征并不能很精确地估计抽取短语对的质量，因此我们可以设想提出一种新的特征，来帮系统助抽取更加可靠的短语对。

我们考虑脱离以词频统计为基础的特征设置，从更高一层语义的角度出发，考量短语对之间语义的相似度，并以此作为评判短语对质量的标准之一，可能会帮助我们更好地评价短语对。

第 3 章　递归自动编码器

递归自动编码器（Recursive Autoencoder）是一种具有层次化递归特性的深度学习神经网络结构（图3.1），可以算作循环神经网络（RNN）的一种。其最大特点是每一个输入单元与输出的维度相同，因此当输入层有多个输入单元时，递归自动编码器可以被看作一种降维的方法，非常适合处理输入与输出的表示隶属于同一类型或过程需要递归循环的任务（如句法分析，短语向量表示等）。就本文而言，我们主要论述的就是递归编码器在短语向量表示中的应用，相关讨论也会根据该内容展开。

Socher et al. (2011)[8] 将递归自动编码器 (Recursive Autoencoder) 应用于表示短语甚至句子的向量中，并以此来获取情感信息。Socher et al. (2013)[9] 又使短语向量可以学习语句结构知识。Li et al. (2013)[10] 将递归自动编码器应用于 ITG 翻译模型[11] 的短语调序中。

我们可以将底层的输入 x 看作一个有序单词序列 $\{x_i\}_1^n$，它可以代表一个短语（句子），而 x_i 则为每个单词的词向量，作为编码器二叉树的叶节点。每两个子节点构成一个三元组 $(p \to c_1 c_2)$，其中 p 为父节点，c_1，c_2 为子节点。子节点既

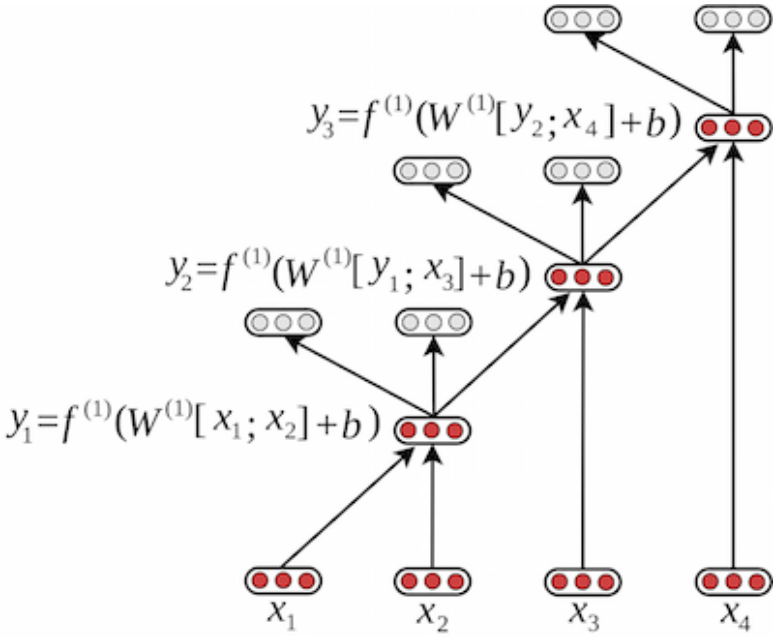

图 3.1　递归自动编码器结构

可以是叶节点的词向量，也可以是非叶节点的短语向量。一棵二叉树就可以用这样的方法由下至上生成。在图3.1中，我们能找到下列三元组 $\Big((y_1 \to x_3x_4), (y_2 \to x_2y_1), (y_3 \to x_1y_2)\Big)$。为了能使得递归循环的过程持续进行，我们必须要求 y_i 与 x_i 拥有相同的维数，因此，由图3.1中可见各个节点向量由输入到输出维数都应为 3，即递归自动编码器的维数为 3。

对于一个维数为 n 的递归自动编码器，其二叉树输入叶节点的词向量需要事先初始化，即在 n 维连续空间内为每个单词找到一个向量。我们可以事先根据 Benjio et al. (2003)[12] 训练好的词向量或 Word2Vec 生成的词向量[13] 作为递归自动编码器的输入向量，也可以根据 $x \sim \mathcal{N}(0, \sigma^2)$ 的高斯分布随机为每个单词生成一个输入向量。通过上述两种方法，我们可以得到一个输入词向量的矩阵 $L \in \mathbb{R}^{n \times |V|}$，其中 $|V|$ 为训练集中所有的词汇数目，每一列即为某个单词的 n 维词向量。我们为每个单词建立一个索引，则取出一个单词的词向量 x_i 就可以用中类似查询的操作得到，如式3-1所示，其中 b_k 为一个 n 维向量，其第 k 个分量为 1，其余分量均为 0。

$$x_i = Lb_k \in \mathbb{R}^n \tag{3-1}$$

虽然通过事先训练好的词向量本身就可以捕获某些语义和语句信息，但是由于在训练中矩阵 L 随着迭代需要进行更新，因此这两种初始化方法的性能近似，优劣并不明显，需要根据不同情景通过实验得到结果。在本文的后续实验中，为了统一起见，除非特殊说明，我们都采用随机生成词向量的方法对输入向量进行初始化。

根据应用任务的不同，递归编码器二叉树的生成方法也层出不穷，本文在短语向量表示的任务中分别介绍最基本的非监督式递归自动编码器（URAE）与双语短语递归自动编码器（BRAE）两种主要的结构。

3.1 非监督式递归自动编码器

在文献 [8] 中，Socher 首先提出了一种最基本的非监督式的训练方式，通过贪心算法生成编码器二叉树，通过最小化重建错误训练得到其中的参数。

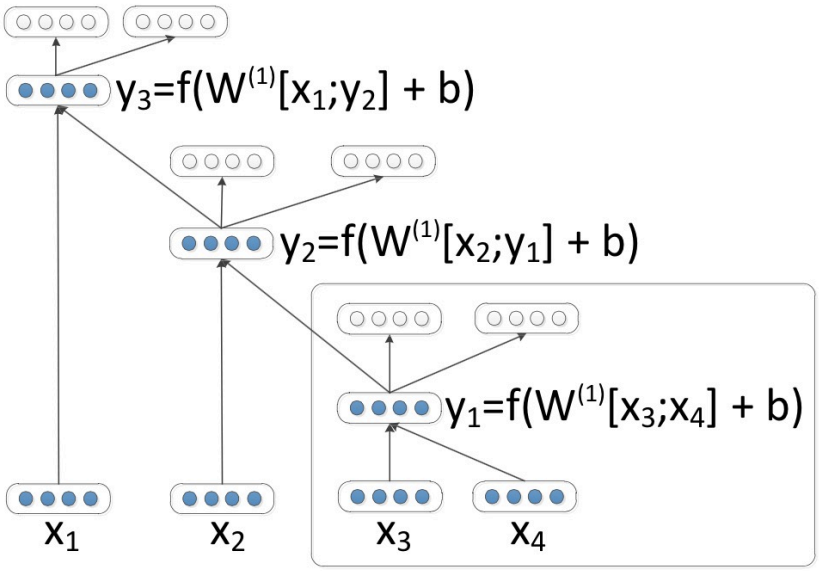

图 3.2　非监督式递归自动编码器

以图3.2为例，我们将方框中的结构看作递归编码器的一个递归单元，定义下述公式可以用来计算单元中由子节点编码为父节点的过程，其余节点的编码过程与该单元中的过程完全一致。

$$\begin{aligned} p' &= g(W^{(1)}[c_1; c_2] + b^{(1)}) \\ p &= \frac{p'}{\| p' \|} \end{aligned} \tag{3-2}$$

其中 p 为最终的父节点。

如果令 $(c_1, c_2) = (x_3, x_4)$，则 $p = y_1$。假设短语向量的维数为 n，则 $W^{(1)} \in \mathbb{R}^{n \times 2n}$ 为一个参数矩阵，$b^{(1)} \in \mathbb{R}^n$ 为一个偏置向量。最后我们利用 $tanh$ 函数作为神经元的激发函数，便得到父节点的向量表示 $y_1 \in \mathbb{R}^n$。

为了衡量父节点是否较好地涵盖了两个子节点向量所代表的语义信息，我们可以采用一个形式类似的反变换从该父节点中重建（Reconstruction）出两个子节点。

$$[c_1'; c_2'] = \frac{g(W^{(2)}p + b^{(2)})}{\| g(W^{(2)}p + b^{(2)}) \|} \tag{3-3}$$

类似地，$W^{(2)} \in \mathbb{R}^{2n \times n}$ 为一个参数矩阵，$b^{(2)} \in \mathbb{R}^{2n}$ 为一个偏置向量，最终的结果是由父节点重建出的子节点 c_1', c_2' 拼成的列向量。图3.1中空心的向量即为重建后的子节点。

对于每次递归编码，我们用原子节点 (c_1, c_2) 与重建子节点 (c_1', c_2') 的欧式距离来表示重建错误（Reconstruction Error）。

$$E_{rec}([c_1; c_2]) = \frac{1}{2} \parallel [c_1; c_2] - [c_1'; c_2'] \parallel^2 \tag{3-4}$$

因此当经过若干次递归编码，当生成整个二叉树时，我们就得到了整个短语 x 的向量表示（二叉树根节点的向量表示）和所有非叶节点的重建错误之和。

3.1.1 最小重建错误二叉树

在图3.2树结构已经确定的情况下，重建错误的计算十分容易。而对于非监督式的递归编码器，其目标函数就是通过优化各个参数，在没有确定树结构的情况下，生成一个使各节点平均重建错误最小的二叉树。对于不同的参数集合，我们每次生成的二叉树的结构也有可能不同。

在给定一个短语输入 x 的情况下，我们定义 $A(x)$ 为由 x 可以生成的所有二叉树的集合，$y \in A(x)$ 是某一棵特定结构的二叉树，$T(y)$ 为生成二叉树过程中所有节点编码后而产生的所有三元组 $(p1 \rightarrow c1, c2)$ 的集合。利用式 (3-4) 我们可以得到非监督式自动递归编码器的最优二叉树：

$$y_\theta^*(x) = \arg \min_{y \in A(x)} \sum_{s \in T(y)} E_{rec}([c_1; c_2]_s) \tag{3-5}$$

其中 θ 为参数集合。

假设该短语由 m 个有序单词构成 $x = (x_1, ..., x_m)$，根据式 (3-5) 所表示的目标函数，我们可以通过算法1的贪心算法来生成最小重建错误二叉树。

