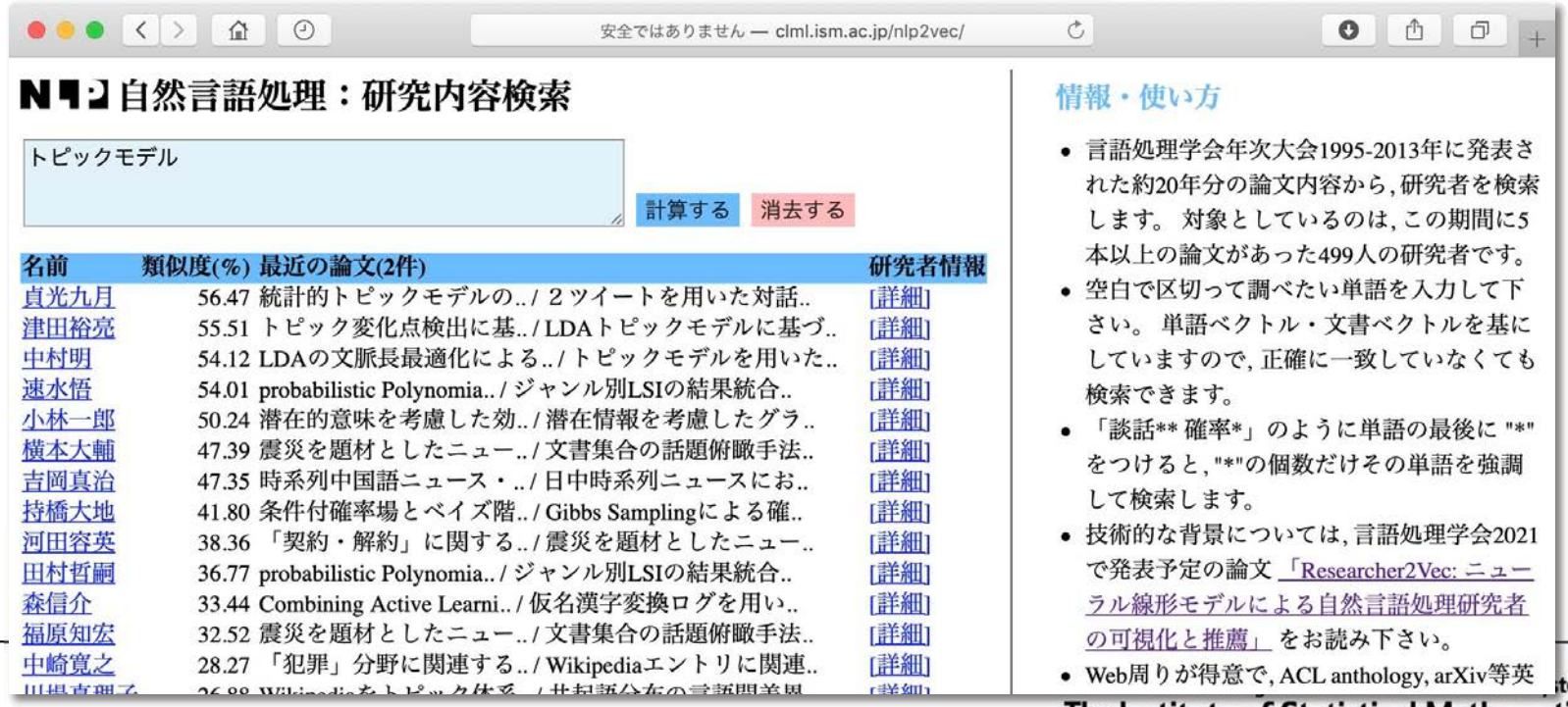
# Researcher2Vec: ニューラル線形モデルによる自然言語処理研究者の可視化と推薦

持橋大地
統計数理研究所 数理・推論研究系
日本学術振興会 学術情報分析センター
daichi@ism.ac.jp

言語処理学会年次大会 2021
2021-3-16 (火)

# 概要

- 論文の内容から研究者をベクトル化し、キーワードで検索できるサーバを公開
  - Word2vecと同様のニューラル文書ベクトルを、特異値分解によりDoc2Vecの20倍の速度で計算
  - Doc2Vec, LDAによる検索を超えて最高精度

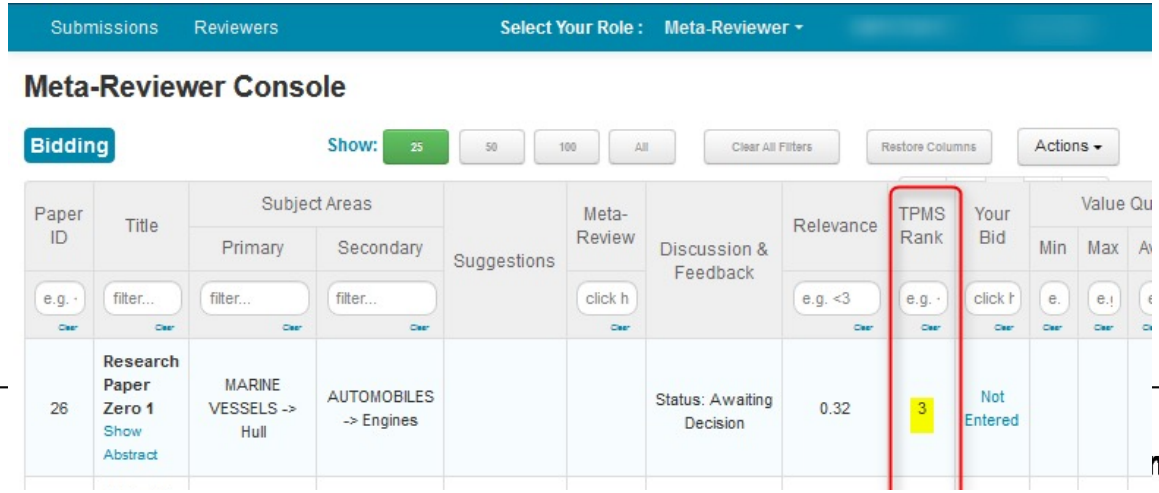


# 概要

- 研究者ベクトルを2次元に可視化することで、研究内容の近い研究者が客観的に可視化できる
(既存のシステムはヒューリスティックが多い)

