56th **COW**: Code Review and Continuous Inspection/Integration

Towards Automated Supports for Code Reviews using Reviewer Recommendation and Review Quality Modelling

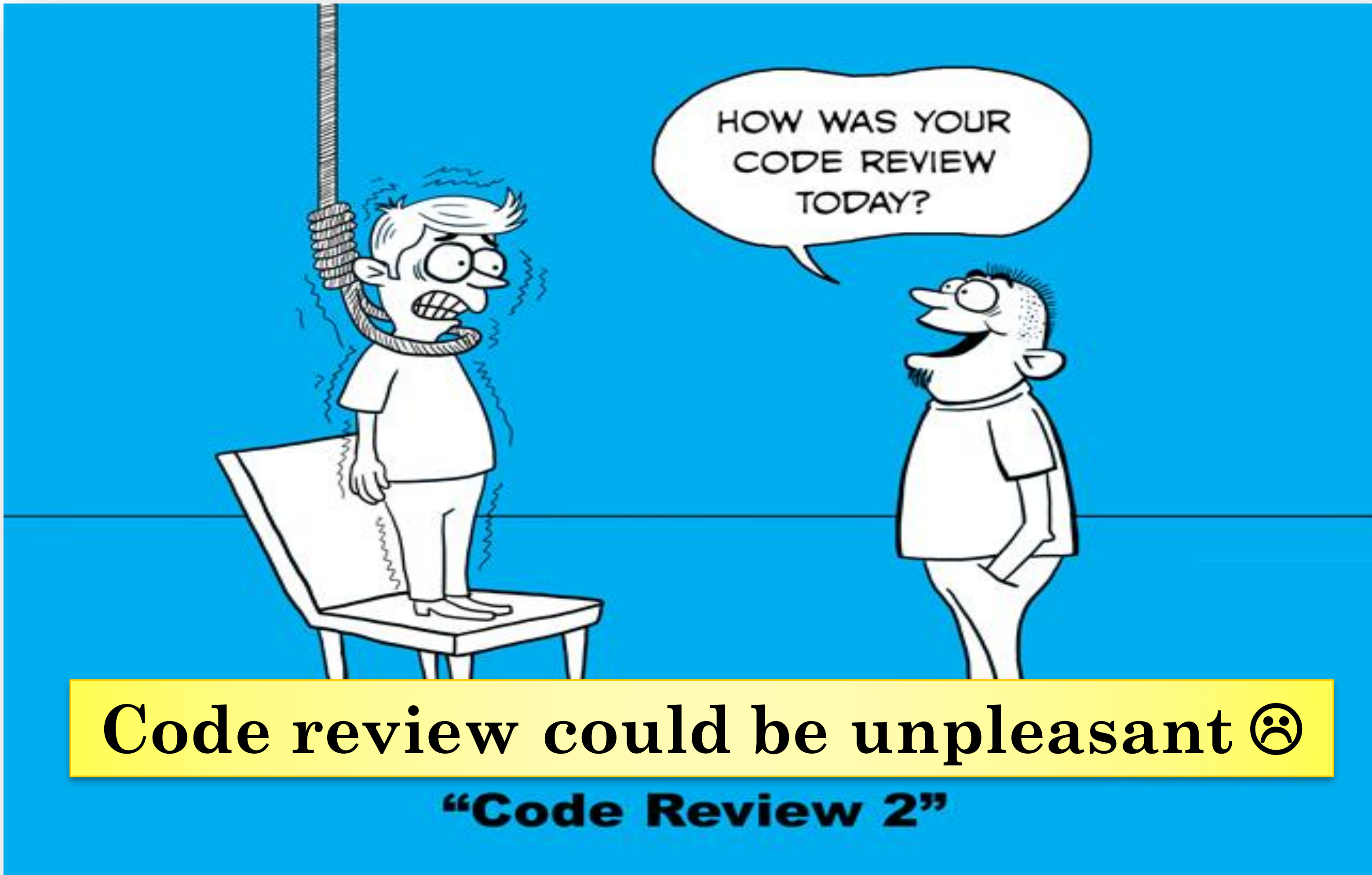
    

Mohammad Masudur Rahman, **Chanchal K. Roy**,
Raula G. Kula, **Jason Collins**, and **Jesse Redl**

University of Saskatchewan, Canada, Osaka University, Japan

Vendasta Technologies, Canada

Code Review



Code review could be unpleasant ☹

Recap on Code Review

Formal inspection

Peer code review

Modern code review (MCR)

Code review is a systematic examination of source code for detecting **bugs** or **defects** and **coding rule violations**.

Early bug **detection**

Stop coding rule violation

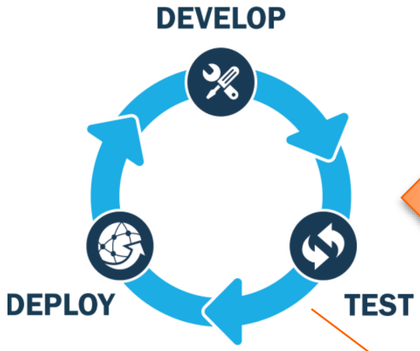
Enhance developer skill

Today's Talk Outline

Part I: Code Reviewer Recommendation System (ICSE-SEIP 2016)

Part II: Prediction Model for Review Usefulness (MSR 2017)

Today's Talk Outline



Part III: Impact of Continuous Integration on Code Reviews (MSR 2017 Challenge)

Part I: Code Reviewer Recommendation (ICSE-SEIP 2016)

- **FOR**

Novice developers

Distributed software development

Delayed 12 days
(Thongtanunam et al, SANER 2015)

Existing Literature

- **Line Change History (LCH)**
  - *ReviewBot* (Balachandran, ICSE 2013)
- **File Path Similarity (FPS)**

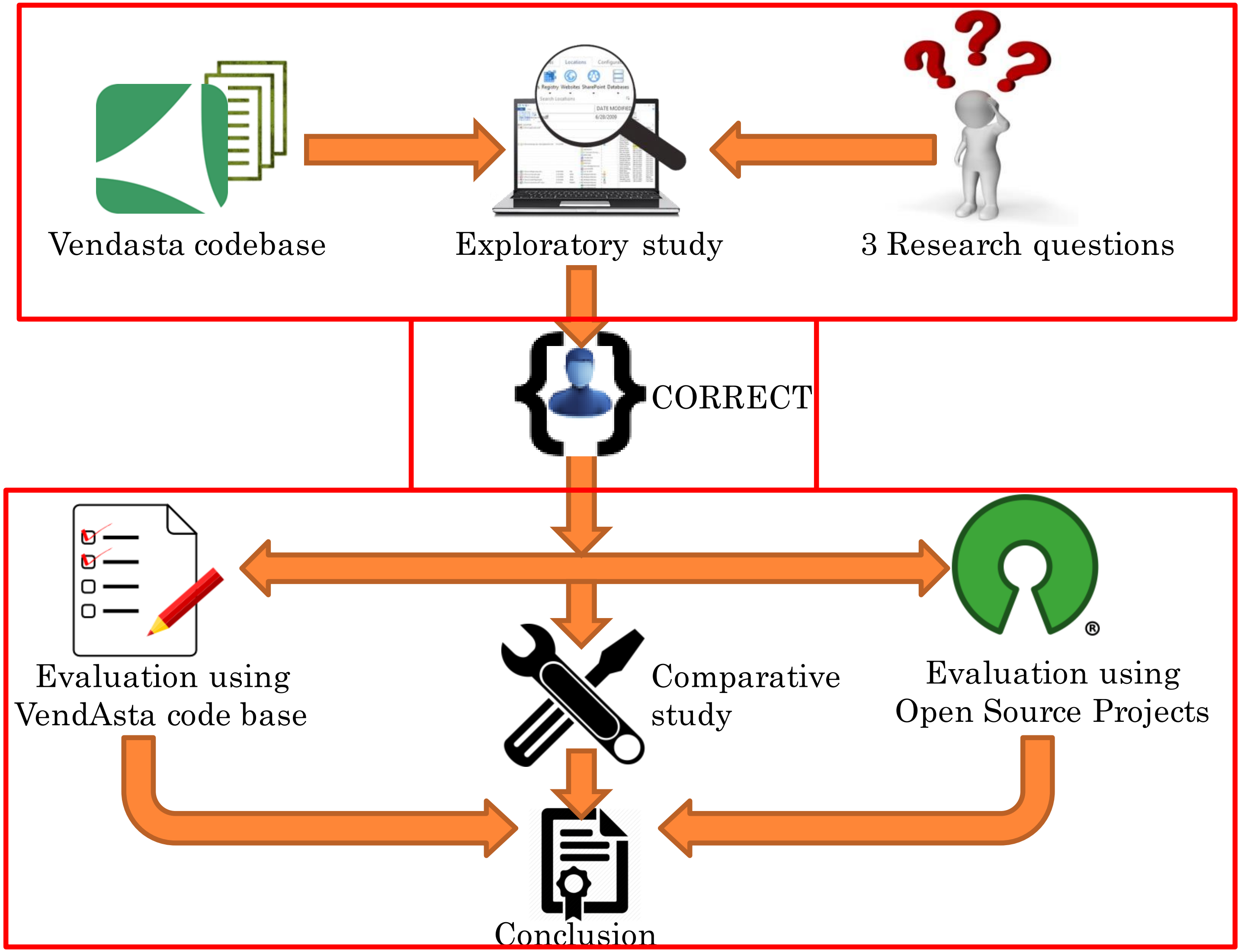
- **Issues & Limitations**

- **Library & Technology Similarity**

Library

Technology

Outline of this Study



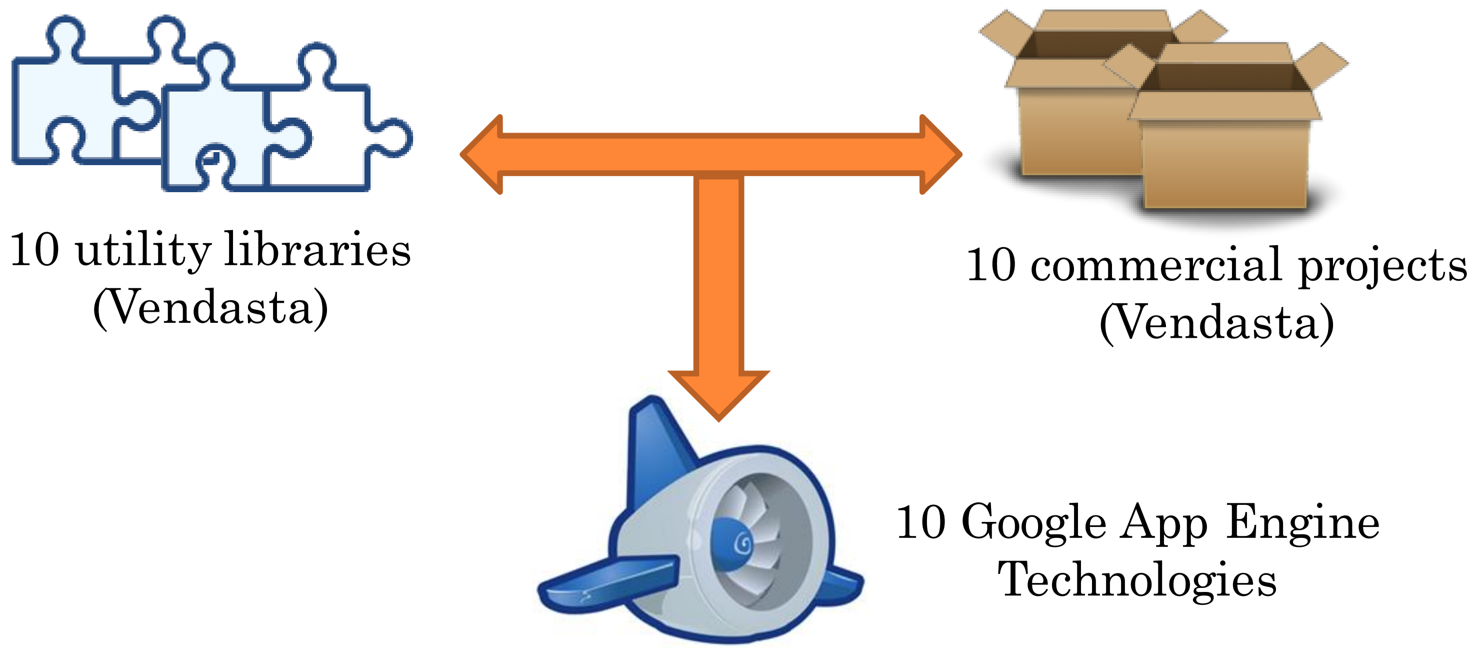
Exploratory Study (3 RQs)

- **RQ$_1$**: How **frequently** do the commercial software projects **reuse external libraries** from within the codebase?

- **RQ$_2$**: Does the **experience** of a developer with such libraries matter in **code reviewer selection** by other developers?

