Supervised Word Embedding

sjtu

2019年9月16日

Contents

1 Introduction **3**

1.1 Background Knowledge 3

1.2 Symbol Declaration 4

2 Related Works **6**

3 Supervised Learning of the Language Model **7**

3.1 The Language Model 7

3.2 The Semantic Matrix(SM) 8

3.3 Word Embedding as Matrix Smoothing 9

3.4 Supervised Learning of the Language Model 11

4 Dealing with Uncertainty **15**

4.1 Uncertainty in the Semantic Matrix 15

4.2 Gaussian Process 16

5 Gaussian Process Reweighting **18**

5.1 Completing the Semantic Matrix with GP 18

5.2 Learning the Embedding 19
5.3 Posterior Analysis* 20
5.4 Pseudocode 21

6 Experiment **25**

6.1 Graphical Model 25
6.2 Dataset 27
6.3 Language Model Selection 27
6.4 Word Similarity 29

7 Conclusion and some Discussions **33**

7.1 Conclusion 33
7.2 Discussion on Uncertainty 33
7.3 Discussion on Dimensionality 33

8 Reference **35**

1 Introduction

1.1 Background Knowledge

本文提出了通过监督学习的方式选取语义矩阵，继而获得词向量编码的一套方法。

词向量提取是一种利用代数知识从语料库中提取出单个词汇的数字表征的算法。它一方面可以用于提取语义信息（同义词、类比等），一方面可以为后续的高级语义问题（主题分类、情感分析等）提供预处理。词向量提出的起始动机是：语义上相似的词汇应该在代数空间具有可计算的相似性。但是这种动机其实给词向量提取法则施加了一种微妙的枷锁：词汇的向量表征本来是一种非监督学习，但是这种学习的结果却不得不经历一种监督式的评判（即从人工知识构建的同义词库出发评价词向量提取算法的优劣性）。**这一矛盾的根源在于语言模型，即映射函数的选择被认为是一个经验性的、不可自动优化的步骤，本文提出了一种变分语言模型的方法，使得这一过程（指映射函数的选择）可以遵循监督学习的模式被优化，而不再是一个武断、任意的步骤。**

另外，在现在的学习模型中，计算结果的不确定性值得更多关注，因为如果能够显式地给出最终计算的不确定性，将能够对于后续的风险决策提供支撑。词向量的相关处理中其实也有这种关于不确定性的需求，以助于信息检索、敏感词测评等等。**出于建模不确定性的需求，本文设计了一整套流程以计算与词向量相关的不确定性，比如出于范数度量的同义词查找结果的不确定性等等。**

本文的主要贡献有如下三点：

1. 将词向量编码算法所依赖的映射函数选择参数化，使得这一过程不再过于依赖人工经验和形式化的预设，从而使得最优映射函数可以被监督学习的方式得出。我们继而指出语料库的统计属性对于语言模型选择的影响，并讨论了最优语言模型的存在性；

2. 基于高斯过程构造了一种更好的权重分配方案，比原来的经验权重分配方案更有理可循；

3. 基于高斯过程，写出了词向量编码的后续处理中如何显式地考虑不确定性的影响，使得

基于词向量的所有后续处理算法都可以将不确定程度作为可用信息。

1.2 Symbol Declaration

我们假设考虑的语言是分词式的，否则可以利用许多现成的方法先将被研究语言的语料库进行分词，分词后的语料库记为$\mathcal{D}$。

一个语料库是$N(\mathcal{D})$个句子的集合，一个句子是单词的序列，记第i个句子为$s(i)$，包含$l(i)$个单词，其中第j个单词记为$w(i,j)$：

$$\mathcal{D} = \left\{ s(i) = \{w(i,j)\}_{j=1}^{l(i)} \right\}_{i=1}^{N(\mathcal{D})}.$$

以后我们也用括号$w(\cdot), s(i)$表示语料库中的查询，而用下标w_i表示词汇本身的形式信息——拼写，它是自然场景中一个词汇唯一的标识符。

语境记为$c[\cdot]$，它是含有一个占位符的单词序列。

一个自然语言元素v的向量表征为$\mathrm{vec}(v) \in \mathbb{R}^M$，其中$v$可以为单词、语境甚至短语或者句子。

一种词向量算法就是这样一个过程：它利用语料库的信息，把单个语言元素，特别是词汇，映射成一个数组：

$$\text{Embedding: } \mathcal{D} \times w \to \mathrm{vec}(w)$$

评价一个词向量算法好坏的标准一般是：它能否正确发掘出语言中的同义词关系。具体地，语用者会给出一门语言中同义词对的集合，我们需要计算这些同义词对关系能否被词向量的代数相近性所反映出来。这种可计算的相似指标一般为余弦相似度：

$$\mathrm{CosSim}(w_1, w_2) = \frac{\mathrm{vec}(w_1) \cdot \mathrm{vec}(w_1)}{\sqrt{\mathrm{vec}(w_2) \cdot \mathrm{vec}(w_1)}\sqrt{\mathrm{vec}(w_2) \cdot \mathrm{vec}(w_2)}}$$

这等同于考虑归一化后由2-范数引导的度量：

$$|\frac{\text{vec}(w_1)}{\sqrt{\text{vec}(w_1)\cdot\text{vec}(w_1)}}-\frac{\text{vec}(w_2)}{\sqrt{\text{vec}(w_2)\cdot\text{vec}(w_2)}}|_2^2=2-\text{CosSim}(w_1,w_2)$$

2 Related Works

最早的关于自然语言词汇的编码思想出现在[2][3]中，可以在[1]中获得一个简单的概要。自从[3]提出了word2vec后，它引起了各界比较广泛的讨论和研究，有许多陆陆续续的工作旨在理解该模型许多武断假设背后的理论基础[4]。

早在[2][3]之前，就有许多统计语言学家就自然语言的统计问题和编码进行过尝试性的分析[5]，从现在的角度来看，这些早期的分析和较晚出现的神经网络编码模型其实是统一的，[10]是对于过去乃至以后出现的许多词语向量编码模型的最好的综述文献，其中阐明了词向量编码无非是对于某种从语料库中提取出的矩阵之近似。[9]是较晚期出现的，现今比较受认可的词向量模型，虽然时间较晚，但是本质上，它仍然完全被包含在[10]的讨论内。除了[9]之外，也出现过很多类似的尝试去利用或者讨论[3]的word2vec模型中剩余的自由度，例如[6][7][8]。

也有其他一些研究没有继续着手于编码的理论基础，而是转向后续的应用层面，譬如对于短语的编码[11]，以及其他的高层次语义应用[12][13]。现在的深度语言学模型[15]中也能看到词汇编码研究的影响，对于这一课题的基础性讨论将有助于消除我们对于深度模型理解上的困难，以及进一步减少整个模型中的人为因素。

本文的主要改进基于[9]的编码Glove，特别地，我们讨论的对象集中于[9][10]中共同指出的，词向量编码模型的几个方面：语义矩阵、映射函数、权重函数。特别地，[9]在映射函数和权重函数上做了任意性的选择并取得了不错的效果，而我们进一步形式化了这两个步骤，并指出可以以更严格的、监督学习的方式来选取模型中的这两个要素。

如果以下文的语言模型角度而言，则[9]选择以条件概率的对数为语言模型，[7]以出现次数作为语言模型，一些早期编码和以矩阵加语言模型的形式化的关联可见[9][10]。

3 Supervised Learning of the Language Model

3.1 The Language Model

处理统计语言学的问题的出发点是**语言模型的映射函数**，它的基本假设是：语言仅仅是单个词汇和该词汇所处的语境的集合。这是一种植根于结构主义思潮的理论基石。

一个语言模型对应于这样一个函数：

$$f : \mathcal{D} \times w \times c[\cdot] \to D_f$$

它从语料库$\mathcal{D}$中提取信息，并对于一个（词汇，语境）序对给出一个数值结果。其中D_f是语言模型f的值域，它随着f定义的形式而变化。一些典型的例子有：

$$f_{\text{count}}(\mathcal{D}, w, c[\cdot]) = \sum_{v \in \mathcal{D}} \mathbb{I}[v = c[w]], D_f = \mathbb{N}$$

$$f_{\text{conditional probability}}(\mathcal{D}, w, c[\cdot]) = p(v = w | c[v] \in \mathcal{D}), D_f = \mathbb{R}^+$$

$$f_{\text{Bayesian}}(\mathcal{D}, w, c[\cdot]) = \frac{p(c[w] \in \mathcal{D})}{p(c[\cdot] \in \mathcal{D}) p(w \in \mathcal{D})}, D_f = \mathbb{R}^+$$