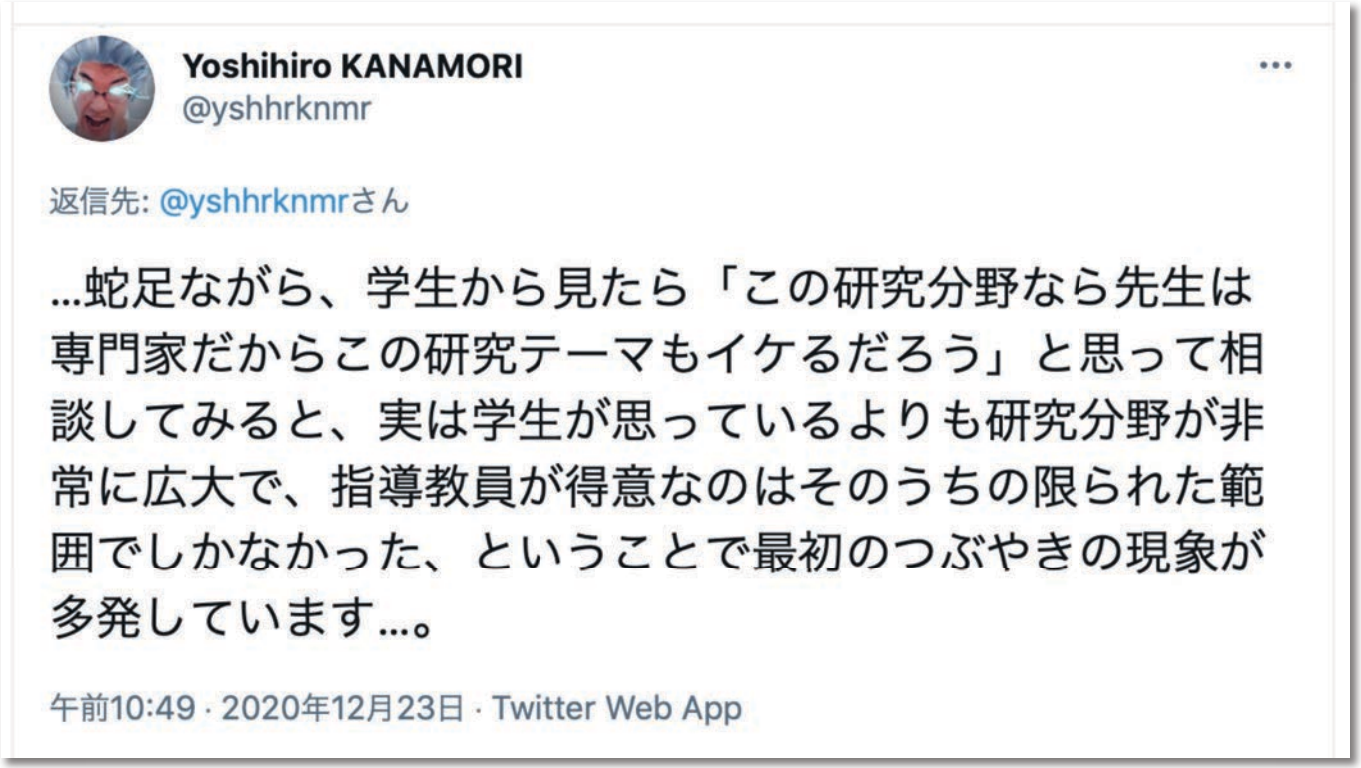
# 背景

- コンピュータサイエンス分野では、論文数が激増
  - 言語処理学会2020…338本, ACL 2020…779本, CVPR 2020…1470本, NeurIPS 2020…1,900本以上
- 人手による査読割り当ては、もはや限界
  - 上の採択数の4倍～10倍程度×3～5程度の査読数が必要 (CVPR 2020では19000個の査読が必要)
  - TPMSが使われているが、研究者別情報は非公開
  - 未だに研究者の専門性を人手で調べる必要がある



# 背景 (2)

- 研究の興味が多様化しており、自然言語処理なら誰に聞いても同じ、ではもはやない



# 背景 (2)

- この分野の研究をよく知っている先生は誰か？
  → 適切な指導者の発見 (学生、企業とも)
- 会議やジャーナルへの適切な研究者のリクルート
  → コネがなくてもコミュニティに加われるようにしたい

# 既存システム

- 「日本の研究.com」, JDream Expert Finder, JSTサイエンスマップなど
  - 論文の内容ではなく、引用などメタ情報がベース
  - 本当の詳しい専門性は分からない、共著関係に引きずられる (コネ)


← 鈴木さんが心理学??

# 既存研究

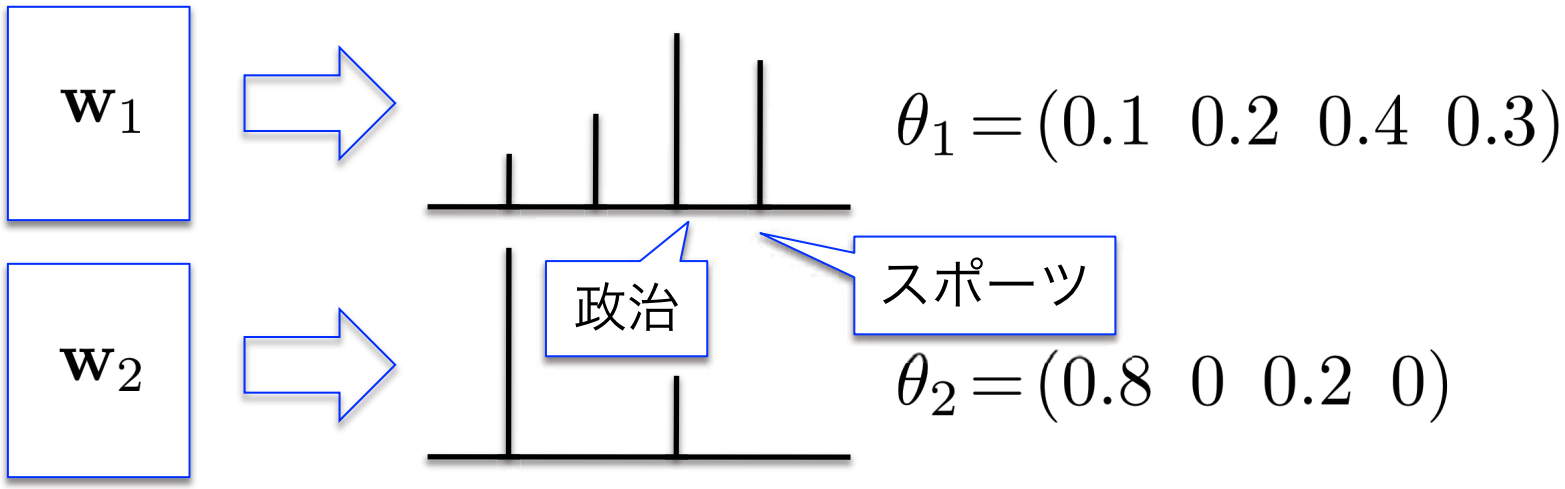
- 桂井ら (2016)：CiNIIの10万件の論文概要、300万語のテキストをトピック数K=500のLDAで解析
  - 次元圧縮しないベクトル空間モデルより高精度
- 持橋 (2019)：学振内部の11万件の科研費申請書、3億語のテキストをトピック数K=4000の巨大なLDAで解析
  - LightLDA(WWW 2015)で高速化しても数日かかる
  - 科研費特別推進、基盤Sの審査に利用

# ちなみに..


- 本研究とは、これとは完全に独立 (学振での必要性)
  - まだ論文が出ていない(はず)
  - 「様々な情報を埋め込む」そうなので、今後に期待

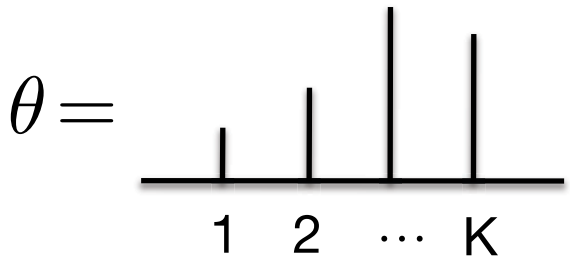
# LDAによる論文テキストの分析

- 確率的潜在意味解析 (LDA, Blei+ 2003) …
  各文書に対し、潜在的な話題 (トピック)分布を推定



# LDAの欠点

- 結果として、LDAは大変有効なモデルではあるが、
  - 負の相関を扱えない
  - 細かい意味的な違いに対処するのに限界がある
- 研究者や申請書の内容は和が1のトピック分布 $\theta$



