Scholarfy: Recommendation system for scientific publications

Titipat Achakulvisut, Daniel E. Acuna, Konrad Kording

Rehabilitation Institute of Chicago and Northwestern University

**Abstract**

An important part of being a professional scientist is to be able to keep a current account of the scholarly material. This task, however, has become increasingly impossible to perform manually due to the large number of publications produced. We could improve this process by using readily available algorithms that are known to work relatively well on music and movie recommendations. However, how these algorithms work with scholarly material has largely been unexplored. Here, we develop a Python library that implements a recommendation system based on the content of voted relevant and irrelevant articles. The library provides real-time and accurate suggestions of new articles based on these votes and it scales to large corpus. We tested the library on 15K posters from the Society of Neuroscience Conference 2015. We tuned the algorithm to provide suggestions that were closed in topic classifications manually provided by editors. We show that our algorithm significantly outperforms suggestions based on keywords while being similarly fast. The work presented here promises to make the exploration of scholarly material faster and more accurate than what it is possible today.

# Introduction

Since the inception of Internet and large online retailers, recommendation systems have automated suggestions for users based on their preferences. They have been proven useful for music, movies, news, and retail in general [1]. In scientific literature search, however, researchers rely mostly on author-provided keywords and citations to find new material. These sources of information hinder their exploration because they are known to be poor and potentially biased [2]. This problem is more prominent during conferences where appropriate keywords may not even exist, let alone citations. An application of recommendation systems based on the researcher’s preferences thus promises to speed literature search and increase relevance of the findings.

There are multiple recommendation systems that use either the personal preferences of a new user (e.g., content-based recommendations) or bootstrap the similarity between the new users’ preferences and previous users’ preferences (e.g., collaborative filtering). A large portion of this research is available in commercial software, such as news [3], movie [4], and music [5] applications. There are also more specific systems for scientific literature search. In [6], the authors present a content-based recommendation system that works on PubMed datasets. In [7], *Scienstein* system combines a large set of criteria for providing literature recommendation. In [8], the authors present a topic-based recommendation system based on a Latent Dirichlet Allocation (LDA) model. It is unclear, however, how these systems scale and the test of their performance is generally indirect.

Here we introduce Scholarfy (http://www.github.com/titipata/scholarfy), an open source Python library that implements a fast and accurate recommendation system for literature search. Briefly, the library uses a scalable vectorization of documents through online Latent Semantic Analysis (LSA) [1]. For the recommendation part, it pairs the Rocchio Algorithm [1] with a large-scale approximate nearest neighbor search based on ball trees [9]. The library aims at providing content-based recommendations using only a user’s votes rather than collaborative filtering. The Scholarly software, then, provides an open source solution to content-based scientific recommendation.

We tune the parameters of the algorithm to fit the statistical nature of a large set of scientific literature. First, we tune the LSA modeling and find the optimal number of SVD components. Second, we test the accuracy of the recommendations. We use around 150K documents from the Pubmed Open Access and obtain their expertly curated keywords known as Medical Subject Headings (MeSH). We tuned the Rocchio parameters to recommend items close in MeSH tag space. We additionally tested the algorithm on a broader dataset from CiteULike and found similar results. The algorithm implementation is available online at <http://sf.scienceofscience.org> where we use data from Society for Neuroscience (SfN) conference ([http://www.sfn.org](http://sfn.org)) and Neural Information Processing System (NIPS) confernece (http://nips.cc).

# Design and Implementation

*Scholarfy* is made to be a repository for general purpose of making a suggestion model based on list of abstracts specifically scientific abstract. Scholarfy provides 4 main algorithm to create suggestion algorithm including 1) Text preprocessing 2) Term frequency-Inversed document frequency (tf-idf) transformation 3) Latent Semantic Analysis 4) *Rocchio Algorithm and Nearest Neighbor Assignment*