当然还存在复合型的语言模型，它支持参数化的变分：

$$f_{\log\ [\text{type}]} = \log f_{[\text{type}]}, D_{f_{\log}} = \log D_f$$

$$f_{g([\text{type A}],[\text{type B}])} = g(f_{[\text{type A}]}, f_{[\text{type B}]}), D_{f_g} = g(D_{f_A}, D_{f_B})$$

其中$p(v \in \mathcal{D})$一律采用计数作为概率测度的近似，[type]可以是任一种语言模型对应的映射函数，特别地：

$$f_{\log \text{ Bayesian}} = f_{\text{pointwise mutual information}}$$

是[10]中介绍的早期语言模型所使用的，完全基于信息量的映射函数。而

$$f_{\log \text{conditional probability}} = f_{\text{Glove}}$$

是[9]中汇报的效果最好的映射函数。

一个语言模型的映射函数自然预设了w和$c[\cdot]$的定义域，w一般就是分词语言的词汇总集合$\mathcal{V}$，而$c[\cdot]$则可以有多种的定义，譬如n-gram系语言模型认为（语境中的词汇顺序作为备选可考察因素）：

$$c[\cdot] \in \mathcal{V}^n$$

非深度模型一般预设一个相对较小的$c[\cdot]$定义域$\mathcal{C}$，否则整个模型的计算负荷太大。

在下文中，我们可能会交替地使用语言模型和映射函数这两个词汇，它们是等价的。

3.2 The Semantic Matrix(SM)

因为w的定义域$\mathcal{V}$和$c[\cdot]$的定义域$\mathcal{C}$一般都是可数的，所以可以不失一般地将一个语言模型f记为一个矩阵$f_{\mathcal{D}}$，其分量即对应映射函数的值，即满足：

$$f_{\mathcal{D}}(i,j) = f(\mathcal{D}, w_i, c_j[\cdot])$$

我们将这个矩阵称为语言模型f对于特定语料库$\mathcal{D}$的**经验语义矩阵（Empirical Semantic Matrix）**。一个语言模型对于一门语言的**语义矩阵**定义为对于“理想的”语料库$\mathcal{D}_I$的经验语义矩阵：

$$f_{D_I}$$

在实际操作中，我们很可能需要面对流形式的语料输入，即$\mathcal{D}$的规模逐渐扩大并单向地收敛向$\mathcal{D}_I$（因为它是理想的语料库），所以我们自然也希望$f_{\mathcal{D}}$向$f_{\mathcal{D}_I}$收敛，这种收敛符合直觉的体

现方式是逐项的收敛：

$$f_{\mathcal{D}}(i,j) \to f_{\mathcal{D}_I}(i,j) \text{ when } \mathcal{D} \to \mathcal{D}_I$$

这一个收敛性的指标从某种意义上判别了语言模型的可靠性，譬如f_{count}在这种意义上就是不好的（所有项发散至无穷），继而$f_{g(\text{count})}$系列模型大都不做好（因为对于任一个单词w和语境$c[\cdot]$的组合，$f(w,c[\cdot]) \to g(\infty)$）。所以基于概率的语言学模型是更好的选择，但它们也面临一个问题：考虑这样一种对于语料库$\mathcal{D}$的扩展，把$\mathcal{D}$中的某个特定句子重复至无穷多次，这种扩展是向$\mathcal{D}_I$的扩展，但是它破坏了常理上的概率测度。当然这是一种相对比较极端的情况，在现实中语料库的在线扩展一般不会遇到这种特例，但它确实是对于概率语言模型和衍生的词向量算法的有效攻击手段。本节的末尾我们会再回到这个问题，并探讨最优语言模型理论上的存在性。

3.3 Word Embedding as Matrix Smoothing

已经有很多文献指出，词向量的本质就是语义矩阵的标准化，最单纯的标准化就是对角化即奇异值分解，将语义矩阵分解：

$$f_{\mathcal{D}} = Q\Lambda P^{\mathrm{T}}$$

其中Q和P分别是$|\mathcal{V}|^2$和$|\mathcal{C}|^2$的正交矩阵，而Λ是只有前$|\mathcal{V}|^2$分块矩阵的对角线有值的$|\mathcal{V}| * |\mathcal{C}|$矩阵（一般$|\mathcal{C}| > |\mathcal{V}|$），对角上分量按照绝对值的大小从大到小依次记为$\lambda_1, \cdots, \lambda_{|\mathcal{V}|}$。上式左侧的分量可以写成：

$$f_{\mathcal{D}}(i,j) = \sum_{k=1}^{|\mathcal{V}|} Q(i,k)\lambda_k P(j,k)$$

如果在SVD时选择对$f_{\mathcal{D}}$进行低秩近似：

$$f_{\mathcal{D}} \approx Q\Lambda_M P^{\mathrm{T}}$$

其中Λ_M只复制Λ的前M个对角元素，则：

$$f_{\mathcal{D}}(i,j) \approx \sum_{k=1}^{M} Q(i,k)\lambda_k P(j,k)$$

这就说明，如果取Q的第i行的前M列分量分别乘以λ_k作为单词w_i的词向量，取P的第j行前M列作为语境$c_j[\cdot]$的向量表征，则可以从向量近似地复现出语义矩阵的信息。

这种记法也是词向量算法评估标准的本质，考虑同义词w_1和w_2，则出于它们语义上的相似性，它们对于任何语料库、任何语境的语义模型输出应该相近：

$$\forall \mathcal{D}, \forall c[\cdot], f(\mathcal{D}, w_1, c[\cdot]) \approx f(\mathcal{D}, w_2, c[\cdot])$$

而$f(\mathcal{D}, w_i, c[\cdot]) = f_{\mathcal{D}}(w_i, c[\cdot]) \approx \mathrm{vec}(w_i) \cdot \mathrm{vec}(c[\cdot])$ 所以应该有：

$$\forall c[\cdot], \mathrm{vec}(w_1) \cdot \mathrm{vec}(c[\cdot]) \approx \mathrm{vec}(w_2) \cdot \mathrm{vec}(c[\cdot])$$

由于$\mathrm{vec}(c[\cdot])$可以看做正交向量的前M个分量，其范数有界，所以柯西不等式指出$\mathrm{vec}(w_1) \approx \mathrm{vec}(w_2)$是上式的充分不必要条件。即是说，前一章节中所述的词向量算法评估标准中的同义词向量相似性是语言模型的等价性、即语义矩阵行的相似性的条件的强化。

必要性的考察涉及P的前M列的秩，或者说它的特征值谱的性质。

由于$|\mathcal{V}|$的值往往会很巨大，而$|\mathcal{C}|$是$|\mathcal{V}|$的指数，所以直接使用代数方法分解矩阵$f_{\mathcal{D}}$很困难，何况还可能有数值不稳定的现象。所以一般可以转而考虑随机矩阵分解方法。由于SVD的M秩近似可以写成这样一个最优化任务：

$$Q', P' = \arg\min \mathcal{L}(Q', P') = \arg\min \sum_{i,j} (f_{\mathcal{D}}(i,j) - Q'_i \cdot P'_j)^2$$

那么其实可以直接写出：

$$\frac{\partial \mathcal{L}}{\partial \mathrm{vec}(w_i)_k} = 2\sum_j (f_{\mathcal{D}}(i,j) - Q_i' \cdot P_j')\mathrm{vec}(c[\cdot]_j)_k$$