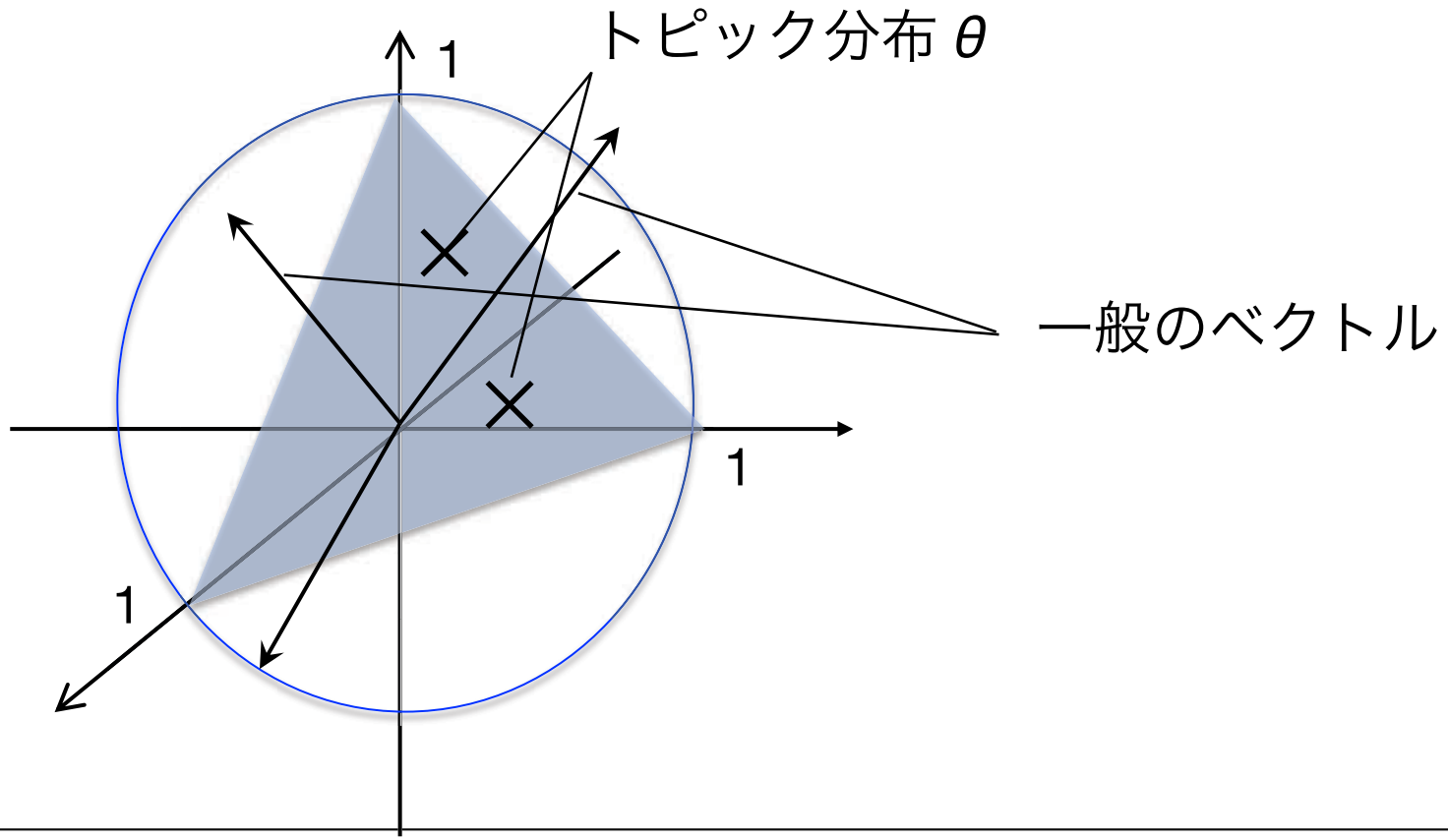

で表現されるため、

- 「経済学者だが、数理的には解析学がベース」
  「経済学者だが、数理的にはゲーム理論がベース」
  などの細かい違いを$\theta$で捉えるのは、非常に難しい

→ 研究者の推薦精度にある程度限界がある

# LDAの欠点 (2)

- 数学的には、LDAは単体の上でしか文書をモデル化していない
  → ベクトル空間のごく一部しか使っていない

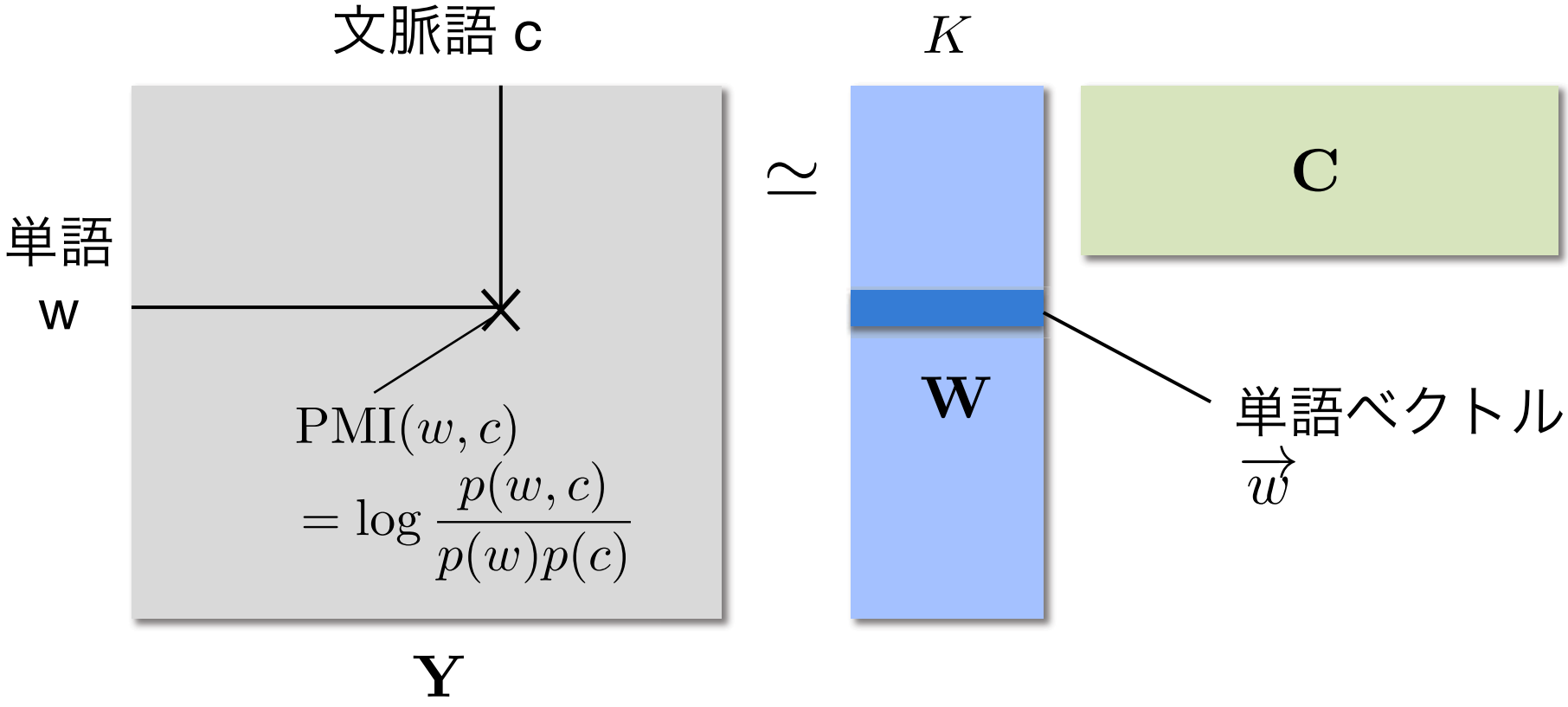


# 文書のベクトル表現

- 確率分布 $\theta$ で文書を表現するのをやめて、
  一般のベクトル $\overrightarrow{d}$ で文書や研究者を表現すればよい?
  → RaP (Gehler+ 2006), RSM (Salakhutdinov 2009),
  Doc2Vec (Le and Mikolov 2014) , NVDM (Miao+ 2016)
  など多数あるが..
- ニューラル手法なので、一般に学習が難しい
- しかし...