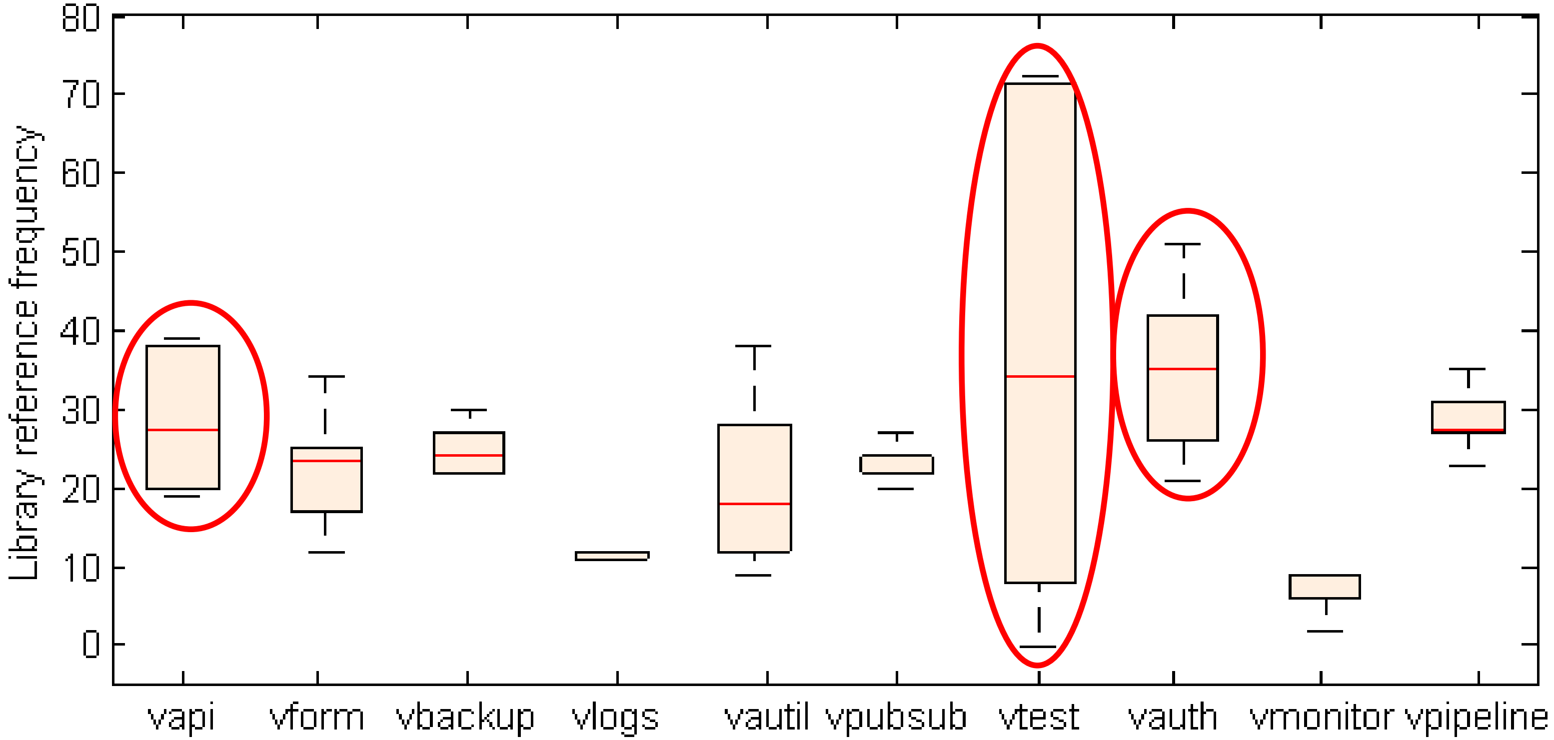
- **RQ$_3$**: How **frequently** do the commercial projects adopt **specialized technologies** (e.g., taskqueue, mapreduce, urlfetch)?

DATASET: EXPLORATORY STUDY



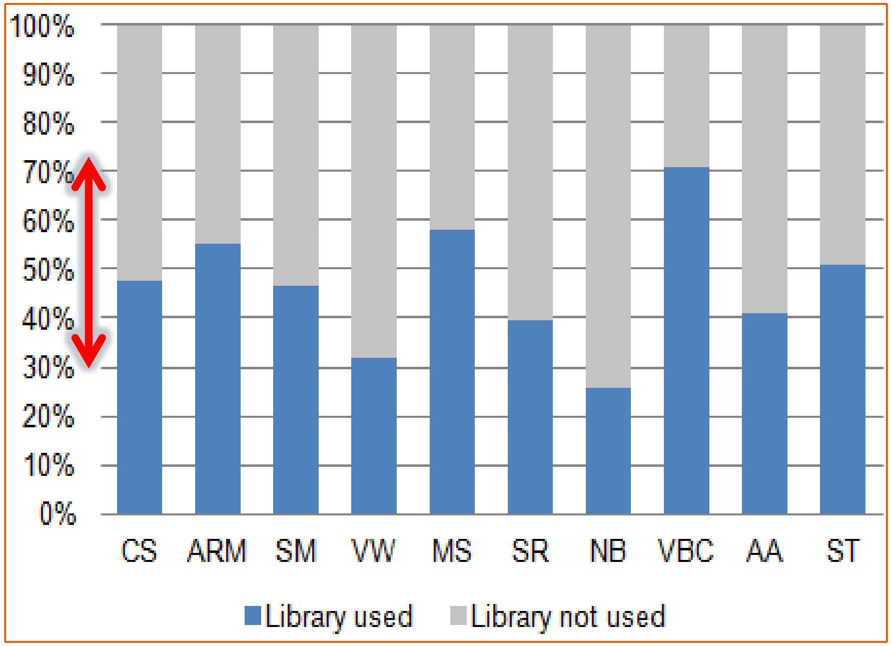
- Each project has at least **750 closed** pull requests.
- Each library is **used** at least **10** times on average.
- Each technology is **used** at least **5** times on average.

Library Usage in Commercial Projects (Answered: Exp-RQ_1)

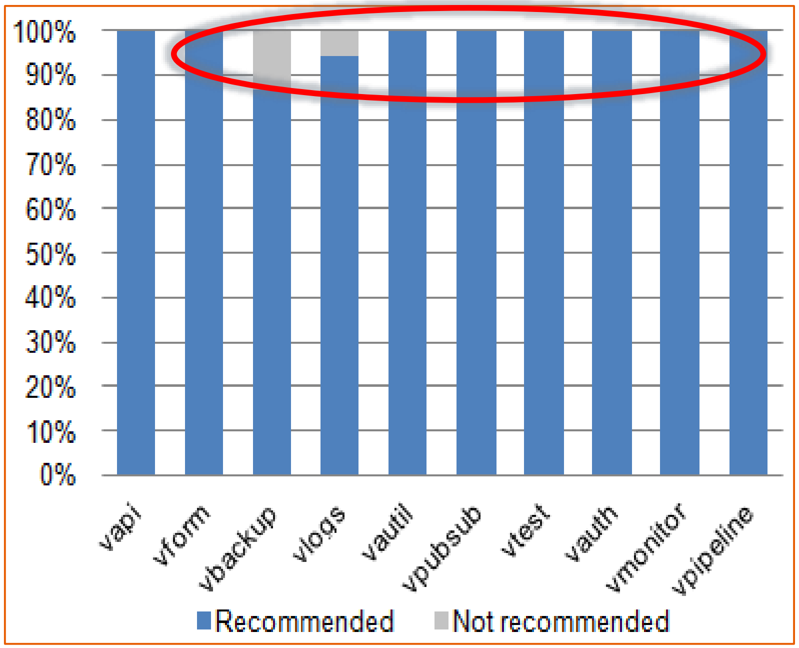


- Empirical library usage frequency in 10 projects
- Mostly used: ***vtest, vauth***, and ***vapi***
- Least used: ***vlogs, vmonitor***

Library Usage in Pull Requests (Answered: Exp-RQ_2)



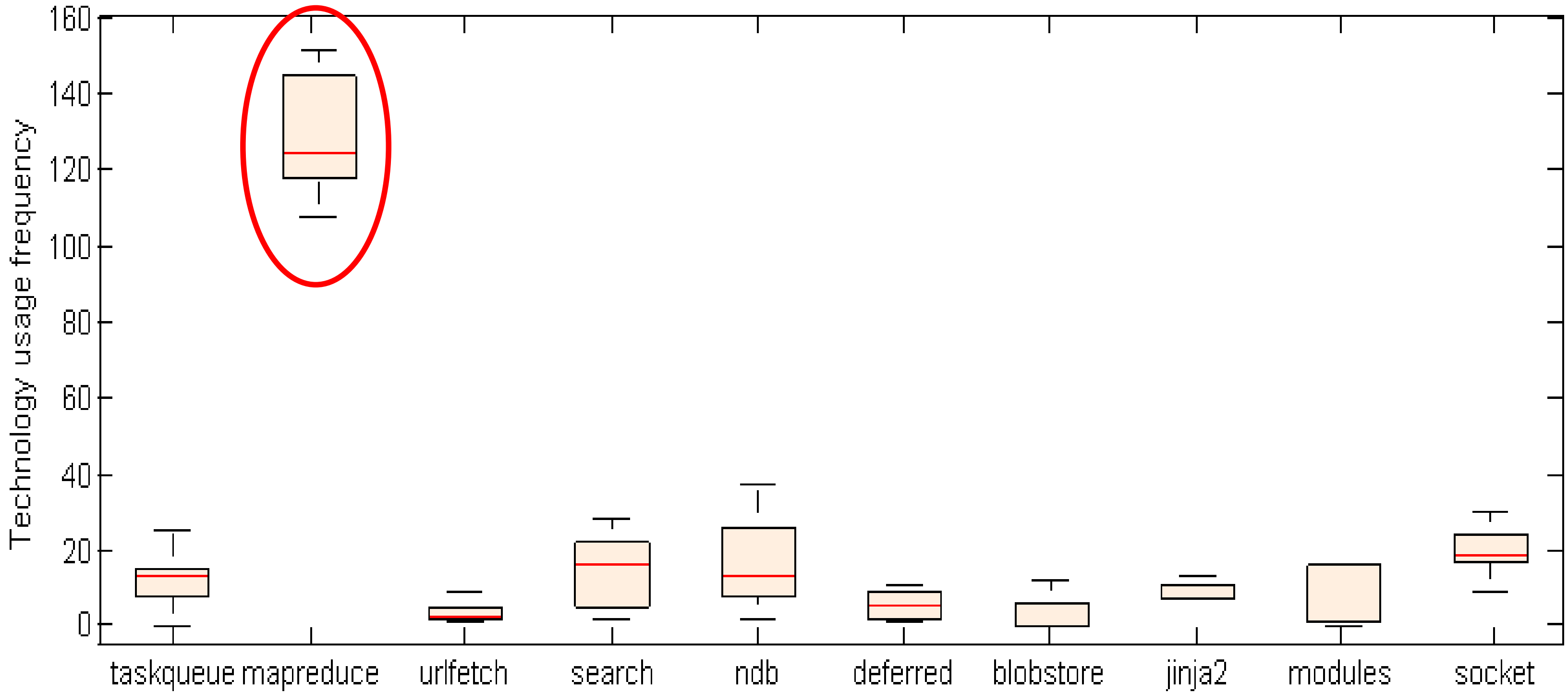
% of PR using selected libraries



% of library authors as code reviewers

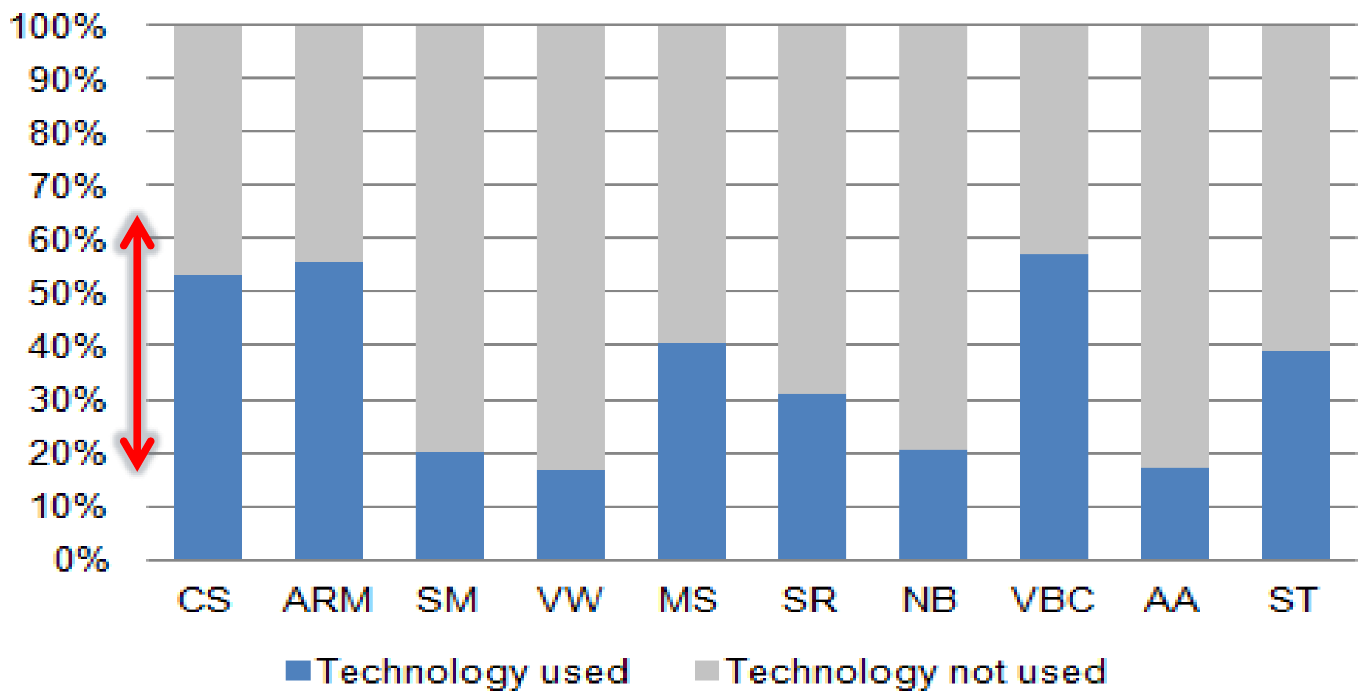
- **30%-70%** of pull requests used at least one of the 10 libraries
- **87%-100%** of library authors recommended as **code reviewers** in the projects using those libraries
- **Library experience** really **matters**!

