论文汇报

《Using code reviews to automatically configure static analysis tools》	2022	ESE
《RepresentThemAll: A Universal Learning Representation of Bug Reports》	2023	ICSE

Fiorella Zampetti[1] · Saghan Mudbhari[2] · Venera Arnaoudova[2] · Massimiliano Di Penta[1] · Sebastiano Panichella[3] · Giuliano Antoniol[4]

Sen Fang
Macau University of Science and Technology
Macau, China
fangsen1996@gmail.com

Tao Zhang *
Macau University of Science and Technology
Macau, China
tazhang@must.edu.mo

Youshuai Tan
Macau University of Science and Technology
Macau, China
tanyoushuai123@gmail.com

He Jiang
Dalian University of Technology
Dalian, China
jianghe@dlut.edu.cn

Xin Xia
Huawei
Hangzhou, China
xin.xia@acm.org

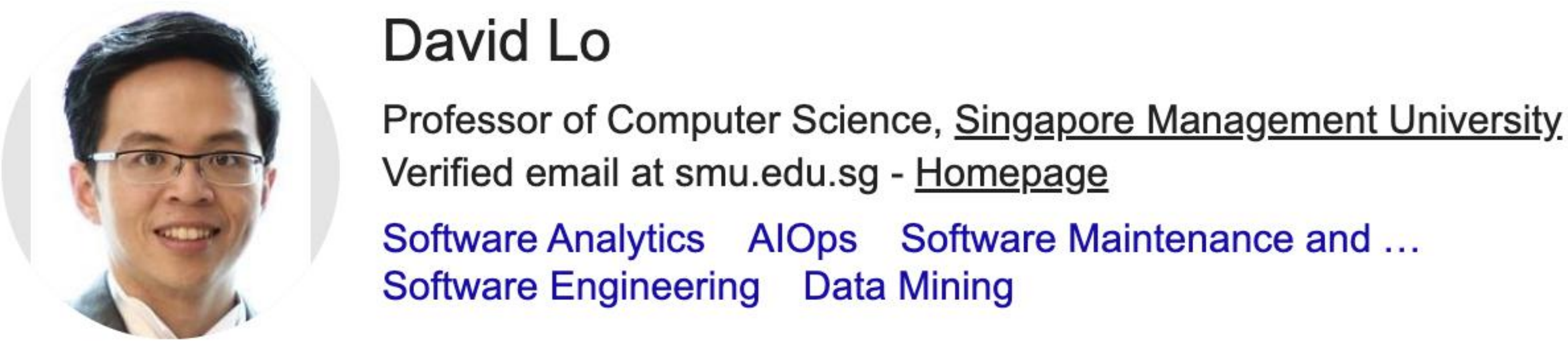
Xiaobing Sun
Yangzhou University
Yangzhou, China
xbsun@yzu.edu.cn

陈冰婷

2023.2.27

知名研究员

David Lo



David Lo

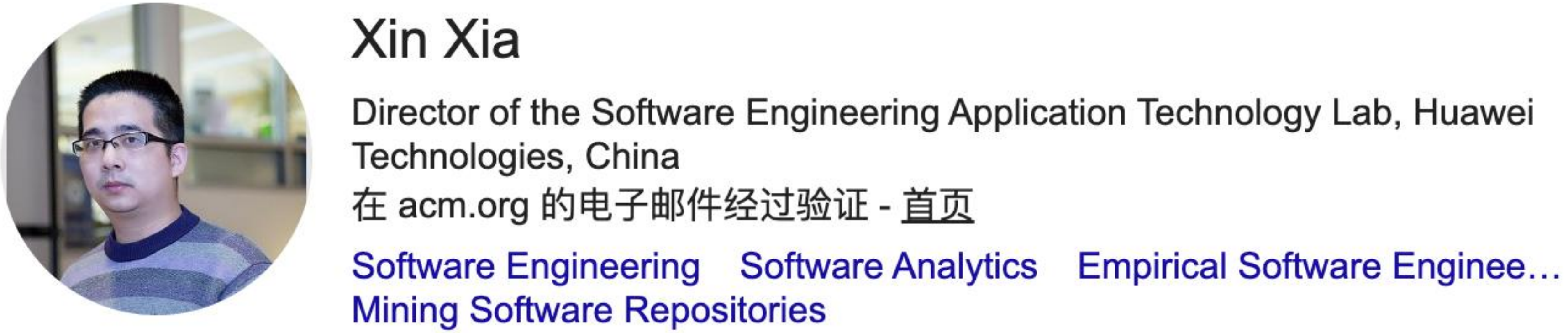
Professor of Computer Science, Singapore Management University

