


Automated Reviewer Assignment: Tutorial and Demonstration

Nihar B. Shah

Machine Learning Department &
Computer Science Department

Carnegie Mellon University

A man in a dark suit stands on a dark platform, viewed from behind. He is looking down at a vast, endless sea of yellow Minions that stretches to the horizon. The sky is a mix of purple and blue. A speech bubble originates from the man.

My computer is going
to assign 3 proposals to
each of you...

Slides available on PRUR website

Quick links

- [Home](#)
- [Important dates](#)
- [SOC/LOC](#)
- [Invited Speakers](#)
- [Code of Conduct](#)
- [Registration](#)
- [Registration Fee Payment](#)
- [Programme \[enodo\]](#)
- [Participants](#)
- [Accommodation](#)
- [Travel Information](#)
- [Local and Practical Information](#)

Wednesday February 8 th	
	SESSION II: METHODOLOGIES
09:00 - 10:30	Nihar Shah: "Automated Reviewer Assignment: Tutorial and Demonstration" (Hands-on session)
10:30 - 11:00	Coffee & Tea
11:00 - 11:40	Rachel Heyard: "Let's talk about uncertainty in funding allocation" (K)

References and links on these slides are clickable

Computer Science Conferences...

- review full papers, and not just abstracts
- reputation at least at par with journals
- are typically a terminal venue for publication



Process is similar to distributed proposal reviews

Growth of submissions in astronomy

[Peer Review Under Review](#)



[Peer Review Under Review](#)

[February 6 - 10, 2023](#)

[ESO Garching](#)

“staggering increase of publications and proposals (doubling every 14 years in astronomy)”

Many aspects of modern peer review have not changed from its inception in the 18th century despite drastic changes in the scientific community. Specifically, contrary to the early days of peer review, it has become a significant challenge to identify experts who can effectively review the more and more specialized fields of science.

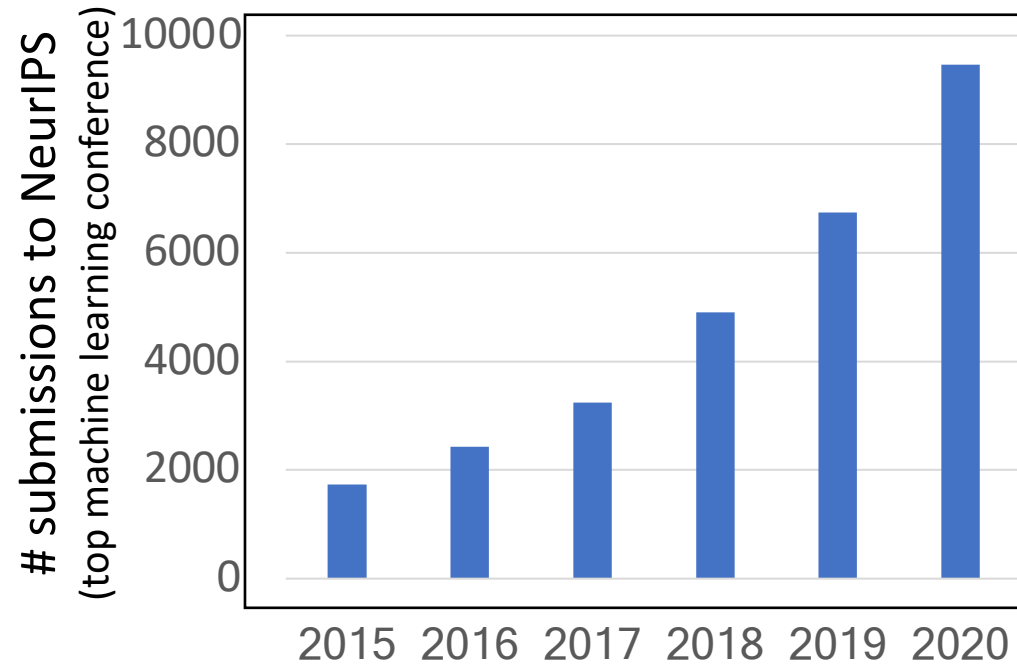
The problem is exacerbated by the ever-rising number of researchers (having grown by 15% between 2014 and 2018 according to a UNESCO report) also seen through the staggering increase of publications and proposals (doubling every 14 years in astronomy). Some say that peer review has not adequately innovated as technology has advanced and the dissemination of publications has surged, creating a space for stagnant and biased reviews.