Specialized Technology Usage in Projects (Answered: Exp-RQ$_3$)



- Empirical technology usage frequency in top **10** commercial projects
- Champion technology: ***mapreduce***

Technology Usage in Pull Requests (Answered: Exp-RQ3)



- **20%-60%** of the pull requests used at least one of the 10 specialized technologies.
- Mostly used in: **ARM, CS** and **VBC**

Summary of Exploratory Findings

About **50%** of the pull requests use one or more of the selected libraries. (Exp-RQ_1)

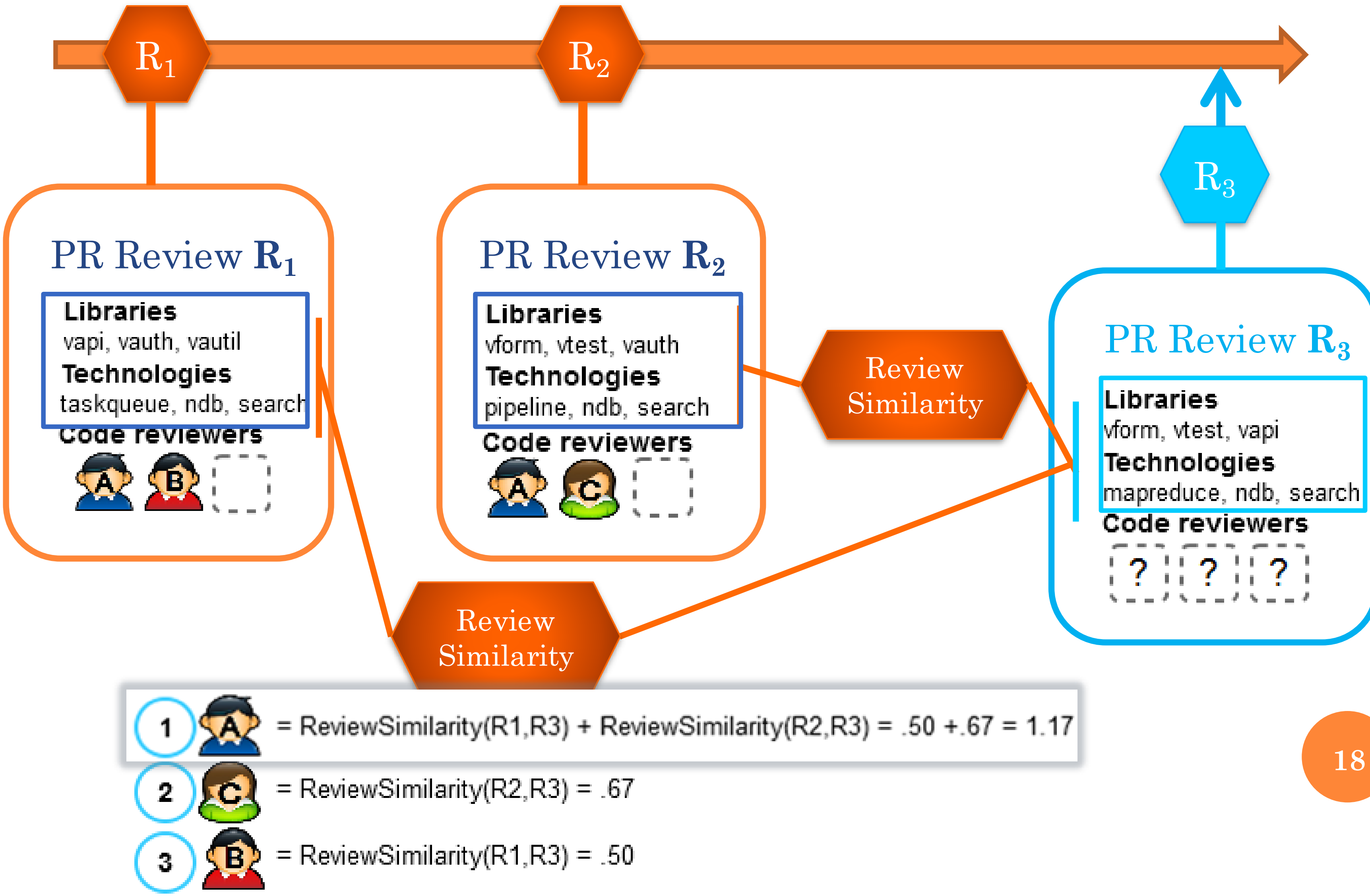
About **98%** of the library authors were later recommended as pull request reviewers. (Exp-RQ_2)

About **35%** of the pull requests use one or more specialized technologies. (Exp-RQ_3)

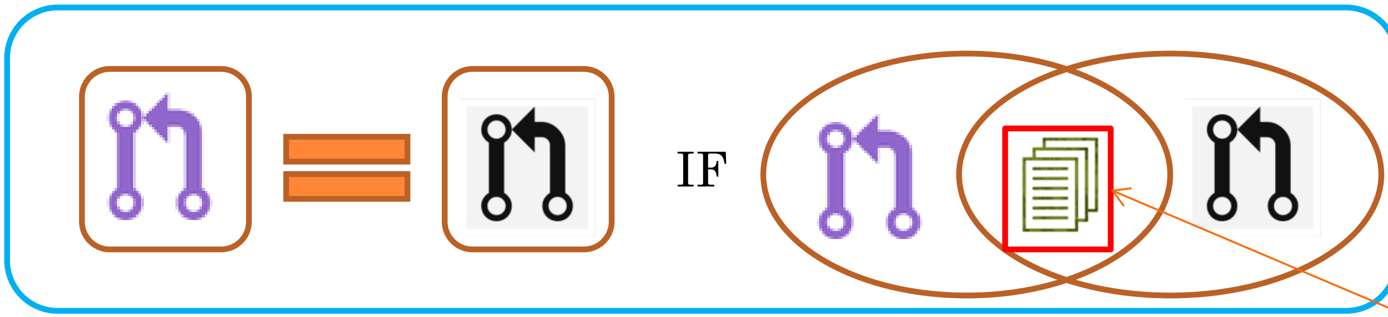
Library experience and **Specialized technology experience** really matter in code reviewer selection/recommendation

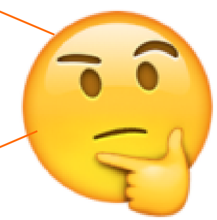
CoRReCT: Code Reviewer Recommendation in GitHub using Cross-project & Technology experience

CORRECT: Code Reviewer Recommendation

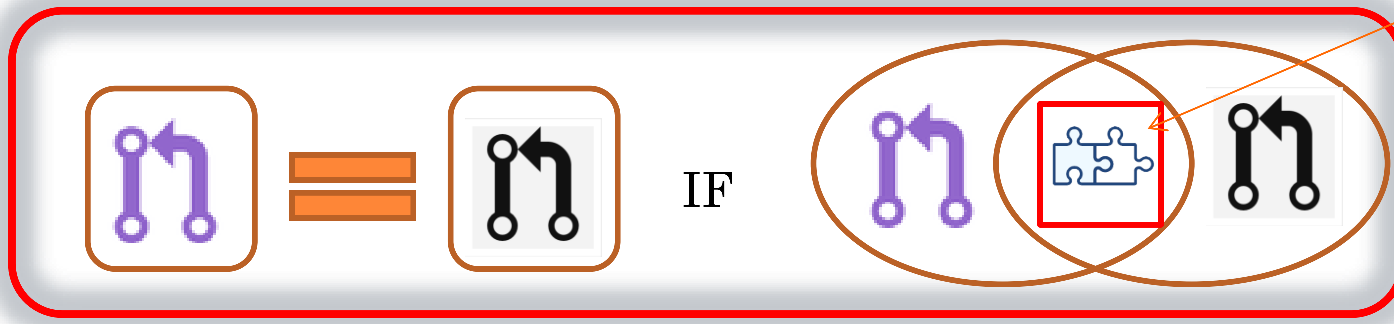


Our Contributions





State-of-the-art (Thongtanunam et al, SANER 2015)



Our proposed technique--CORRECT

= New PR = Reviewed PR = Source file

= External library & specialized technology

Evaluation of CORRECT

- **Two** evaluations using-- (1) Vendasta codebase (2) Open source software projects

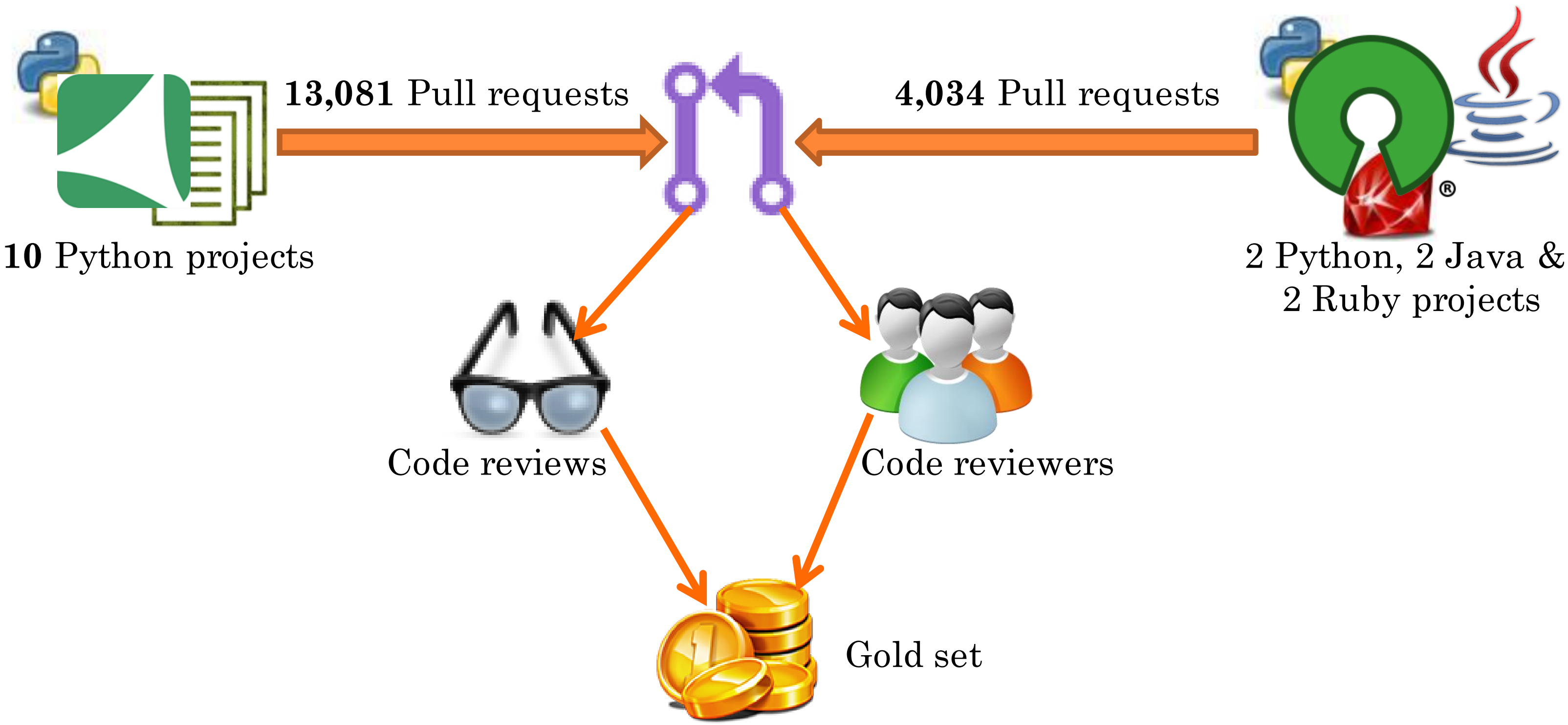
1: Are **library experience** and **technology experience** useful proxies for code review skills?

