Defusing Popularity Bias in Recommender Systems for Software Engineering

Dr. Phuong Nguyen, Tenure-track Assistant Professor
Department of Information Engineering, Computer Science and Mathematics,
University of L'Aquila, Italy
Email: phuong.nguyen@univaq.it; Website: https://www.disim.univaq.it/ThanhPhuong

A research proposal submitted to the University of L'Aquila on February 5th, 2024

This proposal represents an opportunity to contribute to the advancement of Machine Learning in Software Engineering, and aligns with the commitments to innovative research in the field of ethics and fairness for Artificial Intelligence (AI) systems. Should this proposal receive funding, would it pave the way for the formation of a dedicated team at the University of L'Aquila. Being Principal Investigator (PI), I will lead the team to work diligently to advance ongoing research efforts, and propose practical solutions that address unresolved issues in the field of Machine Learning for Software Engineering.

1 Introduction

When building new software, developers usually have to deal with an overload of information from heterogeneous and rapidly evolving open sources. Recommender systems for software engineering (RSSEs) [30] serve as an effective means to provide developers with instant support consisting of different items, e.g., code examples [15, 25], possible third-party components [24, 27, 36], documentation, to name a few. Nevertheless, while the main effort has been spent to make RSSEs more effective and efficient, there are several issues that have not attracted enough attention from the research community.

Recommender systems are among the most widespread applications of machine learning, playing a significant role in assisting human decision-making processes. The quality of the recommendations is closely linked to user satisfaction and the interests of the platforms. However, being heavily reliant on data and algorithms, recommender systems can be susceptible to biases that may lead to unfair outcomes, potentially jeoparadizing trust in them. For recommender systems, the *long tail effect* indicates that a handful of items are extremely popular, whilst most of the remaining ones, so-called the long tail, are not seen by users [3]. Essentially, products belonging to the long tail are considered a social good [2] and recommending rare but useful items benefits both customers and shop owners [37]. *Popularity bias* is a common phenomenon of general purpose recommender systems [1, 2, 9], i.e., providing to users only frequently seen items. Likewise, RSSEs are no exception, while they become more effective in suggesting handy recommendations, RSSEs also suffer from popularity bias by presenting artifacts used by several developers [26]. While this favors artifacts that are possibly more reliable and well-maintained, it would essentially mean that systems fail to recommend some relevant goals, architecture- or solution-specific artifacts.

In our recent work [26], by means of mixed methods research, i.e., performing both a qualitative and quantitative evaluation, we studied popularity bias recommender systems for mining third-party libraries (TPLs). First, following existing guidelines for such type of study software engineering [19], we investigated whether state-of-the-art research has already tackled the issue of popularity bias. Interestingly, the literature review on major software engineering venues reveals that the issue of dealing with popularity bias has not received enough attention from the community. All of the surveyed studies tackled different issues in library recommendation, with the main aim of improving the relevance of the final ranked list, only one work attempts to tackle popularity, unfortunately, it fails to maintain a trade-off between fairness and accuracy.

Then, we performed a quantitative evaluation on four existing TPL RSSEs, exploring their capability to deal with popular artifacts. The experiments showed that three among the considered systems provide to developers highly popular TPLs. The remaining system, while being able to lessen the effect of frequent TPLs, suffers from a low accuracy. Altogether, we see that cutting-edge research in software engineering neglects the issue of popularity bias in TPL recommender systems, leaving a research gap that has to be bridged.

Our work was positively received by the reviewers, and accepted for publication in a CORE Rank A conference[1] with the following details:

- Phuong T. Nguyen, Riccardo Rubei, Juri Di Rocco, Claudio Di Sipio, Davide Di Ruscio, Massimiliano Di Penta "*Dealing with Popularity Bias in Recommender Systems for Third-party Libraries: How far Are We?*", in Proceedings of the IEEE/ACM 20th Int. Conf. on Mining Software Repositories (MSR 2023), DOI: 10.1109/MSR59073.2023.00016.

Research gap. In our previous work [26], we found out that while RSSEs have become more effective at suggesting relevant items, they are prone to popularity bias. Among the considered studies, only one approach tries to improve diversity in the recommendation results, nevertheless it suffers from a low prediction accuracy. In this respect, we see an urgent need to thoroughly study the related issues, with the ultimate aim of *devising effective mechanisms to improve the fairness and robustness of RSSEs*.