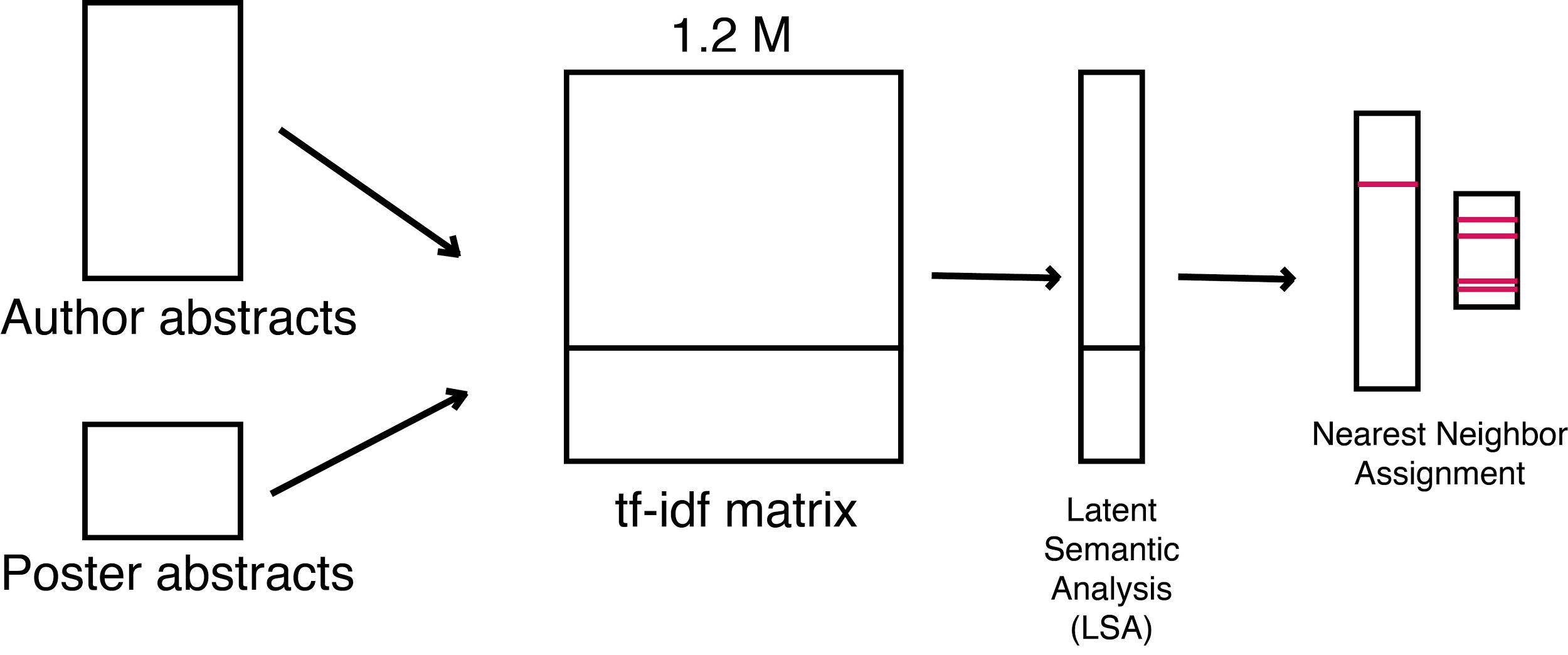


Fig. 1 The schematic of Scholarfy is provided in fig 1. (schematic of whole process)

## Text Preprocessing Part

Words from abstract is tokenized then transform all unicode character to ascii format. Also, we apply Snowball Stemmer [cite] to tokenized word in order to unify some of the words such as neuron and neurons.

We choose distance in topic space and allow feedback using Rocchio Algorithm in this particular problem to solve the problem 1) LSA based on singular value decomposition which is scalable to big dataset 2) Rocchio algorithm is basically vector addition and subtraction which is fast and intuitive 3) The recommendation part is based on nearest neighbor which is scalable for larger dataset also.

## Term frequency-Inversed document frequency transformation

Tokenized words from will then transform into sparse matrix so called tf-idf matrix. Each row will represent each documents or abstracts and column represent dictionary of words. Stop words in English are also removed before the transformation.

tf = ...., idf = ... , tf-idf = …

## Latent Semantic Analysis

Latent semantic analysis (LSA) is a topic modeling technique based on singular value decomposition. We can represent tf-idf matrix (X) as product of left singular vector (U), diagonal singular value matrix (S) and right singular vector (V) i.e. X = USV where diagonal singular value are ranked from highest to lowest value.