In this workshop, we aim at bringing together experts from a large number of organizations and facilities (ESO, ESA, ALMA, STScI, NASA, NOIRLab) to discuss the state of peer review and the ways forward for a digital and interconnected science community. The workshop will be divided in 4 main sessions (Peer review at large, Methodologies, Diversity, Equity and Inclusion in peer review, and Concrete examples) and it will include ample time for discussion.

The workshop will take place at the ESO Headquarters in Garching (Germany) from February 6 to 10, 2023.

Conference email: peer_review_conf@eso.org

Meanwhile in my field...



5-fold increase in 5 years

Automation in reviewer assignment helps with scale

Used for 3 decades in computer science

Stages

(0) Reviewer pool

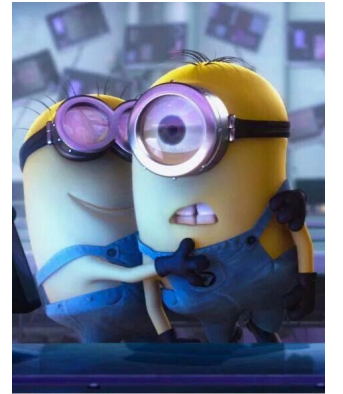
Construct a pool of potential reviewers



(1) Compute similarities

For every (proposal, potential reviewer) pair, compute a “similarity score”

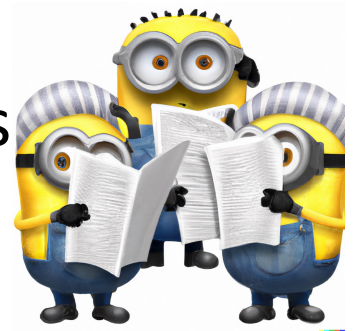
- Topic intersection
- Text matching (demo)
- Bidding



(2) Assignment

Use similarity scores to assign reviewers to proposals

- Sum-similarity algorithm
- PeerReview4All algorithm (demo)



Stages

(0) Reviewer pool

Construct a pool of potential reviewers



(1) Compute similarities

For every (proposal, potential reviewer) pair, compute a “similarity score”

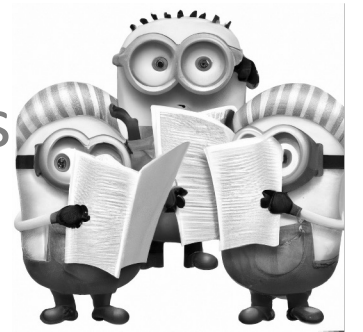
- Topic intersection
- Text matching (demo)
- Bidding



(2) Assignment

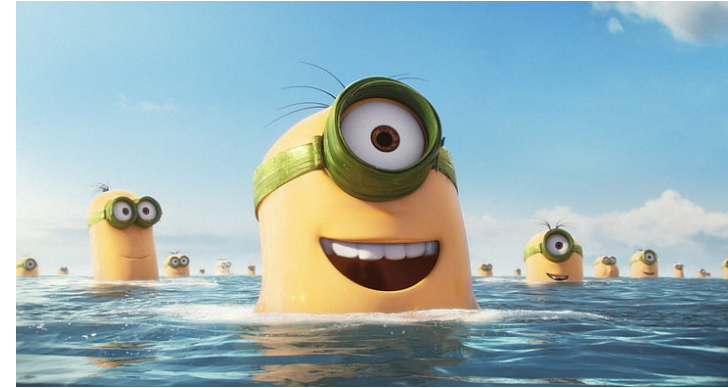
Use similarity scores to assign reviewers to proposals

