# Researcher2Vec: Neural Linear Model of Scholar Recommendation for Funding Agency

Daichi Mochihashi[1]

[1]*daichi@ism.ac.jp*
The Institute of Statistical Mathematics / Center for Science Information Analysis,
Japan Society for the Promotion of Science (JSPS)
5-3-1 Kojimachi, Chiyoda-ku, Tokyo 102-0083 Japan

## Abstract
This study proposes Researcher2Vec, which efficiently computes the neural document vectors of each paper using a simple linear algebra (SVD) to obtain a "researcher vector" based on the theory of Levy et al. (2014), who reported that the famous word2vec is equivalent to the matrix factorization. Researcher2Vec allows to objectively grasping and visualizing the expertise of each researcher in a low-dimensional space very efficiently. It also allows efficient search of researchers who are most associated with a given paper or a proposal as an analytical solution to ordinary least squares regression. Experimental results on academic paper datasets confirmed that our Researcher2Vec performs better than Doc2Vec and LDA in terms of both training speed and accuracy.

## Introduction
The number of papers and researchers has been constantly increasing as science and technology is advancing, thus rendering increasingly difficulty to grasp their full picture. This issue is particularly severe for funding agencies that receive numerous research proposals each year: 43,614 for NSF in the USA (2021), 100,084 for NSFC in China (2019) and 95,208 for JSPS in Japan (2021). Every proposal must be reviewed by expert researchers with subject matter expertise. However, finding such experts is challenging because even the most experienced funding manager has a limited scope over numerous candidates in various disciplines. Currently most funding agencies find relevant reviewers by areas and keywords; but this approach has a clear limitation because the agency does not examine the actual contents of the papers of each reviewer when they handling novel research proposals that typically do not have any existing authorities.
This problem is already noticed by the computer science community, which is mainly operated primarily by large international conferences with strict review processes. In 2021, NeurIPS received 9,122 submissions in machine learning and CVPR received 7,500 in image processing, both of which require over 20,000 reviews for each conference in a short period of time because each paper requires multiple reviewers. These peer review assignments of papers are no longer possible without the automatic scoring of appropriate candidates by the Toronto Paper Matching System (TPMS) (Charlin and Zemel 2013) that internally uses a probabilistic topic model, namely latent Dirichlet allocation (LDA) (Blei et al. 2003). However, LDA is insufficiently flexible as described below, and training LDA with numerous topics is computationally quite expensive. Springer Nature Reviewer Finder is a similar system; however, it basically operates on keywords and abstracts as well as being proprietary and not open to the public.
To solve this issue, this study proposes Researcher2Vec, which efficiently computes the neural document vectors of each paper through a simple linear algebra to obtain the “researcher vector” of each author. This allows representing the expertise of each researcher in a low dimensional embedding space very efficiently, and enables quickly searching the most relevant researchers for each proposal (or a paper) as an analytical solution to ordinary least squares (OLS) regression using word vectors as we show in this paper. Experimental results on the author prediction task indicate that Researcher2Vec is considerably superior to both Doc2Vec (Le and Mikolov 2014) and LDA in terms of computational speed and accuracy by a large margin.

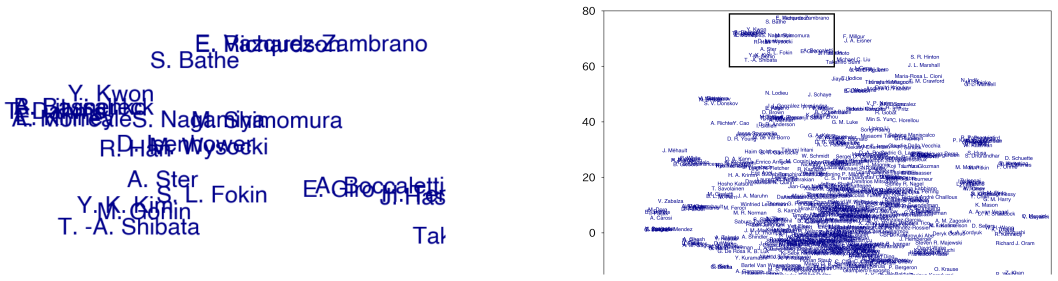

Figure 1: Visualization of embeddings of researchers, *i.e.* arXiv authors in the unarXive dataset (Saier and Färber 2020). The box in the right plot is zoomed into the left figure. Here, 100-dimensional "researcher vector" is mapped to 2D by *t*-SNE.