LSA will select only high singular value with corresponded left and right singular vector which will give lower dimension X\_lsa = U’S’V’. Here each row or LSA matrix will represent each document vector (so called poster vector) corresponded to original plain abstract.

## Rocchio Algorithm and Nearest Neighbor Assignment

Rocchio algorithm is an algorithm that is incorporate to let scientists or user to define relevant and non-relevant documents so we can compute user preference based on the selected documents. User preference vector can be computed by following,

import numpy as np

import pandas as pd

import scholarfy as sf

abstracts = ['Sciene of science ...', 'is ...', 'awesome ...']

abstracts\_preprocess = map(lambda abstract: sf.preprocess(abstract), abstracts)

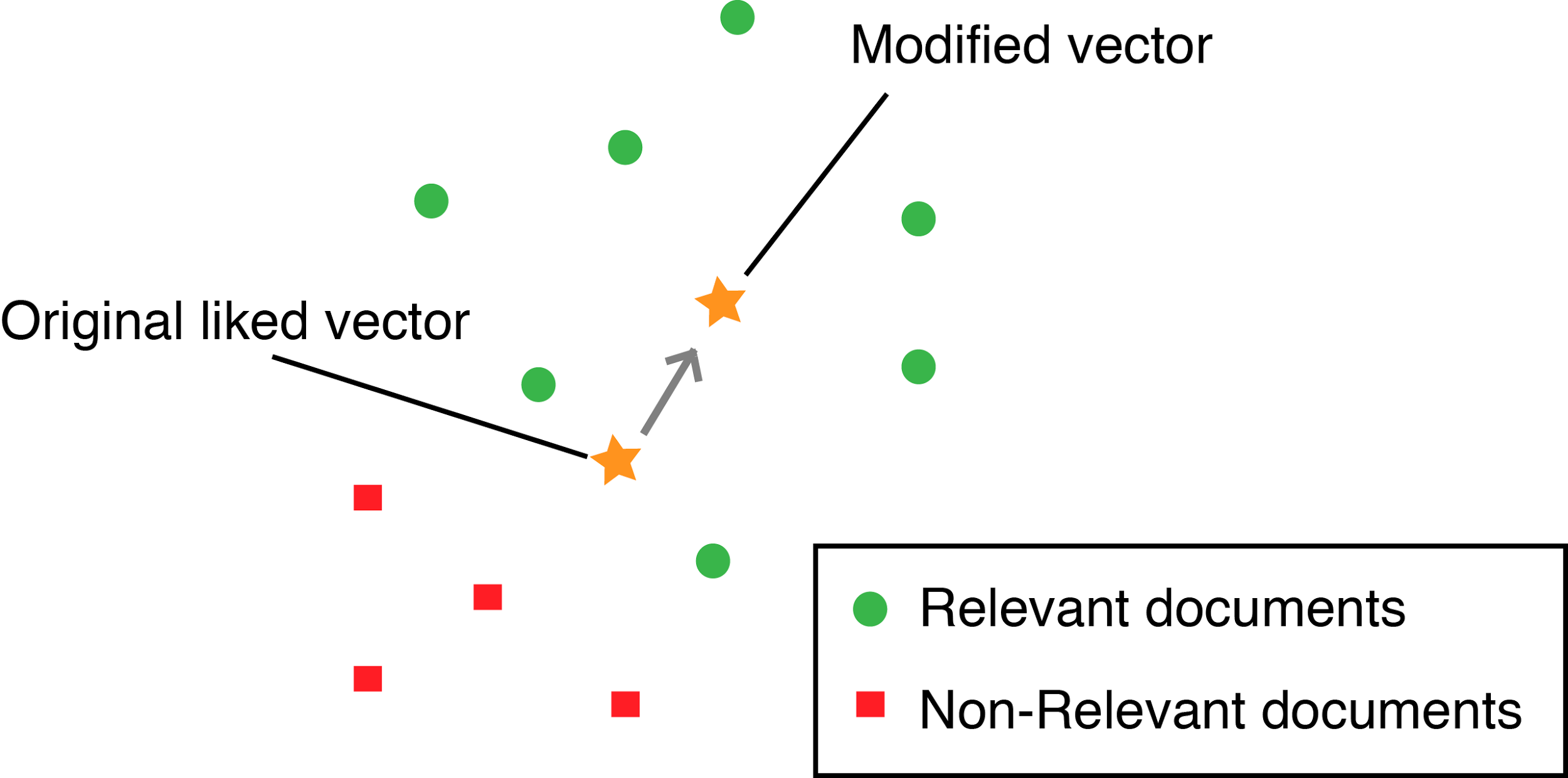
# stemming string

tfidf\_matrix = sf.tfidf\_vectorizer(abstracts\_preprocess)