对于对偶的$\frac{\partial \mathcal{L}}{\partial \mathrm{vec}(c[\cdot])_k}$有完全对称的结果。所以向量编码可以不使用SVD，而使用迭代的最小二乘法解出，因为损失函数是二次型，所以最终一定能收敛到唯一的全局最优解。在实际的操作中，由于遍历i,j的过程太冗长，且其中可能包含一些未被良好定义的数值（譬如$\log 0$），所以可以只对于$f_{\mathcal{D}}$中的部分分量进行求和，一般只对于出现在语料库中的$(w, c[\cdot])$对求和。

word2vec中强调的提升整体编码效果的negative-sampling的本质就是将求和式的集合——语料库中出现的$(w, c[\cdot])$对集合扩展一些未出现的对，这些对的语义矩阵对应值被统一设置为某个先验量。

很容易看出词向量编码和推荐系统的相似性，譬如电影推荐系统处理的矩阵就是由用户空间和电影空间构成的张量积，所以类似的技术如随机矩阵分解和其他协同过滤算法都可以直接套用在词向量编码问题中。

Remark: 但需要注意到，两个词是“同义词”这个概念本身能多大程度上由两个词对应的语义矩阵行相似？这完全取决于语言模型的选取是否把握住了同义词间的特征。如果一种语言模型本身就无法捕捉同义词的这种代数相似性（举一个极端的例子，f是常函数），那么后续的矩阵分解也只会让这种反推的相似性变得更模糊。然而过去的工作一般把语言模型的选择作为一种经验性的步骤，即通过试错式的选择来进行模型性能的优化，这也是因为同义词的度量标准本身比较模糊，而误差的反向传播也未被系统地研究过，导致词向量模型变成一种：非监督训练、监督式验证的畸形状态。

3.4 Supervised Learning of the Language Model

本文的第一个目标就是提出语言模型的选择方法，用监督学习的方式优化语言模型。形式上，我们选取一种参数化的语言模型f_θ，并通过构造一种损失对于θ可微的、和同义词相关

的损失函数，继而选取出最优的θ取值$\hat{\theta}$，并认$f_{\hat{\theta}}$为当前变分约束下最优的语言学模型。

根据之前的讨论，我们选取这样一种变分形式：

$$f_{\theta=\{a,b\}} = f_{\text{c.p}} + a \cdot f_{\text{c.p}}^2 + b \cdot f_{\text{c.p}}^3, \text{ c.p=conditional probability}$$

即变分的备选语言模型空间是条件概率的二次多项式。进行这种选择的原因有三：

1. 条件概率本身已经被发现是带有较多语义信息的语言模型，但是因为其数值的分布形如指数分布[10]，所以对其采用2-范数有时会导致性能的下降[10]，如果我们想要固定余弦相似度的比较方法，即固定2-范数为度量函数，则必须考虑条件概率的变分形式，如[9]就取$f_{\text{Glove}} = f_{\log \text{c.p}}$；
2. 因为$D_{\text{c.p}} = [0, 1]$，对该变分在0处不失一般地进行泰勒展开，有理由舍去高阶项，保留相对低阶的项即可；
3. 这样的变分形式使得θ的最优化可以约简为一个线性回归问题，而不需要再诉诸复杂的梯度传播（否则，考虑一般的$f = g(\theta, f_{\text{c.p}})$，求解$\theta$的工作将变得复杂而且缺乏直观）。

我们预先调节了语言模型的整体放缩以使得语义矩阵中所有的列都近似归一化。输入的训练数据应该由正样本即同义词对，负样本即无关词对组成，我们从TOEFL同义词列表中随机选取了664个同义词对，再从出现频率前10000的词汇中随机抽取了664个词对，认为它们是无关词对。在当前形式变分的语言学模型下，两个词汇的余弦相似度应为（假设都归一化，这可以通过将整个语义矩阵乘以常数来近似地满足）：

$$\begin{aligned}
\text{Sim}(w_1, w_2) =& \sum_{c[\cdot]} f(w_1, c[\cdot]) \cdot f(w_c, c[\cdot]) \\
\approx& \sum_{c[\cdot]} f_{\text{c.p}}(w_1, c[\cdot]) \cdot f_{\text{c.p}}(w_1, c[\cdot]) + a[f_{\text{c.p}}^2(w_1, c[\cdot]) f_{\text{c.p}}(w_2, c[\cdot]) + f_{\text{c.p}}(w_1, c[\cdot]) f_{\text{c.p}}^2(w_2, c[\cdot])] \\
&+ b[f_{\text{c.p}}^3(w_1, c[\cdot]) f_{\text{c.p}}(w_2, c[\cdot]) + f_{\text{c.p}}(w_1, c[\cdot]) f_{\text{c.p}}^3(w_2, c[\cdot])] \\
=& x_{1,w_1,w_2} + a \cdot x_{2,w_1,w_2} + b \cdot x_{3,w_1,w_2}
\end{aligned}$$

其中我们略去了阶数高于4的3项因为受$D_{\text{c.p}}$的性质它们取值较小，再略去a, b的交错项以简化计算。对于所有的词对w_1, w_2，我们首先从$f_{\text{c.p}}$的语义矩阵中计算出$x_{i,w_1,w_2}, i = 1, 2, 3$存入设计矩阵$\mathbf{X}$，再根据该词对是否属于TOEFL的同义词对来指定回归结果为1（语义同义词）或者0（无关词）存入回归目标$\mathbf{y}$，最后，我们对这一线性回归任务进行常规求解：

$$(a, b)\mathbf{X} = \mathbf{y}$$

其中X是如上述由同义词对生成的$2 * 1328$的矩阵，$\mathbf{y}$是$1 * 1328$的矩阵，则解得：

$$(a, b) = (\mathbf{X}\mathbf{X}^{\mathrm{T}})^{-1}\mathbf{y}\mathbf{X}^{\mathrm{T}}$$

应该注意到，我们并不能证明监督学习所得的一定是最优的语言模型，但它是在当前变分形式以及当前语料库下是最似然（即对于训练集合来说）最优的。使用其他变分形式或者其他语料库有可能导出效果更好的模型，如果使用神经网络可以得到最一般的结果，但是会面临训练样本以及输入信息不足的问题。

综上所述，给定同义词/无关词库以后，可以以监督的方式针对一个语料库选择一个最好的语言模型。在以往的工作中，语料库和语言模型被认为是独立的分量，因为前者往往是客观存在的，所以优化往往只是对于语言模型的离散化、试错化调优。但是现在我们指出语言模型应该是语料库（特别地，是语料库统计量）的函数，那么就有可能解决之前工作中常常面临的“语料库较小、或者有领域相关偏颇，导致性能不好”的问题，从而可能**从一个很小的语料库中、利用最合适它的语言模型，得到和其他语言模型在大型语料库中引导的词向量编码性能相似的编码。**

现在我们再回到这个议题上：是否存在一个最优的语言模型，以至于对于语料库的任何拓展（即增加某些合法的句子），其最优性（w.r.t.如上的监督学习任务）都不改变？

我们认为不存在这样一个模型，考虑这样一种扩展，它只重复语料库中已有的某个句子多次并将它们添加入语料库中，这种扩展只会改变回归模型中设计矩阵$\mathbf{X}$的部分分量，而通

过筛选训练集合的元素，就可以导致这种改变被反映在最终解出的权值上。换言之，对于任何一个语料库导出的最优权值（即当前的最优语言模型），都可以通过扩展语料库本身来将其改变，所以不存在最优的语言模型。

对于这种恶意的扩展而言，条件概率本身已经不再能详尽地表示语义信息，但是对于平凡、典型的大型语料库而言，即其满足：本身足够大，以消除非典型模式的影响；无恶意扩展两个条件时，从中训练出的语言模型可以被认为几乎是最优的。

4 Dealing with Uncertainty

4.1 Uncertainty in the Semantic Matrix

在协同过滤中一个常规任务是：给定一个矩阵的部分元素，要求对某些未给出的元素进行估计。譬如电影推荐系统会根据某个用户对老电影的评价，以及其他用户对老电影和新电影的评分，来估算该用户对于某部新电影的评分，从而做出营销策略的决策。在语义矩阵中我们面临着同样的问题，即我们需要对于语义矩阵的某些分量计算不确定性，它的效果是：