2: Does CoRReCT **outperform** the baseline technique for reviewer recommendation?

3: Does CoRReCT perform **equally/comparably** for both private and public codebase?

4: Does CoRReCT show **bias** to any of the development frameworks

EXPERIMENTAL DATASET



- **Sliding window** of **30** past requests for learning.
- **Metrics:** Top-K Accuracy, Mean Precision (MP), Mean Recall (MR), and Mean Reciprocal rank (MRR).

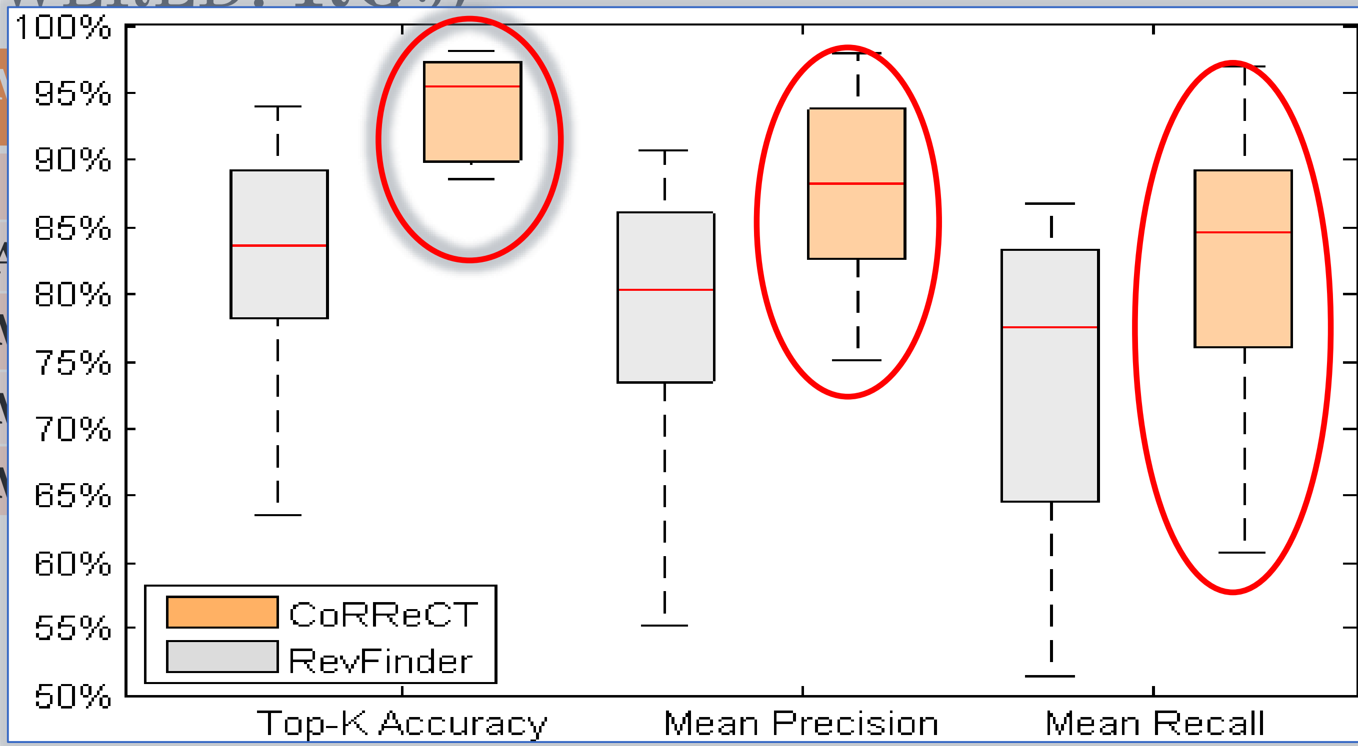
Library Experience & Technology Experience (Answered: RQ_1)

Metric	Library Similarity		Technology Similarity		Combined Similarity	
	Top-3	**Top-5**	**Top-3**	**Top-5**	**Top-3**	**Top-5**
Accuracy	83.57%	92.02%	82.18%	91.83%	83.75%	**92.15%**
MRR	0.66	0.67	0.62	0.64	0.65	0.67
MP	65.93%	85.28%	62.99%	83.93%	65.98%	**85.93%**
MR	58.34%	80.77%	55.77%	79.50%	58.43%	**81.39%**

[**MP** = Mean Precision, **MR** = Mean Recall, **MRR** = Mean Reciprocal Rank]

- Both **library experience** and **technology experience** are found as good proxies, provide over **90% accuracy**.
- Combined experience provides the **maximum** performance.
- **92.15%** recommendation accuracy with **85.93%** precision and **81.39%** recall.
- Evaluation results align with exploratory study findings.

Comparative Study Findings (Answered: RQ_2)

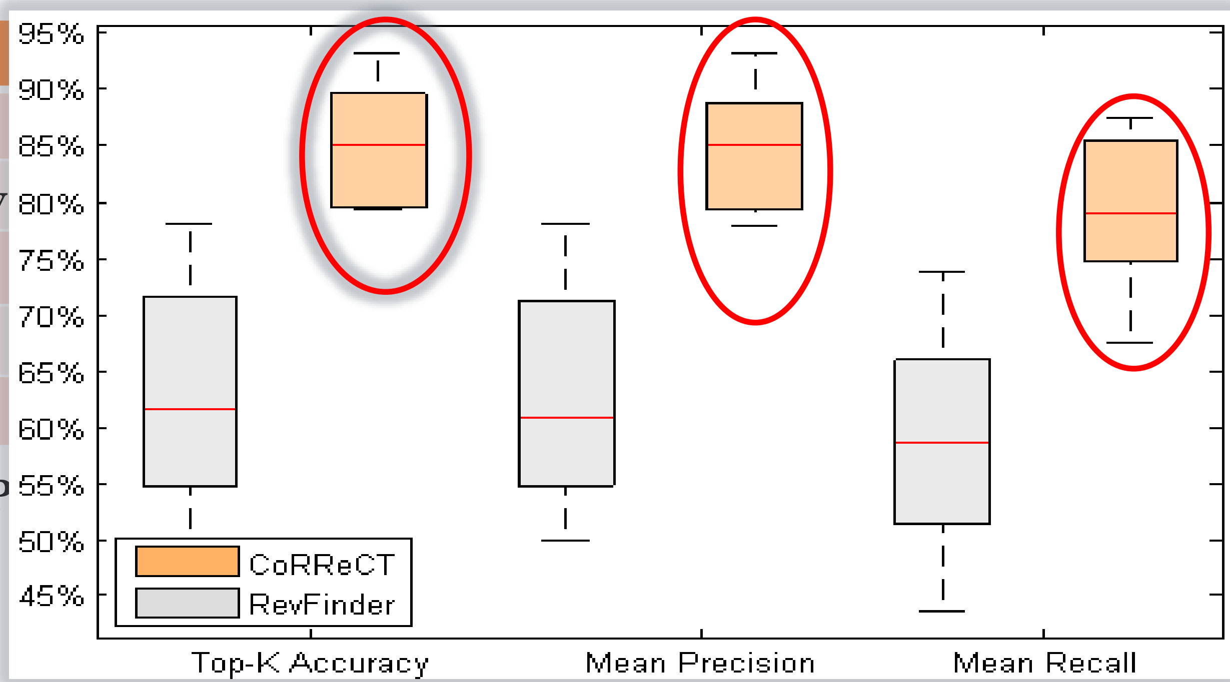


- **CoRReCT** performs better than the competing technique in **all metrics** (*p-value*=**0.003<0.05** for Top-5 accuracy)
- Performs better both **on average** and **on individual** projects.
- RevFinder uses **PR similarity** using source file **name** and file's **directory** matching

Comparison on Open Source Projects (Answered: RQ_3)

Metric	eCT (VA)
	Top-5
Accuracy	92.15%
MRR	0.67
MP	85.93%
MR	81.39%

[MP al Rank]



- In **OSS** projects, CoRReCT also **performs better** than the baseline technique.
- **85.20% accuracy** with **84.76% precision** and **78.73% recall,** and not significantly different than earlier (*p-value*=**0.239>0.05** for precision)
- Results for **private** and **public** codebase are **quite close**.

Comparison on Different Platforms (Answered: RQ_4)

Metrics	Python			Java			Ruby		
	Beets	St2	Avg.	OkHttp	Orientdb	Avg.	Rubocop	Vagrant	Avg.
Accuracy	93.06%	79.20%	**86.13%**	88.77%	81.27%	**85.02%**	89.53%	79.38%	**84.46%**
MRR	0.82	0.49	0.66	0.61	0.76	0.69	0.76	0.71	0.74
MP	93.06%	77.85%	**85.46%**	88.69%	81.27%	**84.98%**	88.49%	79.17%	**83.83%**
MR	87.36%	74.54%	**80.95%**	85.33%	76.27%	**80.80%**	81.49%	67.36%	**74.43%**

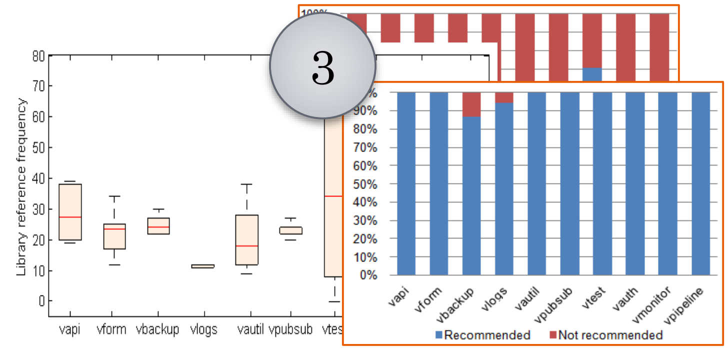
[**MP** = Mean Precision, **MR** = Mean Recall, **MRR** = Mean Reciprocal Rank]

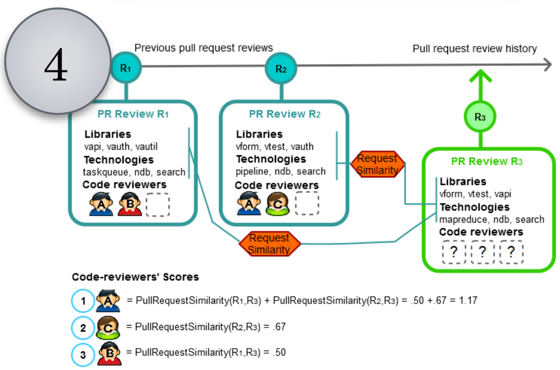
- In OSS projects, results for **different platforms** look **surprisingly close** except the recall.
- Accuracy and precision are close to **85%** on average.
- CORRECT **does NOT** show **any bias** to any particular platform.

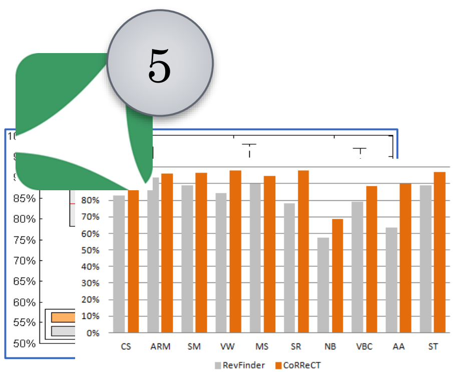
Threats to Validity

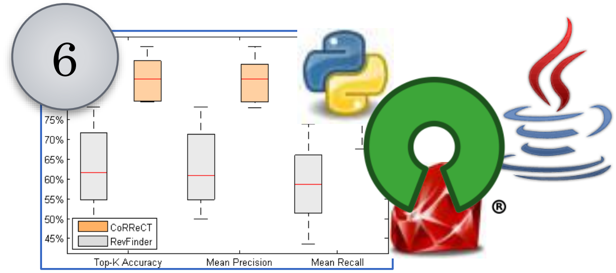
- **Threats to Internal Validity**
  - *Skewed dataset:* Each of the 10 selected projects is medium sized (i.e., 1.1K PR) except **CS.**
- **Threats to External Validity**
  - *Limited OSS dataset:* Only 6 OSS projects considered—not sufficient for generalization.
  - *Issue of heavy PRs:* PRs containing hundreds of files can make the recommendation **slower**.
- **Threats to Construct Validity**
  - *Top-K Accuracy:* Does the metric represent *effectiveness* of the technique? Widely used by relevant literature (Thongtanunam et al, SANER 2015)

TAKE-HOME MESSAGES (PART I)