还以图3.2为例，但是此时我们需要假设树的结构是不确定的，即图3.2中叶节点以上的结构都是不存在的。我们在给定 4 个叶节的词向量以及一组参数之后，需要进行 $4 - 1 = 3$ 次递归编码生成根节点的向量表示。

（1）建立子节点集合 $\{x_1, x_2, x_3, x_4\}$，$m = 4, RAE_\theta = 0$；

（2）分别计算 $E_{rec}([x_1; x_2]), E_{rec}([x_2; x_3]), E_{rec}([x_3; x_4])$，发现 $E_{rec}([x_3; x_4])$

Algorithm 1 Greedy Unsupervised RAE

1: 子节点个数 $m \leftarrow$ 叶节点个数，$RAE_\theta \leftarrow 0$

2: **while** $m > 1$ **do**

3: 更新当前所有相邻子节点之间的重建错误 $\{E_{rec}\}_1^{m-1}$

4: 获取构成最小重建误 $\arg\min\{E_{rec}\}_1^{m-1}$ 的两个子节点，编码并生成父节点 $[p^*, c_1^*; c_2^*]$

5: $RAE_\theta = RAE_\theta + \min\{E_{rec}\}_1^{m-1}$

6: 将 c_1^*, c_2^* 从子节点集合中删除，将 p^* 加入子节点集合

7: $m \leftarrow m - 1$

最小，编码生成三元组 $[y_1, x_3; x_4]$，$m = 3, RAE_\theta = E_{rec}([x_3; x_4])$。更新子节点集合 $\{x_1, x_2, y_1\}$；

（3）计算各相邻子节点间的错误 $E_{rec}([x_1; x_2]), E_{rec}([x_2; y_1])$，发现 $E_{rec}[x_2; y_1]$ 最小，于是编码 $[y_2, x_2; y_1]$，$m = 2, RAE_\theta = E_{rec}([x_3; x_4]) + E_{rec}([x_2; y_1])$。更新子节点结合；

（4）由于子节点集合只有两个元素 x_1, y_2，所以我们直接编码 $[y_3, x_1; y_2]$，$m = 1, RAE_\theta = E_{rec}([x_3; x_4]) + E_{rec}([x_2; y_1]) + E_{rec}([x_1; y_2])$。

这样经过 3 次迭代就可以生成一棵最小重建错误的编码二叉树。

3.1.2 参数训练

我们可以将最小重建错误二叉树的生成视为正向传播，通过 BP 算法[14]，利用梯度下降的方法来优化网络中的各个参数。

我们将主要分析3.1.1节中所提到的递归单元这样一个小的网络结构，整棵树的错误传递过程和递归单元基本相同。将子节点看做输入，父节点为隐层输出，重建节点为输出层，为了方便计算定义以下若干式子：

$$
\begin{aligned}
[c_1'; c_2'] &= [y_1; y_2] \\
W^{(1)} &= [W_i^1, W_i^2] \\
b^{(1)} &= b_i \\
W^{(2)} &= [W_o^1; W_o^2] \\
b^{(2)} &= [b_o^1; b_o^2]
\end{aligned}
$$

其中为区分 c' 与 c 我们将输出变量改写为 y，并将式 (3-2) 中的 $W^{(1)}$ 分解为了 W_i^1, W_i^2 这两个参数，式 (3-3) 中的 $W^{(2)}$ 分解为 W_o^1, W_o^2，这样就使得所有参数矩阵均为 $\mathbb{R}^{n\times n}$，并且在计算时可以分别对 c_1'，c_2' 及与其有关的参数优化，有利于向量化的操作。

$$
\begin{aligned}
g(x) &= tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \\
f(x) &= \frac{g(x)}{\| g(x) \|_2} \\
X_i &= W_i^1 c_1 + W_i^2 c_2 + b_i \\
p' &= g(X_i) \\
p &= f(X_i) \\
X_o^1 &= W_o^1 p + b_o^1 \\
X_o^2 &= W_o^2 p + b_o^2 \\
y_1' &= g(X_o^1) \\
y_2' &= g(X_o^2) \\
y_1 &= f(X_o^1) \\
y_2 &= f(X_o^2)
\end{aligned}
$$

我们将 g 记为 $tanh$ 的激发函数，将 f 记为归一化后的函数，同时定义了若干输入与输出层的中间变量 X 以简化偏导数的表示。

有了上述若干公式之后，我们的目标函数就变为式 (3-6)。

$$
J_{RAE} = \frac{1}{N}\sum_s fE_t(s;\theta) + \frac{\lambda_{rec}}{2} \| \theta_{rec} \|^2 + \frac{\lambda_L}{2} \| \theta_L \|^2 \tag{3-6}
$$

其中 E_t 为一个短语的重建错误之和，f 为该短语在训练集中出现的频度，N 为训练集的所有短语的总内节点个数。第二，三项为正则项，用于抑制过学习的情况，λ_{rec}，λ_L 为超参数。因此式 (3-6) 的实质就是最小化训练集中的平均节点重建错误，而我们所要优化的参数为 $\theta = \theta_{rec} \cup \theta_L, \theta_{rec} = \{W_i^1, W_i^2, b_i, W_o^1, W_o^2, b_o^1, b_o^2\}, \theta_L = \{L\}$。需要说明的是，我们需要同时优化 L 使得训练完成后每个单词的词向量能捕获更多的语义及情感信息，而为了优化 L，我们可以将一个有 m 个单词的网

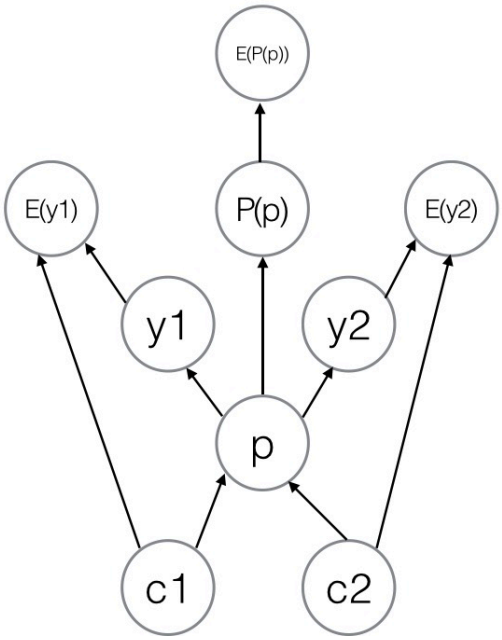

图 3.3　中心节点不为叶节点的错误关系

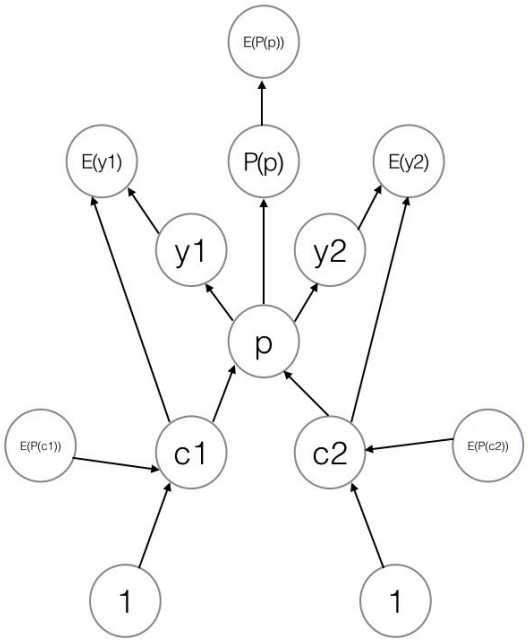

图 3.4　中心节点为叶节点的错误关系

络输入向量看做式 (3-1) 中的 $\{b_k\}_1^m$，将矩阵 L 看作一层的权重矩阵，通过该层网络后，$b_k{}_1^m$ 就会变为实际的输入词向量 $\{x_i\}_1^m$。

计算每个短语重建错误 E_t 的过程实际就是递归计算图3.2中每个递归单元（节点）重建错误 E 的过程。我们可以用式 (3-7) 表示一个递归单元的重建错误。

$$E = \frac{f}{2N} \parallel y_1 - c_1 \parallel^2 + \frac{f}{2N} \parallel y_2 - c_2 \parallel^2 + E_{P(p)} \tag{3-7}$$

三项中第一，二项为两个重建节点的重建错误，第三项 $E_{P(p)}$ 为由 p 的父节点 $P(p)$ 传递下来的错误。

图3.3，图3.4展示了一个计算单元中各个节点间的错误传递关系，我们可以根据式 (3-7) 以及图中所展示的关系计算单元中各节点的梯度。

首先由递归单元的顶层输出层开始，计算 W_o，b_o 的梯度。

$$\begin{aligned} \frac{\partial E}{\partial W_o^1} &= \frac{f}{N}\frac{\partial y_1^T}{\partial X_o^1}\frac{\partial E}{\partial y_1}\frac{\partial X_o^1}{\partial W_o^1} = \frac{f}{N}\frac{\partial y_1^T}{\partial X_o^1}(y_1 - c_1)p^T \\ \frac{\partial E}{\partial b_o^1} &= \frac{f}{N}\frac{\partial y_1^T}{\partial X_o^1}\frac{\partial E}{\partial y_1}\frac{\partial X_o^1}{\partial b_o^1} = \frac{f}{N}\frac{\partial y_1^T}{\partial X_o^1}(y_1 - c_1) \end{aligned} \tag{3-8}$$

其中 $\frac{\partial y_1^T}{\partial X_o^1}$ 为 y_1^T 对 X_o^1 求导后的 Jacobi 矩阵。如果我们展开 $\frac{\partial f(x)}{\partial x}$，则可以得到

$$\frac{\partial f(x)}{\partial x} = \begin{pmatrix} sech^2(x_1) & & 0 \\ & \ddots & \\ 0 & & sech^2(x_n) \end{pmatrix} \times \left(\frac{1}{\|tanh(x)\|} \boldsymbol{I} - \frac{1}{\|tanh(x)\|^3} tanh(x) tanh(x)^T \right)$$