1. 迭代地补全语义矩阵的分量，使得线上学习（online learning）成为可能，具体地，我们期望当输入语料慢慢扩大时，语义矩阵分量上的方差逐渐减小，即语义矩阵每个分量都随着语料的输入经历一个贝叶斯估计的过程；
2. 将最优化的损失函数一般化为一个加权的和，就像一些工作中所指出的，使用加权和可以获得更好的向量表达效果；
3. 如果能够从向量信息中估计出语义矩阵分量的不确定性，那么就可以将不确定性信息代入之后的处理环节，从而提供一种模糊的后续处理步骤，在一些有关拒绝的任务中有可能有效果。

这一权重函数可以用于消除语义矩阵中不良项对于矩阵分解的干扰：比方说，对于一个极小的条件概率，它的生成可能是由于语料库的不充分，所以在回归时我们理应基于它一个较小的权重，即该分量对应的高斯分布具有一个较大的方差。譬如[9]中以类似relu的方式进行了各个分量的权重制衡，输入信息为该分量对应对的出现次数（而非频率）。

给语义矩阵中元素引入不确定性的天然方案是使用高斯过程。具体而言，当矩阵中某个元素$f_{\mathcal{D}}(i,j)$的值可以被这样估计：将语义矩阵的第i行建模为一个高斯过程，而分量间的协方差矩阵利用核方法构造，此行中对应语境k和l的两个元素的核函数可以取为语义矩阵的第k列和第l列之内积，这种内积的构造显然符合协方差矩阵要求的半正定性，同时从理念上语义矩阵的列也是语境的一种特征表示法。在这种取内积过程中可能还会遭遇到未知分量，此时可

以引入先验的方差，并通过迭代多次来获得稳定的估计。

应该注意到出于单词和语境的对偶性，也可以将语义矩阵的第j列作为高斯过程。

在回顾和统一一下高斯过程的记号之后，我们再下一章会详细地给出高斯过程词向量的计算框架。

Remark:应该注意到，[9]中的权重函数对于巨大语料库是无效的，因为它基于$(w, c[\cdot])$的出现次数而不是频率，当语料库扩展到一定大小时，它会变得无意义。而GP的优点在于：它和语料库的扩展无关，某个分量的不确定性仅取决于该分量对应的单词是否在语料库中有极相似的词汇（从语义矩阵行作为向量意义上的相似），如果有和它相似的词汇，那么它的分量不确定性就较小，否则，如果一个词汇不和语料库中的任何词汇相似，则其分量会具有较大的不确定性。

4.2 Gaussian Process

高斯过程是一种单值回归模型，它的概率图是：

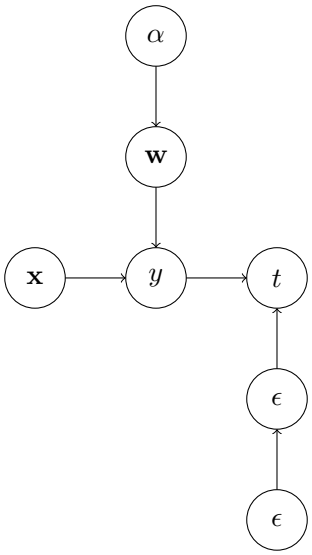

Figure 1: The PGM of GP.

其中$y = \mathbf{w}^{\mathrm{T}}\mathbf{x}$，$t = y + \epsilon$，而且

$$\mathbf{w} \sim \mathcal{N}(0, \alpha^{-1}\mathbf{I}),$$

$$\epsilon \sim \mathcal{N}(0, \beta^{-1})$$

对于$\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$对应的$\mathbf{y} = \mathbf{X}^{\mathrm{T}}\mathbf{w}$，有：

$$\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \alpha^{-1}\mathbf{K}), \mathbf{K}_{n,m} = \alpha^{-1}\mathbf{x}_n^{\mathrm{T}}\mathbf{x}_m$$

所以：

$$\mathbf{t} \sim \mathcal{N}(\mathbf{0}, \mathbf{C} = \mathbf{K} + \beta^{-1}\mathbf{I})$$

对于某个新的输入$\mathbf{x}_{N+1}$，将联合高斯分布做贝叶斯处理可得：

$$t_{N+1} \sim \mathcal{N}(m, \sigma^2)$$

$$m = \mathbf{k}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{t}$$

$$\sigma^2 = c - \mathbf{k}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{k}$$

其中：

$$\mathbf{k} = \alpha^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{x}_{N+1}, k_i = \alpha^{-1}\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_{N+1}$$

$$c = \alpha^{-1}\mathbf{x}_{N+1}^{\mathrm{T}}\mathbf{x}_{N+1} + \beta^{-1}$$

其中所有的内积可以用满足Mercer定理的核函数替代。

5 Gaussian Process Reweighting

现在我们开始具体地说明如何利用高斯过程来给语义矩阵引入不确定性，对比的出发点是word2vec中负采样的技巧和Glove中的损失函数对于语义矩阵分量的加权和。

5.1 Completing the Semantic Matrix with GP

选定一个语言模型，并对于出现在$\mathcal{D}$中的对$(w, c[\cdot])$计算相应的语义矩阵分量，对于未出现的分量，以N/A标记语义矩阵中的相应分量。

我们的初衷是：不同的N/A分量可以被区别对待，为了估计一个分量$f_{\mathcal{D}}(i,j)$的值与方差$m(i,j), \sigma^2(i,j)$，我们把语义矩阵的第j列当做一个单值回归来处理，即对于该列中的非N/A元素$f_{\mathcal{D}}(i',j)$：

$$\mathbf{x}_{i'} \to w_{i'}$$

$$t_{i'} \to f_{\mathcal{D}}(i',j)$$

给定α, β后，我们只需要设法写出此时的核函数：

$$k(w_{i'}, w_{i''})$$

因为核函数的本质是第i'和第i''个数据项在某个特征空间中的表示的内积，而语义矩阵本身就是一个特征空间，所以可以令：

$$k(w_{i'}, w_{i''}) = \sum_{c[\cdot]_{j'}} f_{\mathcal{D}}(i',j') f_{\mathcal{D}}(i'',j')$$

在这个求和的过程中，可能会遇到$f_{\mathcal{D}}(i',j')$为N/A的问题，为了避免相互依赖性的耦合，我们先采样：

$$f_{\mathcal{D}}(i',j') \sim \mathcal{N}(0, \gamma^{-1})$$

如果$f_{\mathcal{D}}(i',j')$已经被估计过，那么就以估计后的数值采样：

$$f_{\mathcal{D}}(i',j') \sim \mathcal{N}(m(i',j'),\sigma^2(i',j'))$$

整个过程需要被重复多次以消除随机性的影响和初始化的影响。这是用GP补全语义矩阵的流程，在实验中还有一些可利用的技巧来减少计算量：

1. 没有必要对于所有的N/A项进行补全，只随机地取一些分量补全即可，word2vec也只是进行了部分分量的盲猜式补全；

2. 计算的主要性能瓶颈在于：对每一个分量而言，均需要求一个很大矩阵的逆。为了提高效率，可以做如下简化：

   选取一个“常用词汇”构成的固定典型集合，它们应该满足：对于大部分语境，取值不为N/A；出现频次相对较高。对于第j列作为一个单值回归，我们不再以所有取值非N/A的分量作为证据，而只以典型集合引导的分量集合作为证据，那么补全所有的N/A分量只需要逆一次矩阵即可。

5.2 Learning the Embedding

在已经获得了诸多原本为N/A分量的语义矩阵项的后验均值和方差以后（记截止目前非N/A元素之集合为$\mathcal{E}$，在实验中，我们取$\mathcal{E}$为语义矩阵的前10000列），写出改良过的损失函数：

$$\mathcal{L}(\mathrm{vec}(w),\mathrm{vec}(c[\cdot])) = \sum_{(w_i,c[\cdot]_j)\in\mathcal{A}} \frac{(m(i,j)-\mathrm{vec}(w_i)^{\mathrm{T}}\mathrm{vec}(c[\cdot]_j))^2}{\sigma^2(i,j)}$$

它是一个加权的最小二乘法最优化目标函数，和当前最新的Glove编码方式相比，权重$\frac{1}{\sigma^2(i,j)}$是一个更加智能的选取方案。（[9]G中love的权重是$w_i, c_j[\cdot]$出现次数的类似relu的变换，一个经验性的取值）