Verified email at smu.edu.sg - Homepage

Software Analytics AIOps Software Maintenance and …

Software Engineering Data Mining

夏鑫



Xin Xia

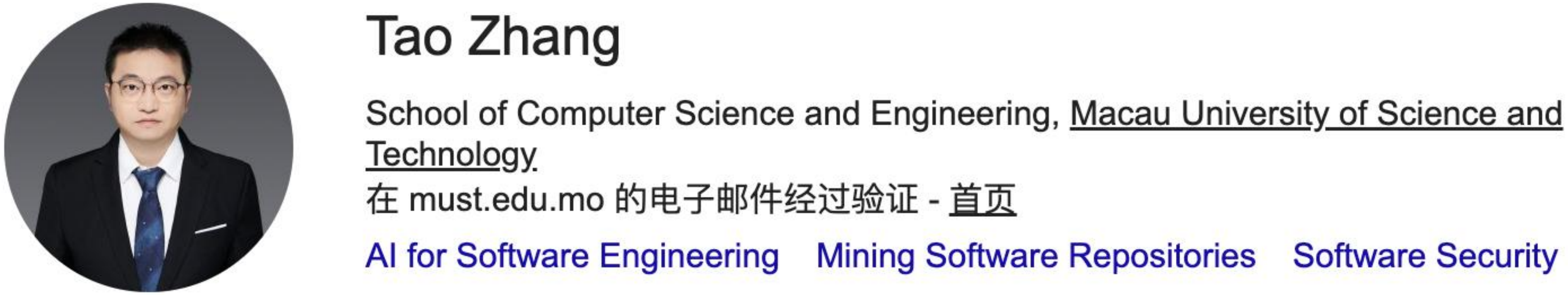
Director of the Software Engineering Application Technology Lab, Huawei Technologies, China

在 acm.org 的电子邮件经过验证 - 首页

Software Engineering Software Analytics Empirical Software Enginee…

Mining Software Repositories

张涛



Tao Zhang

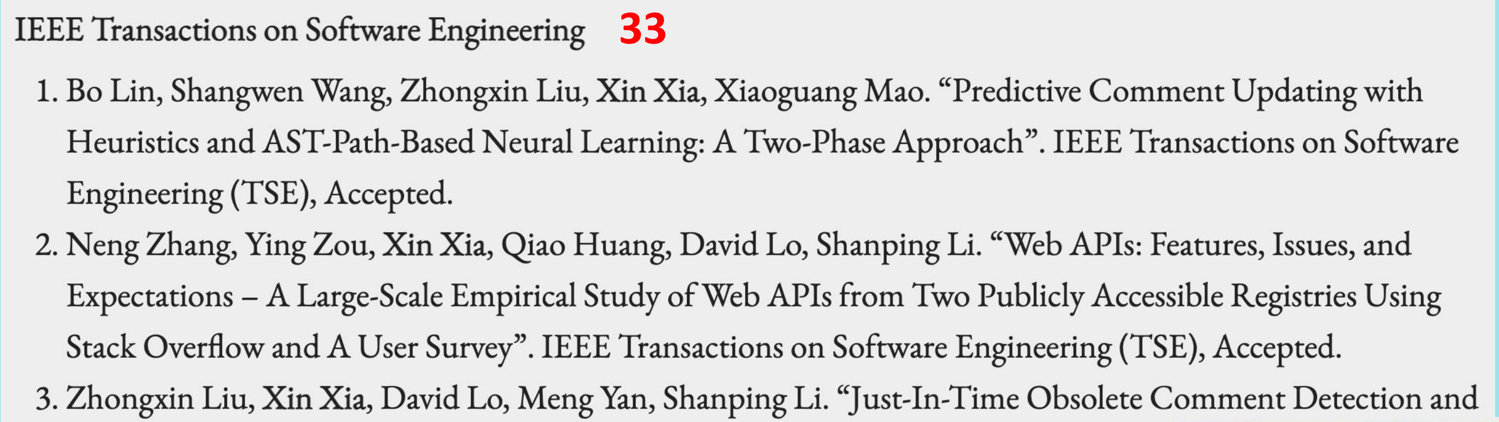
School of Computer Science and Engineering, Macau University of Science and Technology