其中 n 为短语向量的维度。

由式 (3-8) 可得 W_o^2，b_o^2 的梯度类似。

$$\frac{\partial E}{\partial W_o^2} = \frac{f}{N} \frac{\partial y_2^T}{\partial X_o^2} \frac{\partial E}{\partial y_2} \frac{\partial X_o^2}{\partial W_o^2} = \frac{f}{N} \frac{\partial y_2^T}{\partial X_o^2} (y_2 - c_2) p^T$$
$$\frac{\partial E}{\partial b_o^2} = \frac{f}{N} \frac{\partial y_2^T}{\partial X_o^2} \frac{\partial E}{\partial y_2} \frac{\partial X_o^2}{\partial b_o^2} = \frac{f}{N} \frac{\partial y_2^T}{\partial X_o^2} (y_2 - c_2)$$

之后错误由输出层与 p 的父节点传到 p 处，由此可以得到 p 的梯度。

$$\frac{\partial E}{\partial p} = \frac{f}{N} \frac{\partial X_o^1}{\partial p} \frac{\partial y_1^T}{\partial X_o^1} \frac{\partial E}{\partial y_1} + \frac{f}{N} \frac{\partial X_o^2}{\partial p} \frac{\partial y_2^T}{\partial X_o^2} \frac{\partial E}{\partial y_2} + \frac{\partial E_{P(p)}}{\partial p} \tag{3-9}$$

求得 p 的梯度后，我们便可以将错误再次向下传播至输入节点，计算 W_i，b_i 的梯度。

$$\begin{aligned}
\frac{\partial E}{\partial W_i^1} &= \frac{\partial p^T}{\partial X_i} \Big(\frac{f}{N} W_o^{1T} \frac{\partial y_1^T}{\partial X_o^1} (y_1 - c_1) + \frac{f}{N} W_o^{2T} \frac{\partial y_2^T}{\partial X_o^2} (y_2 - c_2) + \frac{\partial E_{P(p)}}{\partial p} \Big) c_1^T \\
\frac{\partial E}{\partial W_i^2} &= \frac{\partial p^T}{\partial X_i} \Big(\frac{f}{N} W_o^{1T} \frac{\partial y_1^T}{\partial X_o^1} (y_1 - c_1) + \frac{f}{N} W_o^{2T} \frac{\partial y_2^T}{\partial X_o^2} (y_2 - c_2) + \frac{\partial E_{P(p)}}{\partial p} \Big) c_2^T \\
\frac{\partial E}{\partial b_i} &= \frac{\partial p^T}{\partial X_i} \Big(\frac{f}{N} W_o^{1T} \frac{\partial y_1^T}{\partial X_o^1} (y_1 - c_1) + \frac{f}{N} W_o^{2T} \frac{\partial y_2^T}{\partial X_o^2} (y_2 - c_2) + \frac{\partial E_{P(p)}}{\partial p} \Big)
\end{aligned} \tag{3-10}$$

然后我们还需要将错误传到输入层 c_1，c_2，即求 $\frac{\partial E_P(c)}{\partial c}$，使错误传至下层单

元。

$$\frac{\partial E_P(c_1)}{\partial c_1} = W_i^{1T}\frac{\partial p^T}{\partial X_i}\Big(\frac{f}{N}W_o^{1T}\frac{\partial y_1^T}{\partial X_o^1}(y_1-c_1)+\frac{f}{N}W_o^{2T}\frac{\partial y_2^T}{\partial X_o^2}(y_2-c_2)+\frac{\partial E_{P(p)}}{\partial p}\Big)-\frac{f}{N}(y_1-c_1)$$
$$\frac{\partial E_P(c_2)}{\partial c_2} = W_i^{2T}\frac{\partial p^T}{\partial X_i}\Big(\frac{f}{N}W_o^{1T}\frac{\partial y_1^T}{\partial X_o^1}(y_1-c_1)+\frac{f}{N}W_o^{2T}\frac{\partial y_2^T}{\partial X_o^2}(y_2-c_2)+\frac{\partial E_{P(p)}}{\partial p}\Big)-\frac{f}{N}(y_2-c_2) \tag{3-11}$$

由于编码器二叉树的根节没有父节点，因此 $E_{P(p)}=0$，与其有关的梯度值也均为 0。而对于整个网络的输入词向量 x_i 来说，还需要利用式 (3-1) 对 L 求导以更新 L。

$$\frac{\partial E_P(x_i)}{\partial L} = \frac{\partial E_P(x_i)}{\partial x_i}\frac{\partial x_i}{\partial L} = \frac{f}{n}\frac{\partial E_P(x_i)}{\partial x_i}b_{k(i)}^T \quad i=1,2,...,m \tag{3-12}$$

至此，各参数梯度都已经求得，我们可以采用梯度下降的方法，利用 L-BFGS[15] 优化算法对各参数进行优化。

3.1.3 相关实验

非监督式的递归自动编码器在 Github 上有一份开源代码①，基本实现了文献 [8] 中所描述的非监督式自动编码器的编码树生成，以及参数训练时求解梯度的框架。

我首先需要做的工作就是读懂代码，完全搞懂代与递归自动编码器相关的核心代码。我推导出了3.1.2节中求各节点梯度的公式，这对于之后将此代码扩展至下节的双语结构有很大帮助。

此份代码采用了消息传递接口（Message-Pssing Interface, MPI）进行分布式多线程编程，使得大规模数据的训练成为了可能，还包括了一些如梯度检查②的一些小工具。除此之外，它预留了一些接口与数据结构供我修改，扩展性比较好，让我可以修改以适应文献 [16] 中的训练过程。

由于文献 [8] 并没有完全列举所有参数的求法，所以在推导公式时我都是依照自己和网络的帮助完成的。我在阅读文献 [8] 时忽略了对底层对词向量矩阵 L

① 网址为：https://github.com/pengli09/str2vec
② 梯度检查的工具是后续实验中验证公式推导以及代码正确性的最重要的工具。

的优化，而且代码中也没有对其进行优化[①]，我在后面的实验中才发现这一点，这也反映出我的不自信与读论文的不仔细。

1. 向量可视化

利用非监督式递归自动编码器可以在单语短语中进行短语距离的可视化，以验证其编码语义信息的功能。

首先我利用 Moses[②]在平行语料中抽取短语翻译表。之后选择短语表一端的（如选择 *s* 端）短语，利用它们训练非监督式自动编码器的参数，求出短语向量并将其可视化。

2. 实验设置

本次实验的数据集是 120 万句对 LDC 中英双语平行语料的很小一部分，*s* 端为中文，*t* 端为英文，共 3 万个句对，抽取了 38 万个双语短语对。词向量的词典是在整个 120 万个句对中生成的，共有 8 万中文单词与 7.2 万英文单词。在这 38 万双语对中找出每个 *s* 端短语对应的翻译概率最高的短语对（1 Best）进行过滤，保留了 3.6 万短语对，将其 *s* 端作为最终处理后的训练语料。

考虑到深度学习的实验周期非常长，因此缩小向量空间，将短语向量维度设为 10 维，最大训练迭代次数 60 次。经过实验，设置重建错误正则项超参数为 0.15，词向量矩阵 L 的正则项超参数为 10^{-2}。

实验环境为 Ubuntu12.04 服务器，训练时通过 MPI 接口使用 6 个 Intel(R) Xeon(R) 2.60GHz 处理器。

3. 实验结果与讨论

由于短语对中存在很多没有实际语义的短语，因此随机选择 40 个左右有实际语义信息的短语，进行可视化。

利用训练后的参数可以求得每一个短语的向量。由于维度为 10 维，故利用主成分分析（PCA）对其降至 2 维，如图3.5所示。将下方两个重叠较多的部分放大，如图3.6，图3.7所示。

可以看到大部分相近的词语聚类的情况都比较令人满意，可以基本验证这种方法的有效性，例如图3.5中最左侧有关军事的“伊拉克军队”，“副总参谋长”和“军事设施”，以及最右侧与美国有关的短语；图3.7中的大部分银行，与中国有关的词语也都互相接近。