## Understanding Expertises of Researchers

Finding appropriate experts for a given problem, *i.e.* a paper or a funding proposal in our case, has been pursued as an "expert recommendation" in artificial intelligence (see Nikzad-Khasmakhi et al. (2019) for a recent review). It can be based on two approaches: (1) using meta-information of a paper such as citations, areas and keywords, and (2) analyzing the actual contents of the paper. Approach (1) is generally susceptible to subjective noise on the areas or keywords that are assigned by humans, and the amount of available information is limited in practice. Furthermore, citation information is not available for a new paper or a funding proposal.

By contrast, approach (2) directly models the scientific content of a paper, which enables more specialized recommendation of experts; in fact, TPMS follows this approach. Katsurai et al. (2016) performed LDA with $K$=500 topics on approximately 100,000 paper abstracts registered in Japan to show that LDA is superior to naive vector models, *i.e.* matching keywords, using probabilistic dimensionality reduction. We also trained a huge LDA of $K$=2000 topics on the full text of approximately 110,000 grant-in-aid proposals with 300 million words in total to help the funding agency to choose the appropriate reviewers for large grant proposals (JSPS, 2018). However, LDA is computationally intensive even if we employ the latest Gibbs sampler that has $O(1)$ computation per word (Yuan et al. 2015). Furthermore, it has the following theoretical limitations:

- It cannot deal with papers that have equally strong latent topics, because the latent topic distribution $\theta$ must sum to one;
- It cannot represent flexible (possibly negative) correlations between latent topics because each element of $\theta$ must be always positive.

In summary, LDA can estimate a "broad genre" of a paper, but cannot represent the fine-grained differences found in each research discipline.

## Document and Researcher Vectors by *Neural Linear Model*

To handle this problem, in this study documents (papers) and their authors were embedded in a more flexible real vector space for representation. Unlike neural networks such as Doc2Vec (Le and Mikolov 2014), the proposed method is equivalent to a neural network but allows a fast and efficient computation and recommendation through a simple linear algebra.

Levy et al. (2014) reported that the computation of word embeddings in word2vec (Mikolov et al. 2013) is equivalent to the matrix factorization with shifted pointwise mutual information (SPPMI) defined as follows. Let $n(w,c)$ be the number of times a word $w$ and a context word $c$ co-occurring within a surrounding window (say, within 10 words before and after $w$). Let the frequency of $w$ and $c$ be $n(w)$ and $n(c)$, respectively, and $N=\sum_w \sum_c n(w,c)$. Then the following (shifted) Positive PMI (PPMI)

$$\mathrm{PPMI}(w, c) = \max\left(\log \frac{n(w, c) \cdot N}{n(w) n(c)} - \log k, 0\right) \tag{1}$$

can be arranged into a matrix $\mathbf{Y}$ that is decomposed as $\mathbf{Y} \simeq \mathbf{W}\mathbf{C}^T$ by a singular value decomposition (SVD), as shown in Figure 2.

Each row of $\mathbf{W}$ can be shown to be mathematically equivalent to the word vector obtained by word2vec, specifically, skip-gram with negative sampling. Here, $\mathbf{C}$ is an auxiliary matrix of context vectors, $K$ is the embedding dimensions, and $k$ corresponds to the number of negative examples; however, Levy et al. (2015) experimentally reported that $k = 1$ , *i.e.* no shift, is optimal when using SVD. Therefore, we employed $k = 1$ hereafter.

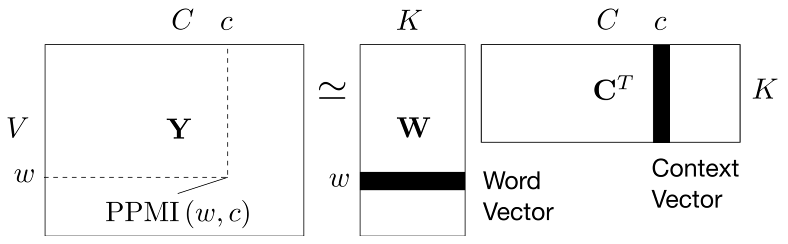

**Figure 2: Neural linear model of word vectors. Famous word2vec is equivalent to linear decomposition of PPMI matrix *Y* that can be easily obtained by co-occurrence statistics.**