在 must.edu.mo 的电子邮件经过验证 - 首页

AI for Software Engineering Mining Software Repositories Software Security

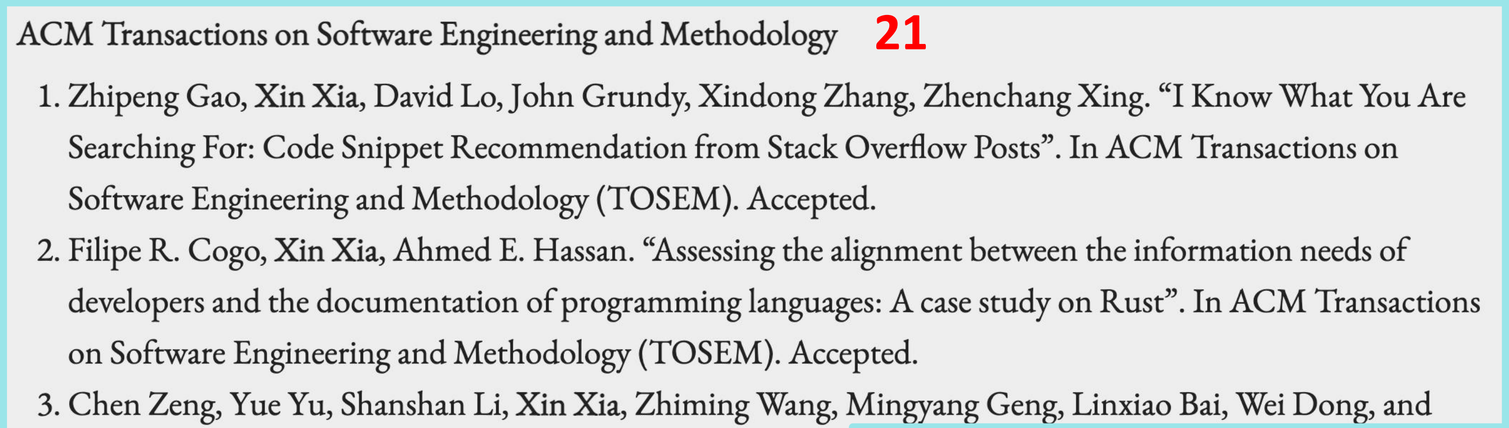
知名研究员

https://xin-xia.github.io/



IEEE Transactions on Software Engineering **33**

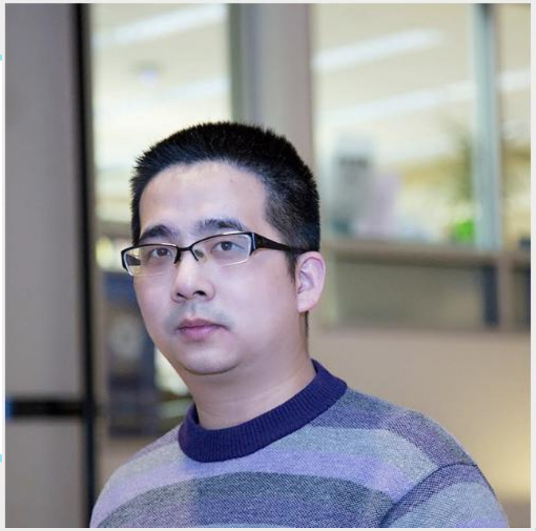
1. Bo Lin, Shangwen Wang, Zhongxin Liu, **Xin Xia**, Xiaoguang Mao. “Predictive Comment Updating with Heuristics and AST-Path-Based Neural Learning: A Two-Phase Approach”. IEEE Transactions on Software Engineering (TSE), Accepted.
2. Neng Zhang, Ying Zou, **Xin Xia**, Qiao Huang, David Lo, Shanping Li. “Web APIs: Features, Issues, and Expectations – A Large-Scale Empirical Study of Web APIs from Two Publicly Accessible Registries Using Stack Overflow and A User Survey”. IEEE Transactions on Software Engineering (TSE), Accepted.
3. Zhongxin Liu, **Xin Xia**, David Lo, Meng Yan, Shanping Li. “Just-In-Time Obsolete Comment Detection and Update”. IEEE Transactions on Software Engineering (TSE), Accepted.