① 通过与原作者沟通，我了解到这份代码是一份简化版的非监督式自动递归编码器，原始版本实现了文献 [10] 的工作，并非开源。

② 开源翻译系统，在后文中将具体介绍。

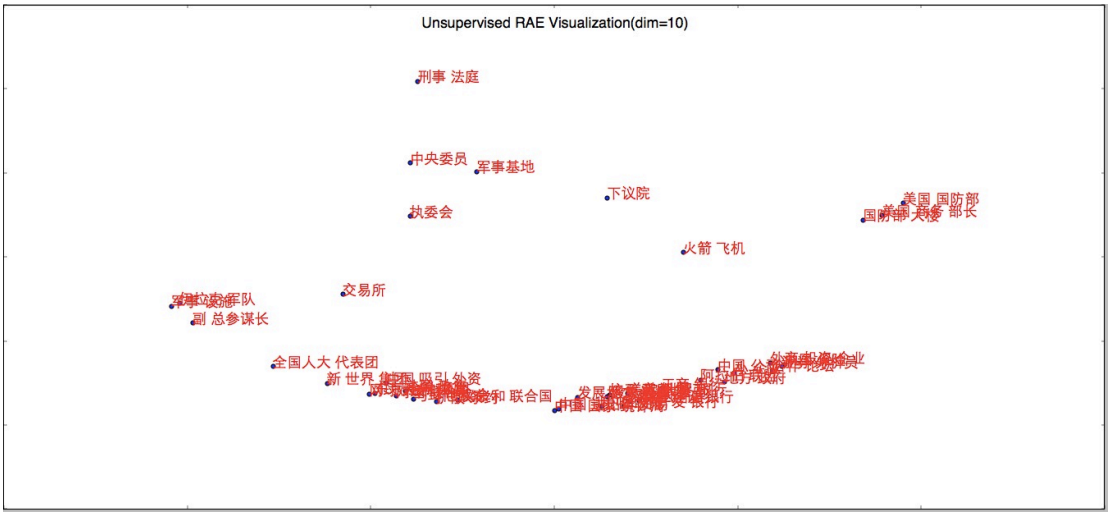

图 3.5　非监督式自动编码器可视化

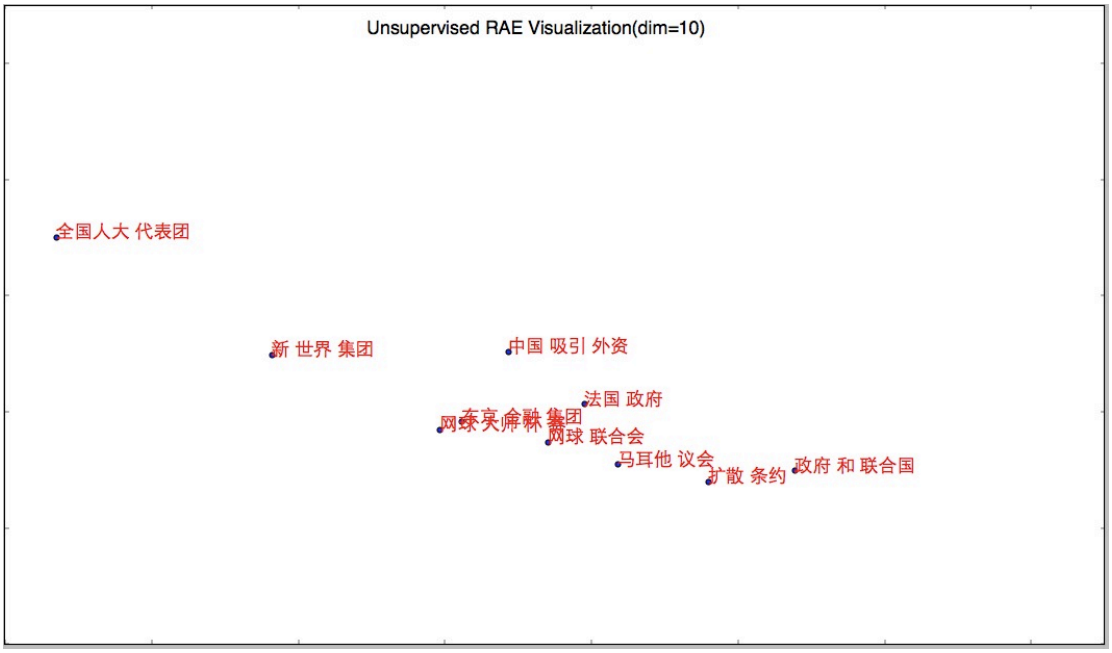

图 3.6　可视化左侧放大部分

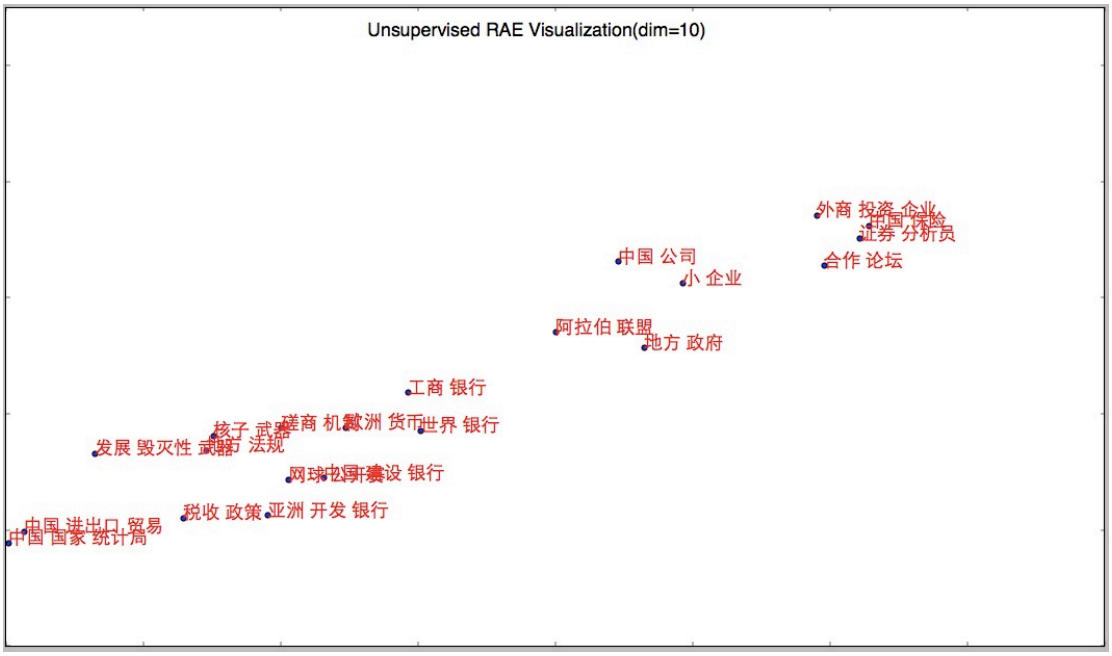

图 3.7　可视化右侧放大部分

然而我们也应该看到一些问题，例如图3.5上方的军事基地并没有与其他与军事有关的词语聚集在一起，还有一些孤立的，单独存在的词语，其语义并没有与其他词语有很强的关联性，但是它们又与这些词类距离很近，例如图3.7的“磋商机制”等。

我认为以下原因可能会使最后的实验结果不准确：

（1）训练语料所用的短语仅仅是通过对齐抽取出来的，很多并没有实际的语义，这些短语在训练时会干扰实意短语的聚类。

（2）训练集数目过小，仅有 3 万个短语对。

（3）训练过程中迭代次数不足，参数优化时没有收敛。

（4）PCA 降维所造成的误差。本实验中信息量最大的 2 维特征在正则化后的协方差矩阵中的值为 0.3，0.2，即大致有 50% 的信息丢失。

总的来说在粗略实验的情况下，以聚类的标准我们可以看到非监督式递归编码器还是可以将语义相近的短语向量聚集在一起，较好获得短语的语义信息。

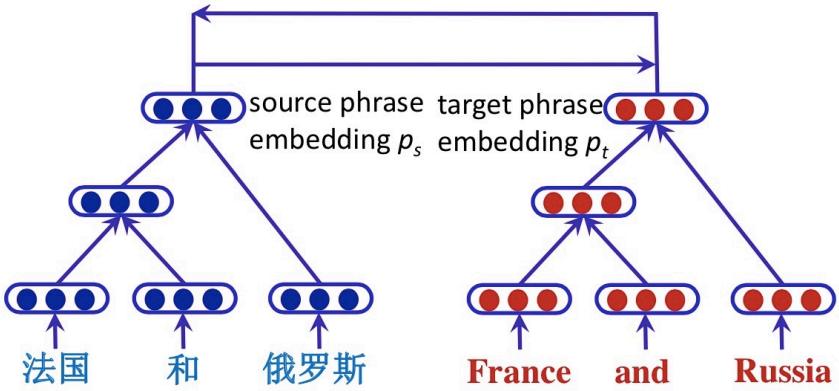

图 3.8　双语短语递归自动编码器举例

3.2　双语短语递归自动编码器

非监督式递归编码器有着语料容易获取，训练方法直接，模型连续，包含语义信息丰富等诸多优点，而且可以通过比较短语向量间的距离衡量短语之间语义上的相似程度 (Similarity)，那么如何将其拓展到机器翻译领域，利用递归编码器的优势，比较双语间某个短语对的相似度，作为短语翻译系统的一个特征帮助解码便成了一个非常自然却不失创新的想法。于是 Zhang et al. (2014)[16] 便在的递归自动编码中更进一步，将其改进后应用于双语体系中，在双语的两个不同向量空间中分别建模，并提出一种新的方法计算双语间短语向量的相似度，获取双语中语义相近的短语对，并在短语表剪枝，短语语义比较中得到了成功的应用。

3.2.1　双语语义错误

如图3.8所示设中文为源语言端（s 端），英文为目标语言端（t 端），左右两侧英语与中文短语互为翻译短语。如果我们在 s 端通过非监督式自动编码器学习得到该短语的向量表示，就可以把它作为标签用来帮助学习 t 端对应短语的向量表示，反之也是一样。将该问题视为一个半监督式学习的问题，采用高质量的短语对作为训练数据，在非监督式递归编码器的基础上，使 s，t 端的短语向量互为对方的标签，通过最小化短语对向量表示的语义距离我们可以让两个向量在包含尽可能多语义信息的同时在“语义距离”的衡量下尽可能接近，使语义相近的双语短语距离小。

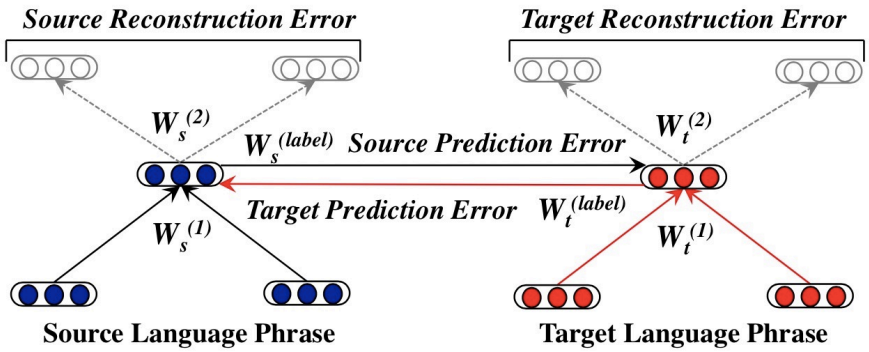

图 3.9　双语短语递归自动编码器网络结构

这样我们就需要解决两个问题：如何选取训练短语对使得短语对间的语义相近？如何计算双语间的“语义距离”？

不同于一般的监督式学习，短语对间不存在绝对的“黄金标签”，即 s 端的短语在 t 端可能有多个对应的语义相近的短语。不过，根据前述我们可以从高质量双语平行语料库中抽取短语规则，抽取出的短语对由于传达的是完全相同的信息因此其语义信息必然相近，所以我们可以将这些短语对作为训练集，使其互为标签同时进行训练。

如果我们假设两端短语对的向量表示维度均为 $\mathbb{R}^n$，则计算二者语义距离的方法就可以用最简单的欧式距离得到。但是需要注意的是，由于我们是在两端分别进行建模，因此二者不必处在同一个向量空间中。我们假设二者处在两个不同的向量空间中，则 s 端的短语向量表示可以通过一个空间变化映射到 t 端向量空间中，同理对 t 端也适用，并可以用式 (3-13) 表示。

$$\begin{aligned} y_{lt} &= f(W_t^l p_t + b_t^l) \\ y_{ls} &= f(W_s^l p_s + b_s^l) \end{aligned} \tag{3-13}$$

y_{ls} 为 s 端短语在 t 端向量空间中的向量表示，y_{lt} 的定义也相似。

有了式 (3-13) 的两个向量，我们再计算 s（t）端变换后的向量与 t（s）端的原向量间的欧氏距离，得到两个向量间的语义距离。根据图3.9可见，两侧编码器二叉树根节点除了存在重建错误之外，还存在两个标签预测错误，即所谓的语义距离，用式 (3-14) 表示。

$$
\begin{aligned}
E_{sem}(s \mid t, \theta) &= \frac{1}{2} \parallel p_t - f(W_s^l p_s + b_s^l) \parallel^2 \\
E_{sem}(t \mid s, \theta) &= \frac{1}{2} \parallel p_s - f(W_t^l p_t + b_t^l) \parallel^2
\end{aligned}
\tag{3-14}
$$