# Word2vecの数理

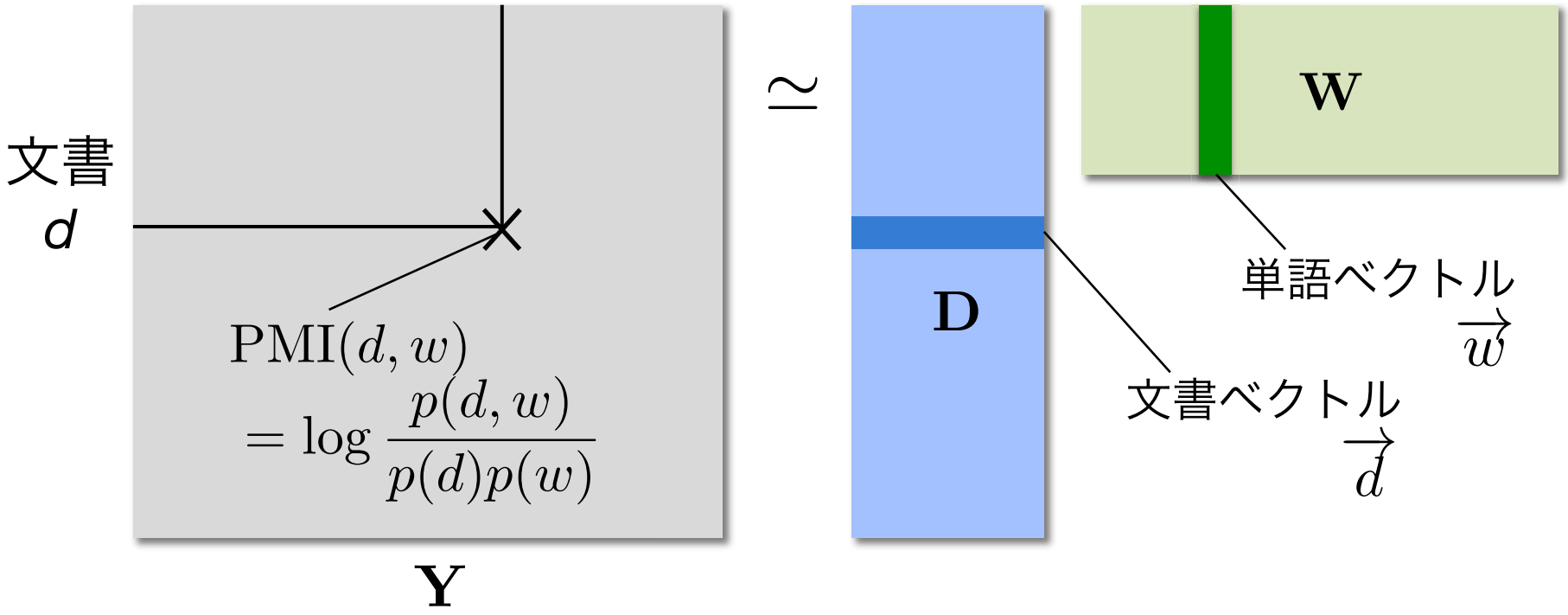
- 単語をベクトル化する、有名なWord2vec (Mikolov+ 2013) は、以下の自己相互情報量行列の行列分解と等価であることが示されている (Levy and Goldberg 2014)



# Word2vecから文書ベクトルへ

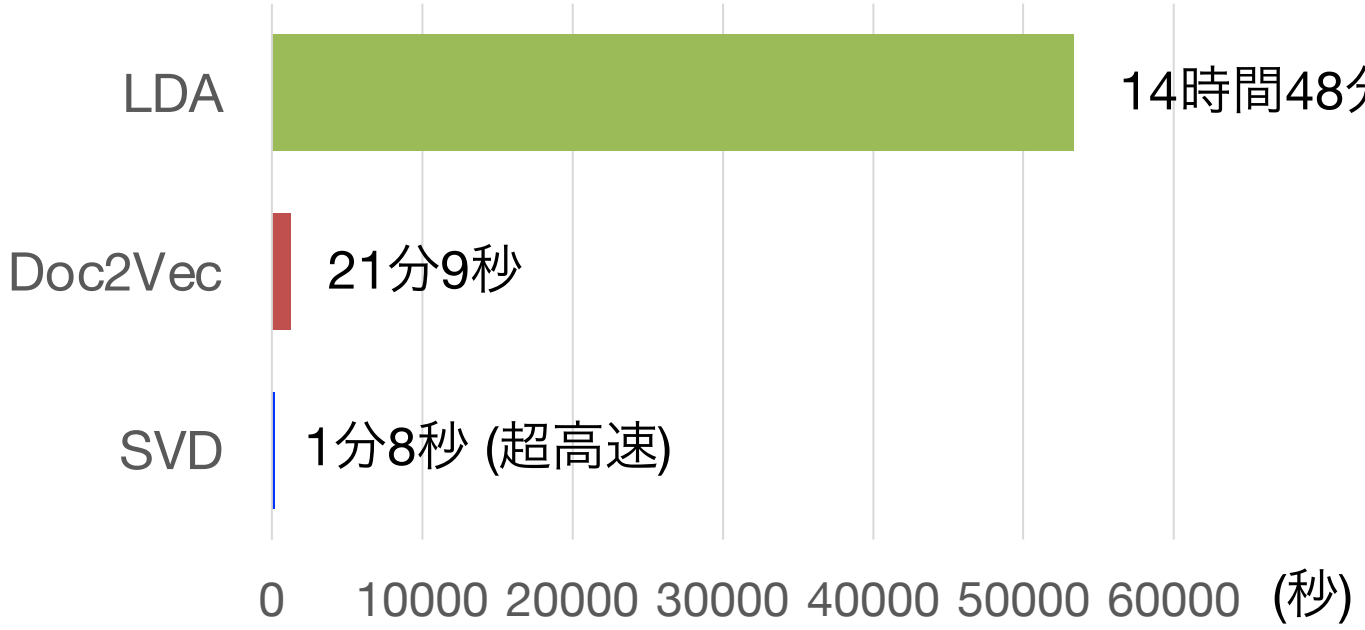
- 単語→文書、文脈語→含まれる単語 に置き換えれば、SVDで簡単に「文書ベクトル」と「単語ベクトル」を計算できる

(注：$\log \frac{p(d,w)}{p(d)p(w)} = \log \frac{p(w|d)}{p(w)}$)



