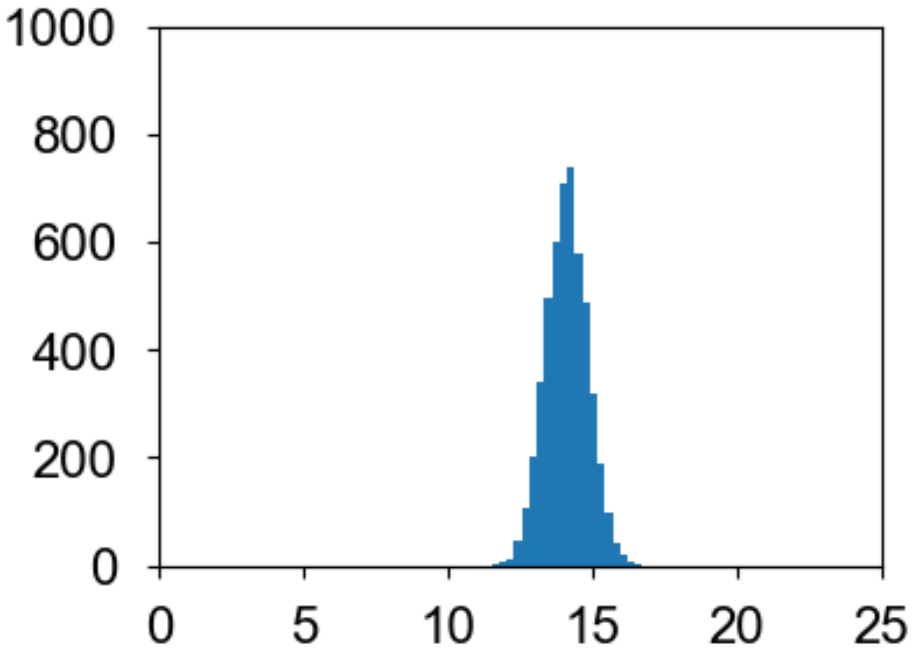
A Simple but Tough-to-beat baseline for sentence embeddings
Sanjeev Arora, Yingyu Liang, Tengyu Ma
ICLR 2017

持橋大地
統計数理研究所
daichi@ism.ac.jp

最先端NLP 2018

High-dimensional Gaussian

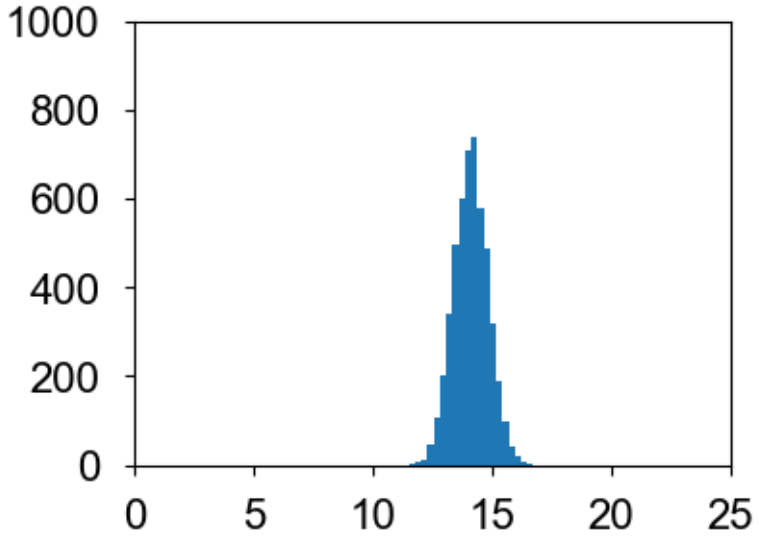
```
from pylab import *
hist(map(norm,randn(5000,200)),bins=20)
axis([0,25,0,1000])
show()
```



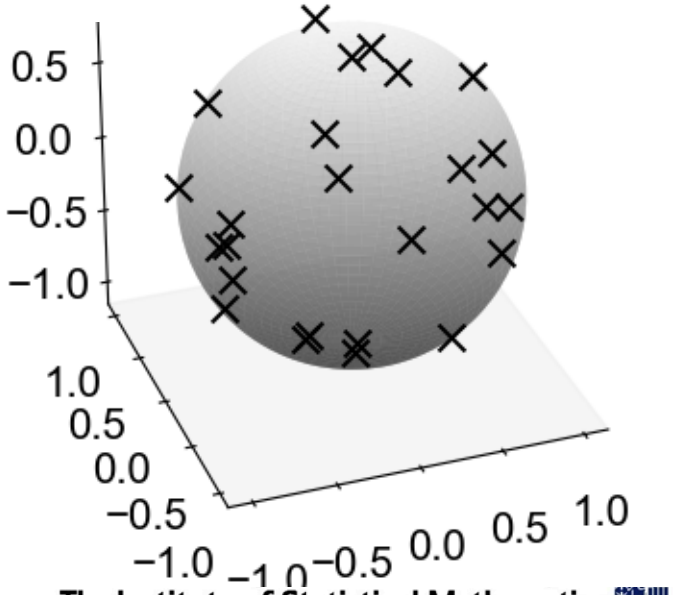
理論値(平均):