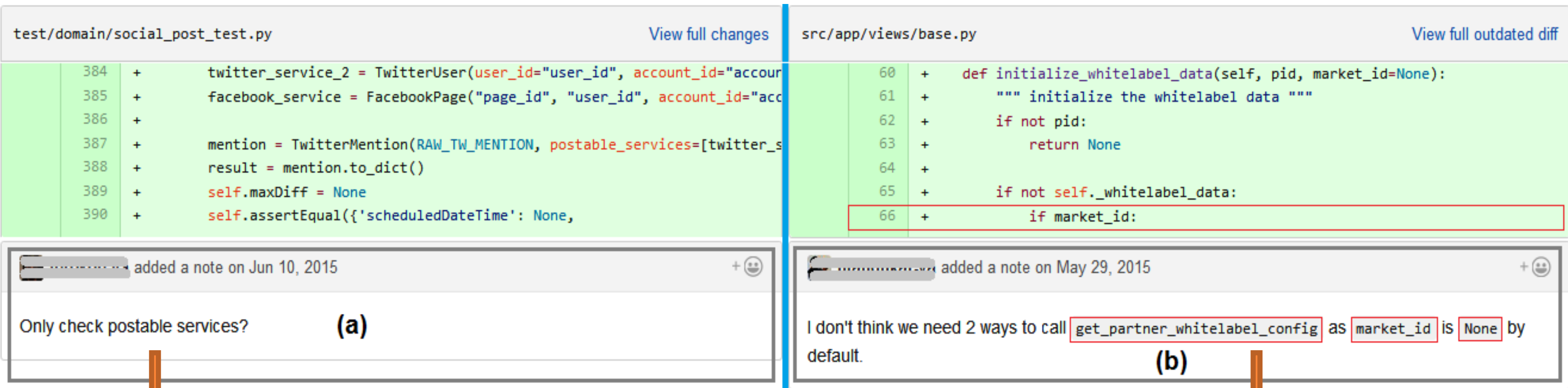






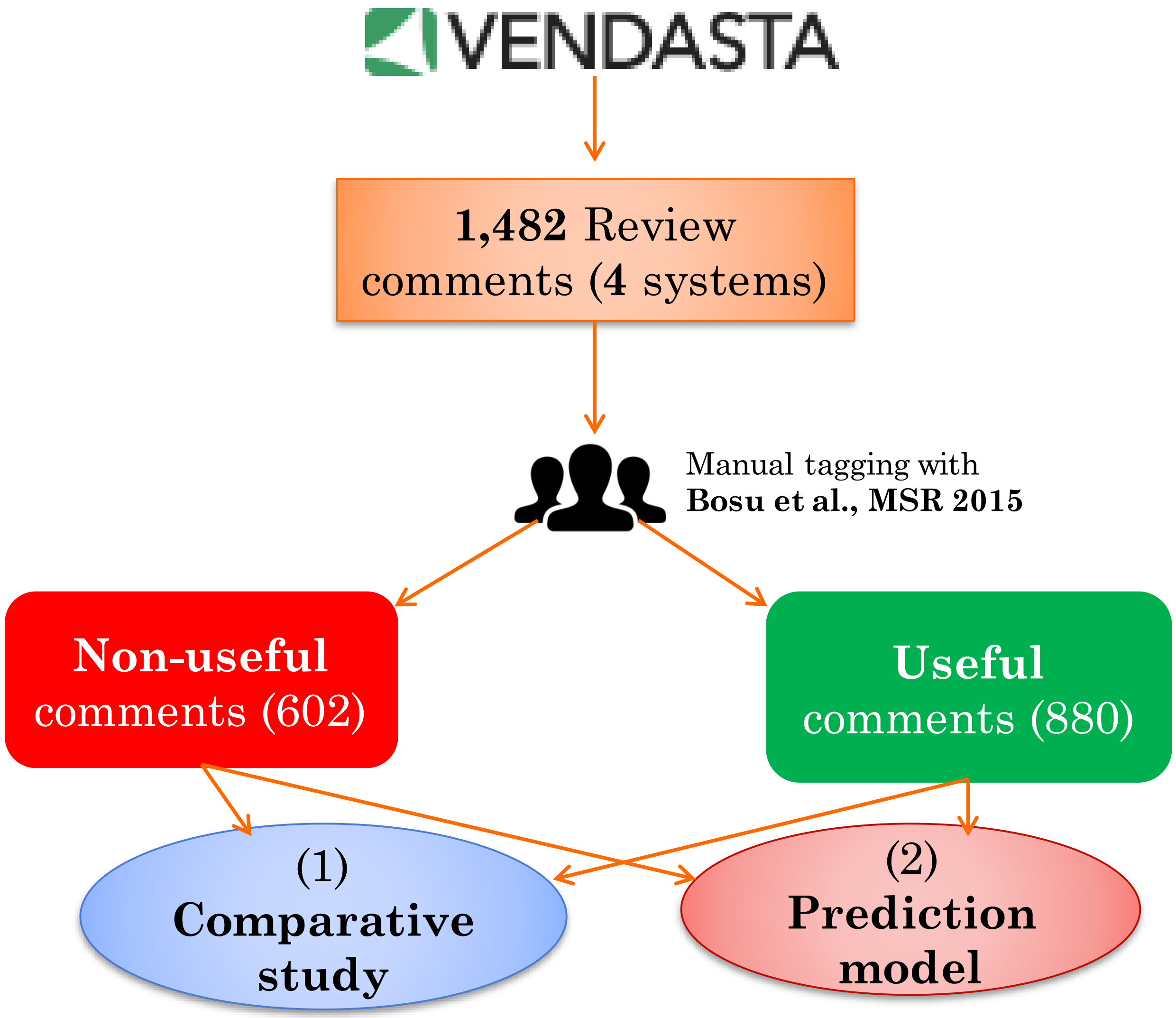
Part II: Prediction Model for Code Review Usefulness (MSR 2017)

Research Problem: Usefulness of Code Review Comments



- What makes a review comment **useful** or **non-useful**?
- **34.5%** of review comments are **non-useful** at Microsoft (Bosu et al., MSR 2015)
- No automated support to **detect** or **improve** such comments so far

Study Methodology

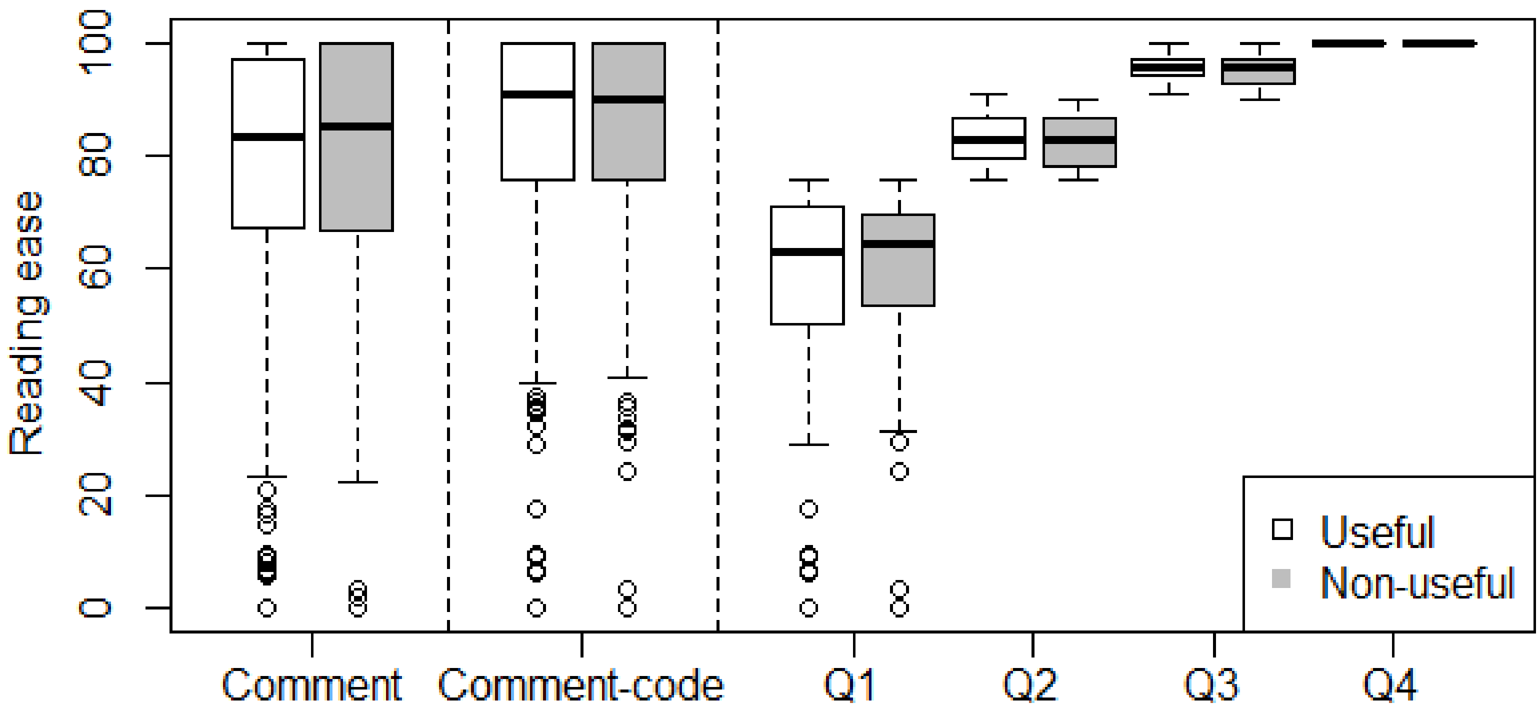


Comparative Study: Variables

- **Contrast** between **useful** and **non-useful** comments.
- **Two** paradigms– **comment texts**, and **commenter's/developer's experience**
- Answers **two RQs** related to two paradigms.

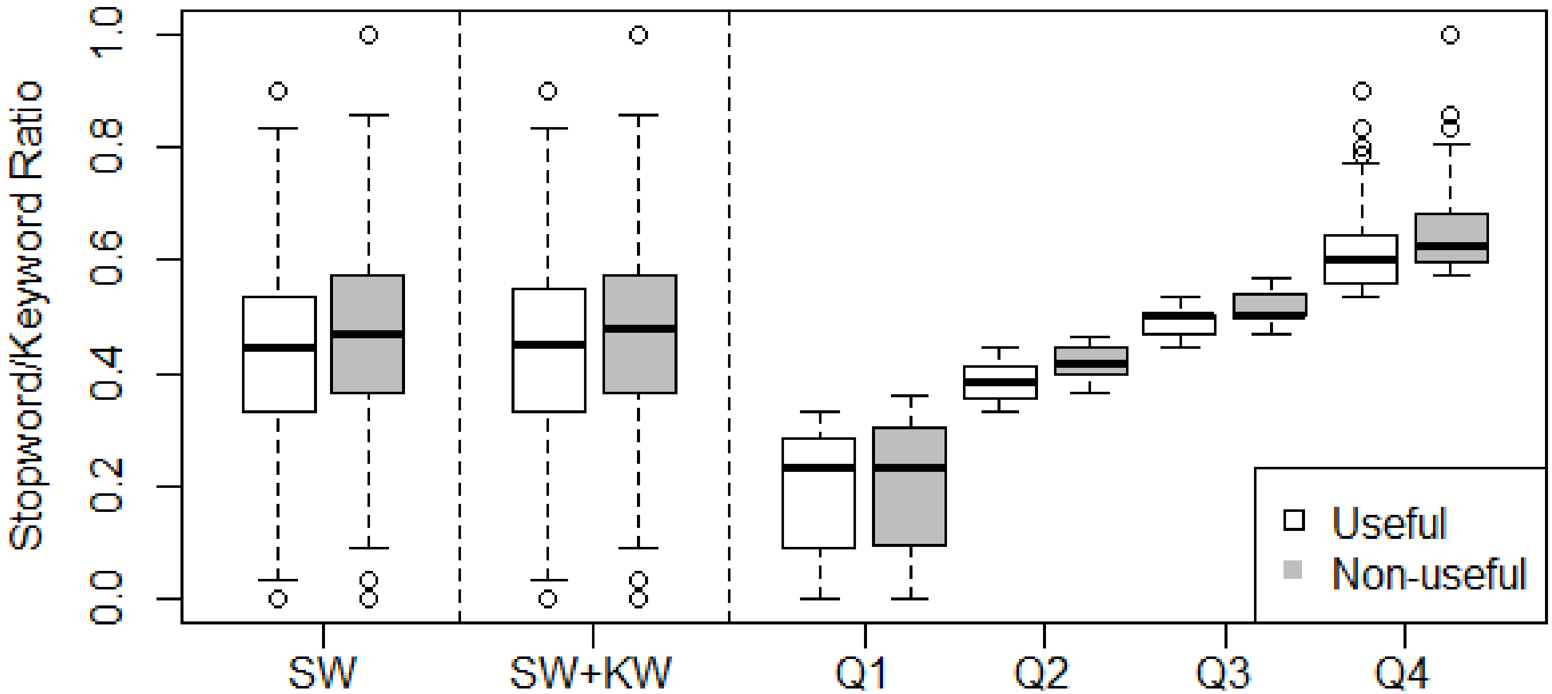
Independent Variables (8)		Response Variable (1)
Reading Ease	*Textual*	Comment Usefulness **(Yes / No)**
Stop word Ratio	*Textual*	
Question Ratio	*Textual*	
Code Element Ratio	*Textual*	
Conceptual Similarity	*Textual*	
Code Authorship	*Experience*	
Code Reviewership	*Experience*	
External Lib. Experience	*Experience*	