Expected contributions. Our work aims to empower RSSEs with the capability to effectively deal with popularity bias, while maintaining or even enhancing their accuracy. The main contributions are as follows:

- Pioneering exploration. We take the initiative to bring attention to the previously overlooked issues of popularity bias affecting RSSEs. This represents a pioneering effort in the field.
- Holistic approach. We propose a holistic and integrated approach to fortify RSSEs, leveraging tailored reinforcement learning. This comprehensive strategy is designed to enhance their fairness.
- Applicability. The techniques we develop are not limited to RSSEs only, they can be readily applied to fine-tune pre-trained deep learning models in the broader context of software engineering, contributing to the advancement of AI-driven solutions in the field.

2 Methodology

Figure 1 depicts the architecture with the proposed module being printed in the cyan color, which can be independently assembled to any existing recomender systems, being padded right before the interface to developers to defuse popularity bias. This section presents our proposed approach to fill the research gap introduced in Section 1.

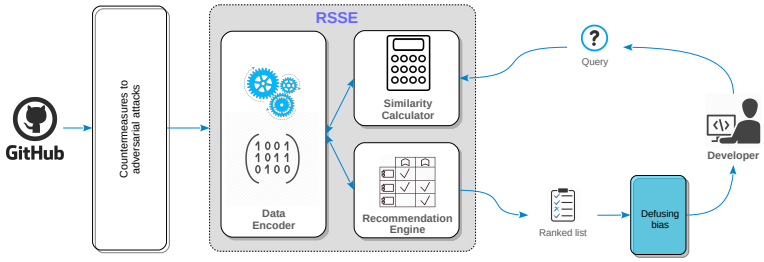

Figure 1: The proposed architecture.

2.1 Research objectives

Within the funded project, we are going to answer the following research questions.

– **RQ**: "*How can we make RSSEs less biased towards popular items, while still preserving accuracy?*" We plan to develop novel methods to defuse popularity bias. Existing mechanisms conceived for other domains might not work well for recommender systems for software engineering, as we showed in our recent work [26]. We anticipate that Reinforcement Learning [35] can be applied to improve re-ranking techniques, taking into consideration the similarity between projects when a rare library needs to be moved up in the ranked list.

To answer the research question, correspondingly we divide the tasks into two tasks, namely **T1** and **T2** (see Figure 1), and they are explained in the succeeding subsections.

[1] http://portal.core.edu.au/conf-ranks/?search=MSR&by=all&source=CORE2023&sort=atitle&page=1

2.2 Defusing popularity bias

Looking to other domains, we see that there are three major methods to defuse bias in general, i.e., the pre-processing, in-processing, and post-processing paradigms [10]. We suppose that these can also be adopted in software engineering for the same purpose. We are going to perform the following tasks:

- **T1: Re-implementation of existing re-ranking mechanisms.** In our recent work [26], we applied xQuAD [33] to improve diversity in the recommendations. We plan to investigate and re-implement other techniques, with the aim of finding suitable mechanisms to be adopted for RSSEs. Another possible candidate is PFAR [21], a practical approach conceived to allow items to have a fair chance of being recommended. We will considered also personalized ranking [1], which promotes unpopular items in the ranked list, while maintaining a trade off between fairness and accuracy. Once re-implemented, these techniques can be used as baselines for comparison with our proposed approach presented in **T2**.
- **T2: Reinforcement learning for reducing bias.** Existing re-ranking algorithms such as xQuAD [33] attempt to reduce the number of popular items, as well as to increase the number of unpopular (but useful) ones in the results. However, our empirical evaluation on two recommender systems showed that while introducing diversity in the recommendation results, it also introduces a setback in the prediction accuracy. Our findings suggest that further research should be conducted to propose effective countermeasures. We suppose that it is crucial to consider additional factors, e.g., the degree of specificity (to certain solutions) of a library, when it comes to providing recommendation. Thus, we will deploy Reinforcement Learning to implement effective post-processing techniques to defuse popularity bias, aiming to improve diversity in the recommendations, while keeping a reasonable prediction accuracy.

3 Expected results and plans

The proposed research on fairness in recommender systems for software engineering has the potential to yield significant outcomes, positively impacting the software engineering field, artificial intelligence, user privacy, and the broader software industry.