但是在实际实验中，式 (3-14) 只能通过最小化语义错误使得两个语义相近的短语向量相近，但并没有语义不相近的短语反例来保证其短语向量语义距离较远，因此我们需要在学习正例的同时学习反例。Zhang et al. (2014)[16] 在式 (3-14) 的基础上扩展出了最大化裕量错误 (max-margin error) 的方法，使得正例与反例的语义错误之差始终在一个裕量之上 (在文献 [16] 中定义为 1)，可用式 (3-15) 表示。

$$
\begin{aligned}
E_{s,sem}^* &= \max\{0, E_{s(t),sem} - E_{s(t'),sem} + 1\} \\
E_{t,sem}^* &= \max\{0, E_{t(s),sem} - E_{t(s'),sem} + 1\}
\end{aligned}
\tag{3-15}
$$

以 s 端为例，生成反例 s' 的方法就是在词向量矩阵 L 中随机选取 m 个词向量构成 s' 的输入词向量，m 为 s 的单词个数，s' 的编码过程所用参数与 s 完全相同。同理对 t 端同样适用。

3.2.2 参数训练

在3.1.2节所需训练的参数基础上，依据图3.9，我们归纳出以下三类参数 θ。

（1）θ_L：词向量参数矩阵；

（2）θ_{rec}：非监督学习中所需的参数，与3.1.2节的参数完全相同，具体即为 $\{W_i^1, W_i^2, b_i, W_o^1, b_o^1, W_o^2, b_o^2\}$；

（3）θ_{sem}：与语义错误相关的参数 $\{W^l, b^l\}$。

由于需要优化 s 端与 t 端，所以上述参数数量会加倍，在 s 端与 t 端各存在一个参数集。

因此式 (3-6) 会改写为式 (3-16)。

$$
J_{BRAE} = \frac{f}{N} \sum_{(s,t)\in(S,T)} E(s, t; \theta) + \frac{\lambda}{2} \parallel \theta \parallel^2
\tag{3-16}
$$

$E(s,t;\theta)$ 为双语短语对的总体错误，第二项为正则项，包括之前的三类参数。$E(f,e;\theta)$ 可以分为 s，t 端的重建错误与语义错误，展开写即为:

$$\begin{aligned} E(s,t;\theta) &= \alpha E_{rec}(s,t;\theta) + (1-\alpha)E_{sem}(s,t;\theta) \\ &= \alpha\big(E_{rec}(s;\theta) + E_{rec}(t;\theta)\big) + (1-\alpha)\big(E^*_{sem}(s \mid t,\theta) + E^*_{sem}(t \mid s,\theta)\big) \\ &= \big(\alpha E_{rec}(s;\theta) + (1-\alpha)E^*_{sem}(s \mid t,\theta)\big) + \big(\alpha E_{rec}(t;\theta) + (1-\alpha)E^*_{sem}(t \mid s,\theta)\big) \end{aligned} \tag{3-17}$$

我们可以将 s, t 端的反例视为常数，直接利用梯度下降的方法优化式 (3-17)，也可以对 s，t 端同时优化，采用同时训练（co-training）的方法，将式 (3-17) 分解为两个子式 (3-18)。

$$\begin{aligned} E(s \mid t;\theta) &= \alpha E_{rec}(s;\theta) + (1-\alpha)E^*_{sem}(s \mid t,\theta) \\ E(t \mid s;\theta) &= \alpha E_{rec}(t;\theta) + (1-\alpha)E^*_{sem}(t \mid s,\theta) \end{aligned} \tag{3-18}$$

当采用同时训练的方法时，需要优化的目标函数就变成了两个，但是我们仍通过式 (3-17) 来判断两个目标函数是否收敛。文献 [16] 中采用了这种同时训练的方法。[①]

类比式 (3-7)，我们用式 (3-19) 表示双语某一端一个递归单元的错误。

$$\begin{aligned} E =& \frac{\alpha f}{2N}\Big(\parallel y_1 - c_1 \parallel^2 + \parallel y_2 - c_2 \parallel^2 \Big) \\ &+ \frac{(1-\alpha)f}{2N} \max\{0, (\parallel y_l - p_o \parallel^2 - \parallel y_l - p_{o'} \parallel^2 + 1)\} + E_{P(p)} \end{aligned} \tag{3-19}$$

其中 p_o 为另一端 (Other Side) 短语的向量表示，p'_o 为其反例向量表示。

由于第二项语义错误是基于裕量错误的 hinge loss，其在定义域内存在不可导的点，因此我们可以先对式 (3-20) 求梯度，在确保梯度正确后再加入 hinge loss

① 第一种方法是我在推导文献 [16] 中参数梯度时采用的，在和原作者讨论后发现他在文中间接指出了第二种方法；鉴于两种方法在小测试集的表现近似，而且第二种方法的思想更明确，表达更简洁，故后续代码均采用该方法。

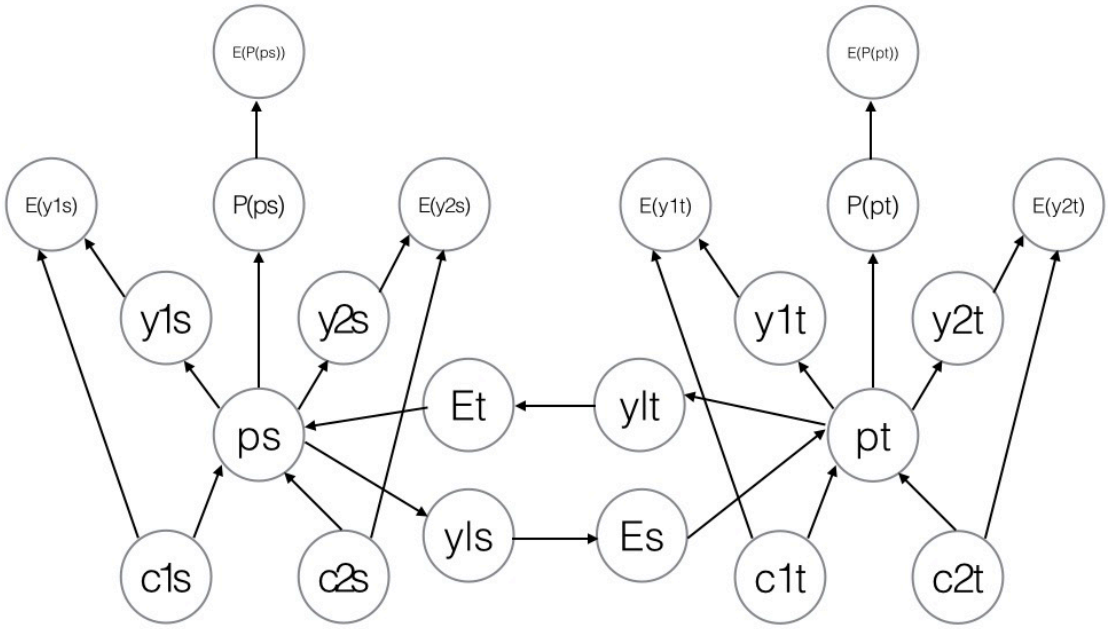

图 3.10　中心节点不为叶节点的错误关系

的形式。

$$\begin{aligned} E =& \frac{\alpha f}{2N}\left(\| y_1 - c_1 \|^2 + \| y_2 - c_2 \|^2 \right) \\ &+ \frac{(1-\alpha) f}{2N}(\| y_l - p_o \|^2 - \| y_l - p_{o'} \|^2 + 1) + E_{P(p)} \end{aligned} \tag{3-20}$$

由于双语结构的自动编码器与非监督式的自动编码器的结构大致相同，故许多参数的求导都相同。在计算各参数梯度前，我们在3.1.2节的基础上再添加以下记号。

$$\begin{aligned} X_l &= W_l p + b_l \\ y_l' &= g(X_l) \\ y_l &= f(X_l) \end{aligned}$$

节点为根节点时的错误关系可以用图3.10表示，其余节点的错误关系不变，即图3.3和图3.4。其中 E_t, E_s 即为 t 端与 s 端的语义错误。

首先依旧是由顶层传来的的重建错误。由于此处的结构没有变化，因此 $\frac{\partial E}{\partial W_o^1}, \frac{\partial E}{\partial W_o^2}, \frac{\partial E}{\partial b_o^1}, \frac{\partial E}{\partial b_o^2}$ 的公式不变，即为式 (3-8)。

同时语义错误也会从顶层传来，我们可以计算 W_l, b_l 的梯度。

$$\begin{aligned}\frac{\partial E}{\partial W_l} &= \frac{(1-\alpha)f}{N}\left(\frac{\partial y_l^T}{\partial X_l}(y_l - p_o) - \frac{\partial y_l^T}{\partial X_l}(y_l - p_{o'})\right)p^T \\ \frac{\partial E}{\partial b_l} &= \frac{(1-\alpha)f}{N}\left(\frac{\partial y_l^T}{\partial X_l}(y_l - p_o) - \frac{\partial y_l^T}{\partial X_l}(y_l - p_{o'})\right)\end{aligned} \tag{3-21}$$

因此当错误由输出层与 p 的父节点传到 p 处时，与式 (3-9) 不同，需要增加由语义错误传来错误而得到 p 的梯度。

$$\begin{aligned}\frac{\partial E}{\partial p} =& \frac{\alpha f}{N}\frac{\partial X_o^1}{\partial p}\frac{\partial y_1^T}{\partial X_o^1}\frac{\partial E}{\partial y_1} + \frac{\alpha f}{N}\frac{\partial X_o^2}{\partial p}\frac{\partial y_2^T}{\partial X_o^2}\frac{\partial E}{\partial y_2} \\ &+ \frac{(1-\alpha)f}{N}\frac{\partial y_l^T}{\partial X_l}(y_l - p_o) - \frac{(1-\alpha)f}{N}\frac{\partial y_l^T}{\partial X_l}(y_l - p_{o'}) + \frac{\partial E_{P(p)}}{\partial p}\end{aligned} \tag{3-22}$$