ANSWERING RQ$_1$: READING EASE



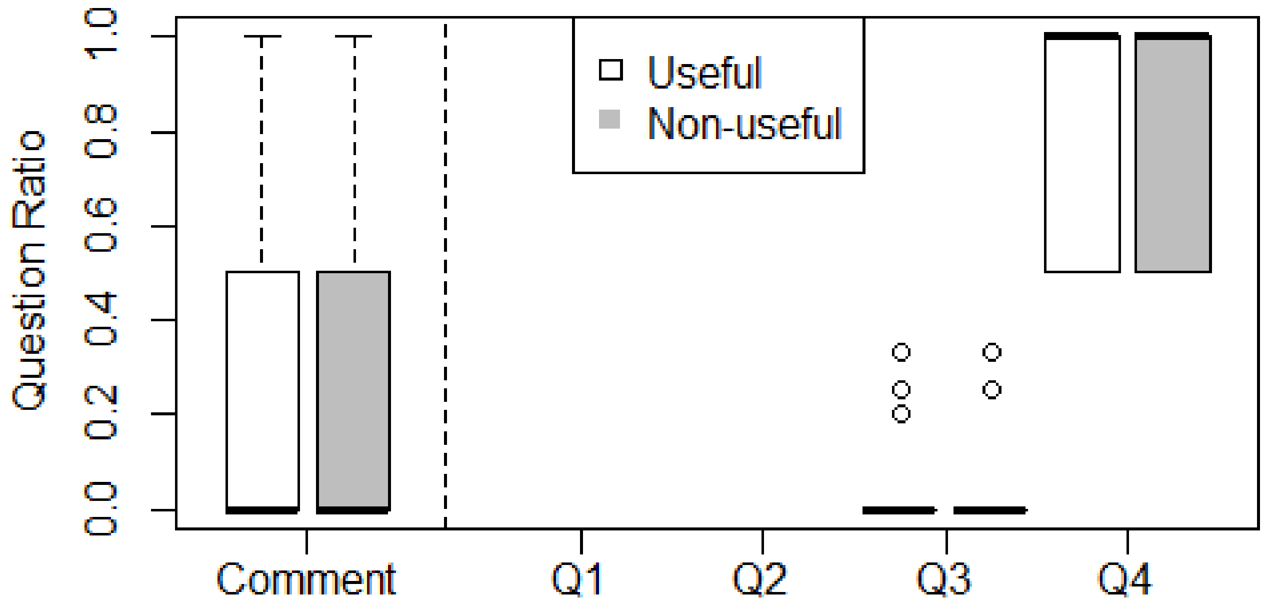
- **Flesch-Kincaid Reading Ease** applied.
- **No significant difference** between useful and non-useful review comments.

Answering RQ_1: Stop word Ratio



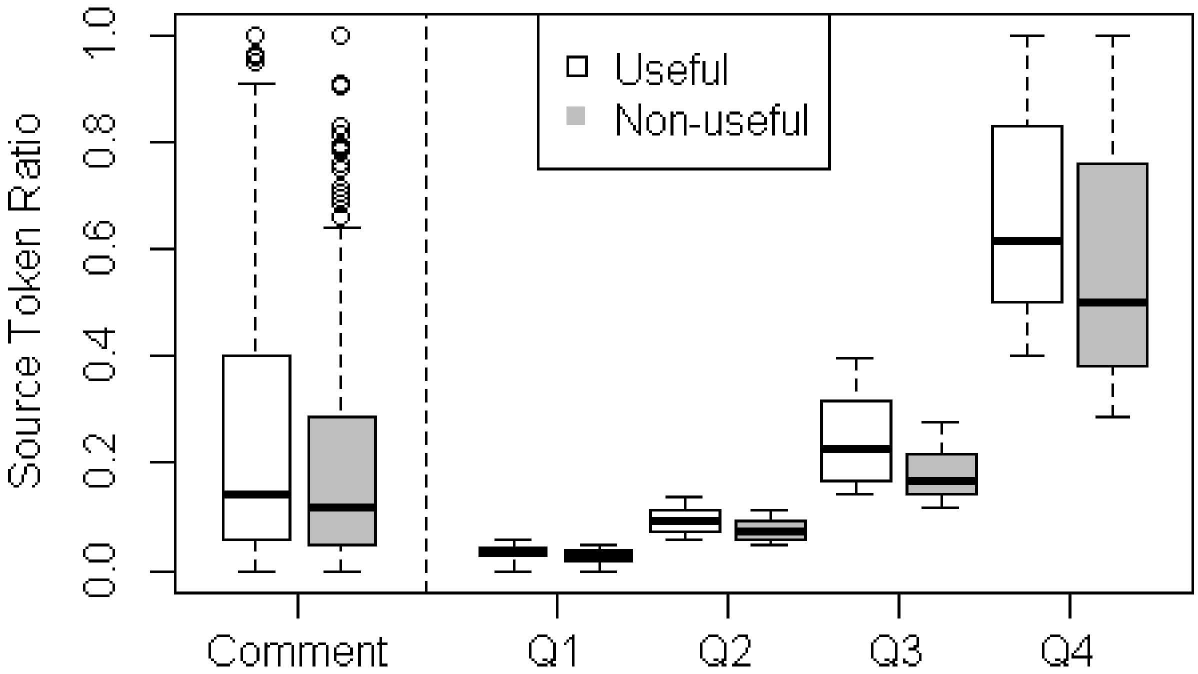
- Used Google stop word list and Python keywords.
- **Stop word ratio = #stop or keywords/#all words** from a review comment
- **Non-useful comments** contain **more stop words** than **useful comments**, i.e., *statistically significant*.

ANSWERING RQ$_1$: QUESTION RATIO



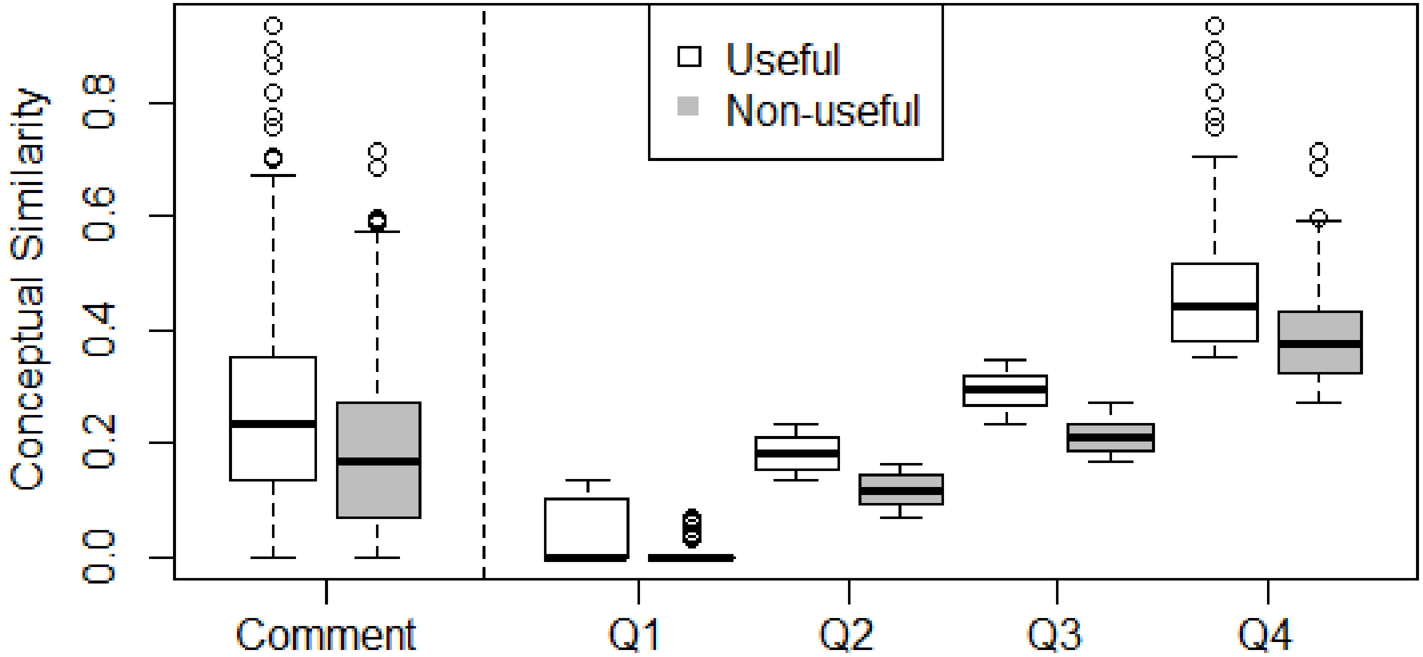
- Developers treat **clarification questions** as non-useful review comments.
- **Question ratio = #questions/#sentences** of a comment.
- **No significant difference** between useful and non-useful comments in **question ratio**.

ANSWERING RQ$_1$: CODE ELEMENT RATIO



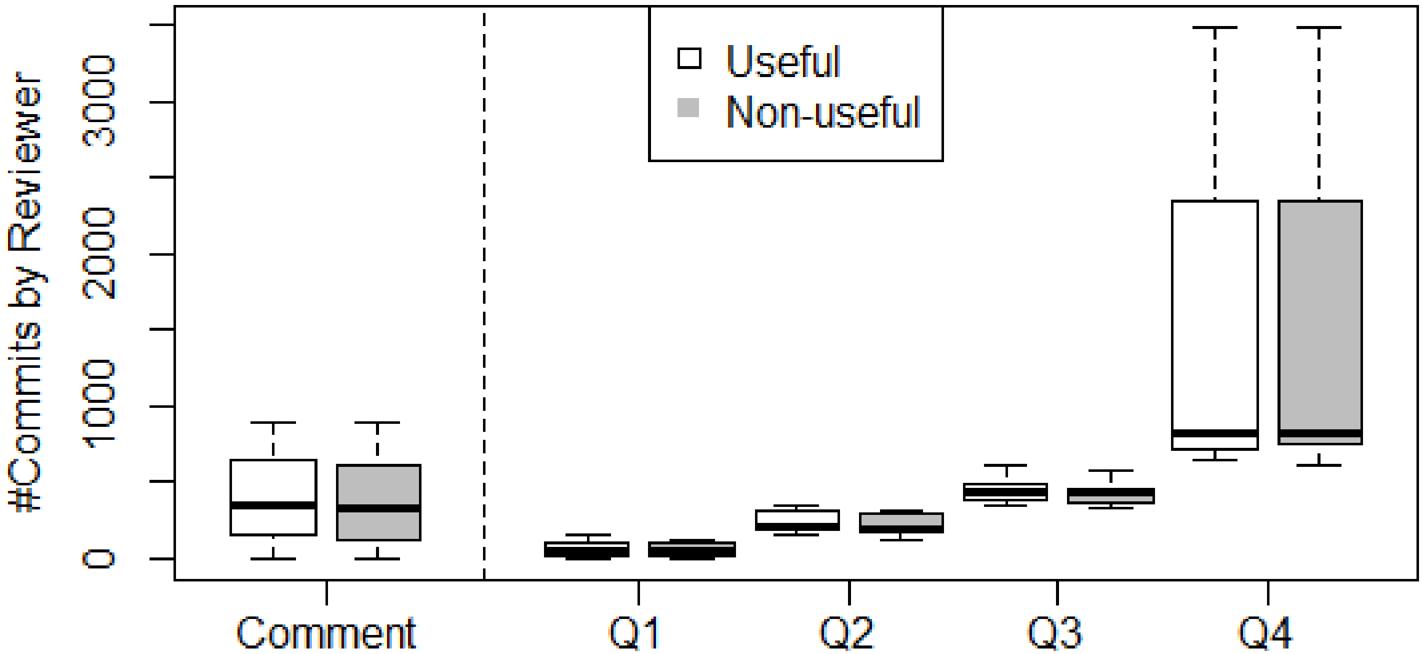
- Important code elements (e.g., identifiers) in the comments texts, possibly trigger the code change.
- **Code element ratio = #source tokens/#all tokens**
- **Useful comments > non-useful comments** for code element ratio, i.e., *statistically significant*.