- Sum-similarity algorithm
- PeerReview4All algorithm (demo)



Construct pool of potential reviewers

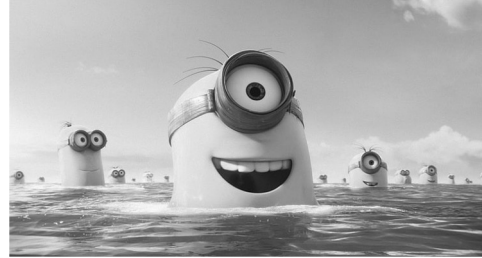
- **Invited based on** publication history, previous reviewing experience, references, open calls
- **Obtain their published papers**
 - Scrape: usually freely available in computer science
 - Ask reviewers to submit their relevant papers
- **Obtain conflicts of interest (Col) information**
 - Co-author information from publication history
 - Reviewers and authors asked to provide conflict domains (e.g., “cmu.edu”) and other individual conflicts



Stages

(0) Reviewer pool

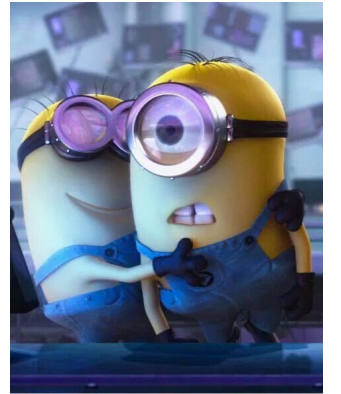
Construct a pool of potential reviewers



(1) Compute similarities

For every (proposal, potential reviewer) pair, compute a “similarity score”

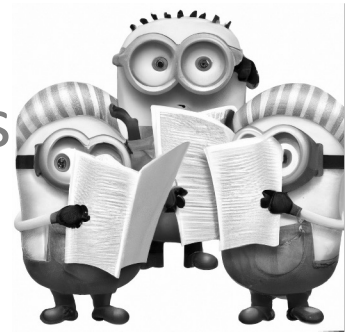
- Topic intersection
- Text matching (demo)
- Bidding



(2) Assignment

Use similarity scores to assign reviewers to proposals

- Sum-similarity algorithm
- PeerReview4All algorithm (demo)



(1)

Compute
similarities

Topic intersection

- Pre-defined list of topics



- ☐ Bananas
- ☒ Apples
- ☐ Potatoes
- ☐ Plums
- ☒ Cauliflower
- ⋮

- Authors select topics pertaining to their proposal
- Reviewers select topics of their expertise

Topic similarity = amount of intersection in selections

- **Challenges:**
 - Coarse granularity
 - Manually decided topics may not be exhaustive

(1)

Compute
similarities

Text matching

Submitted proposal:

We propose to find exoplanets where the atmosphere rains bananas or if there are bananas in the ground. Our methods include the design of novel telescopes that can detect even the tiny amounts of banana radiation...

Potential reviewer's published paper:

In this paper, we study the banana-atmosphere phenomenon on the moon. Using a prototype version of our banana-radiation detectors, we are able to sniff banana peels from the moon's core...

Objective: Compute similarity between these texts

Studied extensively in computer science

- Topic models / latent semantic indexing
 - Dumais et al. 1992, Mimno et al. 2007, Anjum et al. 2019
- Term frequency - inverse document frequency (TF-IDF)
 - Basu et al. 1999, Hettich and Pazzani 2006, Charlin et al. 2013
 - Open source implementation: <https://github.com/niharshah/TFIDFsimilarity>
 - Recently rediscovered in astrophysics [Kerzendorf et al. 2020]
 - https://github.com/wkerzendorf/deepthought_web
- Deep-learning based language models
 - Peters et al. 2018, Wieting et al. 2019, Cohan et al. 2020
 - <https://github.com/openreview/openreview-expertise>

(1)

Compute
similarities

TF-IDF matching: High-level idea

Submitted proposal:

We propose to find exoplanets where the atmosphere rains bananas or if there are bananas in the ground. Our methods include the design of novel telescopes that can detect even the tiny amounts of banana radiation...