5.3 Posterior Analysis*

使用GP并将语义矩阵的行作为单词的特征向量允许我们从向量表示直接计算出语义矩阵某分量的方差，只需要代入对于典型集合中的元素：

$$f_{\mathcal{D}}(w, c[\cdot]) \approx \mathrm{vec}(w)^{\mathrm{T}}\mathrm{vec}(c[\cdot])$$

对于非典型集合中的元素，利用：

$$k(w_{i'}, w_{i''}) \approx \mathrm{vec}(w_{i'})^{\mathrm{T}} \left(\sum_{c[\cdot]} \mathrm{vec}(c[\cdot])\mathrm{vec}(c[\cdot])^{\mathrm{T}} \right) \mathrm{vec}(w_{i'})$$

可以近似推理为：

$$m(w, c[\cdot]) = \alpha^{-1}\mathrm{vec}^{\mathrm{T}}(w)\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{X}(\alpha^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{X} + \beta^{-1}\mathbf{I})^{-1}\mathbf{X}^{\mathrm{T}}\mathrm{vec}(c[\cdot]) \approx \mathrm{vec}^{\mathrm{T}}(w)\mathrm{vec}(c[\cdot])$$

其中采取近似β远大于α，此时可以统一非典型集合中的元素和典型集合中元素的均值之近似的表达式。

对于方差的近似，代入GP的方差推断，并将上述的内积形式代入，易得：

$$\sigma^2(w, c[\cdot]) = \alpha^{-1}\mathrm{vec}^{\mathrm{T}}(w)\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathrm{vec}(w) + \beta^{-1} - \alpha^{-1}\mathrm{vec}^{\mathrm{T}}(w)\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{X}(\alpha^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{X} + \beta^{-1}\mathbf{I})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}}\mathbf{Y}\mathrm{vec}^{\mathrm{T}}(w)$$

这一式在β远大于α的近似下（即在求逆处舍去第二项）退化为β^{-1}，如果想要获得和w有关的方差估计，可采取近似：

$$(A + B)^{-1} = A^{-1}(1 + \frac{B}{A})^{-1} \approx A^{-1} - \frac{B}{A^2}$$

得到：

$$\sigma^2(w, c[\cdot]) \approx \beta^{-1}(1 + \mathrm{vec}^{\mathrm{T}}(w)(\mathbf{X}\mathbf{X}^{\mathrm{T}})^{-1}\mathrm{vec}(w))$$

其中

$$\mathbf{X} = (\mathrm{vec}(w_1), \cdots, \mathrm{vec}(w_T))$$

是典型集合的词向量集合。类似地$\mathbf{Y}$是语境列向量集合出的设计矩阵。

这种后验误差的定量化是提供进一步的模糊模型可行性的基础。特别是在一些场合下，我们需要诸项地比对两个单词在各个语境中的语言模型数值时，此时我们需要语境的向量以及误差的计算以辅助决策。

5.4 Pseudocode

我们把现在的模型提案总结为如下的过程，规定如下记号：

语料库$\mathcal{D}$；语言模型f；单个词汇w；单个语境$c[\cdot]$；向量表示$\mathrm{vec}(\cdot)$；随机选取的经验语义矩阵的分量集合（其中可以包含N/A分量）$\mathcal{E}$；典型的词汇集合$\mathcal{W}$。

额外模型参数：α, β, γ是随机过程中的参数，τ是语料库采样中的参数。

1. 利用当前语料，计算条件概率经验语义矩阵：

$$\forall w, c[\cdot], f_{\mathcal{D},\mathrm{c.p}}(w, c[\cdot]) = p(w|c[\cdot]).$$

2. 计算出3.4节的设计矩阵和回归目标，并求解出参数，以得当前最优语言模型f；

3. 计算当前最优语言模型对应的经验语义矩阵：

$$\forall w, c[\cdot], f_{\mathcal{D}}(w, c[\cdot]) = f(\mathcal{D}, w, c[\cdot]).$$

对于出现次数少于某个阈值τ的$w, c[\cdot]$对，$f_{\mathcal{D}}(w, c[\cdot])$的值设置为N/A。

4. 选取一些词汇作为典型集合$\mathcal{W}$，它们应该是出现频次较高的词汇，总的数量约为500-1000。

5. 随机选取$f_{\mathcal{D}}$的部分分量，进入集合$\mathcal{E}$，对于一组$(w, c[\cdot]) \in \mathcal{E}$且$f_{\mathcal{D}}(w, c[\cdot]) = \mathrm{N/A}$：

   将经验语义矩阵中的这些部分视作一个高斯过程：$c[\cdot]$所对应的一列中关联于$\mathcal{W}$的分量。

$$m(w, c[\cdot]) = \mathbf{k}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{t}$$

$$\sigma^2(w, c[\cdot]) = c - \mathbf{k}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{k}$$

   具体地，$\mathbf{k}$，$\mathbf{t}$均为$|\mathcal{W}|$维的列向量，$\mathbf{C}$为$|\mathcal{W}|^2$的矩阵：

$$\mathbf{k}_i = \mathrm{kernel}(\mathcal{W}_i, w)$$

$$\mathbf{t}_i = f_{\mathcal{D}}(\mathcal{W}_i, c[\cdot])$$

$$\mathbf{C}_{i,j} = \alpha^{-1}\mathrm{kernel}(\mathcal{W}_i, \mathcal{W}_j) + \beta^{-1}\delta_i^j$$

$$c = \mathrm{kernel}(w, w)$$

   其中：

$$\mathrm{kernel}(w_1, w_2) = \sum_{c[\cdot]} f_{\mathcal{D}}(w_1, c[\cdot]) f_{\mathcal{D}}(w_2, c[\cdot])$$

   上述计算中如果需要对原本为N/A的分量进行取值，则分情况：(i)如果已经计算出了它的后验均值和方差，则按照该值采样；(ii)否则按照0均值，γ为方差的正态分布采样。

6. 重复上一步骤多次，消除初始化和采样中的随机性影响。

7. 至此，已经获得了所有$\mathcal{E}$元素的均值和方差，总的损失函数是：

$$\mathcal{L}(\mathrm{vec}(w), \mathrm{vec}(c[\cdot])) = \sum_{(w_i, c[\cdot]_j) \in \mathcal{E}} \frac{(m(i,j) - \mathrm{vec}(w_i)^{\mathrm{T}}\mathrm{vec}(c[\cdot]_j))^2}{\sigma^2(i,j)}$$

先对$\mathrm{vec}(w)$和$\mathrm{vec}(c[\cdot])$随机初始化，然后迭代：

$$\mathrm{vec}(w_i)_k- = \lambda \left(\sum_{(w_i,c[\cdot]_j)\in\mathcal{E}} \frac{2}{\sigma^2(i,j)} [m(i,j) - \mathrm{vec}(w_i)^{\mathrm{T}}\mathrm{vec}(c_j[\cdot])](-\mathrm{vec}(c_j[\cdot])_k) \right)$$

$$\mathrm{vec}(c_j)_k- = \lambda \left(\sum_{(w_i,c[\cdot]_j)\in\mathcal{E}} \frac{2}{\sigma^2(i,j)} [m(i,j) - \mathrm{vec}(w_i)^{\mathrm{T}}\mathrm{vec}(c_j[\cdot])](-\mathrm{vec}(w_i)_k) \right)$$

直到收敛为止（既可以使用固定学习速率的梯度下降，也可以使用最速梯度下降，因为损失函数是编码中每一个分量的二次型）就获得了词汇和语境的编码，它是经验语义矩阵中$\mathcal{E}$分量集合的近似。

8. 现在可以删除内存中关于语义矩阵的信息，对于一个查询对$(w, c[\cdot])$，它的均值和方差由向量后验地计算近似为：

$$m(w, c[\cdot]) \approx \mathrm{vec}(w)^{\mathrm{T}}\mathrm{vec}(c[\cdot])$$

$$\sigma^2(w, c[\cdot]) = \mathrm{vec}(w)^{\mathrm{T}}[\mathbf{C} - \mathbf{C}^{\mathrm{T}}\mathbf{X}(\alpha^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{X} + \beta^{-1}\mathbf{I})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{C}]\mathrm{vec}(w)$$