Answering RQ_1: Conceptual Similarity between Comments & Changed Code



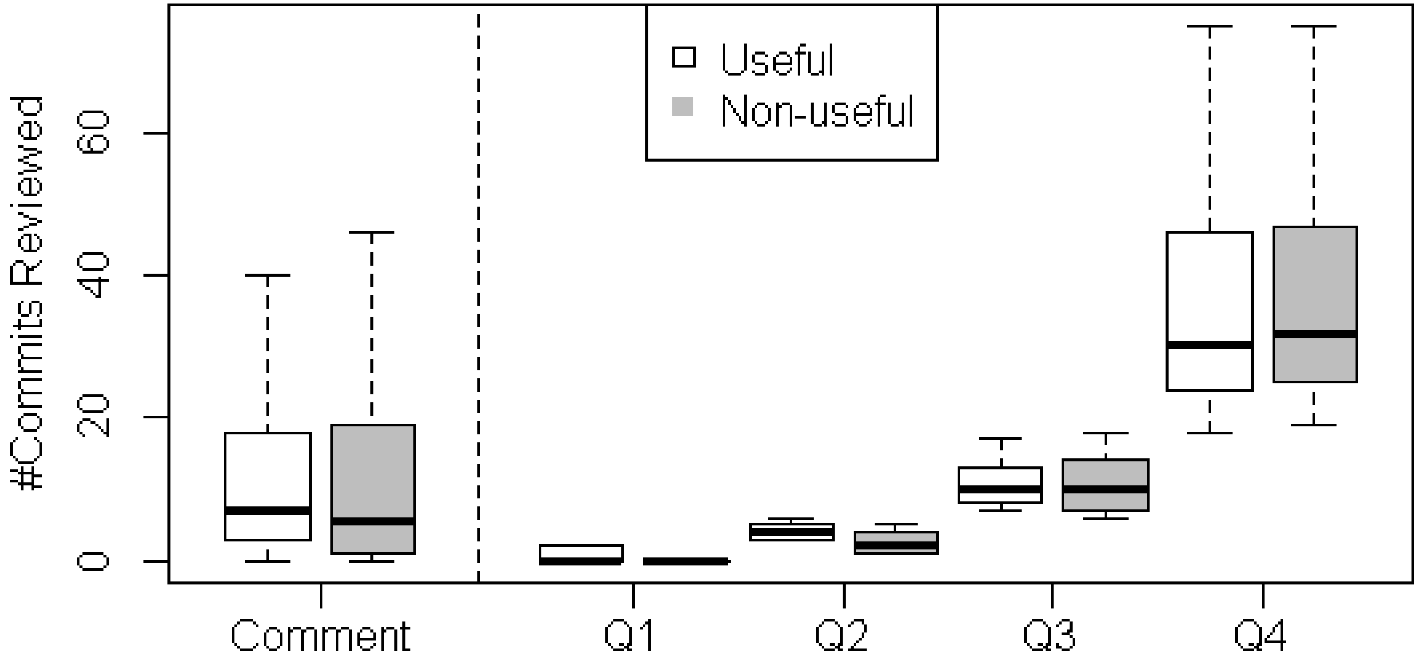
- How **relevant** the **comment** is with the **changed code**?
- Do comments & changed code **share vocabularies**?
- **Yes**, useful comments **do more sharing** than non-useful ones, i.e., *statistically significant*.

ANSWERING RQ$_2$: CODE AUTHORSHIP



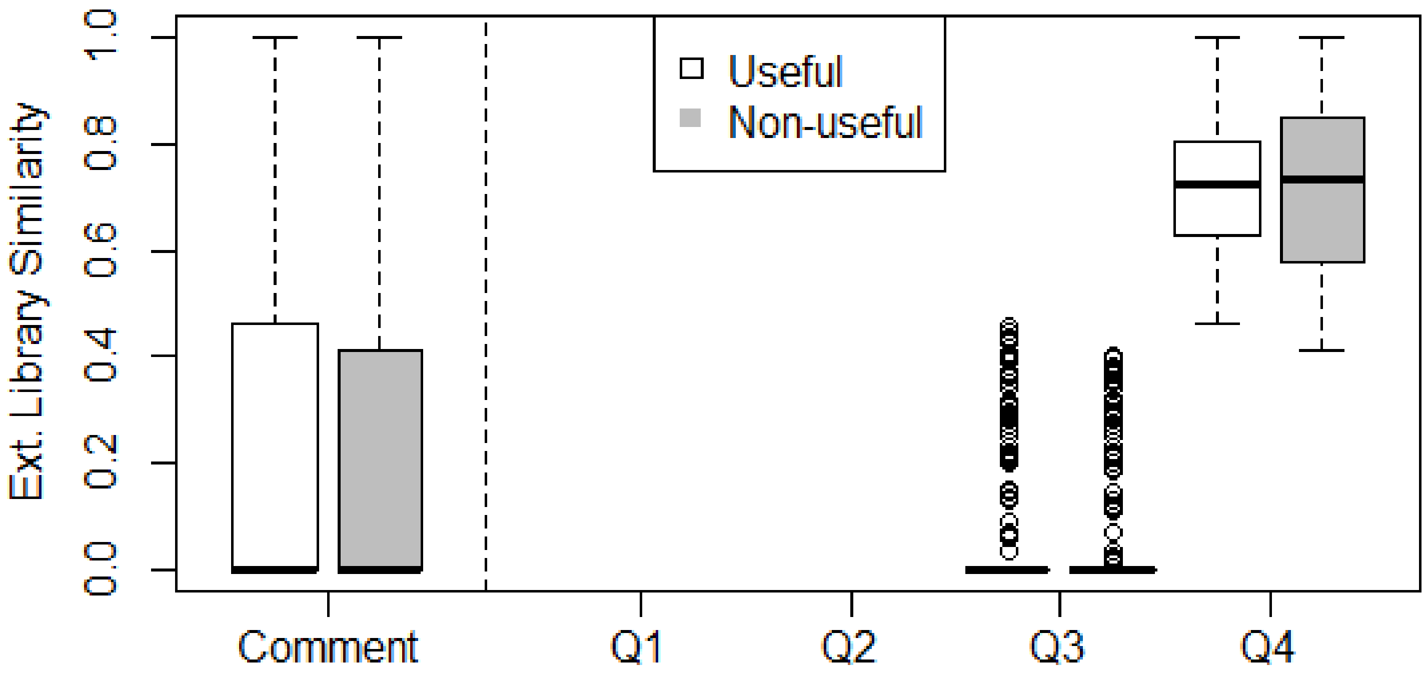
- **File level authorship** did not make much difference, a bit counter-intuitive.
- **Project level authorship** differs between useful and non-useful comments, mostly for Q2 and Q3

ANSWERING RQ$_2$: CODE REVIEWERSHIP



- Does **reviewing experience** matter in providing **useful comments**?
- **Yes**, it does. **File level reviewing experience matters**. Especially true for Q2 and Q3.
- **Experienced reviewers** provide more useful comments than non-useful comments.

ANSWERING RQ_2: EXT. LIB. EXPERIENCE



- **Familiarity** with the library used in the changed code for which comment is posted.
- *Significantly higher* for the authors of useful comments for Q3 only.

Summary of Comparative Study

RQ	Independent Variables	Useful vs. Non-useful Difference
RQ_1	Reading Ease	Not significant
	Stop word Ratio	**Significant**
	Question Ratio	Not significant
	Code Element Ratio	**Significant**
	Conceptual Similarity	**Significant**
RQ_2	Code Authorship	Somewhat significant
	Code Reviewership	**Significant**
	External Lib. Experience	Somewhat significant

Experimental Dataset & Setup

1,482 code review comments

Evaluation set (**1,116**)

Model training & cross-validation

Validation set (**366**)

Validation with unseen comments

REVHELPER: USEFULNESS PREDICTION MODEL

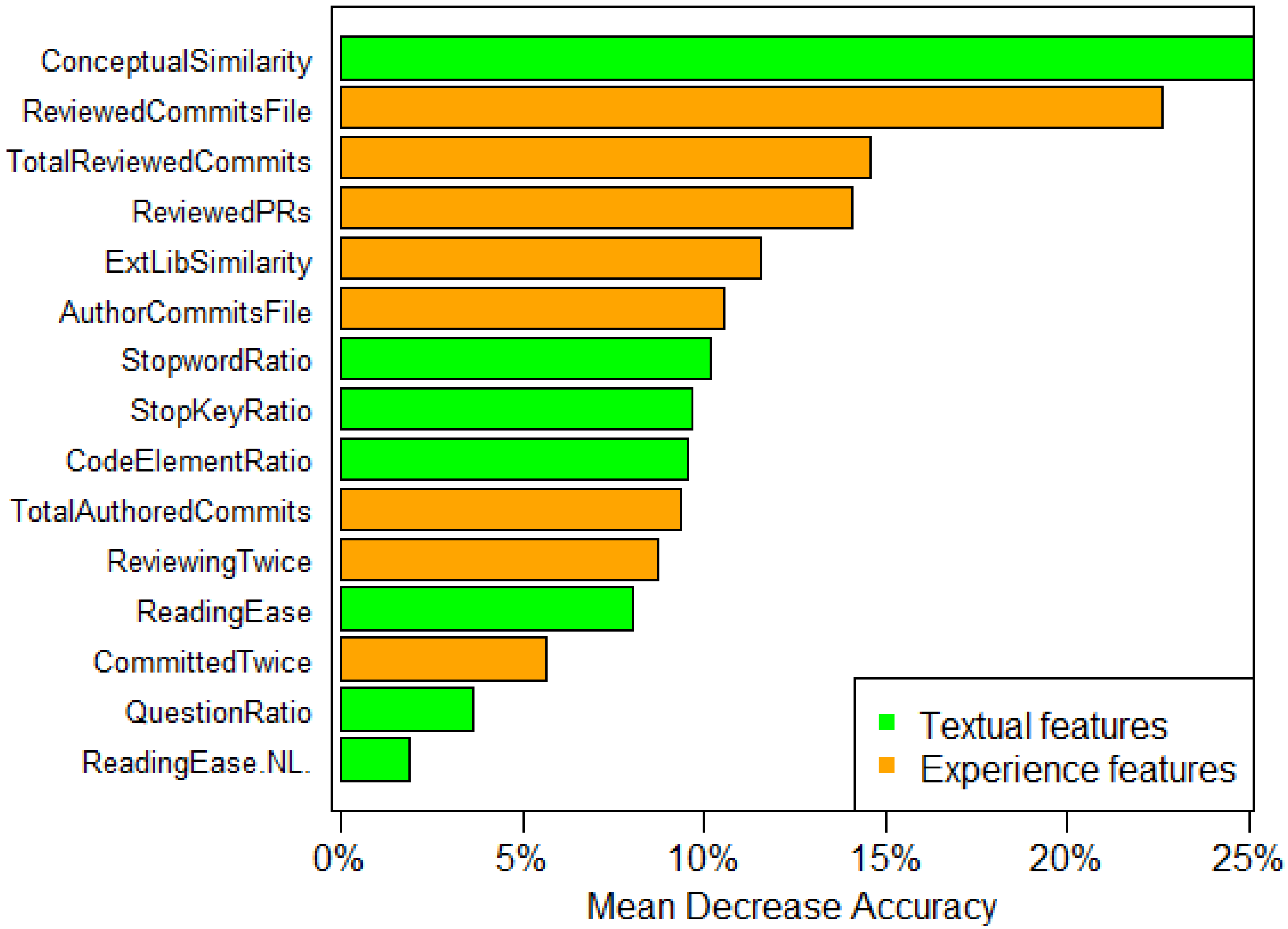
- **Prediction of usefulness** for a **new review comment** to be submitted.
- Applied **three** ML algorithms– **NB, LR,** and **RF**
- Evaluation & validation with different data sets
- Answered **3** RQs– $\mathbf{RQ_3}$, $\mathbf{RQ_4}$ and $\mathbf{RQ_5}$

ANSWERING RQ$_3$: MODEL PERFORMANCE

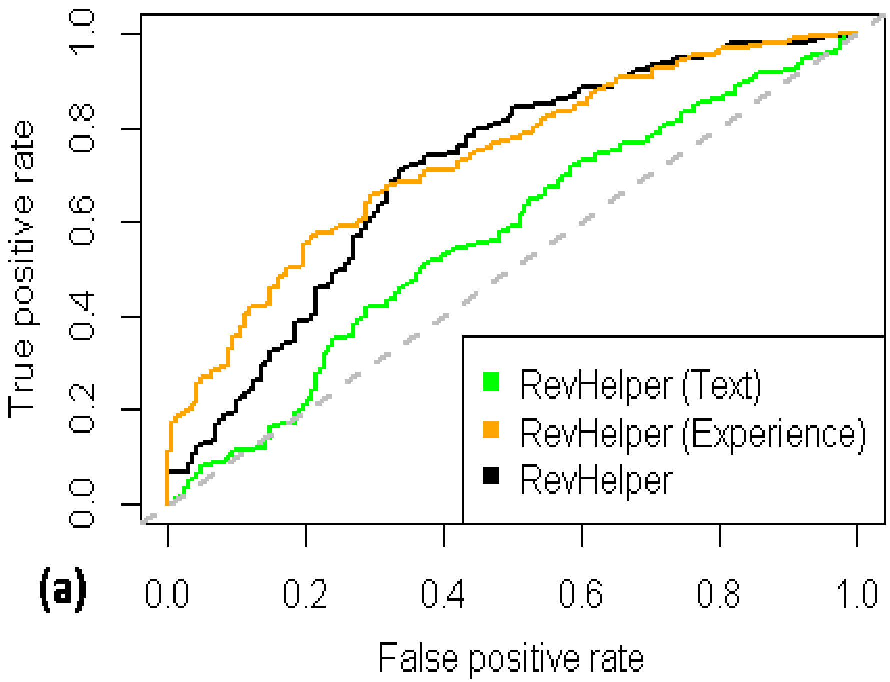
Learning Algorithm	Useful Comments		Non-useful Comments	
	Precision	Recall	Precision	Recall
Naïve Bayes	61.30%	66.00%	53.30%	48.20%
Logistic Regression	60.70%	71.40%	54.60%	42.80%
Random Forest	**67.93%**	**75.04%**	**63.06%**	**54.54%**