$$\frac{\sqrt{2}\Gamma((n+1)/2)}{\Gamma(n/2)} = 14.124$$

High-dimensional Gaussian (2)

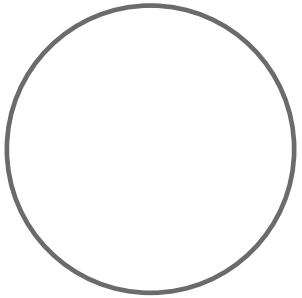


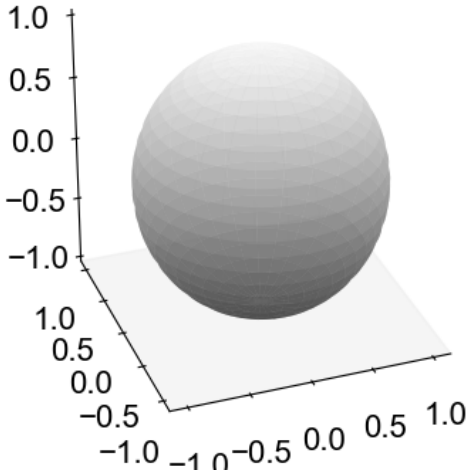
- 高次元のガウス分布からのサンプルは、事実上超球面(±α)上に分布

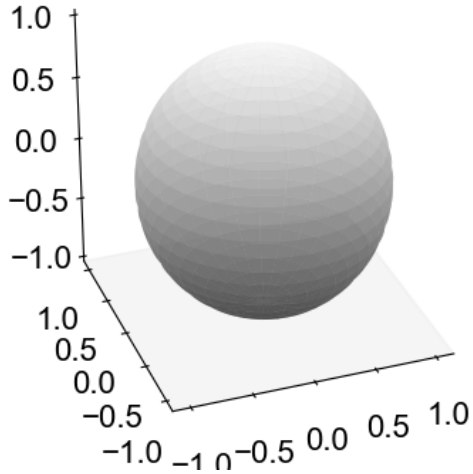


Sphere geometry

- n次元球面の表面積は、半径rで $\dfrac{2\pi^{n/2}}{\Gamma(n/2)}r^{n-1}$







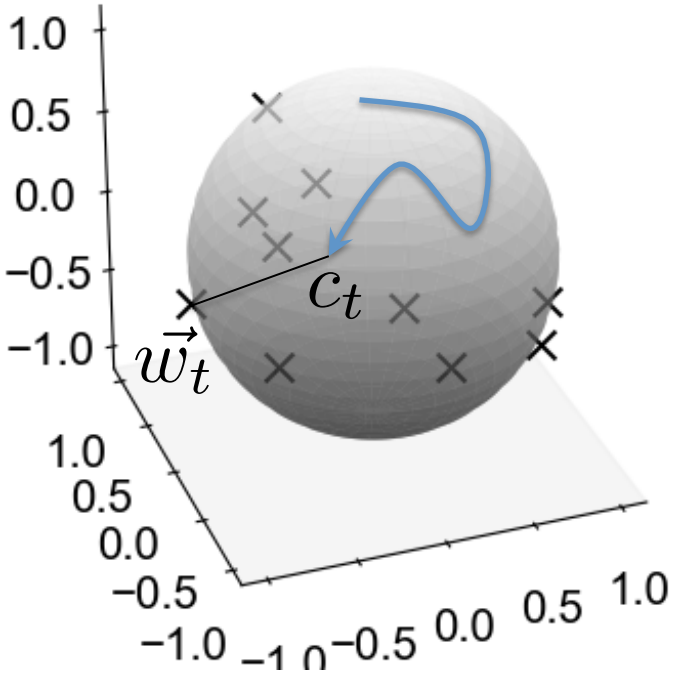
表面積	πr^2	$4\pi r^2$	$\dfrac{2\pi^{n/2}}{\Gamma(n/2)}r^{n-1}$
相対比	1	4	$2.474\times10^{122}\ (n=200)$

概要

- 目的: 文の埋め込み表現を求めること
- 従来のRNNやLSTMなど重い計算を用いなくとも、単純な単語ベクトルの重み付き和＋線形写像でそれら以上の性能を達成
  - 重み $a/(p(w)+a)$: SIF (smoothed inverse frequency) aはだいたい10^{-3}でok
  - 従来手法では教師データが必要だったが、完全教師なし学習
- 理論的な裏付けと導出がある (重要!)