Potential reviewer's published paper:

In this paper, we study the banana-atmosphere phenomenon on the moon. Using a prototype version of our banana-radiation detectors, we are able to sniff banana peels from the moon's core...

- How frequently do words occurring in the submitted proposal also occur in the potential reviewer's published paper(s)?

(1)

Compute
similarities

TF-IDF matching: High-level idea

Submitted proposal:

We propose to find exoplanets where the atmosphere rains bananas or if there are bananas in the ground. Our methods include the design of novel telescopes that can detect even the tiny amounts of banana radiation...

Potential reviewer's published paper:

In this paper, we study the banana-atmosphere phenomenon on the moon. Using a prototype version of our banana-radiation detectors, we are able to sniff banana peels from the moon's core...

- How frequently do words occurring in the submitted proposal also occur in the potential reviewer's published paper(s)?
- Some words are common across a large number of proposals and reviewers. Downweight these words.

TF-IDF matching: High-level idea

Submitted proposal:

We propose to find exoplanets where the atmosphere rains **bananas** or if there are **bananas** in the ground. Our methods include the design of novel telescopes that can detect even the tiny amounts of **banana** radiation...

Potential reviewer's published paper:

In this paper, we study the **banana**-atmosphere phenomenon on the moon. Using a prototype version of our **banana**-radiation detectors, we are able to sniff **banana** peels from the moon's core...

- How frequently do words occurring in the submitted proposal also occur in the potential reviewer's published paper(s)?
- Some words are common across a large number of proposals and reviewers. Downweight these words.
- High intersection of words not common across proposals/reviewers contributes most.

(1)

Compute
similarities

TF-IDF matching: High-level idea

Submitted proposal:

We propose to find exoplanets where the atmosphere rains bananas or if there are bananas in the ground. Our methods include the design of novel telescopes that can detect even the tiny amounts of banana radiation...

Potential reviewer's published paper:

In this paper, we study the banana-atmosphere phenomenon on the moon. Using a prototype version of our banana-radiation detectors, we are able to sniff banana peels from the moon's core...

- How frequently do words occurring in the submitted proposal also occur in the potential reviewer's published paper(s)?
- Some words are common across a large number of proposals and reviewers. Downweight these words.
- High intersection of words not common across proposals/reviewers contributes most.

Text matching: Demonstration of web-based tool

<http://opensupernova.org:9999/>



Vicente Amado Olivo



Wolfgang Kerzendorf

Deepthought Reviewer

Find Reviewers in Astronomy

Find Expertise for given text:

Example Text: It is widely believed that Type Ia supernovae (SNe Ia) originate in binary systems where a white dwarf accretes material from a companion star until its mass approaches the Chandrasekhar mass and carbon is ignited in the white dwarf's core. This scenario predicts that the donor star should survive the supernova (SNe) explosion, providing an opportunity to understand the progenitors of SNe Ia. In this paper, we argue that rotation is a generic signature expected of most nongiant donor stars that is easily measurable. Ruiz-Lapuente et al. examined stars in the center of the remnant of SN 1572 (Tycho SN) and showed evidence that a subgiant star (Star G by their naming convention) near the remnant's center was the system's donor star. We present high-resolution (R $\sim 40,000$) spectra taken with the High Dispersion Spectrograph on Subaru of this candidate donor star and measure the star's radial velocity as $79 \pm 2 \text{ km s}^{-1}$ with respect to the local standard of rest and put an upper limit on the star's rotation of 7.5 km s^{-1} . In addition, by comparing images that were taken in 1970 and 2004, we measure the proper motion of Star G to be $\mu_l = -1.6 \pm 2.1 \text{ mas yr}^{-1}$ and $\mu_b = -2.7 \pm 1.6$

