AutoExtend

使用词汇资源: WordNet

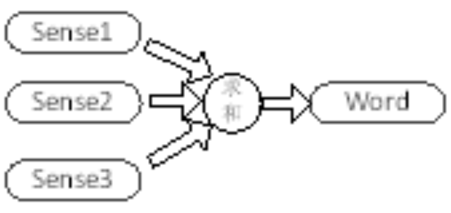
目标：enrich WordNet with embeddings for synsets and lexemes。就是通过AutoExtend方法拓展词向量，从而得到词素和同义词集的向量表示，将三种数据类型（words, lexemes, synsets）表示在同一空间。

因为这些资源对于NLP是有用的，如：词汇资源 WordNet 、 Wiktionary，知识库 Wikipedia and Freebase等。

我们提出的方法的基础是：资源的约束可以被转化为向量形式的约束（embeddings），然后可以将词向量扩展到其他数据类型（如同义词集）的嵌入。

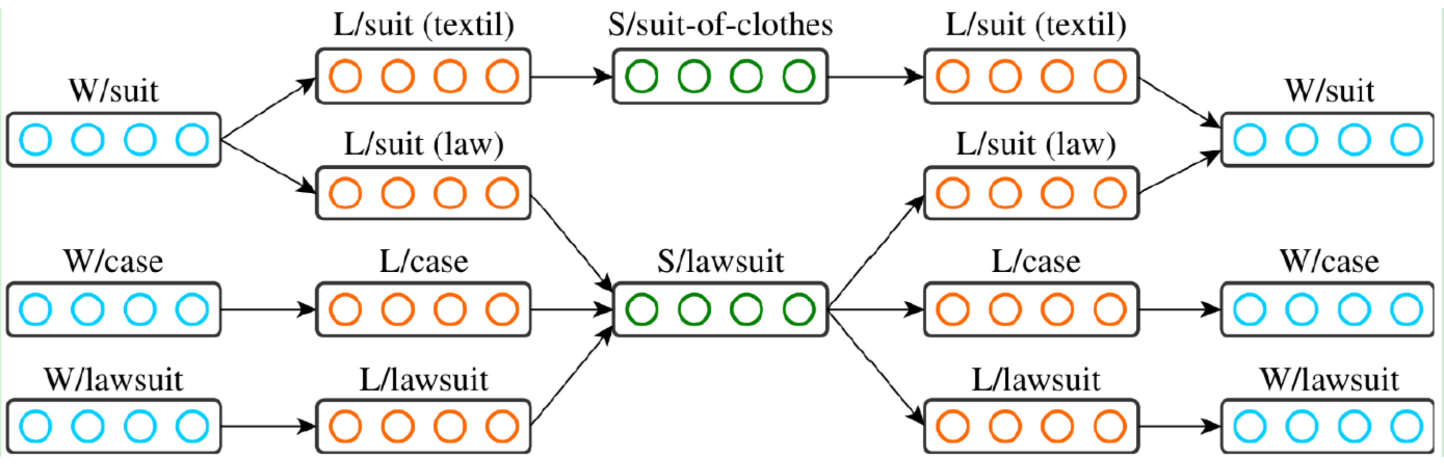
文中提到AutoExtend的替代方法：①利用已标注数据来训练同义词集嵌入，

②**S**$_{\text{naive}}$: add up all word embedding vectors related to a particular node in a resource to create the synset vector of *lawsuit* in WordNet.

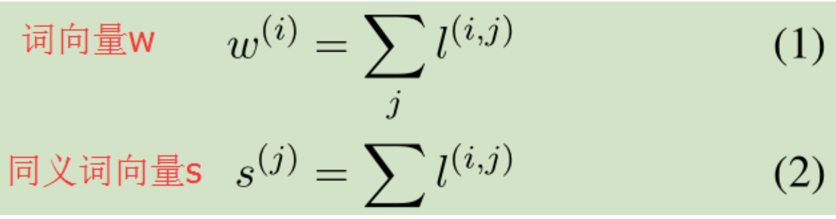


WordNet中的每个词可以看成是由一个或多个词素（lexemes）组成。Lexemes from different words together can form a synset.