Random Walk in Discourse model

- Arora+ (TACL 2016)で提唱, Mnih&Hintonの Log-linear言語モデルがもと



高次元超球面上で、文脈ベクトル c_t がゆっくりランダムウォークしているとする。このとき、単語w_tが出力される確率は

$$p(w_t|c_t) = \frac{\exp(\vec{w}_t \cdot c_t)}{Z}$$

GloVe, word2vecと同じベクトルが得られることが導かれる

Very simple idea

- 1つの文の中でc_tが一定だと仮定すれば、そのc_tを求めれば文の埋め込みになるのでは?
  → 元モデルでは、単語ベクトルを単に平均すればc_tの最尤推定量になる
- 問題あり：
  - “the”や“a”などのベクトルも足されてしまう
    → 単語の重みづけが必要
  - 意味というより文法や慣例によるノイズに引きずられる (例: san francisco)
- どんな重みやノイズ除去にすればよい?

Improved Random walk model

- 単語は文脈ベクトルc_t'との近さ以外に、確率αでランダムにユニグラム分布から出力される

$$p(w_t|c_t') = \alpha p(w_t) + (1-\alpha)\frac{\exp(\vec{w}_t \cdot c_t')}{Z}$$

- 文脈ベクトルc_t'は、定常ベクトルc_0と時変成分c_tに分けられる

$$c_t' = \beta c_0 + (1-\beta)c_t, \quad c_0 \perp c_t$$

  - c_0は、文法的な単語の出やすさを支配
  - 最近の研究でも同じ観察 (“All-but-the-top”, ICLR 2018) : こちらの方が深い議論をしているのでお薦め

Estimation

- 文 $s = w_1 w_2 \cdots w_M$ に対して、対応するcを推定したい
- 最尤推定を考えると、

$$\log p(s|c) = \sum_{w \in s} \log \left[\alpha\, p(w) + (1-\alpha) \frac{\exp(\vec{w} \cdot c)}{Z} \right]$$

- Σの中をf(w,c)とおくと、テイラー展開により

$$\begin{aligned} f(w,c) &\approx f(w,0) + \nabla f(w,0)^T c \\ &= \frac{(1-\alpha)/(\alpha Z)}{p(w) + (1-\alpha)/(\alpha Z)} \langle c, \vec{w} \rangle + (\text{constant}) \end{aligned}$$

Estimation (2)

- よって、近似的に

$$\mathrm{argmax}_c \sum_{w \in s} f(w,c) \propto \sum_{w \in s} \frac{a}{p(w)+a} \vec{w}, \quad a = \frac{1-\alpha}{\alpha Z}$$

  - cの推定量は、文の単語ベクトルの重み付き和
  - Smoothed Inverse Frequency (SIF) weighting

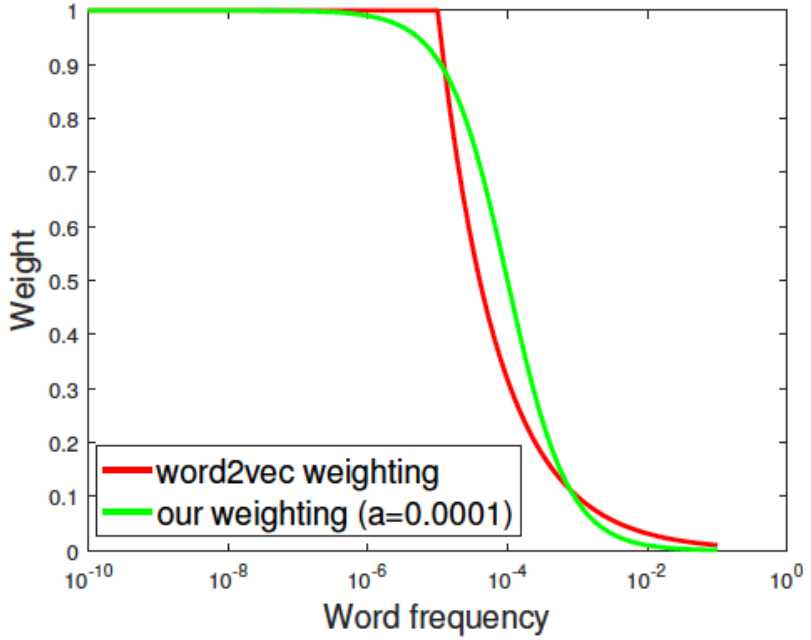
- 注: |c|=1 なので、上の導出には

$$\max_{c:|c|=1} \langle c, g \rangle + C = g/|g|$$

  を使っている

word2vecとの関係

- word2vecは、近傍語wを選ぶときに $1/\sqrt{p(w)}$ に比例して選んでいる
- これは、単語に重み $q(w) = \min\left(1, \sqrt{\frac{10^{-5}}{p(w)}}\right)$ をかけていることに相当
- SIFとq(w)は非常に似ている