其中$\mathbf{X}$的各列为各个单词的词向量，而

$$\mathbf{C} = \sum_{c'[\cdot]} \mathrm{vec}(c'[\cdot])\mathrm{vec}(c'[\cdot])^{\mathrm{T}}$$

是一个二阶张量。应该注意到$\sigma^2(w, c[\cdot])$是和$c[\cdot]$无关，只和w有关的量。换言之，现在的词向量本身之中包含了关于单个词汇的编码是否足够可靠的信息。

Remark:对于同义词比较任务而言，我们更严谨地写出两个高维高斯分布的KL散度之和作为一种对称的度量：

$$\mathrm{distance}(w_A, w_B|c[\cdot]) = \frac{\sigma_A^4 + \sigma_B^4 + (\sigma_A^2 + \sigma_B^2)(m(w_A, c[\cdot]) - m(w_B, c[\cdot]))^2}{\sigma_A^2\sigma_B^2}$$

约简成：

$$|\mathcal{C}|\left((\frac{\sigma_A^2}{\sigma_B^2})^2+(\frac{\sigma_B^2}{\sigma_A^2})^2\right)+(\frac{1}{\sigma_A^2}+\frac{1}{\sigma_B^2})(\mathrm{vec}(w_A)-\mathrm{vec}(w_B))^{\mathrm{T}}\mathbf{C}(\mathrm{vec}(w_A)-\mathrm{vec}(w_B))$$

6 Experiment

6.1 Graphical Model

传统的词向量模型编码方案的图模型如下，其中绿色的部分是输入的部分，红色的部分是武断的，即需要人工经验提供的部分：

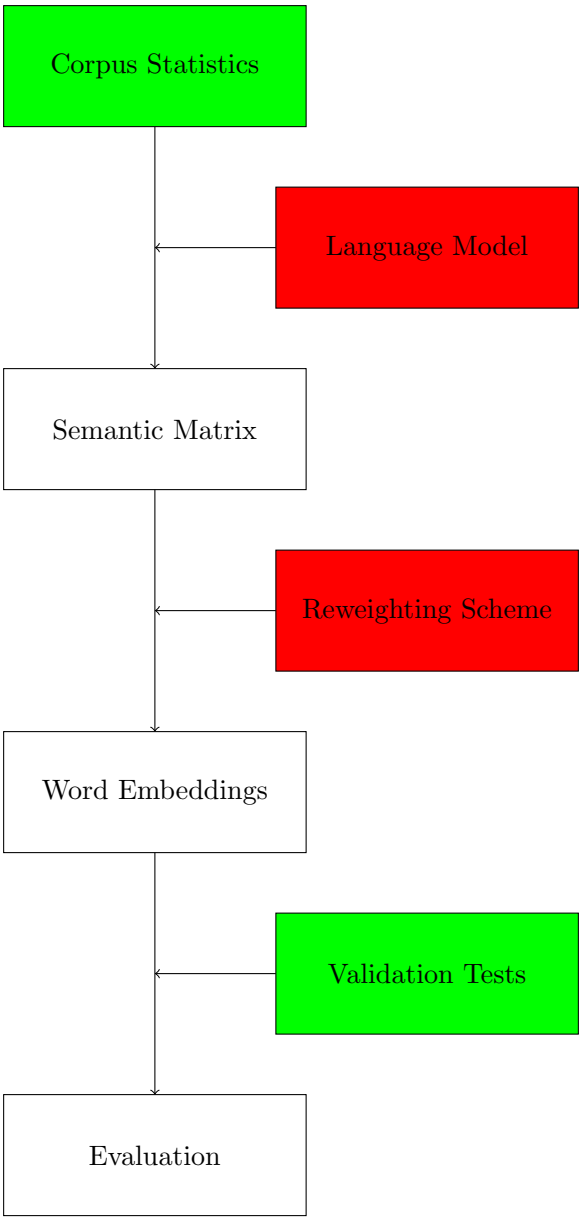

Figure 2: The PGM of ordinary word embeddings.

本文中监督词向量编码的图如下所示，关键改进在于语言模型现在是监督学习的输出，而且权重再分配算法使用了GP，其中语言模型上淡绿色，以表示它很大程度上是和输入相关的：

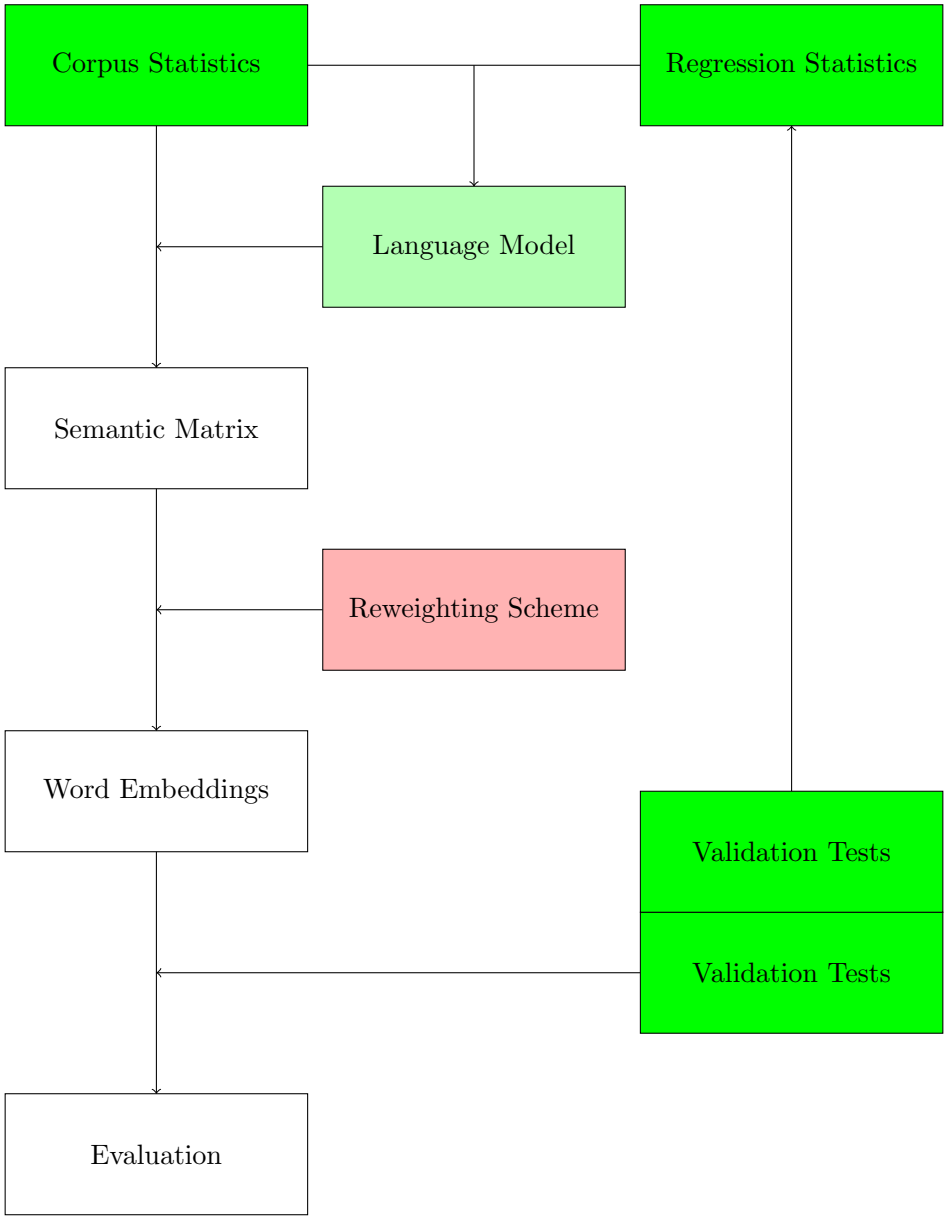

Figure 3: The PGM of supervised word embeddings.