3.1 Implications

- **Enhancing user trust and satisfaction.** By mitigating bias and promoting fairness in recommender systems, our approach aims to enhance user trust and satisfaction with software engineering platforms. As users rely on recommendations to make critical decisions in software engineering, our research helps to preserve user trust by ensuring that recommendations remain trustworthy.
- **Improving system performance.** Fairness-aware recommender systems are likely to generate recommendations that are not only unbiased but also more accurate and relevant to users. This improvement in recommendation quality can positively impact the overall performance of software engineering platforms.
- **Applicability.** The potential of Large Language Models (LLMs) in software engineering has been recently acknowledged [28]. However, these models are trained on vast and diverse datasets, which can introduce biases. Additionally, LLMs often require prompt tuning to tailor them to specific purposes, necessitating fine-tuning with relevant data. In this context, the techniques developed in this research project offer practical solutions by mitigating biases within training data for LLMs.
- **Educational impact.** The ethical implications of AI, including bias mitigation and adversarial counteraction, have gained significant attention in recent years. Our research offers an opportunity to educate future AI professionals about the importance of ethical considerations when developing AI-based systems. It can be integrated into AI ethics courses to promote responsible AI development.

3.2 Prospective publications

The findings and methodologies conceived in this funded research stay will contribute to state-of-the-art research in software engineering and AI ethics. We will target both the software engineering, and the machine learning communities, aiming to have *at least one article* submitted and accepted for publication to a Rank A

Table 1: Task schedule.

		Month											
Task	**Description**	1	2	3	4	5	6	7	8	9	10	11	12
Reading	Reviewing SOTA												
T1	Re-implementation of re-ranking techniques												
T1	Reinforcement learning for reducing bias												
Writing	Deliverable D1												
Writing	Deliverable D2												
Writing	Final report												
Writing	Papers												

or A* conference,[2] or Scimago Q1 journals.[3] The following venues are considered: Int. Conf. on Automated Software Engineering (ASE, Rank A*); Int. Conf. on Software Engineering (ICSE, Rank A*); Int. Conf. on Evaluation and Assessment in Software Engineering (EASE, Rank A); Int. Conf. on Mining Software Repositories (MSR, Rank A); The ACM Conf. on Recommender Systems (RecSys, Rank A); Int. Conf. on Software Analysis, Evolution, and Reengineering (SANER, Rank A); The ACM Int. Conf. on Information and Knowledge Management (CIKM, Rank A); Elsevier Information and Software Technology Journal (IST, Q1); Elsevier Journal of Systems and Software (JSS, Q1); Elsevier Expert Systems with Applications (ESWA, Q1); IEEE Transactions on Software Engineering (TSE, Q1).

3.3 Plan

The Gantt chart in Table 1 depicts a tentative plan for the entire project.

References

[1] H. Abdollahpouri, R. Burke, and B. Mobasher, "Managing popularity bias in recommender systems with personalized re-ranking," in *Proceedings of AAAI 2019, Sarasota, Florida, USA, May 19-22 2019*, R. Barták and K. W. Brawner, Eds. AAAI Press, 2019, pp. 413–418. [Online]. Available: https://aaai.org/ocs/index.php/FLAIRS/FLAIRS19/paper/view/18199

[2] H. Abdollahpouri, M. Mansoury, R. Burke, and B. Mobasher, "The unfairness of popularity bias in recommendation," in *Proceedings of the Workshop on Recommendation in Multi-stakeholder Environments co-located with RecSys 2019, Copenhagen, Denmark, September 20, 2019*, ser. CEUR Workshop Proceedings, R. Burke, H. Abdollahpouri, E. C. Malthouse, K. P. Thai, and Y. Zhang, Eds., vol. 2440. CEUR-WS.org, 2019. [Online]. Available: http://ceur-ws.org/Vol-2440/paper4.pdf

[3] C. Anderson, *The Long Tail: Why the Future of Business Is Selling Less of More*. Hyperion, 2006.

[4] V. W. Anelli, Y. Deldjoo, T. Di Noia, E. Di Sciascio, and F. A. Merra, "Sasha: Semantic-aware shilling attacks on recommender systems exploiting knowledge graphs," in *The Semantic Web*. Cham: Springer International Publishing, 2020, pp. 307–323.

[5] J. Chakraborty, S. Majumder, and T. Menzies, "Bias in machine learning software: why? how? what to do?" in *ESEC/FSE '21, Athens, Greece, August 23-28, 2021*. ACM, 2021, pp. 429–440.

[6] J. Chakraborty, S. Majumder, Z. Yu, and T. Menzies, "Fairway: a way to build fair ML software," in *ESEC/FSE '20, Virtual Event, USA, November 8-13, 2020*. ACM, 2020, pp. 654–665.