Algorithm for sentence embedding

Algorithm 1 Sentence Embedding

Input: Word embeddings $\{v_w : w \in \mathcal{V}\}$, a set of sentences $\mathcal{S}$, parameter a and estimated probabilities $\{p(w) : w \in \mathcal{V}\}$ of the words.

Output: Sentence embeddings $\{v_s : s \in \mathcal{S}\}$

1: **for all** sentence s in $\mathcal{S}$ **do**

2: $\quad v_s \leftarrow \frac{1}{|s|} \sum_{w \in s} \frac{a}{a+p(w)} v_w$

3: **end for**

4: Form a matrix X whose columns are $\{v_s : s \in \mathcal{S}\}$, and let u be its first singular vector

5: **for all** sentence s in $\mathcal{S}$ **do**

6: $\quad v_s \leftarrow v_s - uu^\top v_s$

7: **end for**

- 計算の基となる単語埋め込みは、StanfordのGloVeの300次元埋め込みなどを使用
- 注: 提案手法のモデルから単語埋め込みを求めることも多分可能

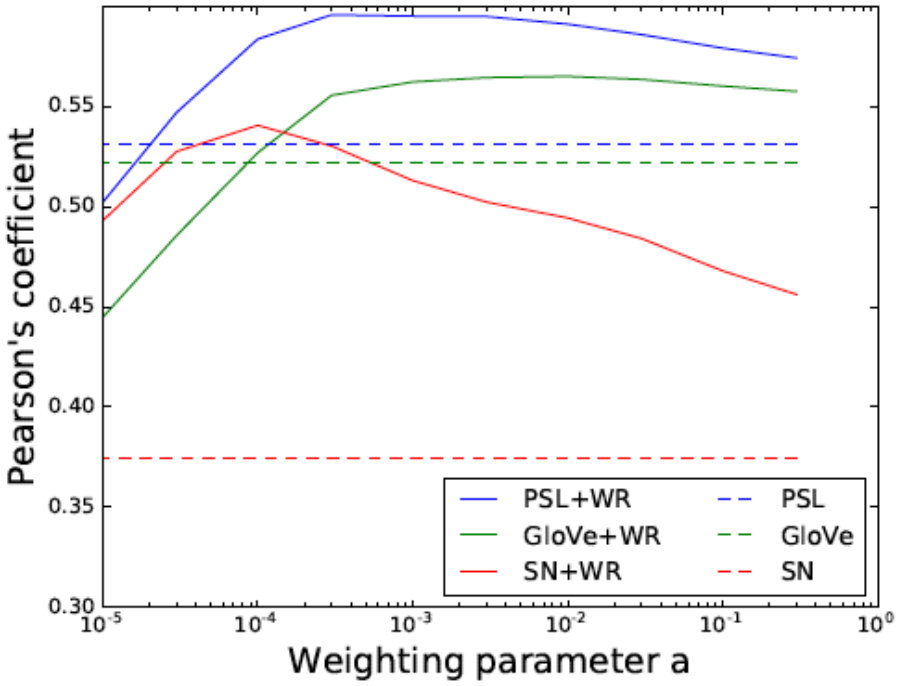
Experiments

- SemEval textual similarityタスク
  - 提案法がRNN, LSTM, SkipThought等を抜き高性能
  - DAN等多くの手法は教師あり＋重い計算が必要