# convert to tf-idf matrix

poster\_vect = sf.svd\_vectorizer(tfidf\_matrix, n\_components=200, n\_iter=150) nbrs\_model = sf.build\_nearest\_neighbors(poster\_vect)

Nearest neighbor will then performed after we get user preference vector to give user feedback of related documents. We employ nearest neighbor model from Python scikit-learn which is very fast (takes about 50 milliseconds to perform one nearest neighbor)



**Fig 2.** Schematic of Rocchio Algorithm

# Result

## Materials/ Dataset Provided

We obtain 150k abstracts from Pubmed Open Access subset in XML format [cite] from year 2006 to 2014. We found around 30k overlap abstracts from Medline database where we can find MeSh tags. We then parse information from XML format to Python Pandas dataframe format using Pubmed parser <http://github.com/titipata/pubmed_parser>. Data is available in Python Pandas pickle format [cite] which can be downloaded through the repository.

## Optimizing SVD components

Even though the suggestion part can be done in real time, singular value decomposition is still a batch computation where we need to compute topic vectors from plain abstract which require computation power. The main

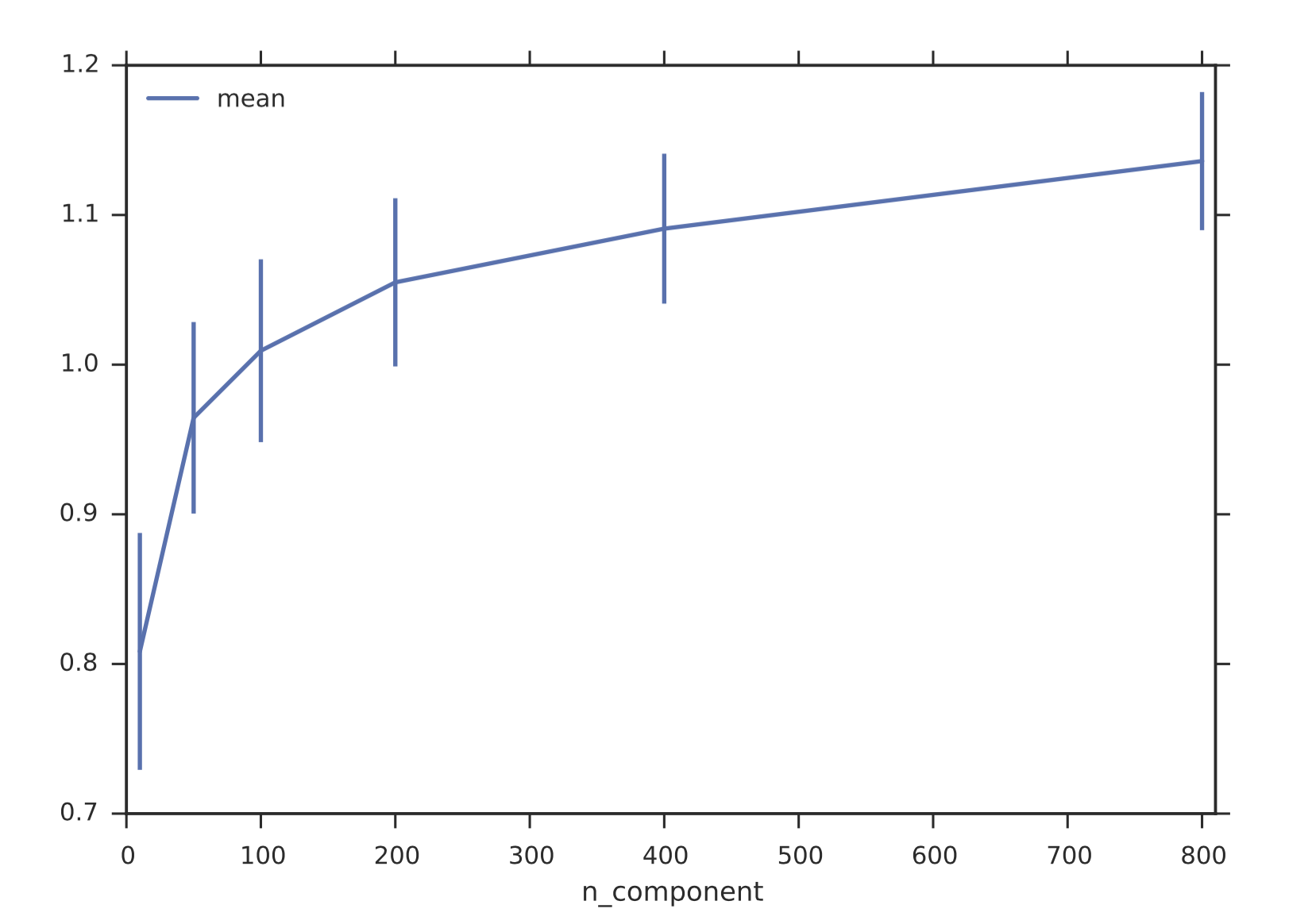


Fig 3. a) log likelihood vs number of SVD components b) number of documents vs. total memory usage

## Optimizing Rocchio algorithm based on MeSH distance

Medline and Pubmed provide Medical Subject Heading (MeSH) terms for article. Here we show that the suggesting perform very well and suggest article that have close MeSH terms to the original article.

Fig 4. Optimizing parameter for Rocchio algorithm

# Availability and Future Directions

Scholarfy is an open-source software under MIT license aim to develop algorithm for suggesting new articles based on user feedback. Therefore, we aim to make both clean dataset publicly available/downloadable through repository and also make the repository more variable. We will add more variability to the algorithm e.g. adding preprocessing function, using customized topic distance such as Latent Dirichlet Allocation (LDA) or Principle component analysis.