[7] C. Chen, Z. Xing, and Y. Liu, "What's spain's paris? mining analogical libraries from q&a discussions," *Empirical Softw. Engg.*, vol. 24, no. 3, p. 1155–1194, jun 2019. [Online]. Available: https://doi.org/10.1007/s10664-018-9657-y

[8] C. Chen, Z. Xing, Y. Liu, and K. O. L. Xiong, "Mining likely analogical apis across third-party libraries via large-scale unsupervised api semantics embedding," *IEEE Transactions on Software Engineering*, vol. 47, no. 3, pp. 432–447, 2021.

[2]CORE Rankings Portal `http://portal.core.edu.au/conf-ranks/`

[3]Scimago Journal & Country Rank `https://www.scimagojr.com/`

[9] J. Chen, H. Dong, X. Wang, F. Feng, M. Wang, and X. He†, "Bias and debias in recommender system: A survey and future directions," *ACM Trans. Inf. Syst.*, oct 2022, just Accepted. [Online]. Available: https://doi.org/10.1145/3564284

[10] B. d'Alessandro, C. O'Neil, and T. LaGatta, "Conscientious classification: A data scientist's guide to discrimination-aware classification," *Big Data*, vol. 5, no. 2, pp. 120–134, 2017, pMID: 28632437. [Online]. Available: https://doi.org/10.1089/big.2016.0048

[11] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, 2002. [Online]. Available: https://doi.org/10.1109/4235.996017

[12] Y. Deldjoo, T. Di Noia, and F. A. Merra, "Adversarial machine learning in recommender systems (aml-recsys)," in *Proceedings of the 13th Int. Conf. on Web Search and Data Mining*, ser. WSDM '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 869–872. [Online]. Available: https://doi.org/10.1145/3336191.3371877

[13] J. Di Rocco, D. Di Ruscio, C. Di Sipio, P. T. Nguyen, and R. Rubei, "Development of recommendation systems for software engineering: the CROSSMINER experience," *Empir. Softw. Eng.*, vol. 26, no. 4, p. 69, 2021. [Online]. Available: https://doi.org/10.1007/s10664-021-09963-7

[14] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96. AAAI Press, 1996, p. 226–231.

[15] J. Fowkes and C. Sutton, "Parameter-free Probabilistic API Mining Across GitHub," in *24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*. New York: ACM, 2016, pp. 254–265.

[16] M. Ge, C. Delgado-Battenfeld, and D. Jannach, "Beyond accuracy: Evaluating recommender systems by coverage and serendipity," in *Proceedings of the Fourth ACM Conference on Recommender Systems*, ser. RecSys '10. New York, NY, USA: ACM, 2010, pp. 257–260. [Online]. Available: http://doi.acm.org/10.1145/1864708.1864761

[17] I. Gunes, C. Kaleli, A. Bilge, and H. Polat, "Shilling attacks against recommender systems: A comprehensive survey," *Artif. Intell. Rev.*, vol. 42, no. 4, p. 767–799, Dec. 2014. [Online]. Available: https://doi.org/10.1007/s10462-012-9364-9

[18] Q. He, B. Li, F. Chen, J. Grundy, X. Xia, and Y. Yang, "Diversified third-party library prediction for mobile app development," *IEEE Transactions on Software Engineering*, pp. 1–1, 2020.

[19] B. A. Kitchenham, P. Brereton, Z. Li, D. Budgen, and A. J. Burn, "Repeatability of systematic literature reviews," in *15th Int. Conf. on Evaluation & Assessment in Software Engineering, EASE 2011, Durham, UK, 11-12 April 2011, Proceedings*, 2011, pp. 46–55. [Online]. Available: https://doi.org/10.1049/ic.2011.0006

[20] B. Li, Q. He, F. Chen, X. Xia, L. Li, J. Grundy, and Y. Yang, "Embedding app-library graph for neural third party library recommendation," in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ser. ESEC/FSE 2021. New York, NY, USA: Association for Computing Machinery, 2021, p. 466–477. [Online]. Available: https://doi.org/10.1145/3468264.3468552

[21] W. Liu and R. Burke, "Personalizing fairness-aware re-ranking," *CoRR*, vol. abs/1809.02921, 2018. [Online]. Available: http://arxiv.org/abs/1809.02921

[22] K. W. Nafi, M. Asaduzzaman, B. Roy, C. K. Roy, and K. A. Schneider, "Mining software information sites to recommend cross-language analogical libraries," in *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2022, pp. 913–924.

