ID	Paper ID	Motivation	Assessment criteria (RQ1)	Metrics (RQ2)	Remarks
4	Rebai 2020	Validation of multi-objective approach	Past reviews and collaboration	Precision@k, Recall@k, MRR@k	Recommendations on average better than state of the art
5	Kovalenko 2019	Is CRR relevant ***in vivo?*** Determining relevance with survey	A mix of criteria based on past activity	top-k accuracy, MRR	An overview of the topic; study whether the CRR is helpful for developers
6	Garousi 2020	Extensive research on the relevance of SE research	-	-	Does not answer RQs
8	Jiang 2019	Time decaying relationships	Past reviews; focus on time-decaying of relevance	accuracy, MRR	Describes the problem of relevance diminishing over time
10	Liao 2019	A new CRR algorithm proposition (TIRR), focus on recommendation accuracy - an opportunity to study how a process of developing new algorithm looks like	Past reviewers	Precision, Recall, F-Measure	Strength of connection between reviewers is based on an absolute value (number of commented PR's), thus it can favour 'busy' reviewers over less active ones
12	Sulun 2019	Automatically suggesting appropriate reviewers have two main benefits: (1) reducing overall review time by automatically assigning reviewers (2) increasing the review quality and thus reducing the potential errors related to the reviewed artifact.	comparison with actual reviewers set	top-k accuracy, MRR	implementation uses Software Artifact Traceability Graphs
14	Asthana 2019	we found that reviewer recommendation systems will often assign a large proportion of reviews to a small set of experienced developers (e.g., 20% of the developers in a project are assigned 80% of the reviews)	matching reviewers who the developers chose manually	precision@m, recall@m, F-score@m, PR Hits (P@n), as the fraction of all pull-requests where we were able to make at least one positive suggestion	load balancing recommended reviewers; there is an (often implicit) assumption that those who participated in the review were in fact the best people to review the change and those who were not invited were not appropriate reviewers.
17	Ye 2017	as of June 2018, Github has over 28 million users and 57 million repositories; Gousios conducted a survey with 749 core members and found that they feel overwhelmed in managing pull requests. Integrators are struggling to maintain the quality of the code and are spending a lot of time to review pull requests	compare the ranking result with the ideal ranking in which the candidates who are reviewers should be listed at the top	Accuracy@k, Mean Average Precision, MRR	Many features that the algorithm may be constructed upon.
19	Fejzer 2017	Commits without assigned reviewers take the longest time to be merged due to the lack of both interest and proper domain knowledge; the number of potential reviewers in GitHub is significantly higher than in a traditional code review system like Gerrit.	Past reviewers; time decaying params	top-k prediction accuracy (precision, recall, F-Measure), MRR	low computational complexity due to the incremental model
20	Peng 2018	Marlow found that GitHub reviewers tend to examine contributors' profiles before deciding on whether to accept or reject the PR review request. Despite the in- sightful findings, they did not study the needs from the contributors' perspective and the process did not involve ARR.	-	-	evaluation of mention bot in GitHub (study how software developers perceive and work with the Facebook mention bot, one of the most popular ARR bots in GitHub) - results show that developers appreciate mention bot saving their effort, but are bothered by its unstable setting and unbalanced workload allocation
21	Mohamed 2018	It is important to predict reopened pull requests immediately after pull requests' first close, and help integrators reopen pull requests in time. If pull requests are reopened a long time after their close, they may cause conflicts with new submitted pull requests, add software maintenance cost, and increase burden for already busy developers.	-	-	Study on how to predict if pul request would be reopened - can be used to enhence reviewer recommendation
23	Lipcak 2018	The automated recommendation of code reviewers can reduce the efforts necessary to find the best code reviewers, cutting time spent by code reviewers on understanding large code changes	Comparison with the actual reviewers	top-k accuracy (the percentage of correct recommendations on the total number of recommendations) and MRR	categorization of recommendation algorithms into four groups: i) Heuristic-based approaches, ii) Machine Learning-based, iii) Social Networks-based, iv) Hybrid ap- proaches; In our experiments, we evaluate whether one of the top-k recommended reviewers actually evaluated the pull request. However, we do not know whether this reviewer was the best candidate for the specific task. Historical data might include wrong reviewers choices.