# 計算時間の比較

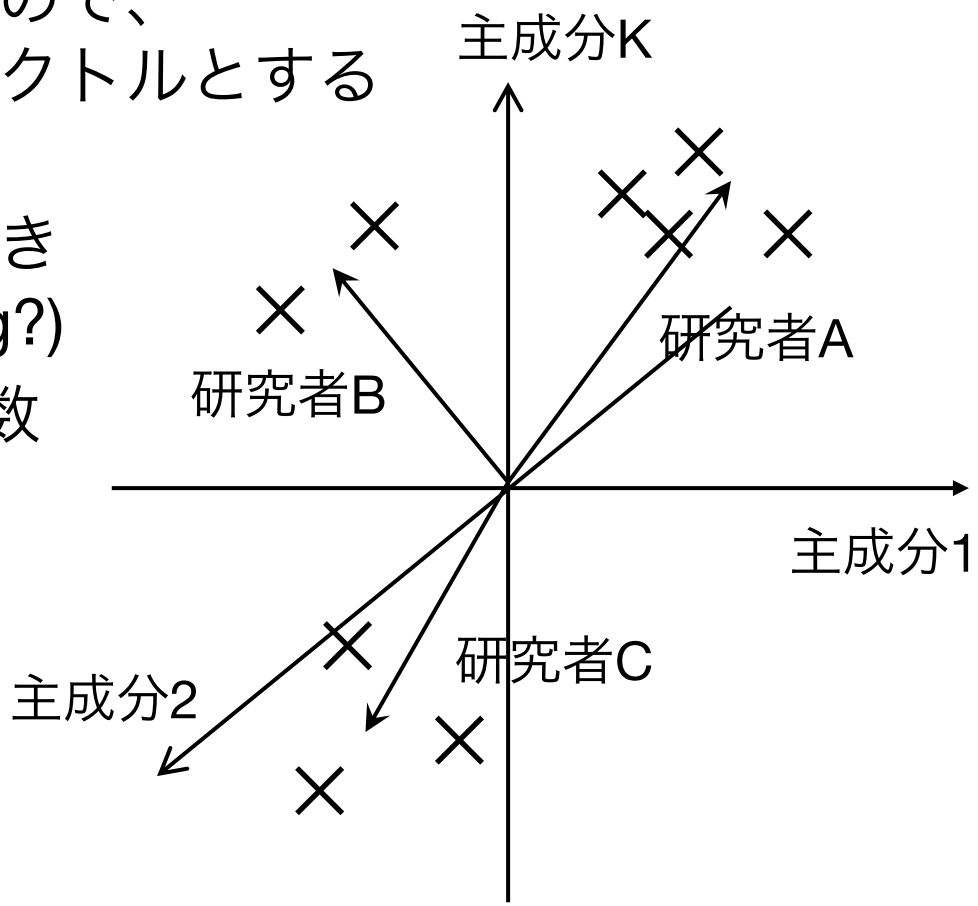
- K=1,000次元(トピック)で同じコーパスから学習した場合、



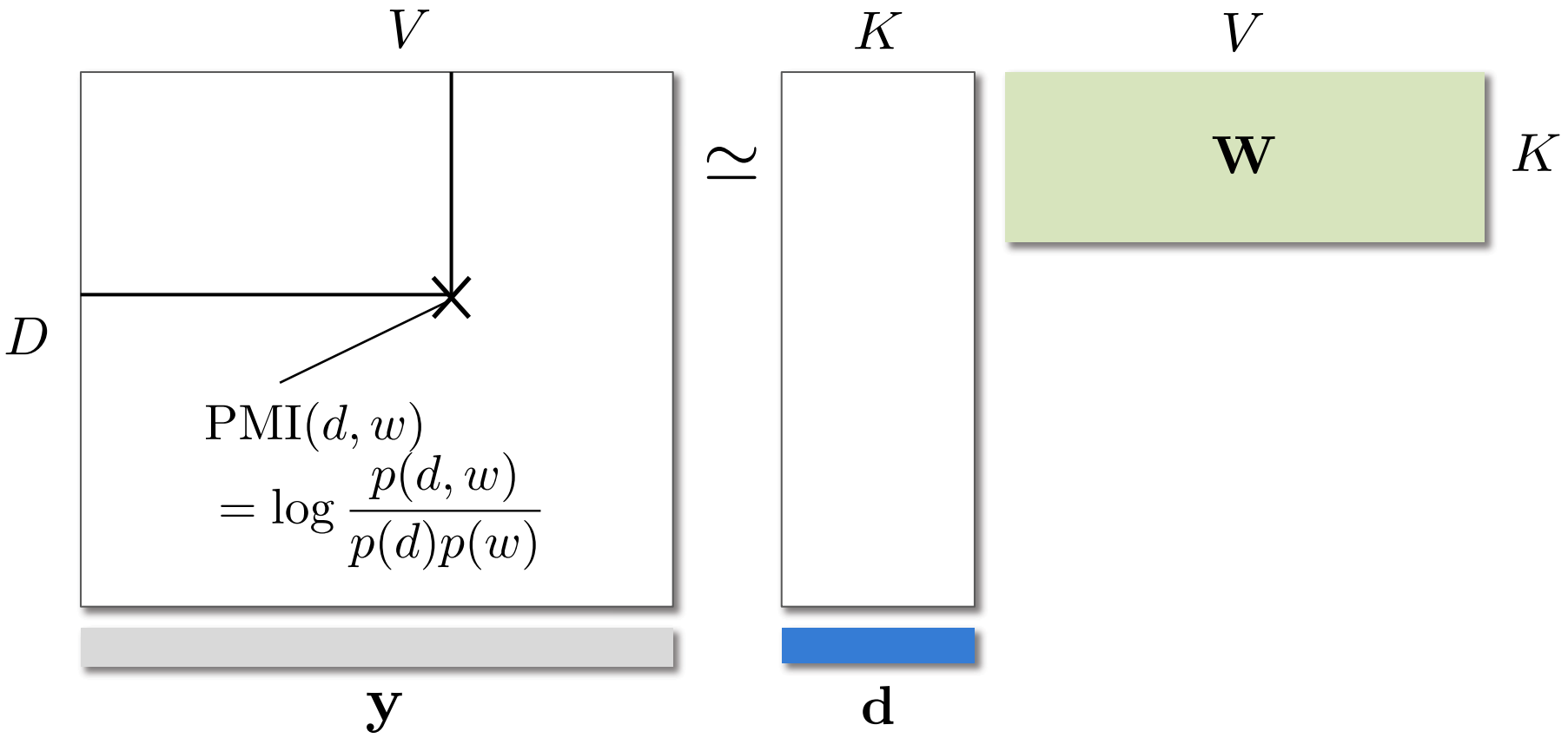

  - LDAはGibbs 1,000 iteration, Doc2Vecは100 epochs
  - データが巨大な場合、提案法はredsvdなども使える

# 文書ベクトルから研究者ベクトルへ

- 各研究者について、書いた論文/科研費申請書の文書ベクトルが得られるので、それらの平均を研究者ベクトルとする(最尤推定)
- 本来は分散も推定するべき(Kernel mean embedding?)
- 次元Kは、最大値は文書数(実験では3582)