Find referees for given text

or

1207.4481

Find referees for given arXiv id

Deepthought Reviewer

Find Reviewers in Astronomy

Expert Identifier	Similarity	Inspire Link
W.E.Kerzendorf.1	0.74	link
S.Justham.1	0.57	link
P.Podsiadlowski.1	0.57	link
B.P.Schmidt.1	0.56	link
R.P.Kirshner.1	0.56	link
S.J.Smartt.1	0.56	link
J.J.Eldridge.1	0.56	link
Nidia.I.Morrell.1	0.56	link
J.Vinko.1	0.56	link
P.Challis.1	0.56	link
P.Podsiadlowski.2	0.55	link
S.Jha.1	0.55	link

Text matching: Demonstration of code

- <https://github.com/niharshah/TFIDFsimilarity>

 TFIDFsimilarity.py

- You may have to install [nltk package](#) “pip install nltk”
- Create two folders:
 - "Paper_Folder": Contains text files where each text file contains the text of a paper
 - "Reviewer_Folder": Contains text files where each text file contains the text of a potential reviewer's past papers
- `compute_similarities(Reviewer_Folder, Paper_Folder)`
- Outputs: reviewer list, author list, similarity matrix
- Example:

```
➤ from TFIDFsimilarity import compute_similarities  
➤ (reviewer_idx, paper_idx, similarity_matrix) = compute_similarities("r", "p")
```


(1)

Compute
similarities

Our independent evaluations

Objective: Collect a publicly releasable “gold standard” dataset (to be released soon)

Method:

- Ask researchers to think of 10 papers they read from the past year
- Give ranking of these papers in terms of their expertise
- Also give ratings in terms of their expertise

Results:

- “SPECTER+MFR” algorithm is the best of the deep-learning based algorithms
 - Uses only title + abstract
- TF-IDF does worse if it uses only title+abstract
- “TF-IDF with full text” performance similar to “SPECTER+MFR with title+abstract”

(1)

Compute
similarities

Text matching: Challenges

- Does get confused (topic of active research)
- Black box (we are designing “explanation” methods)
- Deep-learning based “modern” language models
 - Needs more time and computation
 - Can often handle only abstracts and not full papers/proposals

(1)

Compute
similarities

Bidding

- Reviewers shown list of all proposals
 - May also be shown the computed similarities
- Can “bid” to review or not review any proposal

Proposal title	Not willing to review	Indifferent	Eager to review
<i>Finding bananas in exoplanets</i>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<i>Large apple collider</i>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<i>Radiations from superpotato star</i>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<i>Solving the black guacamole mystery</i>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Challenges:

- Reviewer lethargy [[Section 3.1 of Shah et al. 2017](#)]
- Biases [[Fiez et al. 2020](#)]
- Fraud via coalitions [[Jecmen et al. 2022](#)]

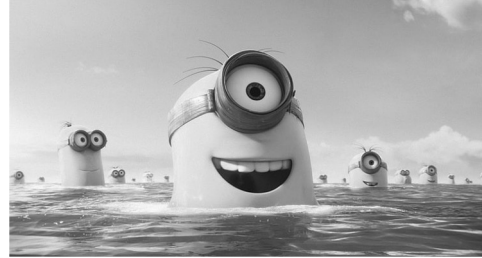
Putting things together

- These similarity measures are then combined to obtain a single similarity score for each (proposal, potential reviewer) pair
- **Notation:** Combined similarity score between proposal p and reviewer r is $s_{pr} \in [0, 1]$
- How are the different similarity measures combined?
 - Combined using some formula based on previous experiences, e.g., [\[Shah et al. 2017\]](#)
$$s_{p,r} = 2^{\text{bid-score}_{p,r}} (\text{subject-score}_{p,r} + \text{text-score}_{p,r}) / 4$$
 - Or via try-eyeball-iterate
 - Or via regression-based methods [\[Leyton-Brown et al. 2022\]](#)
 - See [Saveski et al. 2022](#) for an evaluation
- If conflict of interest, then $s_{pr} = -1$