Similarly, let $n(d, w)$ be the frequency of word $w$ in document $d$, and $n(d)$ be the number of words in $d$. Then, as shown in Figure 3, the document-word matrix $\mathbf{Y}$ that consists of $\mathrm{PPMI}(d, w)$ computed similarly as Equation (1) can be decomposed as

$$\mathbf{Y} \simeq \mathbf{D}\mathbf{W}^T \tag{2}$$

to yield a "document vector" $\vec{d}$ and a "word vector" $\vec{w}$ as each row of matrices $\mathbf{D}$ and $\mathbf{W}$, respectively. This document vector $\vec{d}$ is essentially equivalent to Doc2Vec, which is trained by a neural network. However, here it can be obtained by a simple linear algebra that allows fast computation over a vast number of documents without iterative optimization and tuning hyperparameters. The experiments demonstrated that this document vector, which we call DocVec in this paper, is excessively faster to compute compared with Doc2Vec and has a higher performance as it is a mathematically optimal solution. Moreover, it has an analytical solution for retrieval, which we describe below.

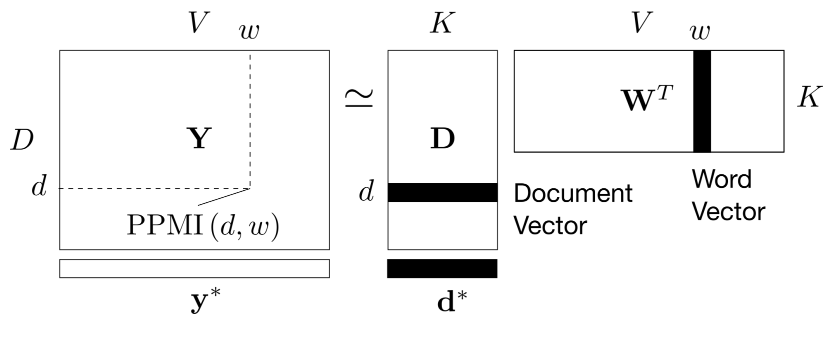

**Figure 3: Neural linear model of document vectors and word vectors as an extension to word2vec. The optimal decomposition is obtained from the SVD of PPMI matrix Y.**

*Computing Researcher Vectors*

Once the document vector $\vec{d}$ of each paper is obtained in this way, the embedding vectors of the researchers who wrote them can be computed. Let $\Omega_r$ denote the set of papers whose authors include researcher $r$, then the researcher vector $\vec{r}$ can be computed as the average of the document vectors of the papers in $\Omega_r$ as follows:

$$\vec{r} = \frac{1}{|\Omega_r|} \sum\nolimits_{\vec{d} \in \Omega_r} \vec{d} \tag{3}$$

where $\vec{d}$ is the associated row vector in $\mathbf{D}$ and $|\Omega_r|$ the size of $\Omega_r$, namely the number of papers researcher $r$ has authored. This implies that each researcher in the embedding space is represented as a $K$-dimensional Gaussian distribution to calculate its mean.

Figure 1 shows a part of an example of two-dimensional visualization by $t$-SNE of a $K = 100$-dimensional researcher vector $\vec{r}$ computed from a corpus of 100,000 arXiv papers used in the experiments. As no meta-information, such as areas and keywords, are included, this is purely an embedding computed from the technical contents of the papers of each author.

*Finding Researchers by Words*

Once the embedding vectors of researchers are obtained in this way, researchers who are closely related to the given paper or proposal can be determined. Unlike conventional methods, the proposed method directly models the paper contents written by each author instead of areas or keywords; thus even highly specialized words can be used to search for researchers related to them.

Let $q$ be a query, such as “reinforcement learning discourse” or an entire document. It can be regarded as a “document” $\mathbf{y}^*$ as shown in the bottom of Figure 3 where the corresponding embedding vector $\mathbf{d}^*$ exists. Given that the document vector and the researcher vectors exist in the same embedding space according to Equation (3), researchers can be searched by comparing the researcher vector $\vec{r}$ with the document vector $d^*$ once we know $\mathbf{d}^*$.