# キーワード検索の方法

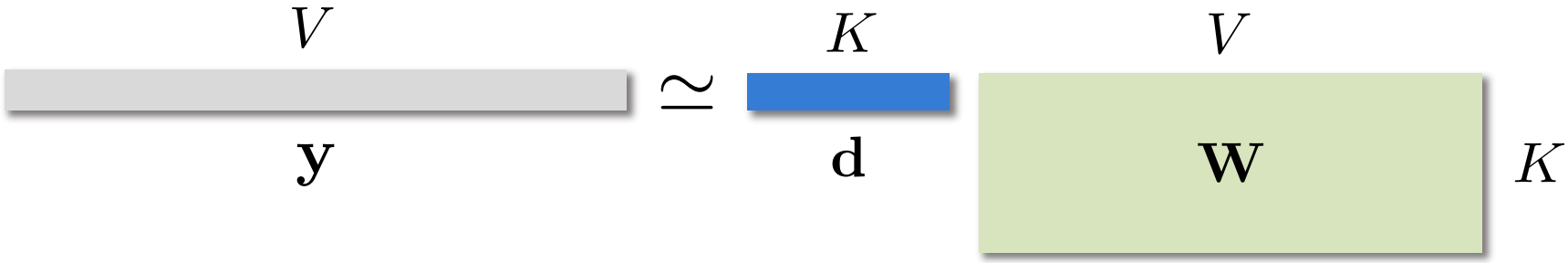



- クエリを仮想的な「文書」yと思うと、

$$\mathbf{y} \simeq \mathbf{dW}$$

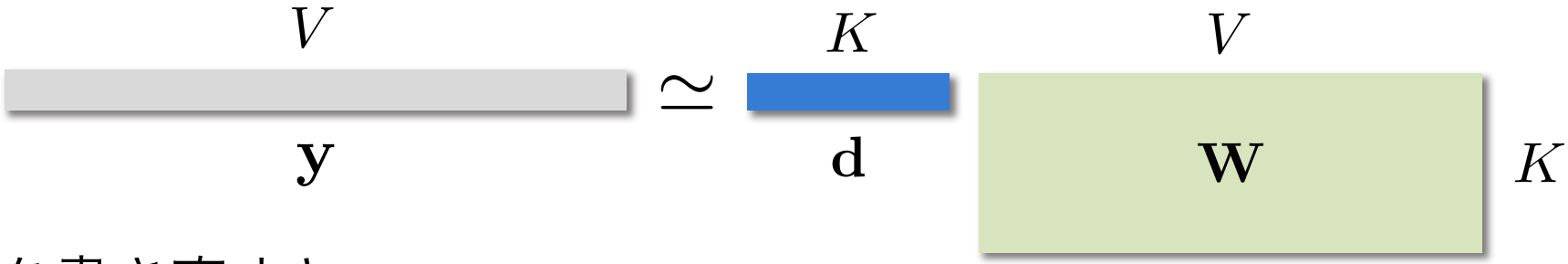
が成り立っている ( $\mathbf{d}$ は対応する文書ベクトル)

# キーワード検索の方法 (2)

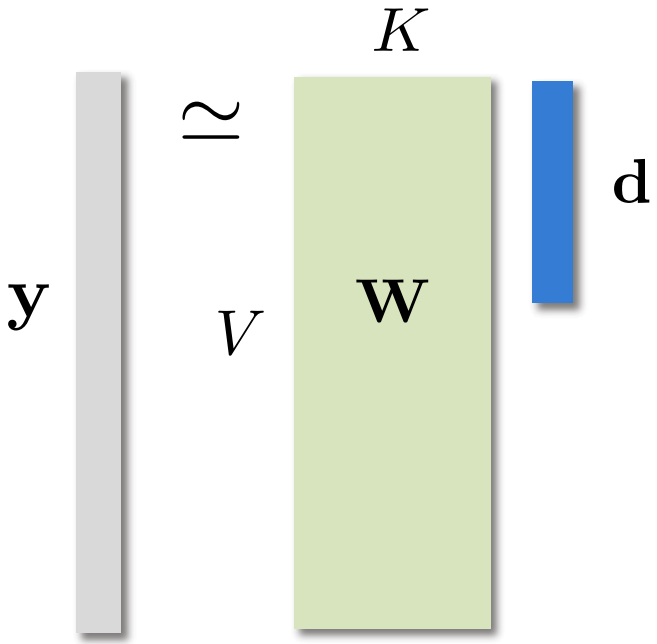



- yの要素はPMI log p(w|y)/p(w) なので、クエリの部分に1を立て、残りは log(1)=0 のベクトル
  - クエリ単語の最後に * を付けると、1ではなく 2,3,... にしてその単語を強調できる (例: “neural** model”)
- y~dW の方程式は等式ではなく近似
- 得られた「文書ベクトル」dを研究者ベクトルと比べればよい

# キーワード検索の方法 (3)




を書き直すと、




- これは線形回帰モデル！

$$\mathbf{y} \simeq \mathbf{W}\mathbf{d}$$

- よって、dの最適解は通常のOLSで、

$$\mathbf{d} = (\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T\mathbf{y}$$

# キーワード検索の方法 (4)

- 線形回帰の基本ですが、二乗誤差を最小化したいので

$$\begin{aligned} E &= |\mathbf{y} - \mathbf{W}\mathbf{d}|^2 \\ &= (\mathbf{y} - \mathbf{W}\mathbf{d})^T(\mathbf{y} - \mathbf{W}\mathbf{d}) \\ &= \mathbf{y}^T\mathbf{y} - 2\mathbf{d}^T\mathbf{W}^T\mathbf{y} + \mathbf{d}^T\mathbf{W}^T\mathbf{W}\mathbf{d} \end{aligned}$$

- よって

$$\frac{\partial E}{\partial \mathbf{d}} = -2\mathbf{W}^T\mathbf{y} + 2\mathbf{W}^T\mathbf{W}\mathbf{d} = 0$$

$$\therefore \ \mathbf{d} = (\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T\mathbf{y}$$

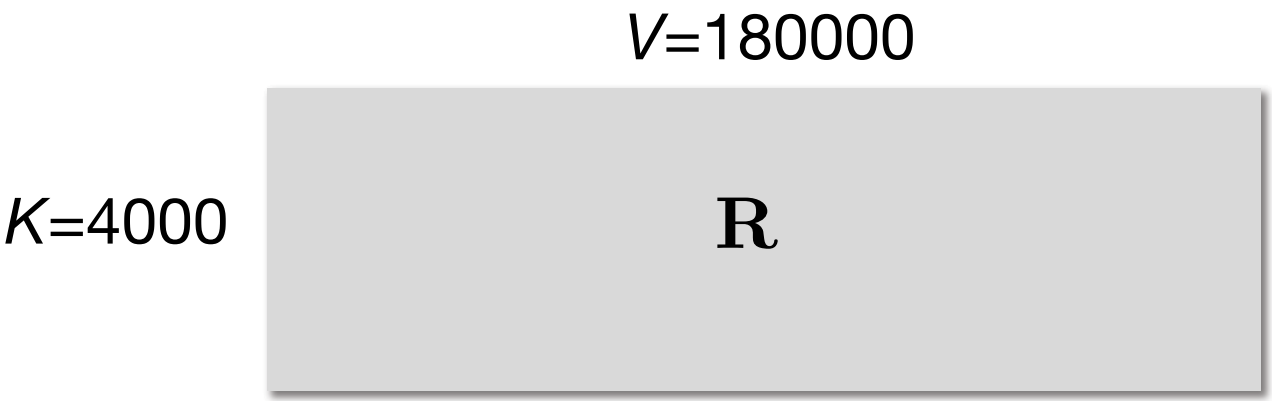
- 事前に $\mathbf{R} = (\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T$ を計算しておけば、