Stages

(0) Reviewer pool

Construct a pool of potential reviewers



(1) Compute similarities

For every (proposal, potential reviewer) pair, compute a “similarity score”

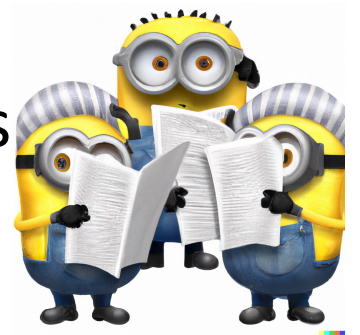
- Topic intersection
- Text matching (demo)
- Bidding



(2) Assignment

Use similarity scores to assign reviewers to proposals

- Sum-similarity algorithm
- PeerReview4All algorithm (demo)



(2)

Assignment

Maximize sum similarity

Intuitive approach: Choose the assignment which maximizes the sum of the similarities of the assigned (reviewer, proposal) pairs

[Goldsmith et al. 2007, Taylor 2008, Tang et al. 2010, Charlin et al. 2012, Long et al. 2013,..]

(2)

Assignment

Maximize sum similarity

Intuitive approach: Choose the assignment which maximizes the sum of the similarities of the assigned (reviewer, proposal) pairs

$$\begin{array}{ll} \text{maximize} & \sum_{p \in \text{Papers}} \sum_{r \in \text{Reviewers}} s_{pr} \mathbb{I}\{\text{paper } p \text{ assigned to reviewer } r\} \\ \text{assignment} & \end{array}$$

1 if p is assigned to r and 0 otherwise

subject to

Every paper gets (at least) certain #reviewers

Every reviewer gets at most certain #papers

Easy to solve (via flow-based algorithms or linear programming relaxations)

Example 1

- One reviewer per paper
- Up to one paper per reviewer

	Paper A	Paper B
Reviewer 1	0.6	0.9
Reviewer 2	0.7	0.8

Sum of similarities = 1.6

For any other assignment, sum of similarities ≤ 1.4

Example 2

- Two reviewers per paper
- Up to one paper per reviewer

	Paper A	Paper B
Reviewer 1	0.9	0.6
Reviewer 2	0.4	0.7
Reviewer 3	0.1	0.9
Reviewer 4	0.9	0.1

Sum of similarities = 3.4

For any other assignment, sum of similarities ≤ 2.8

Example 3

- One reviewer per paper
- Up to one paper per reviewer

	Paper A	Paper B	Paper C
Reviewer 1	1	0	0.5
Reviewer 2	0.7	1	0
Reviewer 3	0	0.7	0

Assignment is unfair to paper C

There exists another more balanced assignment

Example 4

- Two reviewers per paper
- Up to one paper per reviewer

	Paper A	Paper B	Paper C
Reviewer 1	0.9	0	0.5
Reviewer 2	0.6	0	0.5
Reviewer 3	0	0.9	0.5
Reviewer 4	0	0.6	0.5
Reviewer 5	0	0	0
Reviewer 6	0	0	0

Assignment is unfair to (inter-disciplinary) paper C

There exists another more balanced assignment

Problems with maximizing total similarity

- **Unbalanced:** Can assign all relevant reviewers to some papers and all irrelevant reviewers to others [[Stelmakh et al. 2018](#)]
- **Can be particularly unfair** to interdisciplinary papers
- In practice, this algorithm has been found to assign papers with **all reviewers having 0 similarity** [[Kobren et al. 2019](#)]