6.2 Dataset

我们分别取用了维基百科中的两端随机语料作为训练输入，token数量分别为120M和6B。

在语境的定义上面，我们分别考虑了以下几种情况：

1. 出现在某个单词前后两个位置的单个单词;

2. 围绕一个单词的前后两个单词的集合；

3. 一个单词的前方两个或者后方两个的集合；

但是经过对比发现，如果语境作为多个单词的集合，则（经验）语义矩阵会过于稀疏。就算采取第一种最宽泛的定义方法，语义矩阵中非零的元素占比也只有0.245%（对于第二种语境的第一方法，非零分量占比1.8e-6%）。所以我们最终选取单个词汇作为语境，这和当前最优算法Glove的选取方法是一样的。

进一步地，我们试图捕获语料库中的整体信息，而不仅仅是片段信息。具体而言，基于片段信息的编码方法（如word2vec）试图最大化每个局部语段的生成概率，而基于全局信息的编码方法并不求局部语段生成概率的最大化，而试图逼近语料库中语段的统计概率。

举例而言，如果语料库中出现一个片段$abcd$，基于片段信息的编码会试图使得参数化的生成概率$p(d|abc)$等等贴近于一，但是基于全局信息的编码会首先统计d作为一个单词出现在语境abc中的概率，再使得参数化的生成概率贴近这个统计出的数值，而不是一直向1回归。

既然已经选定语境为单词，那么我们就首先统计出每个单词w_j作为语境时，每个单词w_i的出现次数，记录其为$X_{i,j}$，这一频数统计是我们的第一个语义矩阵：计数语义矩阵。

6.3 Language Model Selection

[数据集可见regression.xlsx]代入语料库中的数据导出的设计矩阵和回归目标得到：

$$(a, b) = (-0.540893, 0.333722)$$

这一组系数引导的变换函数在[0,1]区间内和对数函数$\log(1+x)$的泰勒展开[-0.5,0.333]很相近，区别于[9]中的$\log(x)$。$\log(1+x)$，平方根函数（经验告诉我们它们也很类似）和我们回归出的多项式在$D_{\text{c.p}}$上的结果如下图：

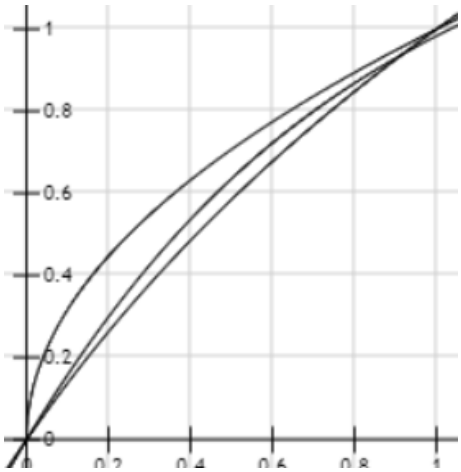

Figure 4: 从上到下分别是平方根函数$\sqrt{x}$，回归多项式，$\log(1+x)$

在Glove[9]中，用一种看似principled的方法提出先统计条件概率$p(w|c)$，再使用对数函数进行变换，再以最小二乘法作为优化方案。但是其中的推理基于文中Eq(1)的形式，虽然后面的步步推理看起来都很不容置疑，但是Eq(1)本身是一个武断的假设。

一个好的语言模型及映射函数F应该满足以下两个条件：

1. 如果两个词汇语义上很相似，那么他们对应的变换后的语义矩阵行作为向量应该也很相似，这第二种相似必须基于某种度量函数；

2. F的变换应该使得上述的度量函数拥有较好的物理意义。

考虑以2-范数作为向量的度量，则此时需要这样的一个F，经它映射之后矩阵分量的数值应该形成近似正态分布，因为2-范数引导的最小二乘法恰恰是正态分布对应的最大似然估计。

那么对数函数$F=\log$能否将条件概率映射为一个符合正态分布的分布呢？遗憾的是，实验证明经过对数变换的条件概率距离一个正态分布相差甚远。可以继而观察到条件概率更贴近一个指数分布，而对于我们利用回归求出的语言模型以及平方根变换，容易看出来，它们

都能比对数函数形成更好的变换结果，即更相似于一个高斯分布。图中橙色为对数变换，蓝色为平方根变换，灰色为变分最优（$\approx \log(1+x)$）的变换。

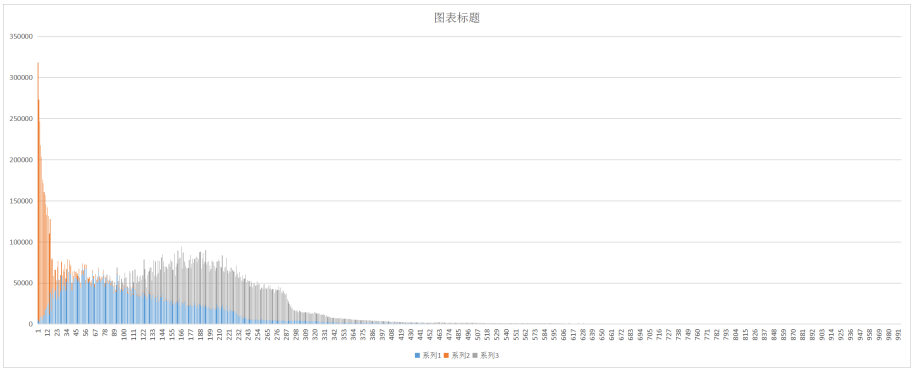

Figure 5: 三种变换后语义矩阵中元素的分布（在120M语料库中）

应该谨记词向量引导的同义词查询恰恰基于向量，继而是语义矩阵中行的2-范数度量，而为了使得这种度量本身足够合理，必须实证回归的目标拥有一个近似正态的分布。Glove中的方案从这种角度而看是缺乏合理性的，形式上这种缺乏性在于它用正态分布对一个明显不是正态分布的随机变量进行了拟合。而使用变分回归所得的变换使得后续的似然化、即最小二乘法符合理性，因为我们确实是在向一个满足正态分布的变量进行回归[10]。

6.4 Word Similarity

在词向量的对比中，相似性一直是最主要的议题。因为文本的主要探讨问题是：能够通过变分语言模型的方式，使得我们可以针对某个语料库求出最优的语言模型，其导出的词向量性能较好。所以我们对比了[9]中state-of-the-art的词向量模型（100维，训练在60亿token的语料库上），和两个在我们的小语料库上训练的词向量（100维，1.2亿token，前者语料库大小是其50倍），训练方法分别是：[9]中的Glove模型（其在一系列经验任务中已被证明优于其他大多数向量编码模型）和我们的监督模型，GP权重配分作为备选项。而我们想要证明的是：虽然语料库要小很多很多，以至于通用的编码策略可能无法得出很好的效果，但是如果使用了监督策略以攫取最适合的语言模型的话，仍旧可以获得很良好的效果。

我们进行了两组验证：一是在Sim353中进行了三种编码的余弦相似度比较，Sim353内涵

Model	Tokens	Spearman Rank Coefficient
Conditional probability	120M	0.366
Square conditional probability	120M	0.533
Glove	120M	0.530
Variational optimal model	120M	0.575
Variational optimal model + GP	120M	0.625
Square conditional probability	6B	0.565
Glove	6B	0.658
Variational optimal model	6B	0.657
Variational optimal model + GP	6B	0.695

Table 1: 不同语言模型在同义词比对任务中的性能比较

了353个词对，以及人工标注的，每对词的相似程度，处于[0,1]之间，而词汇编码之间的余弦相似度和人工经验相似度的相关性是词向量模型的惯用对比方法之一。

上表是Sim353数据集合上的实验结果，右栏的输出是Spearman秩系数，它度量了Sim353中给出的词对相关性序列和从编码结果中复原的相关性序列的相似性，其值越大，说明二序列的相关性越高，即编码的效果越好。其值为零时，二序列几乎无关。可以看出，在120M的较小语料库上，[9]中提出的变换逊于变分所得的模型，但是120M语料库使用变分最优的语言模型则可以获得好得多的效果，并且向在6B的语料库中得到的编码性能接近。另一方面，使用GP进行权重的重新分配几乎总能够获得更优的性能。