$$\mathbf{d} = \mathbf{R}\mathbf{y}$$

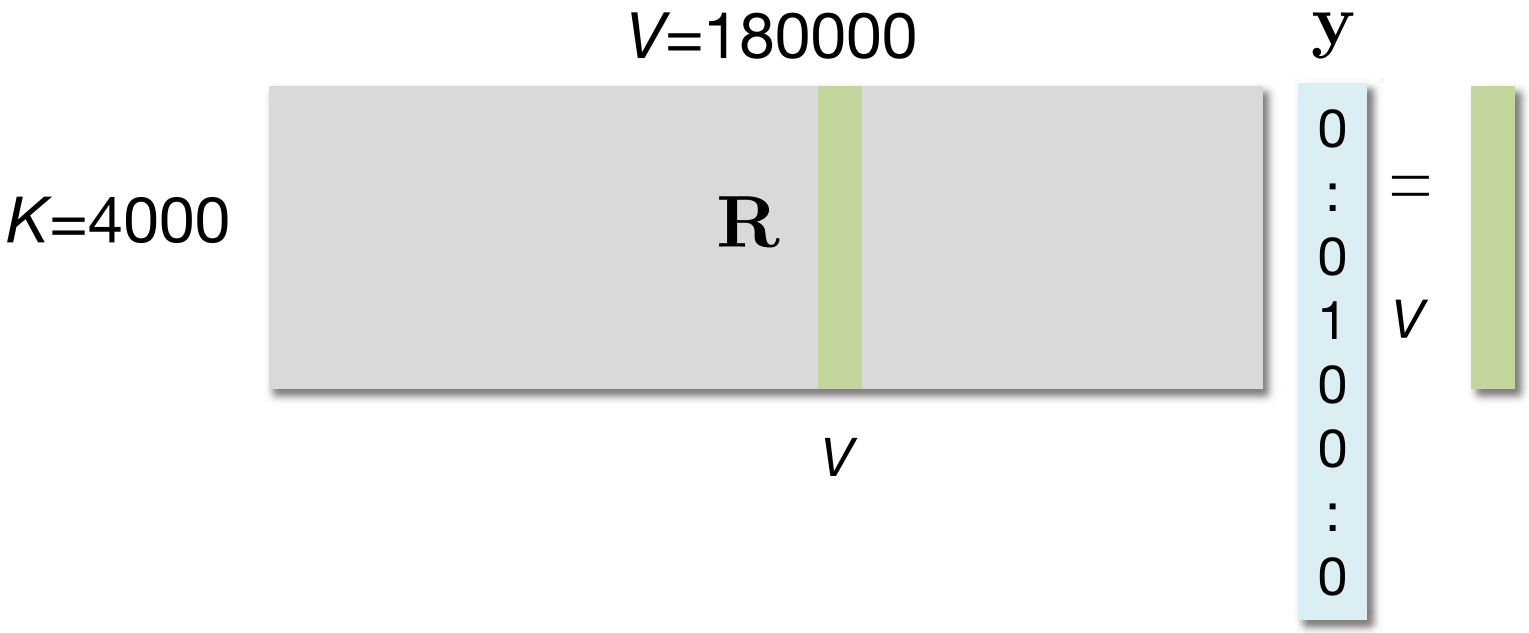
で一瞬で求まる

# キーワード検索の方法 (5)

- $\mathbf{R} = (\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T$ は、numpy.linalg.solve (dot(W.T,W), W.T) で求められる
- WはVxKなので、$W^TW$はKxK, $R=(W^TW)^{-1}W^T$は (KxK)x(KxV)=KxV の行列
  - 言語処理学会ではV=18000, K=2000 → Rは277MB
  - 学振の場合はV=180000, K=4000なので R (およびW)は要素数7億2000万個の巨大な行列、～4GB



# キーワード検索の方法 (6)



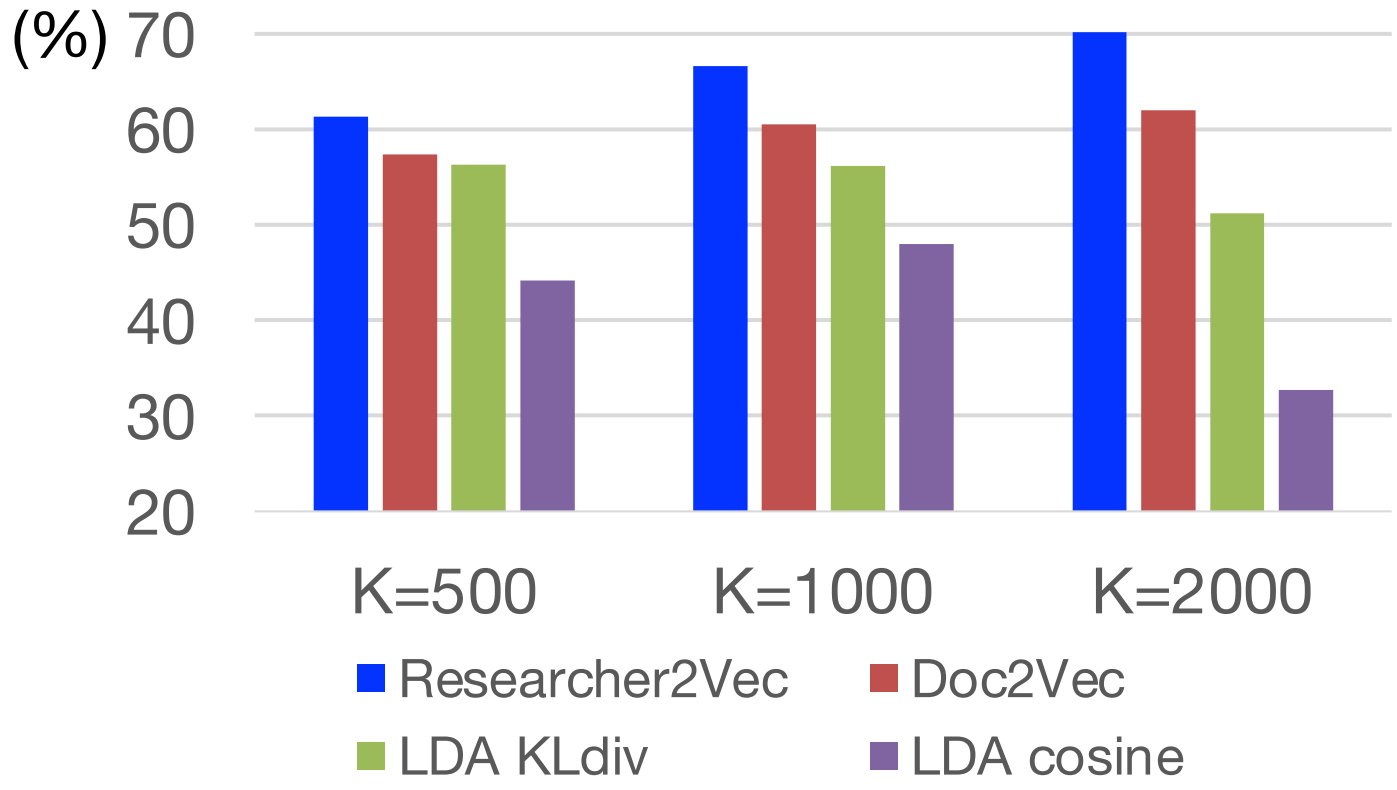

- yの要素はほとんど0なので、Ryの掛け算は結局、Rの対応する列を取り出してくればよい
- Rはディスクにmmap()しておけば、メモリ使用も最小(対応済み)