**Acknowledgements**

Titipat Achakulvisut was supported by Royal Thai Government Scholarship grant #50AC002. DAniel E. Acuna was supposed by the John Templeton Foundation grant #. Tulakan Ruangrong would like to thank… Konrad Kording was supposed by NIH # and John Templeton Foundation #. Thanks to SfN for providing the abstract dataset for testing.

**Author Contributions**

Titipat Achakulvisut (writing, programming, concept), Daniel Acuna (writing, programming, concept), Tulakan Ruangron (programming, concept), Konrad Kording (writing, concept).

**References**

1. Manning, C.D., P. Raghavan, and H. Schütze, *Introduction to information retrieval*. 2008, New York: Cambridge University Press. xxi, 482 p.

2. Chavalarias, D. and J.P. Ioannidis, *Science mapping analysis characterizes 235 biases in biomedical research.* J Clin Epidemiol, 2010. **63**(11): p. 1205-15.

3. Li, L., et al. *A contextual-bandit approach to personalized news article recommendation*. in *Proceedings of the 19th international conference on World wide web*. 2010. ACM.

4. Bell, R.M. and Y. Koren, *Lessons from the Netflix prize challenge.* ACM SIGKDD Explorations Newsletter, 2007. **9**(2): p. 75-79.

5. Ali, M., C.C. Johnson, and A.K. Tang. *Parallel Collaborative Filtering for Streaming Data*. 2011; Available from: <https://github.com/MrChrisJohnson/CollabStream>.

6. Yoneya, T. and H. Mamitsuka. *PURE: a PubMed article recommendation system based on content-based filtering*. in *Genome Inform*. 2007. World Scientific.

7. Gipp, B., J. Beel, and C. Hentschel. *Scienstein: A research paper recommender system*. in *Proceedings of the international conference on emerging trends in computing (ICETiC’09)*. 2009.

8. Wang, C. and D.M. Blei. *Collaborative topic modeling for recommending scientific articles*. in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2011. ACM.

9. Shakhnarovich, G., T. Darrell, and P. Indyk, *Nearest-Neighbors methods in Learning and Vision: Theory and Practice*. 2006: MIT Press.