Since each element of $\mathbf{y}^*$ is a $\mathrm{PPMI}(w, q) = \max\left(\log \frac{p(w|q)}{p(w)}, 0\right)$, $\mathbf{y}^*$ is a sparse vector whose element is 1 when $w$ appeared in $q$ and otherwise 0. We want to find the $\mathbf{d}^*$ that approximates $\mathbf{y}^*$ in a least-squares sense:

$$\mathbf{y}^* \simeq \mathbf{d}^* \mathbf{W}^T \tag{4}$$

Equation (4) can be rewritten as $\mathbf{y}^* \simeq \mathbf{W}\mathbf{d}^*$ when by replacing $\mathbf{y}^*$ and $\mathbf{d}^*$ with their transpose; this is a well-known ordinary least squares (OLS) regression (Boyd and Vandenberghe 2018) whose optimal solution is given by

$$\mathbf{d}^* = (\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T\mathbf{y}^* \,. \tag{5}$$

When precomputing a regression matrix $\mathbf{R} = (\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T$ in advance, $\mathbf{d}^*$ is just given by

$$\mathbf{d}^* = \mathbf{R}\mathbf{y}^* \tag{6}$$

that is computed very efficiently. In Python, $\mathbf{R}$ can be computed without using explicit matrix inversion as `R=numpy.linalg.solve(numpy.dot(W.T,W),W.T)`. From Equation (6) and definition of $\mathbf{y}^*$, $\mathbf{d}^*$ is a sum of a few associated rows in $\mathbf{R}$; therefore, when $\mathbf{R}$ is memory mapped to disk using `mmap()`, memory consumption is also minimal in practice.

Once $\mathbf{d}^*$ is determined from query $q$, the search can be performed by displaying the researchers in the order of cosine distances between their researcher vectors $\vec{r}$ and $\mathbf{d}^*$.

## Experiments

*Materials and Methods*

To validate the proposed approach, experiments were conducted to predict the authors of a paper given its content. As a preliminary analysis, we used two kinds of corpora: (a) a corpus of natural language processing papers in Japan between 1995-2013, ranging over about twenty years, that contains 4,082 papers and 13,654,061 words in total. The size of the lexicon is $V$=18,135 with frequency $\geq$ 10, and (b) collection of 100,000 arXiv papers uniformly sampled from the unarXive dataset (Saier and Färber 2020). This larger dataset contains 460,050,053 words in total over multiple disciplines using 49,402 words of lexicon, and we uniformly dropped the first 100 words of each paper for anonymity. Given that several authors wrote only a few papers, we targeted (a) 499 authors out of 3,660 and (b) 13,989 authors out of 214,583, who wrote over five papers in this period for each dataset.

The papers were randomly split into 90% for training and 10% for test documents. For each case of test papers that contain the target authors, the score of the relevance of each target author was computed to find where the true author appeared in the sorted list of these scores using mean average precision (MAP) (Manning et al. 2008). MAP is higher for better-performing systems, with MAP=1 implying that the all the highest score authors are the true authors of that paper, and 0.5 implying that the predicted author comes second to the true single author out of 499 and 13,989 authors in our experiments.

*Baseline methods*

The reported Researcher2Vec model was compared with Doc2Vec and LDA-based recommendations. For Doc2Vec, a standard implementation in `gensim` was employed. For a fair comparison we used "distributed bag of words" model with hyperparameters epochs=100, hs=0, alpha=0.025, and sample=1e-5. Because Doc2Vec does not have an analytical solution for a new document, $\mathbf{d}^*$ is computed iteratively by `infer_vector()` function of the package. For the case of LDA, we used our open Cython implementation using Gibbs sampling for a better performance. After 1,000 MCMC iterations over training documents, the optimal latent topic distribution $\theta$ for each test document was computed by the variational EM algorithm (Blei et al. 2003) to compare with average topic distribution $\vec{r}$ of each author obtained similarly as Equation (3). Katsurai et al. (2016) computed a cosine distance between $\theta$ and $\vec{r}$ ("LDA cosine"); however, this is inappropriate because they are probability distributions in this case. Therefore, we also conducted experiments using a Kullback-Leibler divergence

$$\mathrm{KL}(\theta||\vec{r}) = \Sigma_{k=1}^{K}\theta_k \log \theta_k/r_k \propto -\Sigma_{k=1}^{K}\theta_k \log r_k \quad (7)$$

as "LDA KLdiv".

*Results*

Table 1 lists the MAP of the author prediction task accompanied by the training time for each model. The proposed Researcher2Vec was almost always superior to Doc2Vec and LDA in performance over different embedding dimension $K$, while training our model is extremely fast using SVD. The distribution of the MAP scores for the NLP papers is shown in Figure 4; for the proposed model, MAP was concentrated to 1.0 (*i.e.* perfect prediction) in most cases, whereas the other noisy cases include English (different language) papers and a student paper whose research interest differs from his supervisor as the last author. From the mathematical property of SVD, $K$ should be smaller than $D$ (=3582), and MAP of $K$=3000 was 71.6%.