# 実験とデータ

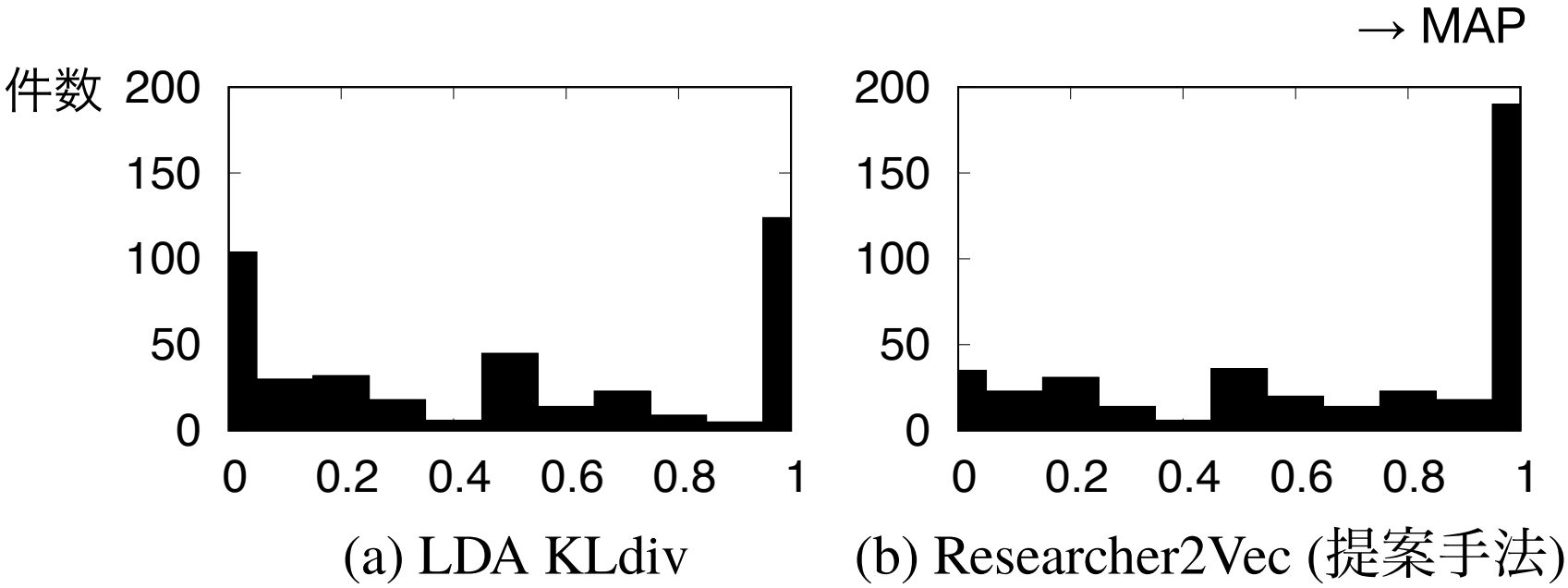
- 言語処理学会年次大会の20年分の論文データ (1995-2013) を使って実験
  - 20周年記念コーパスなので、ニューラル以前なのに注意
- MeCabで形態素解析し、語彙18,135語、論文4,082本で13,654,061語のデータ (1300万単語)
- この期間で5本以上の論文がある著者499人/3660人を実験の対象
- テストデータ500文書の著者を推定し、スコア順に並べた際の平均適合率 (Mean Average Precision, MAP)を計算
  - MAP=1：スコア最上位がすべて真の著者
  - Doc2Vec, LDAと比較

# 論文著者の推薦精度 (平均適合率)

- Doc2Vecを超えて、提案法が常に最高精度
  - LDAは、桂井ら(2016)の確率分布のコサイン類似度よりKLダイバージェンスで測った方がよい



# 平均適合率(MAP)の分布




(a) LDA KLdiv　　(b) Researcher2Vec (提案手法)

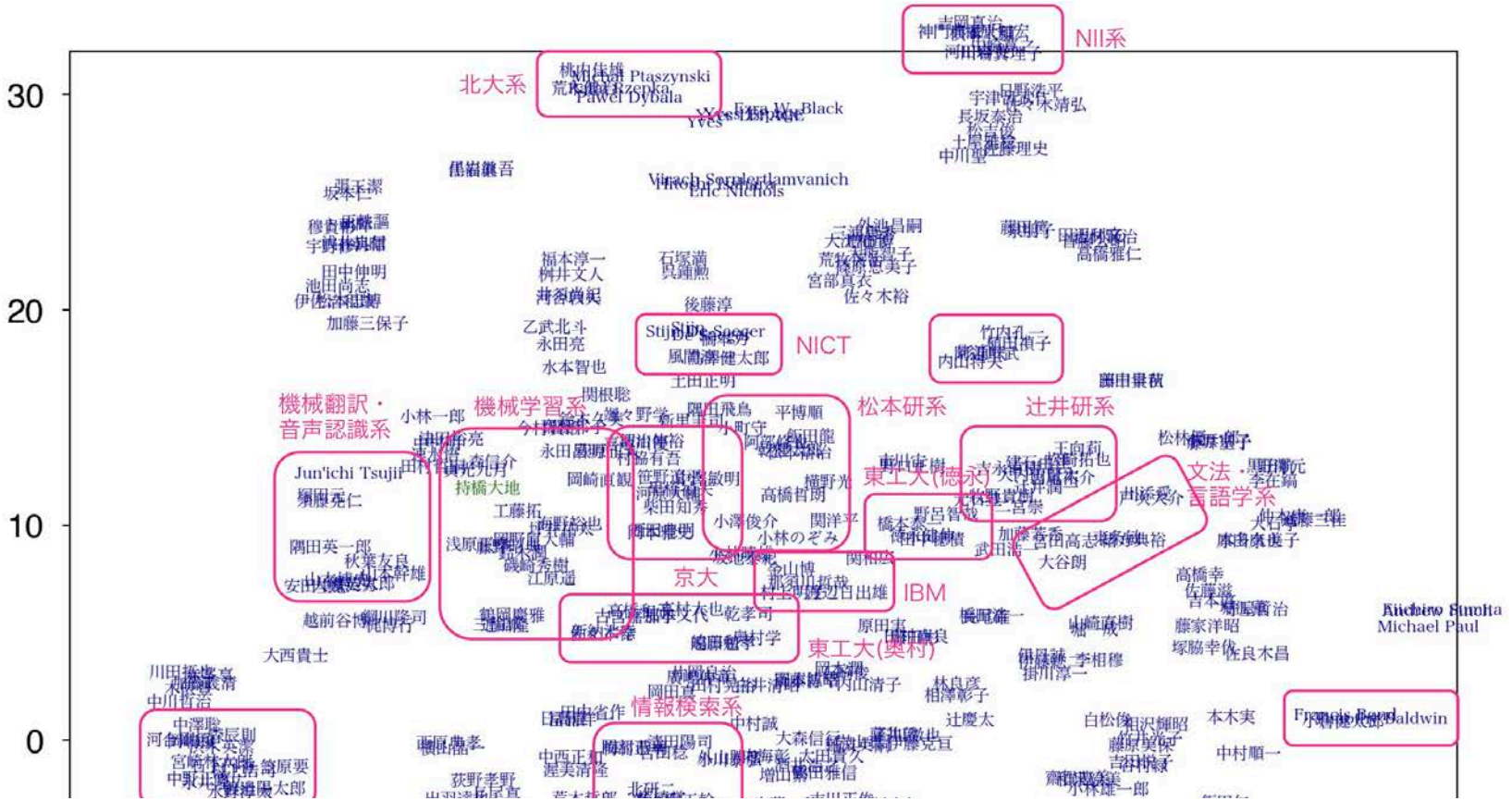
- 提案手法では、ほとんどの場合に平均適合率＝1
  → 著者をスコア順に並び替えたとき、真の著者が最上位を占める
- それ以外は、先生と興味が違う学生の論文や、英語論文

# デモ

http://clml.ism.ac.jp/nlp2vec/

- 公開サーバですので、誰でも検索を試すことができます
  - 言語処理学会の過去の論文のビューアにもなります (著者別)

# 研究者の可視化




- t-SNEによる可視化 (可視化は1通りではありません)
  - 詳しくは論文のAppendixを参照

# 将来の課題

- 今回は直感の働く自然言語処理分野の論文で試しただけで、他の分野へも適用可能
- arXivやACL anthologyの英語論文への適用
  - 査読者や講演者を探すのに実際困っている!
- Web周りが得意で(非常に)役に立つシステムを作りたい学生さんなどを募集しています

- 埋め込み空間での分布全体を考慮した研究者推薦 (kernel mean embedding?)
- 「ある分野全体をカバー」する研究者集合の求め方

# まとめ

- 論文からWord2vecと同等のニューラル文書ベクトルをSVDで高速に求め、それを書いた研究者の “研究者ベクトル”を計算する手法を提案
  - Doc2Vecの20〜40倍高速、解析的な検索解、省メモリ
  - Doc2VecおよびLDAを超えて最高精度
- 実際に言語処理学会の年次大会コーパスから、論文に含まれる単語で研究者を検索できるシステムを公開 (nlp2vec)
- 今後は、日本学術振興会で同様のシステムを開発したい