求得 p 的梯度后，类比式 (3-10)，计算 W_i, b_i 的梯度。

$$\begin{aligned}\frac{\partial E}{\partial W_i^1} =& \frac{\partial p^T}{\partial X_i}\Big(\frac{\alpha f}{N}W_o^{1T}\frac{\partial y_1^T}{\partial X_o^1}(y_1 - c_1) + \frac{\alpha f}{N}W_o^{2T}\frac{\partial y_2^T}{\partial X_o^2}(y_2 - c_2) \\ &+ \frac{(1-\alpha)f}{N}W_l^T(y_l - p_o) - \frac{(1-\alpha)f}{N}W_l^T\frac{\partial y_l^T}{\partial X_l}(y_l - p_{o'}) + \frac{\partial E_{P(p)}}{\partial p}\Big)c_1^T \\ \frac{\partial E}{\partial W_i^2} =& \frac{\partial p^T}{\partial X_i}\Big(\frac{\alpha f}{N}W_o^{1T}\frac{\partial y_1^T}{\partial X_o^1}(y_1 - c_1) + \frac{\alpha f}{N}W_o^{2T}\frac{\partial y_2^T}{\partial X_o^2}(y_2 - c_2) \\ &+ \frac{(1-\alpha)f}{N}W_l^T(y_l - p_o) - \frac{(1-\alpha)f}{N}W_l^T\frac{\partial y_l^T}{\partial X_l}(y_l - p_{o'}) + \frac{\partial E_{P(p)}}{\partial p}\Big)c_2^T \\ \frac{\partial E}{\partial b_i} =& \frac{\partial p^T}{\partial X_i}\Big(\frac{\alpha f}{N}W_o^{1T}\frac{\partial y_1^T}{\partial X_o^1}(y_1 - c_1) + \frac{\alpha f}{N}W_o^{2T}\frac{\partial y_2^T}{\partial X_o^2}(y_2 - c_2) \\ &+ \frac{(1-\alpha)f}{N}W_l^T(y_l - p_o) - \frac{(1-\alpha)f}{N}W_l^T\frac{\partial y_l^T}{\partial X_l}(y_l - p_{o'}) + \frac{\partial E_{P(p)}}{\partial p}\Big)\end{aligned} \tag{3-23}$$

最后同样类比式 (3-11) 计算 c_1, c_2 的梯度。

$$\frac{\partial E_P(c_1)}{\partial c_1} = W_i^{1T}\frac{\partial p^T}{\partial X_i}\Big(\frac{\alpha f}{N}W_o^{1T}\frac{\partial y_1^T}{\partial X_o^1}(y_1 - c_1) + \frac{\alpha f}{N}W_o^{2T}\frac{\partial y_2^T}{\partial X_o^2}(y_2 - c_2)$$
$$+ \frac{(1-\alpha)f}{N}W_l^T(y_l - p_o) - \frac{(1-\alpha)f}{N}W_l^T\frac{\partial y_l^T}{\partial X_l}(y_l - p_{o'}) + \frac{\partial E_{P(p)}}{\partial p}\Big) - \frac{\alpha f}{N}(y_1 - c_1)$$
$$\frac{\partial E_P(c_2)}{\partial c_2} = W_i^{2T}\frac{\partial p^T}{\partial X_i}\Big(\frac{\alpha f}{N}W_o^{1T}\frac{\partial y_1^T}{\partial X_o^1}(y_1 - c_1) + \frac{\alpha f}{N}W_o^{2T}\frac{\partial y_2^T}{\partial X_o^2}(y_2 - c_2)$$
$$+ \frac{(1-\alpha)f}{N}W_l^T(y_l - p_o) - \frac{(1-\alpha)f}{N}W_l^T\frac{\partial y_l^T}{\partial X_l}(y_l - p_{o'}) + \frac{\partial E_{P(p)}}{\partial p}\Big) - \frac{\alpha f}{N}(y_2 - c_2) \tag{3-24}$$

而对词向量参数矩阵 L 的倒数与式 (3-12) 相同。在求出了所有参数的梯度后，我们亦然利用 L-BFGS 算法，采用梯度下降的方法对各个参数进行优化。

最后仿照算法1，我们用算法2表示双语短语递归自动编码器的训练方法。

Algorithm 2 BRAE

1: 预训练（Pre-training）：采用标准非监督式递归自动编码器分别学习双语短语对中 s 端与 t 端的向量表示

2: 参数优化（Fine-tuning）：在 BRAE 的框架下，用 s 端短语向量 p_{s_0} 优化 t 端参数 θ_t，同时用 t 端短语向量 p_{t_0} 去优化 s 端参数 θ_s，之后利用优化后的参数计算得到更新的短语对向量 p_s，p_t 与总错误式 (3-17)

3: 终止条件检查（Termination Check）：如果式 (3-17) 达到局部最优解或者超过一定的迭代次数（如 25 次）则终止迭代，否则我们令 $p_{s_0} = p_s$，$p_{t_0} = p_t$，继续第 2 步迭代进行参数优化。

3.2.3 相关实验

在3.1.3节的非监督式递归自动编码器的基础上，我需要将其完全改造成双语短语递归自动编码器的结构。

依据算法2的过程，我的代码改造过程大致分为以下几步。

（1）在原代码基础上加入单语语义错误；

（2）将单语语义错误拓展到双语语义错误。

其中第 2 步必须利用 co-training 的同时训练的方法，而第 1 步可以分开训练，也可以同时训练。

同时，我需要验证该方法确实有效，因此我实现了文献 [16] 中的短语表剪枝实验。

1. 短语表剪枝

该实验的大致内容就是在3.1.3节的基础上，利用之前在短语翻译系统 Moses 抽取的短语和训练好的非监督式递归编码器的参数，在双语体系下利用 co-training 的方法训练双语短语递归编码器的参数。

通过训练好的双语短语递归编码器我们可以求出整个短语表中每个双语短语对的双向余弦相似度 $Sim(y_{ls}, p_t), Sim(y_{lt}, p_s)$。设定某个阈值，如果两个相似度的任意一个值小于阈值，我们就将此短语对剪枝，否则保留。通过这样的操作我们可以减少解码时翻译短语表的短语数目，继而缩小翻译模型的搜索空间，使得解码速度更快，但同时翻译质量也不会受到很大的影响。

我们将剪枝过的短语表作为训练抽取的短语表，再采用最小化错误率算法（MERT）[17]，测试，与原短语表作为训练短语表的 BLEU 值做比较，以判断其实验效果。

2. Moses 介绍

本实验的短语翻译系统为开源翻译系统 Moses① [18]，它是由以爱丁堡大学为领导的多所大学开发出的当今最流行的统计机器翻译工具之一。它主要包含了训练流程和解码器两个模块，同时还有许多与之相关的工具，支持短语翻译，层次短语翻译，语句翻译等各种翻译模型，经常用来作为各种实验的基线（Baseline）。

然而由于 Moses 的功能非常强大，其操作与设置也相对比较繁琐故而需要一定的时间去熟悉它。由于我没有亲自实现过短语翻译系统，对其中的主要调序，训练算法的掌握也不是很好，因此在这方面花费了大量时间。

本实验需要利用 Moses 的训练和解码部分，翻译系统类型则为短语翻译系统。整个实验大致包括以下三个过程：

（1）从训练语料中抽取短语表，并给短语表打分（短语频率，词汇化权重等）；

（2）利用 MERT 在开发集中学习特征参数；

（3）在测试集上进行解码翻译。

① 网址为 http://www.statmt.org/moses/

3. 实验设置

本次实验的训练语料依旧是从 3 万个句对中抽取并处理过的 3 万个短语对。开发集数据为 NIST06，测试集为 NIST02，04，05，08，其中 NIST 的每个数据集都有 500 至 800 个句对。在得到双向余弦相似度之后，根据阈值分布，在对短语表剪枝时设定阈值为 0.05，0.1，0.15。

词向量矩阵 L，重建错误和语义错误正则项的超参数分别为 10^{-2}，10^{-3} 和。的超参数为重建错误和语义错误之间的超参数 α 为 0.15，最大迭代次数为 25 次，与论文所述相同。改变短语向量的维度测试维度对其影响，分别设置维度为 50，100，200。

4. 实验结果与讨论

实验后 BLEU 值的总体结果见表3.1。

表 3.1　BLUE Score for Different Phrase Tables

Method	n	**Threshold**	**PT**	**MT02**	**MT04**	**MT05**	**MT08**	**ALL**
Baseline			100%	28.06	27.57	26.4	18.25	25.07
BRAE	50(Rd)	0.05	64.5%	22.34	22.15	20.71	14.33	19.88
		0.1	44.5%	18.75	18.60	17.49	10.69	16.38
		0.15	26.6%	13.66	14.16	12.95	7.78	12.13
BRAE	100(Rd)	0.05	65.0%	**22.69**	21.83	20.38	14.18	19.77
		0.1	39.78%	17.93	17.34	15.97	11.20	15.61
		0.15	19.13%	11.33	10.36	9.63	6.48	9.45
BRAE	200(Rd)	0.05	50.6%	21.87	21.56	20.34	13.89	19.42
		0.1	26.93%	15.63	14.87	13.79	9.27	13.39
		0.15	14.4%	8.98	8.11	7.56	4.74	7.35
BRAE	50(W2V)	0.05	67.9%	22.62	**22.67**	**21.91**	**14.85**	**20.51**
		0.1	59.1%	**20.41**	**20.03**	**18.75**	**12.56**	**17.93**
		0.15	46.9%	**16.49**	**16.22**	**14.69**	**9.55**	**14.24**

其中 Rd 为随机初始化词向量，W2V 为利用 Word2Vec 进行预训练。其中加粗的部分是经过剪枝的短语表在相应阈值和数据集上的最高值。

从实验结果可以得到，在短语向量维度一定的情况下，增大阈值，短语表中所剪枝的短语对就会越多；当阈值不变，维度改变时，维度为 50 与 100 的效果近似，普遍比维度为 200 时的效果要好，这两点与文献 [16] 的结果基本一致。

但是可以看到，本次实验的结果与原文的实验结果还相差较远。在原文中，如果采用经过剪枝的短语表作为短语模型的解码短语表，其测试结果的变化随阈值的变化不明显，即在各个阈值区间，测试结果都与基线结果相差较小，基本在 1 以内，而且在有的阈值下其实验结果甚至比基线更好；而我的模型对阈值的敏感度很高，阈值稍微一变化短语表数目的变化就非常明显，而且短语对的相似度普遍较低。这说明我的模型在某种程度上并没有很好编码短语对之间的语义信息。