另一项对比任务是进行在重新挑选（不同于监督训练的材料）的200个同义词对上，此时我们不仅仅希望对于每一对单词的余弦相似度接近于1，更希望在余弦相似度导出的一个单词的近义词队列中，测试数据给出的同义词能够处在靠前的位置。这种新的测试标准检验了编码的全局性能（将所有的词编成相同的编码就能实现在同义词集合上余弦相似度的完美拟合，但这是不良的编码，反观相似性排序就没有这一问题）。具体而言，对于每一个同义词对，我们记录下两个单词在对方的同义词队列中的序位的平均数，一个单词的同义词队列由所有单词与它的语义矩阵行之差的2-范数排序生成，因为每一个词对都是人工认证的同义词，我们自然希望它们互相在对方的同义词队列中排名前列。（但是应该注意到排名作为信息很难作为监督学习的指标，因为很难形式化由它引导的误差并进行反向传播。）在这个实验中，我们希望重点比较三个模型：120M训练的Glove模型、6B训练的Glove模型、120M训练的监督学

习模型，并且希望证明第三者的性能高于第一者，并接近于第二者。

以下三图分别是：

1. 1.2M+Glove引导的排名减去6B+Glove的排名；

2. 1.2M+Glove引导的排名减去1.2M+监督模型选择的排名；

3. 1.2M+监督模型选择的排名减去6B+Glove的排名；

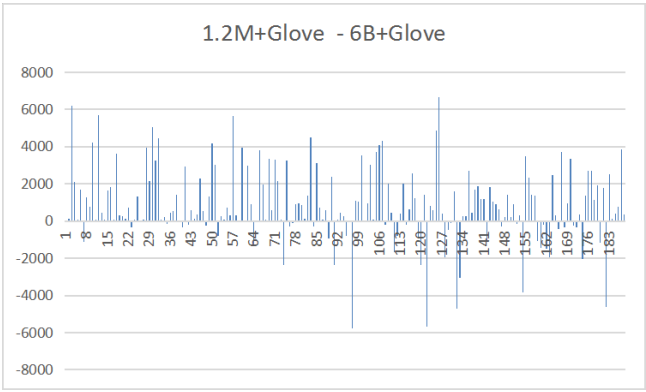

Figure 6: 120M+Glove引导的排名减去6B+Glove的排名

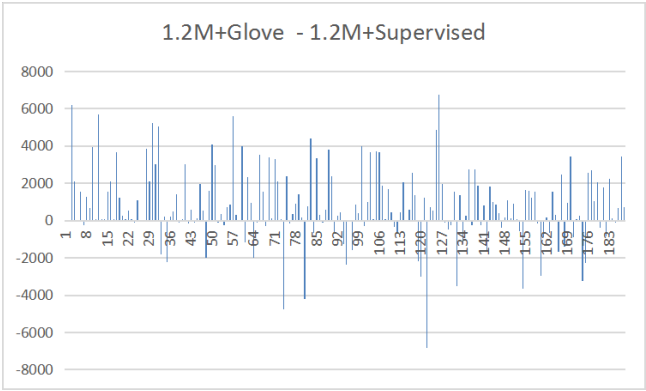

Figure 7: 120M+Glove引导的排名减去120M+监督模型选择的排名

而三者的平均排名分别是：611（6B+Glove）,1476（120M+Glove）,696（120M+Supervised）。综上所述，我们可以认为，通过使用监督的策略，我们从一个很小的语料库中能够提取出和巨大语料库性能类似的词汇编码。不同的语料库拥有不同的统计性质，因而需要配给不同的语言模型，才能针对该语料库提取出最好的编码。对于某个语料库适用的语言模型不一定能

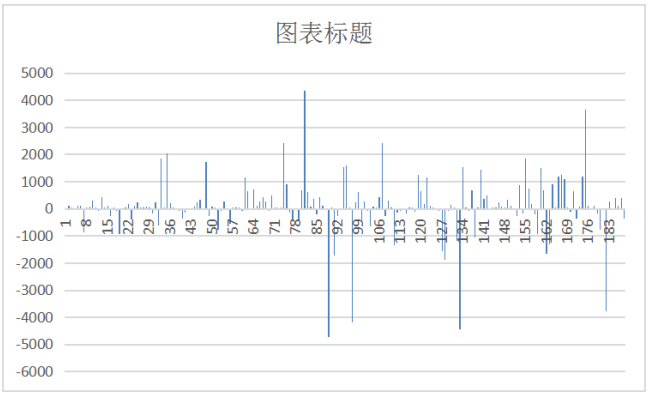

Figure 8: 120M+监督模型选择的排名减去6B+Glove的排名

够在其他语料库中得到好的效果。反之，即便是小的语料库，只要能够在语言模型上进行优化，也可以（相对于大语料库+根据人工经验选取的语言模型）很大幅度提升编码的性能。

7 Conclusion and some Discussions

7.1 Conclusion

本文中我们提出了一套利用监督学习的思路来对于某个特定语料库优化其对应的语言模型，以获得较好的词向量编码的模型。具体地，我们对一个给定的语料库运行了这个流程，并从中提取出了一套词向量，实验表明这套词向量跟现今比较流行的编码方法（以及其语言模型）在一个更大的语料库中获得的编码性能类似，但是直接套用该语言模型于被研究的语料库则会导致性能的下降。这一结果证明了我们的监督方法是有效的，它将一直以来依赖人工直觉或者经验的语言模型检验参数化，并揭示了其中可以被优化的部分，从而强化了词向量算法中的智能学习成分。这一模型还允许我们进一步否定了最优语言模型的存在，并且引导出了可能导致词向量编码性能下降的语料扩充方法，以作为这种自然语言预处理方法的一种深度对抗策略。

本文还继而提出了利用高斯过程来进行权重重新分配的方法，它更有理可循，并且符合数学直觉，并且不会随着语料库规模的变化而有太大的改动。通过高斯过程计算语义矩阵中元素的方差还允许我们将不确定性显式地代入后续高层模型进行计算，作为辅助决策的参考意见。

7.2 Discussion on Uncertainty

对于语义矩阵按分量的不确定性以及引导出的其他不确定性信息的直接应用几乎没有，因为往往来说比较余弦相似度已经足够，不依赖更多的信息，故此处不便继续考量或者比较该方法的效果，但本文已经给出了足够充分的形式化讨论以使得进一步的分析或应用可以直接实现。

7.3 Discussion on Dimensionality

在词向量的编码中，固定维度来进行性能比较已经是一个惯例。此处我们可以进一步讨

论维度之于某个语料库，以及其上语言模型的定量意义。回忆某一固定维度D所保留的信息可以量化为语义矩阵谱的前D个分量的绝对值和该矩阵所有特征值的特征值绝对值之和。所以，对于某一个特定的语料库而言，其对应的最优语言模型的谱不一定是信息量最集中的谱。换言之，对于同一个语料库以及固定的维度D而言，可以保存最多信息的语言模型和对于监督效果而言（即能够最好地捕捉语义信息）最优的语言模型可能不统一。

8 Reference

1. Word representations: A simple and general method for semi-supervised learning, ACL, 07/2010;

2. A Neural Probabilistic Language Model, JMLR, 2003;

3. Efficient Estimation of Word Representations in Vector Space, 2013;

4. word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method, arXiv, 2014;

5. Extracting semantic representations from word co-occurrence statistics: A computational study, Behavior Research Methods, 2007;

6. Evaluation methods for unsupervised word embeddings, ACL, 09/2015;

7. Word2Vec is a special case of Kernel Correspondence Analysis and Kernels for Natural Language Processing, 2018;

8. Breaking the Softmax Bottleneck: A High-Rank RNN Language Model, ICLR, 2018;

9. Glove: Global Vectors for Word Representation， EMNLP，10/2014；

10. Best Summary： From Frequency to Meaning: Vector Space Models of Semantics, JAIR, 2010;

11. Beyond Word2Vec: Embedding Words and Phrases in Same Vector Space, NLPAI, 2017;

12. A Neural Autoregressive Topic Model,

13. From Word Embeddings To Document Distances, ICML, 2015;

14. Dependency-Based Word Embeddings, ACL, 06/2014;

15. Modern state-of-the-art:XLNet: Generalized Autoregressive Pretraining for Language Understanding, arXiv, 19/06/2019;

etc.