[23] A. M. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images." in *CVPR*. IEEE Computer Society, 2015, pp. 427–436. [Online]. Available: http://dblp.uni-trier.de/db/conf/cvpr/cvpr2015.html#NguyenYC15

[24] P. T. Nguyen, J. Di Rocco, D. Di Ruscio, and M. Di Penta, "CrossRec: Supporting Software Developers by Recommending Third-party Libraries," *Journal of Systems and Software*, p. 110460, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121219302341

[25] P. T. Nguyen, J. Di Rocco, D. Di Ruscio, L. Ochoa, T. Degueule, and M. Di Penta, "FOCUS: A Recommender System for Mining API Function Calls and Usage Patterns," in *Proceedings of the 41st Int. Conf. on Software Engineering*, ser. ICSE '19. Piscataway, NJ, USA: IEEE Press, 2019, pp. 1050–1060. [Online]. Available: https://doi.org/10.1109/ICSE.2019.00109

[26] P. T. Nguyen, R. Rubei, J. Di Rocco, C. Di Sipio, D. Di Ruscio, and M. Di Penta, "Dealing with popularity bias in recommender systems for third-party libraries: How far are we?" in *2023 IEEE/ACM 20th Int. Conf. on Mining Software Repositories (MSR)*, 2023, pp. 12–24. [Online]. Available: https://doi.org/10.1109/MSR59073.2023.00016

[27] A. Ouni, R. G. Kula, M. Kessentini, T. Ishio, D. M. German, and K. Inoue, "Search-based software library recommendation using multi-objective optimization," *Inf. Softw. Technol.*, vol. 83, no. C, pp. 55–75, Mar. 2017. [Online]. Available: https://doi.org/10.1016/j.infsof.2016.11.007

[28] I. Ozkaya, "Application of large language models to software engineering tasks: Opportunities, risks, and implications," *IEEE Software*, vol. 40, no. 3, pp. 4–8, 2023.

[29] Y.-J. Park and A. Tuzhilin, "The long tail of recommender systems and how to leverage it," in *Proceedings of the 2008 ACM Conference on Recommender Systems*, ser. RecSys '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 11–18. [Online]. Available: https://doi.org/10.1145/1454008.1454012

[30] M. P. Robillard, W. Maalej, R. J. Walker, and T. Zimmermann, *Recommendation Systems in Software Engineering*. Springer, 2014. [Online]. Available: https://doi.org/10.1007/978-3-642-45135-5

[31] R. Rubei, C. Di Sipio, J. Di Rocco, D. Di Ruscio, and P. T. Nguyen, "Endowing third-party libraries recommender systems with explicit user feedback mechanisms," in *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2022, pp. 817–821.

[32] M. A. Saied, A. Ouni, H. Sahraoui, R. G. Kula, K. Inoue, and D. Lo, "Improving reusability of software libraries through usage pattern mining," *Journal of Systems and Software*, vol. 145, pp. 164 – 179, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121218301699

[33] R. L. Santos, C. Macdonald, and I. Ounis, "Exploiting query reformulations for web search result diversification," in *Proceedings of the 19th Int. Conf. on World Wide Web*, ser. WWW '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 881–890. [Online]. Available: https://doi.org/10.1145/1772690.1772780

[34] Z. Sun, Y. Liu, Z. Cheng, C. Yang, and P. Che, "Req2lib: A semantic neural model for software library recommendation," in *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. Los Alamitos, CA, USA: IEEE Computer Society, feb 2020, pp. 542–546. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/SANER48275.2020.9054865

[35] R. S. Sutton and A. G. Barto, *Reinforcement learning - an introduction*, ser. Adaptive computation and machine learning. MIT Press, 1998. [Online]. Available: https://www.worldcat.org/oclc/37293240

[36] F. Thung, D. Lo, and J. Lawall, "Automated library recommendation," in *2013 20th Working Conference on Reverse Engineering (WCRE)*, Oct 2013, pp. 182–191.

[37] S. Vargas and P. Castells, "Improving sales diversity by recommending users to items," in *Eighth ACM Conference on Recommender Systems, RecSys '14, Foster City, Silicon Valley, CA, USA - October 06 - 10, 2014*, 2014, pp. 145–152. [Online]. Available: http://doi.acm.org/10.1145/2645710.2645744