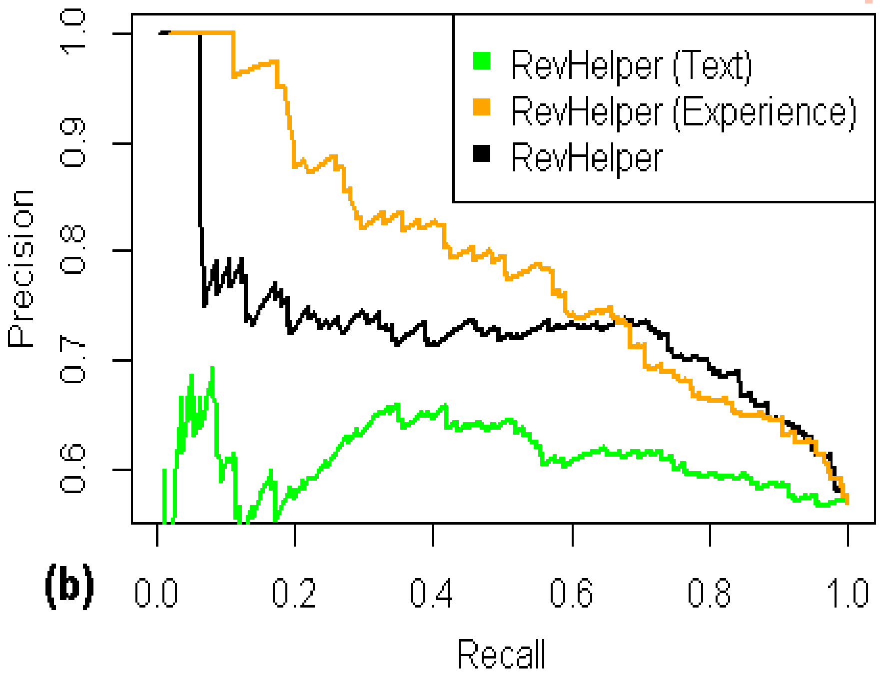
- **Random Forest** based model performs the best.
- Both **F$_1$-score** and **accuracy 66%**.
- Comment usefulness and features are **not linearly correlated.**
- As a **primer**, this prediction could be useful.

Answering RQ_4: Role of Paradigms

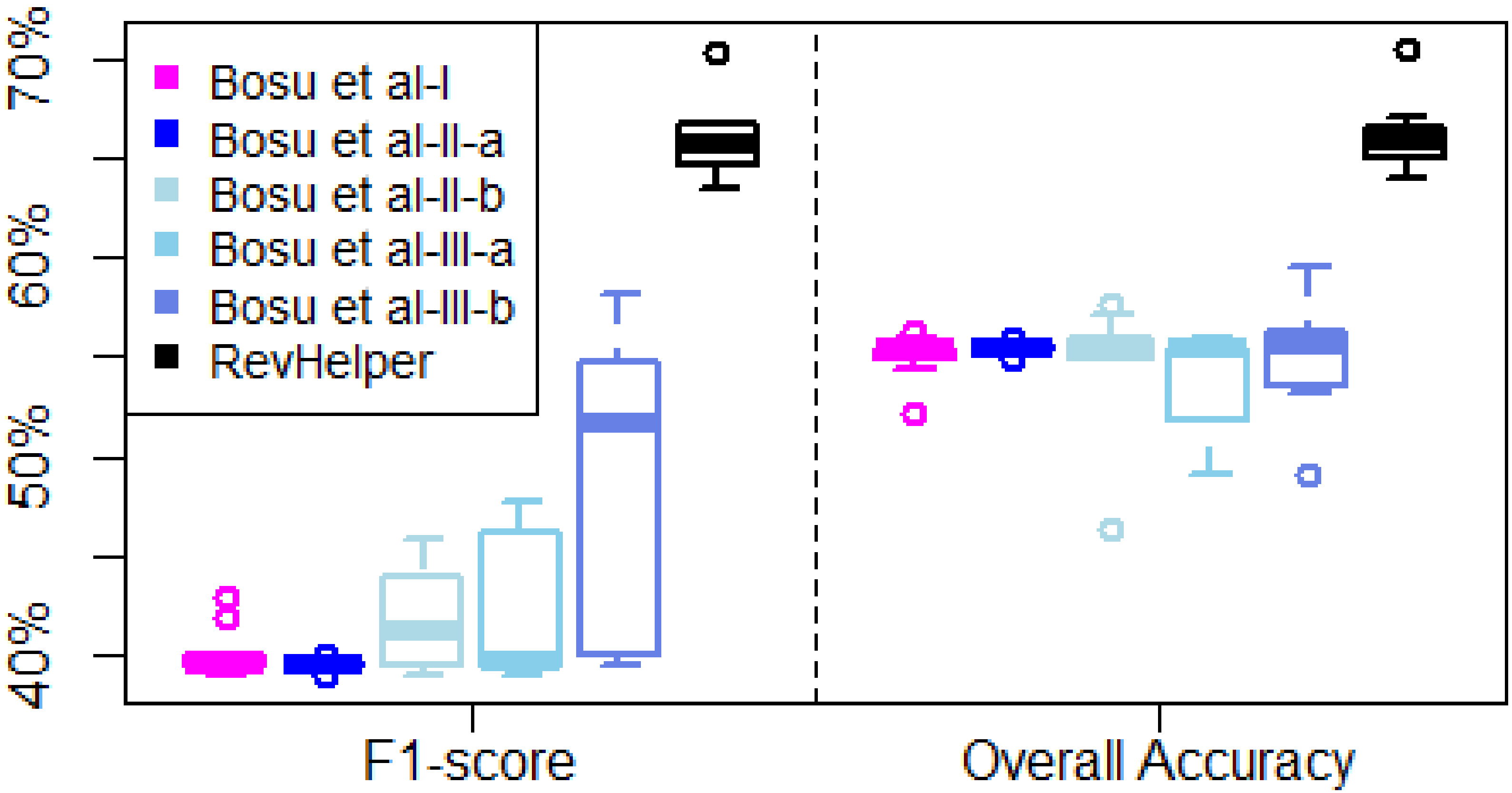


Answering RQ_4: Role of Paradigms

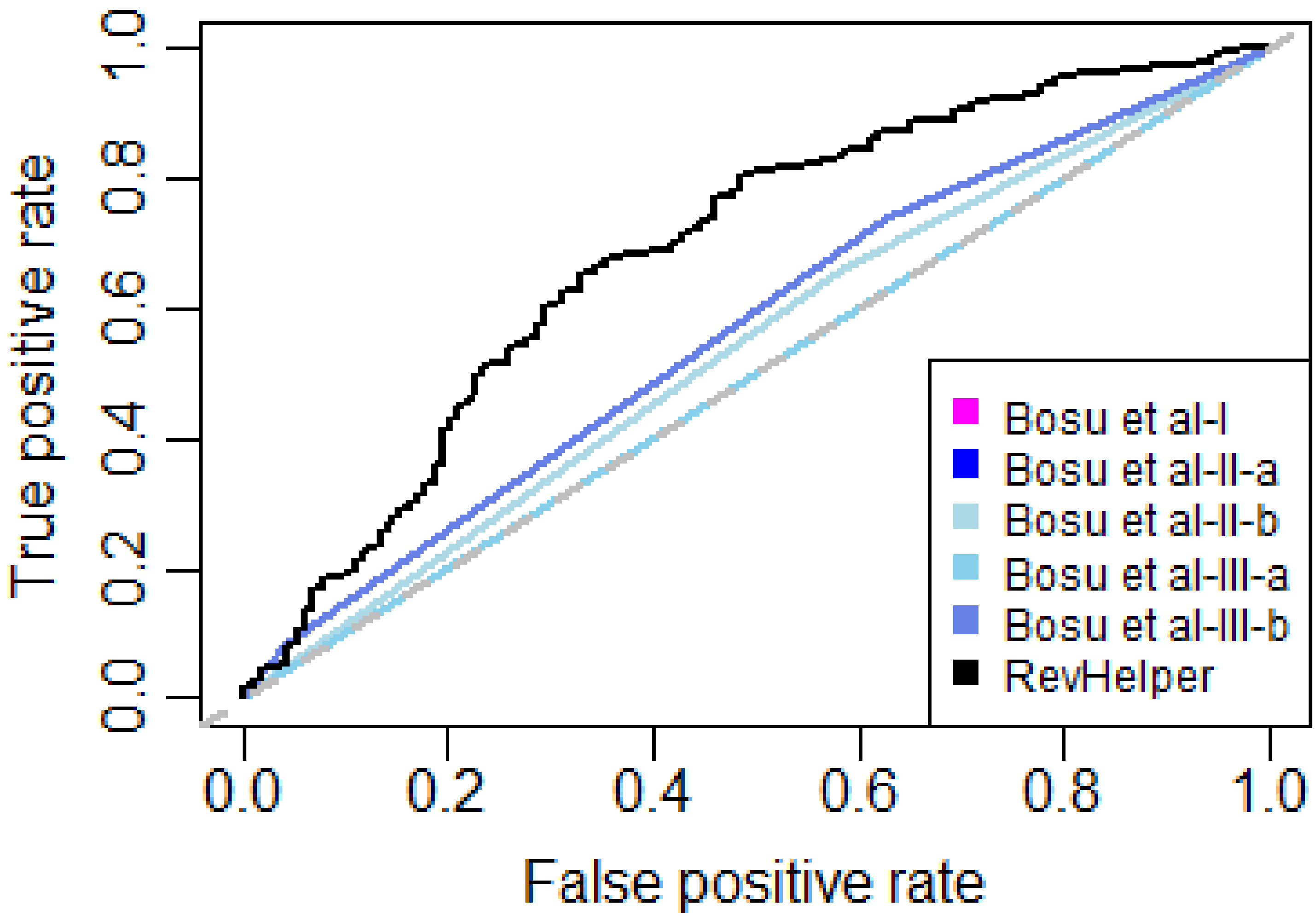




Answering RQ5: Comparison with Baseline (Validation)



Answering RQ5: Comparison with Baseline (ROC)



Take-home Messages (Part II)

- **Usefulness** of **review comments** is **complex** but a much needed piece of information.
- **No automated support** available so far to **predict** usefulness of review comments **instantly.**
- **Non-useful** comments are **significantly different** from **useful** comments in **several textual features** (e.g., **conceptual similarity**)
- **Reviewing experience** matters in providing useful review comments.
- Our **prediction model** can **predict** the **usefulness** of a **new review comment.**
- **RevHelper** performs better than **random guessing** and **available alternatives**.

Part III: Impact of Continuous Integration on Code Reviews (MSR 2017 Challenge)

Take-home Message (Part III)

- **Automated build** might influence **manual code review** since they *interleave* each other in the modern pull-based development
- **Passed builds** more **associated** with review participations, and with new code reviews.
- **Frequently built** projects received **more review comments** than less frequently built ones.
- **Code review activities** are steady over time with frequently built projects. **Not true** for counterparts.
- Our **prediction model** can predict whether a build will **trigger** new code review or not.

Replication Packages

- **CORRECT, RevHelper & Travis CI Miner**
- http://www.usask.ca/~masud.rahman/correct/
- http://www.usask.ca/~masud.rahman/revhelper/
- http://www.usask.ca/~masud.rahman/msrch/travis/

Please contact **Masud Rahman (masud.rahman@usask.ca)** for further details about these studies and replications.

Published Papers

[1] M. Masudur Rahman, C.K. Roy, and Jason Collins, ***"CORRECT: Code Reviewer Recommendation in GitHub Based on Cross-Project and Technology Experience"***, In Proceeding of The 38th International Conference on Software Engineering Companion (**ICSE-C 2016**), pp. 222--231, Austin Texas, USA, May 2016

[2] M. Masudur Rahman, C.K. Roy, Jesse Redl, and Jason Collins, ***"CORRECT: Code Reviewer Recommendation at GitHub for Vendasta Technologies"***, In Proceeding of The 31st IEEE/ACM International Conference on Automated Software Engineering (**ASE 2016**), pp. 792--797, Singapore, September 2016

[3] M. Masudur Rahman and C.K. Roy and R.G. Kula, ***"Predicting Usefulness of Code Review Comments using Textual Features and Developer Experience"***, In Proceeding of The 14th International Conference on Mining Software Repositories (**MSR 2017**), pp. 215--226, Buenos Aires, Argentina, May, 2017

[4] M. Masudur Rahman and C.K. Roy, ***"Impact of Continuous Integration on Code Reviews"***, In Proceeding of The 14th International Conference on Mining Software Repositories (**MSR 2017**), pp. 499--502, Buenos Aires, Argentina, May, 2017

Thank You!! Questions?

Email: chanchal.roy@usask.ca **or** masud.rahman@usask.ca

University of Saskatchewan NSERC CRSNG 大阪大学 OSAKA UNIVERSITY VENDASTA