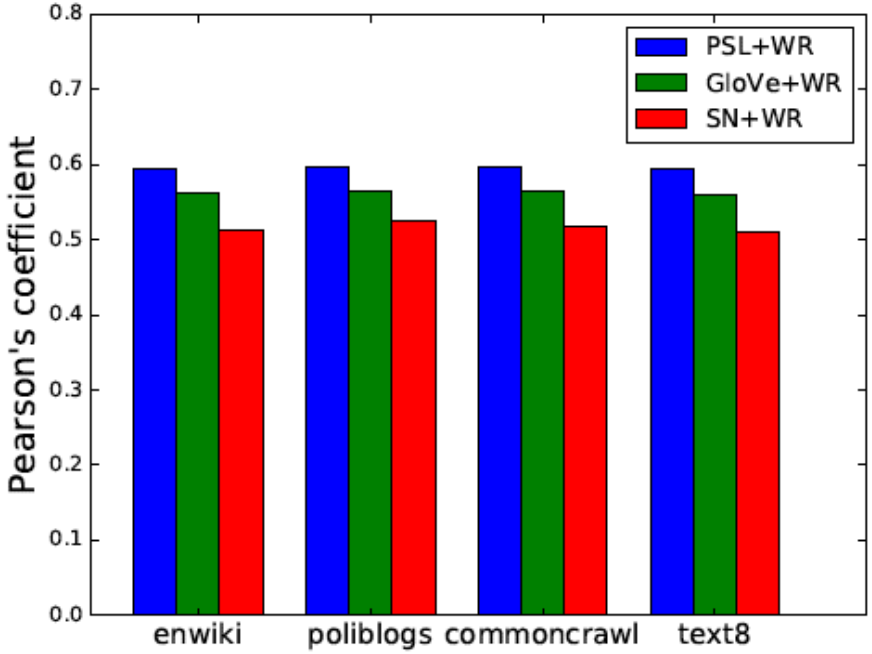
	Results collected from (Wieting et al., 2016) except tfidf-GloVe											Our approach	
Supervised or not	Su.							Un.			Se.	Un.	Se.
Tasks	PP	PP -proj.	DAN	RNN	iRNN	LSTM (no)	LSTM (o.g.)	ST	avg-GloVe	tfidf-GloVe	avg-PSL	GloVe +WR	PSL +WR
STS'12	58.7	**60.0**	56.0	48.1	58.4	51.0	46.4	30.8	52.5	58.7	52.8	56.2	59.5
STS'13	55.8	56.8	54.2	44.7	56.7	45.2	41.5	24.8	42.3	52.1	46.4	56.6	**61.8**
STS'14	70.9	71.3	69.5	57.7	70.9	59.8	51.5	31.4	54.2	63.8	59.5	68.5	**73.5**
STS'15	75.8	74.8	72.7	57.2	75.6	63.9	56.0	31.0	52.7	60.6	60.0	71.7	**76.3**
SICK'14	71.6	71.6	70.7	61.2	71.2	63.9	59.0	49.8	65.9	69.4	66.4	72.2	**72.9**
Twitter'15	52.9	52.8	**53.7**	45.1	52.9	47.6	36.1	24.7	30.3	33.8	36.3	48.0	49.0

Table 1: Experimental results (Pearson's $r \times 100$) on textual similarity tasks. The highest score in each row is in boldface. The methods can be supervised (denoted as Su.), semi-supervised (Se.), or unsupervised (Un.). "GloVe+WR" stands for the sentence embeddings obtained by applying our method to the GloVe word vectors; "PSL+WR" is for PSL word vectors. See the main text for the description of the methods.

αとp(w)の効果



(a)



(b)

- αは10^{-3}〜10^{-4}の広い範囲で高性能 (解析解がある)
- p(w)を計算するコーパスを変えても、結果は同じ

c0の計算と効果

- c0はk-SVDで重み付けした単語ベクトル行列の第一固有ベクトルとして求める
- c0の近傍の語は
  just, when, even, one, up, little, way, there, while, but
  ... 文法的な次元をとらえている

Downstream classification task

	PP	DAN	RNN	LSTM (no)	LSTM (o.g.)	skip-thought	Ours
similarity (SICK)	84.9	85.96	73.13	85.45	83.41	85.8	**86.03**
entailment (SICK)	83.1	84.5	76.4	83.2	82.0	-	**84.6**
sentiment (SST)	79.4	83.4	86.5	86.6	**89.2**	-	82.2

- similarity, entailment, sentiment タスクの文ベクトルとして用いた場合
  - o.g.は出力ゲートあり、noはなし
- similarityとentailmentで提案手法が複雑な手法を凌駕
- sentimentでは既存手法が高性能→(1) SIFは”not”などをほぼ無視 (2) 反意語問題には対応していない

Conclusion

- 文の生成モデルを考えて変更することで、完全教師なしで教師ありの複雑な従来手法と同等以上の性能の文埋め込みを実現
- 文法や論理、感情など複雑な言語的事象を直接モデリングしているわけではないので、それらが必要なタスクでは最高性能ではない(ベースライン)
- 重要な点: 結果ではなく、単語ベクトルの観察と幾何に基づいて、数学的な裏付けのある方法を導いていること.