2 Model



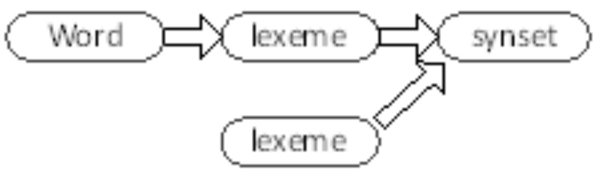
模型的基本前提假设：（1）每个词由其词素（lexemes）组成；（2）同义词集由其词素组成



词向量w $$w^{(i)} = \sum_j l^{(i,j)} \quad (1)$$

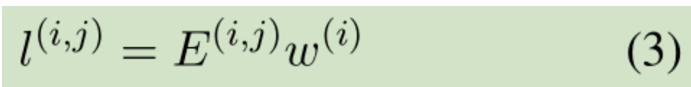
同义词向量s $$s^{(j)} = \sum_i l^{(i,j)} \quad (2)$$

将这两个假设称为约束 *synset constraints*：



***具体做法：Auto-Extend is designed to take word vectors as input and unravel the word vectors to the vectors of their lexemes. The lexeme vectors will then give us the synset vectors.*

引入一个对角矩阵 E 用于训练词素向量：

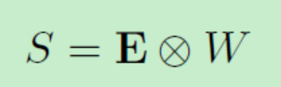


$$l^{(i,j)} = E^{(i,j)} w^{(i)} \quad (3)$$

2.1 Learning

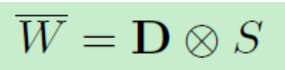
采用自动编码机来学习 lexemes 和 synsets 的向量表示：

Encoding：

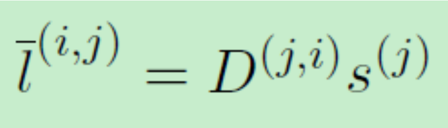


$$S = \mathbf{E} \otimes W$$

Decoding：decodes the synsets into words



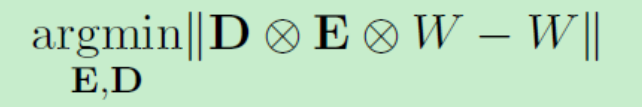
$$\overline{W} = \mathbf{D} \otimes S$$



$$\bar{l}^{(i,j)} = D^{(j,i)} s^{(j)}$$

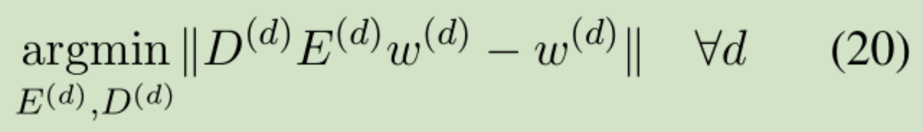
引入一个对角矩阵 D（需要学习得到、4维稀疏张量）

目标函数



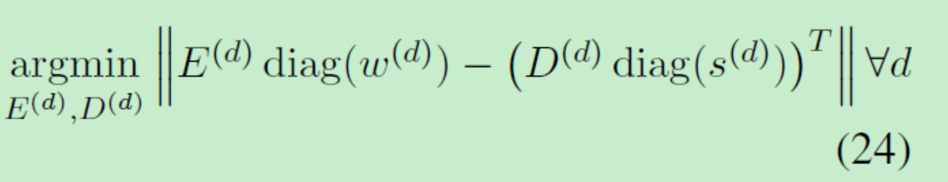
$$\underset{\mathbf{E},\mathbf{D}}{\text{argmin}} \|\mathbf{D} \otimes \mathbf{E} \otimes W - W\|$$

改进



$$\underset{E^{(d)},D^{(d)}}{\text{argmin}} \|D^{(d)} E^{(d)} w^{(d)} - w^{(d)}\| \quad \forall d \qquad (20)$$

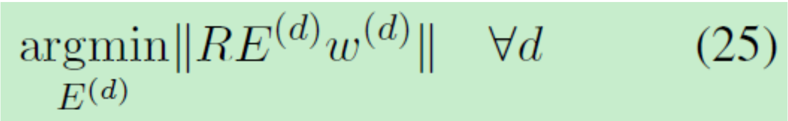
2.3 Lexeme embeddings



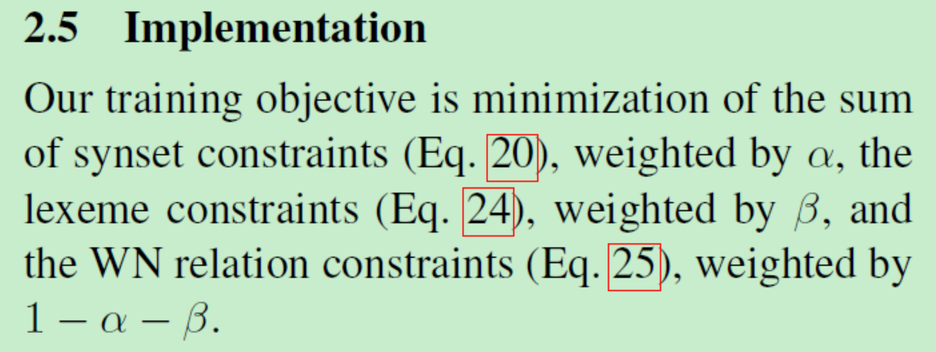
$$\underset{E^{(d)},D^{(d)}}{\text{argmin}} \left\| E^{(d)} \operatorname{diag}(w^{(d)}) - \left(D^{(d)} \operatorname{diag}(s^{(d)})\right)^T \right\| \forall d \qquad (24)$$