**Table 1: Average performance (MAP) of author prediction. Researcher2Vec consistently achieved better prediction with very fast training time. Results of LDA on the arXiv dataset are omitted because it required too long training time (over >4 days) even with efficient $O(1)$ sampler (Yuan et al. 2015). Training time represents the case of $K$=1000.**

| *Model \ K* | 500 | 1000 | 2000 | *Time* |
|---|---|---|---|---|
| Researcher2Vec | **61.3%** | **66.6%** | **70.2%** | **1m** |
| Doc2Vec | 57.4% | 60.5% | 62.0% | 21m |
| LDA KLdiv | 56.3% | 56.2% | 51.2% | 14h48m |
| LDA cosine | 44.2% | 48.0% | 32.7% | 14h48m |

(a) NLP papers (small dataset)

| 500 | 1000 | 2000 | *Time* |
|---|---|---|---|
| 43.3% | **47.2%** | **50.3%** | **38m** |
| **45.2%** | 46.8% | 49.4% | 5h32m |

(b) arXiv papers (large dataset)

## Conclusion

In this study, a system that can visualize and search researchers by the contents of their papers was constructed and evaluated. This system computes document vectors equivalent to neural methods using efficient SVD, and allows efficient search of researchers as an analytical solution to ordinary least squares. Experimental results revealed that the proposed method is very efficient and exhibits higher author prediction performance compared with Doc2Vec and latent topic models. In the future, we aim to consider applying the proposed method on broader datasets like Scopus, which covers wider range of disciplines than used in this paper.

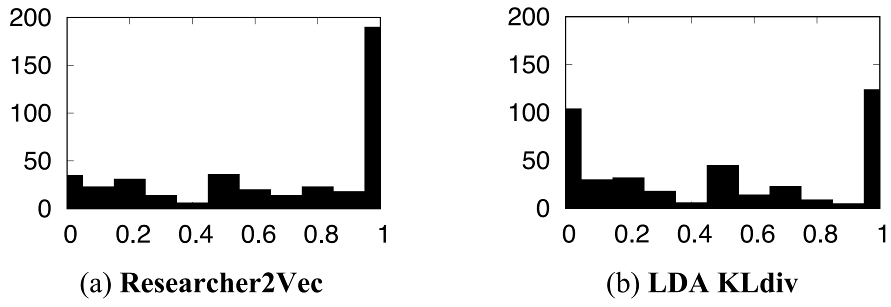

**Figure 4: MAP accuracies of the author prediction for NLP papers. In many cases, the proposed Researcher2Vec achieved perfect prediction (MAP=1) of the authors as compared to LDA.**

## References

David M. Blei, Andrew Y. Ng (2003). and Michael I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Stephen Boyd and Lieven Vandenberghe (2018). *Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares*. Cambridge University Press.

Laurent Charlin and Richard S. Zemel (2013) The Toronto Paper Matching System: An automated paper-reviewer assignment system. In *ICML 2013*.

Jinhui Yuan et al. (2015). LightLDA: Big Topic Models on Modest Computer Clusters. In *WWW 2015*, pages 1351–1361.

Japan Society for the Promotion of Science (2018). *Annual Report of Activities in 2017*, Center for Global Science Information (CGSI) Report No.7 (in Japanese).

Marie Katsurai, Ikki Ohmukai, and Hideaki Takeda (2016). Topic Representation of Researchers' Interests in a Large-Scale Academic Database and Its Application to Author Disambiguation. *IEICE Transactions on Information and Systems*, E99-D(4):1010-1018.

Quoc Le and Tomas Mikolov (2014). Distributed Representations of Sentences and Documents. In *ICML 2014*, pages 1188–1196.

Omer Levy and Yoav Goldberg (2014). Neural Word Embedding as Implicit Matrix Factorization. In *Advances in Neural Information Processing Systems 27*, pages 2177–2185.

Omer Levy, Yoav Goldberg, and Ido Dagan (2015). Improving Distributional Similarity with Lessons Learned from Word Embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze (2008). *Introduction to Information Retrieval*. Cambridge University Press.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean (2013). Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

N. Nikzad-Khasmakhi, M.A. Balafar, and M. Reza Feizi-Derakhshi (2019). The state-of-the-art in expert recommendation systems. *Engineering Applications of Artificial Intelligence*, 82:126–147.

Tarek Saier and Michael Färber (2020). unarXive: a large scholarly data set with publications. *Scientometrics*, 125:3085–3108.