Balanced assignment: “PeerReview4All”

$$\begin{array}{ll} \text{maximize} & \text{minimum} \\ \text{assignment} & p \in \text{Papers} \end{array} \sum_{r \in \text{Reviewers}} s_{pr} \mathbb{I}\{\text{paper } i \text{ assigned to reviewer } j\}$$

subject to

Every paper gets at least certain #reviewers

Every reviewer gets at most certain #papers

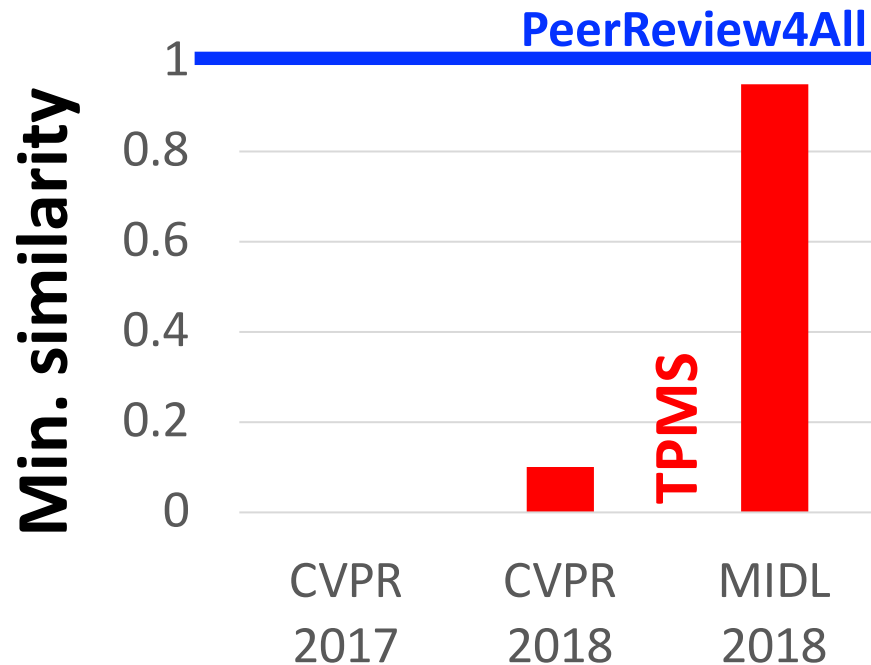
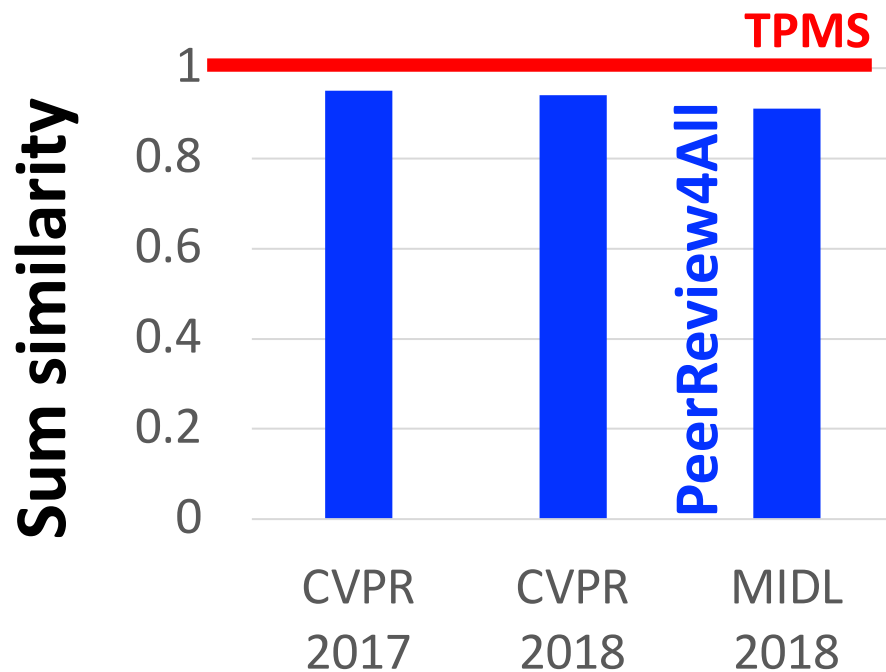
Fix assignment for the worst-off paper $\operatorname{argmin}_{p \in \text{Papers}}$