2.4 WN relations

为解决当同义词集仅包含一个词时无法学习到好的 synsets 向量表示的问题，提出WN关系约束，相近同义词集应该共享相似的向量。



$$\underset{E^{(d)}}{\text{argmin}} \|RE^{(d)} w^{(d)}\| \quad \forall d \qquad (25)$$



2.5 Implementation

Our training objective is minimization of the sum of synset constraints (Eq. 20), weighted by α, the lexeme constraints (Eq. 24), weighted by β, and the WN relation constraints (Eq. 25), weighted by $1 - \alpha - \beta$.

3 Data, experiments and evaluation

300-dimensional embeddings trained on Google News, using word2vec CBOW

WordNet lemmas （很多不在 word2vec 中的词汇导致许多空的同义词集，然而 AutoExtend 能够利用WN关系约束来生成同义词集向量）

set α = 0.2 andβ = 0.5

3.1 Word Sense Disambiguation(WSD)

3.2 Synset and lexeme similarity

a context vector c by adding all word vectors of the context.

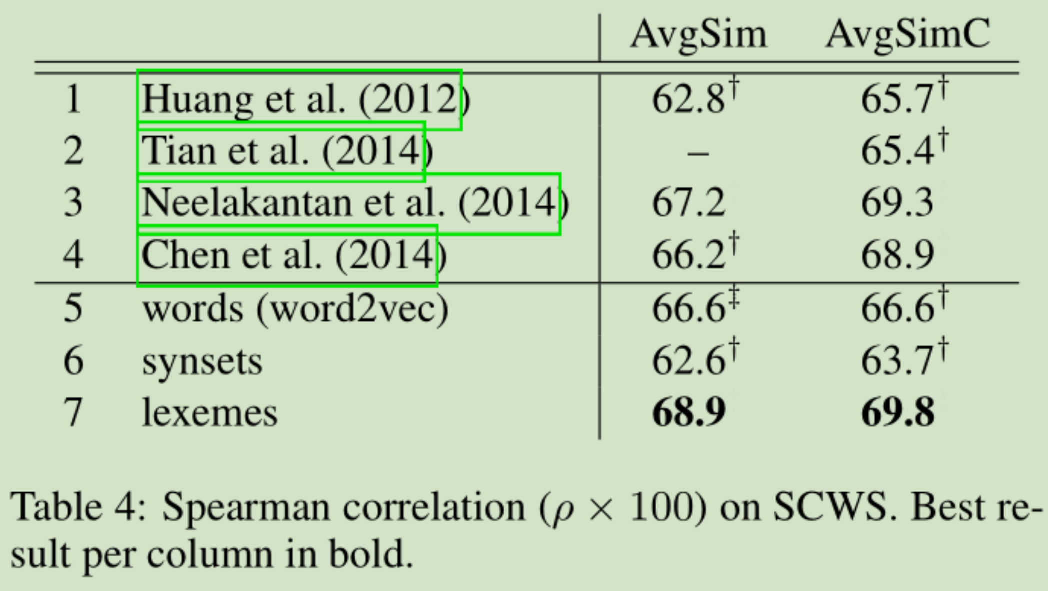
Synset and lexeme embeddings are obtained by running AutoExtend.

AvgSim : We compute the lexeme vector l either as the simple average of the lexeme vectors l (ij).

AvgSimC: as the average of the lexeme vectors weighted by cosine similarity to c.

结论：

对于相似度计算（Spearman correlation），将词向量替换为其词素（lexemes）嵌入的和可以提高词向量的质量。



		AvgSim	AvgSimC
1	Huang et al. (2012)	$62.8^\dagger$	$65.7^\dagger$
2	Tian et al. (2014)	–	$65.4^\dagger$
3	Neelakantan et al. (2014)	67.2	69.3
4	Chen et al. (2014)	$66.2^\dagger$	68.9
5	words (word2vec)	$66.6^\ddagger$	$66.6^\dagger$
6	synsets	$62.6^\dagger$	$63.7^\dagger$
7	lexemes	**68.9**	**69.8**

Table 4: Spearman correlation ($\rho \times 100$) on SCWS. Best result per column in bold.

Lexeme embeddings perform better than synset embeddings