对于实验结果，我认为有以下几点可能使得结果不理想：

（1）模型问题：短语相似度过低，这一点非常可疑；尽管经过梯度检查工具的验证，我的参数训练部分代码应该是正确的，但是很可能在其他部分出现了一些小问题[①]。

（2）训练集差距悬殊：原实验采用 120 万短语对进行训练，本次实验仅仅采用了 3 万短语对进行训练。

（3）迭代次数：本文中为了加速实验，强制第一阶段单语的非监督式自动递归编码器的最大迭代次数为 60，通常情况下该值至少为 100 以上。

（4）设置不同：原文没有采用 Moses 翻译系统，而且其他设置也没有明确说明。另外原文中的训练数据通过强制对齐（Forced Decoding）[②]抽取高质量短语对，而我采用了 Moses 自带的过滤工具，从整个短语表中抽取 *s* 端对应短语翻译概率最高的短语对。

（5）其他原因：其他细节的原因，例如本次实验的平行语料与原文不同，语料质量本身可能存在差异。

本次实验非常耗时，随着短语向量维度的升高，训练集的扩大，其训练时间也会呈指数增长。对于维度为 50 的短语向量，在 3 万短语对上训练需要近 26 个小时，因此我没有足够时间去逐条排除模型效果不佳的实际原因。

而由于双语短语递归编码器的实验效果没有达到要求，因此在双语层次短语递归编码器的实验就不能继续进行。

① 我已将此结果与原作者进行了讨论，他列举了一些可能性，其中就存在此问题。

② 强制对齐利用双语平行预料，采用解码规则使得译文强制为参考译文，并从该过程中抽取高质量的短语对。

第 4 章　双语层次短语递归自动编码器

在完全实现3.2节中的双语短语递归自动编码器后，我们需要将其应用于层次短语当中，为层次短语也生成类似的向量表示。对于层次短语而言，最大的困难就是如何编码非终结符 X^*，即如何为其找到一个合理的向量表示。

每一个层次短语中的 X^* 都代表了一类相似的单词或短语，他们或是在结构方面相似，或是在文法分析中有相似的类型，因此我们可以将 X^* 看作具有相似结构或文法类型的若干短语的集合。

例 4.1：下面我们集中讨论以下例子。

（X_1 and X_2，X_1 和 X_2）‖ 4

（China and Japan，中国 和 日本）‖ 2

（cute and lovely，可爱 和 动人）‖ 1

（have lunch and wash dishes，吃午饭 和 洗盘子）‖ 1

显然，对于每个层次短语的每个 X^*，由于其包含的短语各不相同，故其向量量表示都不应相同，而其出现的频数即为其所包含的普通短语对的频数之和。

一个自然的想法就是该层次短语的向量应当与其所包含的所有普通短语向量存在一定关系，具体来说可以用式 (4-1) 或式 (4-2) 来表示。

$$v(X^*) = f(\cdot) \tag{4-1}$$

$$v(\text{层次短语}) = f(\cdot) \tag{4-2}$$

其中 $v(\cdot)$ 表示层次短语中特定部分的向量，$f(\cdot)$ 则是一个自变量仍待确定的未知函数。而我们的任务就是为上述两种表示方法找到合理的自变量与函数映射。

4.1 获取非终结符的向量表示

根据式 (4-1)，我们需要考虑的仅为非终结符 X^* 的向量表示。一旦确定了每个 X^* 的向量，那么 X^* 也就退化成了一个普通短语，而层次短语的向量表示也就退化成了普通短语的向量表示（即与3.2节所述相同）。

根据例**??**所述，X^* 所代表的短语均表示了其在层次短语下的文法类型，例如“（X_1 and X_2，X_1 和 X_2）”中的 X^* 即可以是形容词，也可以是名词，也可以是动词短语等等，但从另一方面来说它又只能是上述所列的文法类型之一，因此我们需要 X^* 所代表的实际短语的向量表示，它们所包含的信息是 X^* 的一部分。

采用最简单朴素的方法，我们可以使 X^* 为其所代表的所有短语的加权平均。仍以例 4.1 中的 X_1 为例，我们可以用式 (4-3) 来表示。

$$v(X_1) = \frac{1}{4}(2v(\text{China}) + v(\text{cute}) + v(\text{have lunch})) \tag{4-3}$$

其中短语和单词的向量应随着迭代改变。

可以看到此时 X_1 的向量即为其所代表的短语的中心（Centroid），在优化它们的同时也在优化它们的中心。我们可以改进算法2，利用算法3来表示层次短语的训练过程。

Algorithm 3 HBRAE(Non Terminal)

1: 抽取普通短语与层次短语及它们的频度。
2: 采用标准非监督式递归自动编码器分别学习双语短语对中 s 端与 t 端的向量表示，X^* 的向量可以采用式 (4-3) 计算。
3: 采用 BRAE 的标准训练方法，仍然利用式 (4-3) 计算每一个 X^*。

我们可以看到算法3中 X^* 的向量表示在每次迭代时都被已经更新过的普通短语的向量所替换，因此其过程可以看作是在优化普通短语的编码树的基础上，将它们的输出通过函数的变换作为 X^* 的输入，优化另外一棵更大的编码树。

4.2 直接获取层次短语的向量表示

根据式 (4-2)，我们可以绕过 X^* 的表示问题，转而直接求取层次短语的向量表示。这种方法的好处就是可以通过其他普通短语的向量直接得到层次短语的向量，而其缺点则是我们可能会丢失普通某些短语内部的信息。

我们依旧以例 4.1 为例，采用式 (4-3) 中的思想，仿照4.1节中的思想，使层次短语为其所包括的普通短语的中心，用式 (4-4) 表示。

$$\begin{aligned} v(X_1 \text{ and } X_2) = \frac{1}{4}(&2v(\text{China and Japan}) \\ &+ v(\text{cute and lovely}) \\ &+ v(\text{have lunch and wash dishes})) \end{aligned} \tag{4-4}$$

与4.1节中有所区别的是此时的层次短语的向量表示是其代表的整个普通短语的中心而 X^* 所代表的中心，而且此时层次短语的向量不参与算法3中任何阶段的迭代更新，仅仅是作为最后的结果输出。

4.3 相关实验

我的工作是迭代进行的，即开始进行3.2.3节中的实验时，我就利用实验的等待时间实现了4.1节与4.2节所表述的代码。其中具体的过程包括：

（1）在抽取短语对时提取层次短语频度，建立层次短语与普通短语的 ID 映射。

（2）在训练参数时依据方法的不同在层次短语对上进行参数更新。

但是由于3.2.3节中的实验效果不佳，因此大部分的时间就被用到了分析实验数据与代码调试中，我并没有在层次短语上进行相关实验。而且关于层次短语的向量表示，我的想法还比较朴素幼稚，我想利用更多时间来阅读其他文献寻找灵感，以提出一个更加符合语义关系的方法。

第 5 章　结论

本文主要在机器翻译，尤其是短语翻译系统与层次短语翻译系统的背景下，根据基于统计的短语对质量评价方法的缺点，利用递归编码器研究了语义信息对于双语短语对质量评价是否起到帮助的作用。

本文首先详细描述了非监督式递归编码器的最小错误二叉树的生成方法，并推导了反向传播算法所需优化参数的求导公式。在较小数据集中训练出单语模型后，本文利用该模型将单语短语的向量表示进行可视化，以聚类的角度验证了非监督式递归编码器捕获短语语义信息的可行性与有效性。

之后本文在非监督的基础上，依照文献 [16] 所述的方法，扩展出了半监督式的双语短语递归自动编码器，并推导了参数优化的所需公式。将双语短语递归编码器应用于短语表剪枝的实验中，保留语义相似度高的短语，剪枝相似度低的短语，将剪枝后的短语表用于短语翻译系统的训练与解码过程中，验证其对短语对语义信息评价的准确性与可靠性。

从实验结果可以看出，非监督式递归自动编码器可以较好地编码单语短语的语义信息，使得语义相近的短语在向量空间内的距离较近。但是在双语体系下，由于种种可能的原因，我的实验结果与原文[16] 的结果相差较大，这说明我所实现的双语短语递归自动编码器还存在一些问题，需要进一步的调试与优化。

文章最后一部分提出了将短语扩展到层次短语情况下层次短语向量表示的一些简单实现方法，由于双语短语递归编码器的实验情况较差，因此将其作为未来工作，希望在及时解决双语短语递归编码器问题后，再进行下一步的研究工作。

插图索引

图 1.1 Bernard Vauquois 翻译金字塔模型 2
图 2.1 短语对齐 5
图 3.1 递归自动编码器结构 9
图 3.2 非监督式递归自动编码器 11
图 3.3 中心节点不为叶节点的错误关系..... 15
图 3.4 中心节点为叶节点的错误关系 15
图 3.5 非监督式自动编码器可视化 19
图 3.6 可视化左侧放大部分 19
图 3.7 可视化右侧放大部分 20
图 3.8 双语短语递归自动编码器举例 21
图 3.9 双语短语递归自动编码器网络结构..... 22
图 3.10 中心节点不为叶节点的错误关系..... 25
图 A-1 An example of the Basic RAE 44

表格索引

表 3.1　BLUE Score for Different Phrase Tables 29

公式索引

公式 2-1 ……3
公式 2-2 ……3
公式 2-3 ……4
公式 2-4 ……5
公式 2-5 ……5
公式 2-6 ……6
公式 3-1 …… 10
公式 3-2 …… 11
公式 3-3 …… 11
公式 3-4 …… 12
公式 3-5 …… 12
公式 3-6 …… 14
公式 3-7 …… 15
公式 3-8 …… 15
公式 3-9 …… 16
公式 3-10 …… 16
公式 3-11 …… 17
公式 3-12 …… 17
公式 3-13 …… 22
公式 3-14 …… 23
公式 3-15 …… 23

公式 3-16 23
公式 3-17 24
公式 3-18 24
公式 3-19 24
公式 3-20 25
公式 3-21 26
公式 3-22 26
公式 3-23 26
公式 3-24 27
公式 4-1 31
公式 4-2 31
公式 4-3 32
公式 4-4 33

参考文献

[1] Brown P F, Cocke J, Pietra S A D, et al. A statistical approach to machine translation. COMPUTATIONAL LINGUISTICS, 1990, 16(2):79–85