25	Li 2017	each pull request gets 2.6 comments in average and only about 16% of the pull-requests have extended discussions	-	-	Thorough analysis of a code review process in Github; focused mainly on review comments classification
26	Jiang 2017	popular projects receive many pull requests, and commenters may not notice new pull requests in time, or even ignore appropriate pull requests	Comparison with actual commenters; time decaying parameter	top-k prediction accuracy (precision, recall)	Focused mainly on commenter recommendation; The activeness is the most important attribute in the commenter prediction; the mean turnover ratio in GitHub projects is 66% for each quarter
27	Xia 2017	integrators often have difficulty to process the incoming pull requests in time - they can become tired of prioritizing the pull requests due to the heavy workload; 15% of contributors complain the delay of pull request response.	Similarity with the actual reviewer assignment	precision and recall for top-[1,2,3,4,5] recommendations	Vast review of a related work; method focused on social relations
29	Saxena 2017	a method to create a more detailed technology skill profile of a candidate based on her code repository contributions	-	-	SkillMap, a visual representation of candidate skill profile, for quick review and comparison with other candidate profiles
30	Zanjani 2016	it is not always easy to determine who has the most expertise given a particular change for review, especially for newcomers to a codebase or those changing parts of the code with shared ownership by many people	reviewers who contributed in reviews to a given code change and not those reviewers who are assigned to review the code change	precision, recall, F-score for top-[1,2,3,5] recommendation, and MRR	
31	Yu 2016	having access to more potential reviewers does not necessarily mean that it's easier to find the right ones (the "needle in a haystack" problem); three hundred new pull-requests each month	Past reviews; training set consisting of previously prepared set of pull requests (only representative ones are chosen)	top-k prediction accuracy (precision, recall, F-Measure)	comment network making use of social relationships between developers
32	Ouni 2016	Linus's Law "many eyes make all bugs shallow"; Inappropriate reviewers assignment may lead to an inaccurate, time consuming and non effective review process	actual reviewers, that are known, as the ground truth	precision@k, recall@k for top-[1,3,5,10] and MRR	Recent studies showed that when reviewers have a priori knowledge of the context and the code, they complete reviews more quickly and provide more valuable feedback to the author; code review is basically a human process involving personal and social aspects, thus the socio-technical factor plays an extremely improtant role in finding peer reviewers
35	Hannebauer 2016	projects have a large community with a many reviewers, so it can be difficult to decide who should review which patch, especially for newcomers; problems with reviewer assignment in FLOSS projects can defer the acceptance of patches by 6 to 18 days on average; sometimes, submitted patches receive no review at all and therefore are not integrated into the application	Similarity with the actual reviewer assignment	top-k accuracy (the percentage of correct recommendations on the total number of recommendations)	comparison of the prediction performance of eight reviewer recommendation algorithms
36	Rahman 2018	Reliable information on reviewers' expertise (e.g., tech- nology skill) is often not readily available, and it needs to be carefully mined from the codebase; the task of identifying appropriate reviewers is even more challenging and time-consuming for novice developers since they are neither familiar well with the codebase nor are aware of the skills of the hundreds of fellow potential reviewers	evaluatoin using the real code review data from Vendasta codebase.	Top-K Accuracy, Mean Reciprocal Rank, Mean Precision and Mean Recall	solution packaged as a web service and a plug-in for Google Chrome browser

ID	Paper ID	Motivation	Assessment criteria (RQ1)	Metrics (RQ2)	Remarks
40	Thongtanunam 2015	The results show that 4%-30% of reviews have code-reviewer assignment problem. These reviews significantly take 12 days longer to approve a code change.	Similiarity with past reviews	Percentage of correct recommendations (at least one correct reviewer (real choice) returned in the top 10 recommendations) and MRR	RevFinder - one of the earliest algorithms. The original test set - Android, LibreOffice, OpenStack and Qt.
41	Xia 2015	74.4 review requests submitted daily to QT’s Gerrit from 2011 – 2012	Past reviews	top-k prediction accuracy, MRR	Improved version of the RevFinder proposed by Thongtanunam (combination of file based approach and text mining)
42	Yu 2014	the time of the recommended reviewer submitting his first comment on PR is on average 40.8 hours shorter than those without recommendation; the project managers may not completely find out all potential reviewers from crowds	Comparison with actual reviewers	top-k prediction accuracy (precision and recall)	Comment network approach
44	Balachandran 2013	The reviews would be time consuming or inaccurate if appropriate reviewers are not set. Most of the time, novice developers have to reach out to experienced developers or search the file revision history to assign appropriate reviewers.	Revision history	accuracy defined as the number of correct top-k recommendation (at least one correct reviewer) divided by the number of pull requests	
45	Jeong 2009	On average a review takes 1.5 days and between 39% and 45% of patches are accepted; review requests without initial reviewer assignment take longer and have lower chances to be accepted	Comparison with actual reviewers	Accuracy (top N) (The percentage of predictions with at least one correct reviewer in the top N)	Interesting threats to validity presented in the paper - open source projects are specific, reviewer retirement information is hidden.