Repeat for remaining papers

Independent evaluation

[Independent evaluations by Kobren et al. 2019]

- “**TPMS**” algorithm optimizes **sum similarity**
- “**PeerReview4all**” algorithm iteratively optimizes **minimum similarity**



Additional benefits of PeerReview4All

- **Independent evaluations** by Payan et al. 2021
 - Gini index (standard measure of inequality): *PeerReview4All best performing*
 - Envy freeness (standard game theoretic measure of quality): *PeerReview4All best performing*
 - Sum similarity of lowest 10% papers: *PeerReview4All best performing*
 - Sum similarity of lowest 25% papers: *PeerReview4All best performing*
 - Nash Social Welfare: *PeerReview4All nearly best performing*
- Statistical guarantees: **optimal** top-K selection [Stelmakh et al. 2018]

PeerReview4All: Demonstration

- Get a free Gurobi academic license and install Gurobi
<https://www.gurobi.com/academia/academic-program-and-licenses/>
- Execute the PeerReview4All code:
<https://github.com/niharshah/peerreview4all>
(Github repo has instructions)
- Demo...

Gurobi: Always Free for Academics

We make it easy for students, faculty, and researchers to work with mathematical optimization. Whether for use in class or research, academics can **use Gurobi Optimizer at no cost**. Get all the same Gurobi features and performance, with no limits on model size.





Request the academic license that's right for you. Need help choosing? [Let Dr. Kostja Siefen walk you through the options.](#)



Academic Named-User License

Get a free, unlimited-use Gurobi Optimizer license for a single person, on a single machine.

[> Learn More](#)

 LICENSE	Initial commit	last month
 README.md	Update README.md	last month
 autoassigner.py	Add files via upload	last month
 working_example.ipynb	Add files via upload	last month

README.md

peerreview4all: Fair assignment of reviewers to papers

The python class `auto_assigner` in `autoassigner.py` implements the PeerReview4All algorithm (PR4A) of the paper "PeerReview4All: Fair and Accurate Reviewer Assignment in Peer Review" by Ivan Stelmakh, Nihar Shah and Aarti Singh. <https://www.jmlr.org/papers/volume22/20-190/20-190.pdf> This code was written by Ivan Stelmakh with some minor modifications by Nihar Shah.

DEPENDENCIES: This code is written in Python 3. The code uses the `gurobipy` package in python for solving the Linear Programming problems. Gurobi offers free academic licenses for universities. To obtain your license, please go to <http://www.gurobi.com/academia/for-universities> and follow the instructions provided there.

You may also have to install Gurobi python package: <https://support.gurobi.com/hc/en-us/articles/360044290292-How-do-I-install-Gurobi-for-Python->

Code for Example 3 from earlier slide

```
➤ import numpy as np
➤ from autoassigner import auto_assigner

➤ myassignment = auto_assigner(simmatrix=np.array([[1,0,0.5], [0.7,1,0],[0,.7,0]]), demand=1,
    ability=1)
➤ myassignment.fair_assignment()
➤ print(myassignment.fa)
```

INPUTS:

- 'simmatrix' is an #reviewers x #papers matrix with entries in $\{-1\} \cup [0, 1]$
- 'demand' is the number of reviewers required per paper
- 'ability' is the maximum number of papers that reviewer can review

You can also input max time, customized review loads, etc. See documentation.

OUTPUT: A dictionary of the form {paper: [assigned_reviewers]} that encodes the resulting assignment

Concluding comments

- Area of active research!
 - See Chapter 3 of bit.ly/PeerReviewOverview
- PeerReview4All trials at ALMA...
- Don't hesitate to ping me if you would like to know more or want to use automated assignments

<https://cs.cmu.edu/~nihars>

nihars@cs.cmu.edu



Thank you! Questions?