[2] Brown P F, Pietra V J D, Pietra S A D, et al. The mathematics of statistical machine translation: Parameter estimation. Comput. Linguist., 1993, 19(2):263–311

[3] Koehn P, Och F J, Marcu D. Statistical phrase-based translation. Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1, Stroudsburg, PA, USA: Association for Computational Linguistics, 2003. 48–54

[4] Och F J, Ney H. The alignment template approach to statistical machine translation. Comput. Linguist., 2004, 30(4):417–449

[5] Berger A L, Pietra V J D, Pietra S A D. A maximum entropy approach to natural language processing. Comput. Linguist., 1996, 22(1):39–71

[6] Papineni K, Roukos S, Ward T, et al. Bleu: a method for automatic evaluation of machine translation. 2002. 311–318

[7] Chiang D. Hierarchical phrase-based translation. Comput. Linguist., 2007, 33(2):201–228

[8] Socher R, Pennington J, Huang E H, et al. Semi-supervised recursive autoencoders for predicting sentiment distributions. Proceedings of the Conference on Empirical Methods in Natural Language Processing, Stroudsburg, PA, USA: Association for Computational Linguistics, 2011. 151–161

[9] Socher R, Bauer J, Manning C D, et al. Parsing with compositional vector grammars. In Proceedings of the ACL conference, 2013

[10] Li P, Liu Y, Sun M. Recursive autoencoders for ITG-based translation. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, Washington, USA: Association for Computational Linguistics, 2013. 567–577

[11] Wu D. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. Comput. Linguist., 1997, 23(3):377–403

[12] Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model. J. Mach. Learn. Res., 2003, 3:1137–1155

[13] Mikolov T, Chen K, Corrado G, et al. Efficient Estimation of Word Representations in Vector Space. ArXiv e-prints, 2013.

[14] Rumelhart D E, Hinton G E, Williams R J. Neurocomputing: Foundations of research. chapter Learning Representations by Back-propagating Errors, 696–699. Cambridge, MA, USA: MIT Press, 1988: 696–699

[15] Liu D C, Nocedal J. On the limited memory bfgs method for large scale optimization. MATHEMATICAL PROGRAMMING, 1989, 45:503–528

[16] Zhang J, Liu S, Li M, et al. Bilingually-constrained phrase embeddings for machine translation. ACL, 2014

[17] Och F J. Minimum error rate training in statistical machine translation. Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1, Stroudsburg, PA, USA: Association for Computational Linguistics, 2003. 160–167

[18] Koehn P, Hoang H, Birch A, et al. Moses: Open source toolkit for statistical machine translation. Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, Stroudsburg, PA, USA: Association for Computational Linguistics, 2007. 177–180

致　谢

本次毕业设计可以算是我第一个自己独立做的研究项目，虽然没有得到最终预期的结果，但是在实验中我也学到了很多。

在此我衷心感谢刘连臣老师的指导，关心和支持。虽然他不是我直接的指导老师，但是在毕业设计的整个过程中他始终关注我的研究，并在很多方面给我提供了许多宝贵的建议，让我受益很多。另外当我在住院期间他也非常关心我，甚至亲自探望我，让我感到非常感动。可以说不管从学术还是做人，刘老师都是我的榜样。谢谢您!

我还要感谢计算机系的刘洋老师，他是我本次毕业设计的辅导老师。刘老师带的学生很多，工作也非常繁忙，但是他仍然坚持每周都和我讨论毕业设计的问题，指导我研究和实验的方法与技巧，并给我指出了许多新的思路与方向。虽然由于我水平有限没有达到预期的目标，但是正是他的帮助，才使我对自然语言处理与机器翻译产生了更大的兴趣，让我在入门基础很差的情况下得到了较快的提升。谢谢您!

此外我还要感谢实验室的同学，他们在很多细节上都给予了我很大启示与帮助，正是他们的帮助让我的实验进度得以加快。

最后我还想感谢这学期所有帮助过我，鼓励过我的老师，同学们，正是你们的帮助才让我有动力坚持下来。

声　明

本人郑重声明：所呈交的学位论文，是本人在导师指导下，独立进行研究工作所取得的成果。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。

签　名：_____ 日　期：_____

附录 A　外文资料的调研阅读报告

Trying to help computers to understand the meaning of the words, phrases and sentences is always a fascinating task for us. With the development of Machine Learning (ML) and especially in Natural Language Processing (NLP), ways of representing languages and their semantics are explored.

The term *one hot (one on)* derives from the digital circuits where the legal representations of a series of numbers have only one single high signal (1), and others are low (0). It is usually used in representing the current state in the finite state machine (FSM), and in ML the idea is borrowed where every digit is the value of one feature. This method can drastically enlarge the feature space and is successfully used in representing words and calculating the similarity among documents where every word is one-hot encoded as an entry of the feature space.

However we can soon find that this method leads to several problems:

1. reach high-dimensional space easily and cause the curse of dimensionality

2. problem of sparseness for a large vocabulary

3. entries of words are randomly set with no relations, within which we cannot find any semantic information

4. discrete feature space, hard to optimize by using gradient methods

Given such difficulties, researchers started to find out other methods to model the representations of words and phrases better.

In "Distributed representations, simple recurrent networks, and grammatical structure" Jeffrey L. Elman proposed the idea of *embedding* where words are encoded into different layers of neural networks rather than only one entry of the feature space. Therefore every entry contains a bit of information and the whole information spreads out in the space. Representations of a word in such space can be considered as the superposition or the combination of many layers of neural networks.

Since the 21st. century deep learning has aroused great attentions and "A neural probabilistic language model" was claimed to be the first paper to have applied deep

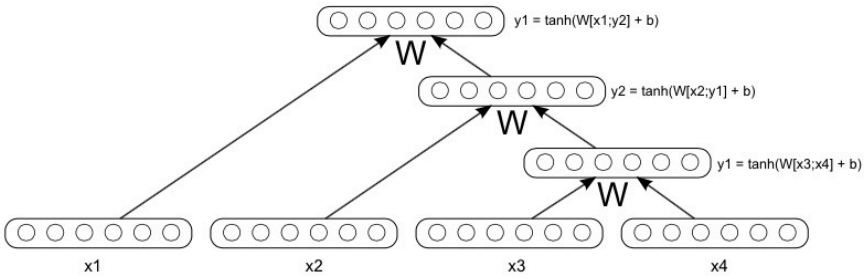

图 A-1　An example of the Basic RAE

learning methods to natural language processing. In this paper, a language model is learned by a recurrent neural network (RNN) and a vector representation for words is learned alongside. Apart from improving the language model, the representations for words can capture the semantics of the words, i.e. similar words in meaning have small distance in the space and some linearity is also maintained. Moreover, the vector space is exactly an example of distributed representation, low-dimensional, continuous and smooth, and this beautiful property is naturally bound with neural networks.

Once we have the representations of words, we can use their vectors to do some math to acquire the vectors of the structured words: phrases. And the essential problem is how we could map the phrases into the same vector space of words while encoding the semantics of the phrases.

Recursive Autoencoder (RAE) is a kind of Deep Learning neural network that can automatically capture the recursive structures of its input to apply the same recursive neural network unit to the structured input. It was firstly introduced by C. Goller et al. in "Learning task-dependent distributed representations by back propagation through structure" to represent the structured objects, and its structure highly resembles the syntactic parsing procedure in NLP where meanings of sentences are analyzed and decomposed.

In "Learning Continuous Phrase Representations and Syntactic Parsing with Recursive Neural Networks", transformations of the similar flavor with neural language model are taken where a parse tree and its phrase representations will be generated concurrently. In this paper, a greedy algorithm of generating the parse tree is proposed, and it is the fundamental forward-propagation process of RAE.

We can see from the Fig.A-1 that every time two nodes that have the minimum

(maximum) score will be encoded to one single node, and this will be performed recursively till the root node. The activation function for every neural is

$$p = tanh(W[c_1; c_2] + b)$$

and the score for every node can be defined according to the tasks and the total score can be seen as the objective function.

"Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions" proposed a hybrid of an unsupervised method and a supervised method to predict the sentiment for the phrases and sentences. The first stage is the unsupervised methods which is to calculate the reconstruction error of every node, which can learn the semantic information of phrases even more than Word2Vec (An improved implementation of the vector representation of neural language model). The second stage is the supervised methods which assigns every non-leaf node a label to represent the distributions of the sentiment, and it is realised by adding a softmax layer upon each non-leaf node.

Partially inspired by the previous work of adding prediction labels to unsupervised RAE, "Bilingually-constrained Phrase Embeddings for Machine Translation" expand the territory of RAE to machine translation.

In this paper, training data is high quality parallel phrase pairs, and the source language phrases and target language phrases are firstly modelled separately by two unsupervised RAEs. After training on both sides, a co-training algorithm is performed by adding two semantic errors on the top of the RAE tree. We could presume that there is a function transformation from source language to target language, and vice versa. And the distance between the transformed source phrase vector and the target phrase vector is the semantic error of source side, and a target side semantic error can be computed equally. During training, we could minimize an objective function that contains construction error and the semantic error, and the result will be that the phrase pairs sharing more semantics are more similar in terms of semantic error, and we could use it to either prune less similar phrases pairs or as a feature of the translation system to help evaluate the phrase pairs in a semantic view.

Besides in NLP filed, RAE is also widely used in computer vision or even con-

necting natural language with images. And in the era of the blossom of deep learning, more and more interesting network structures will flood to the NLP and MT filed in future.

A.1 References

[1] Elman J L. Distributed representations, simple recurrent networks, and grammatical structure. Machine Learning, 1991, 7:195–224

[2] Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model. J. Mach. Learn. Res., 2003, 3:1137–1155

[3] Christoph Goller, and Andreas Küchler. AR-Report, AR-95-02. Institut für Informatik, Technische Universität München, (1995)

[4] Socher R, Manning C D, Ng A Y. Learning continuous phrase representations and syntactic parsing with recursive neural networks

[5] Socher R, Pennington J, Huang E H, et al. Semi-supervised recursive autoencoders for predicting sentiment distributions. Proceedings of the Conference on Empirical Methods in Natural Language Processing, Stroudsburg, PA, USA: Association for Computational Lin- guistics, 2011. 151–161

[6] Zhang J, Liu S, Li M, et al. Bilingually-constrained phrase embeddings for machine translation. ACL, 2014