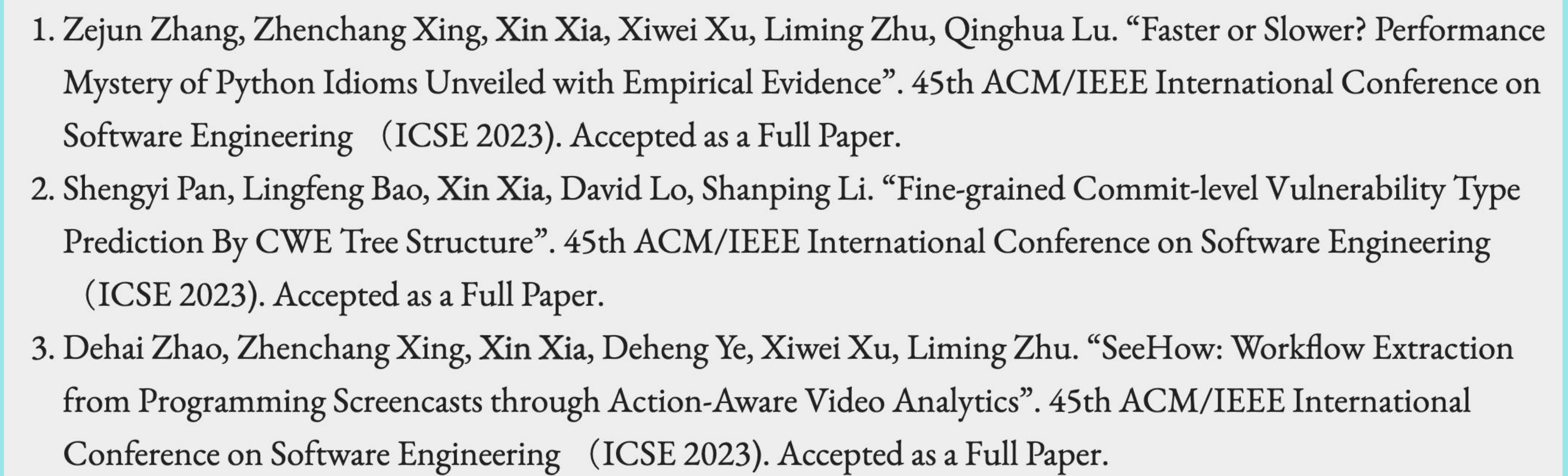


ACM Transactions on Software Engineering and Methodology **21**

1. Zhipeng Gao, **Xin Xia**, David Lo, John Grundy, Xindong Zhang, Zhenchang Xing. “I Know What You Are Searching For: Code Snippet Recommendation from Stack Overflow Posts”. In ACM Transactions on Software Engineering and Methodology (TOSEM). Accepted.
2. Filipe R. Cogo, **Xin Xia**, Ahmed E. Hassan. “Assessing the alignment between the information needs of developers and the documentation of programming languages: A case study on Rust”. In ACM Transactions on Software Engineering and Methodology (TOSEM). Accepted.
3. Chen Zeng, Yue Yu, Shanshan Li, **Xin Xia**, Zhiming Wang, Mingyang Geng, Linxiao Bai, Wei Dong, and Xiangke Liao. “deGraphCS: Embedding Variable-based Flow Transactions on Software Engineering and Methodology (T

夏鑫





ICSE **17**

1. Zejun Zhang, Zhenchang Xing, **Xin Xia**, Xiwei Xu, Liming Zhu, Qinghua Lu. “Faster or Slower? Performance Mystery of Python Idioms Unveiled with Empirical Evidence”. 45th ACM/IEEE International Conference on Software Engineering （ICSE 2023). Accepted as a Full Paper.
2. Shengyi Pan, Lingfeng Bao, **Xin Xia**, David Lo, Shanping Li. “Fine-grained Commit-level Vulnerability Type Prediction By CWE Tree Structure”. 45th ACM/IEEE International Conference on Software Engineering （ICSE 2023). Accepted as a Full Paper.
3. Dehai Zhao, Zhenchang Xing, **Xin Xia**, Deheng Ye, Xiwei Xu, Liming Zhu. “SeeHow: Workflow Extraction from Programming Screencasts through Action-Aware Video Analytics”. 45th ACM/IEEE International Conference on Software Engineering （ICSE 2023). Accepted as a Full Paper.

知名研究员

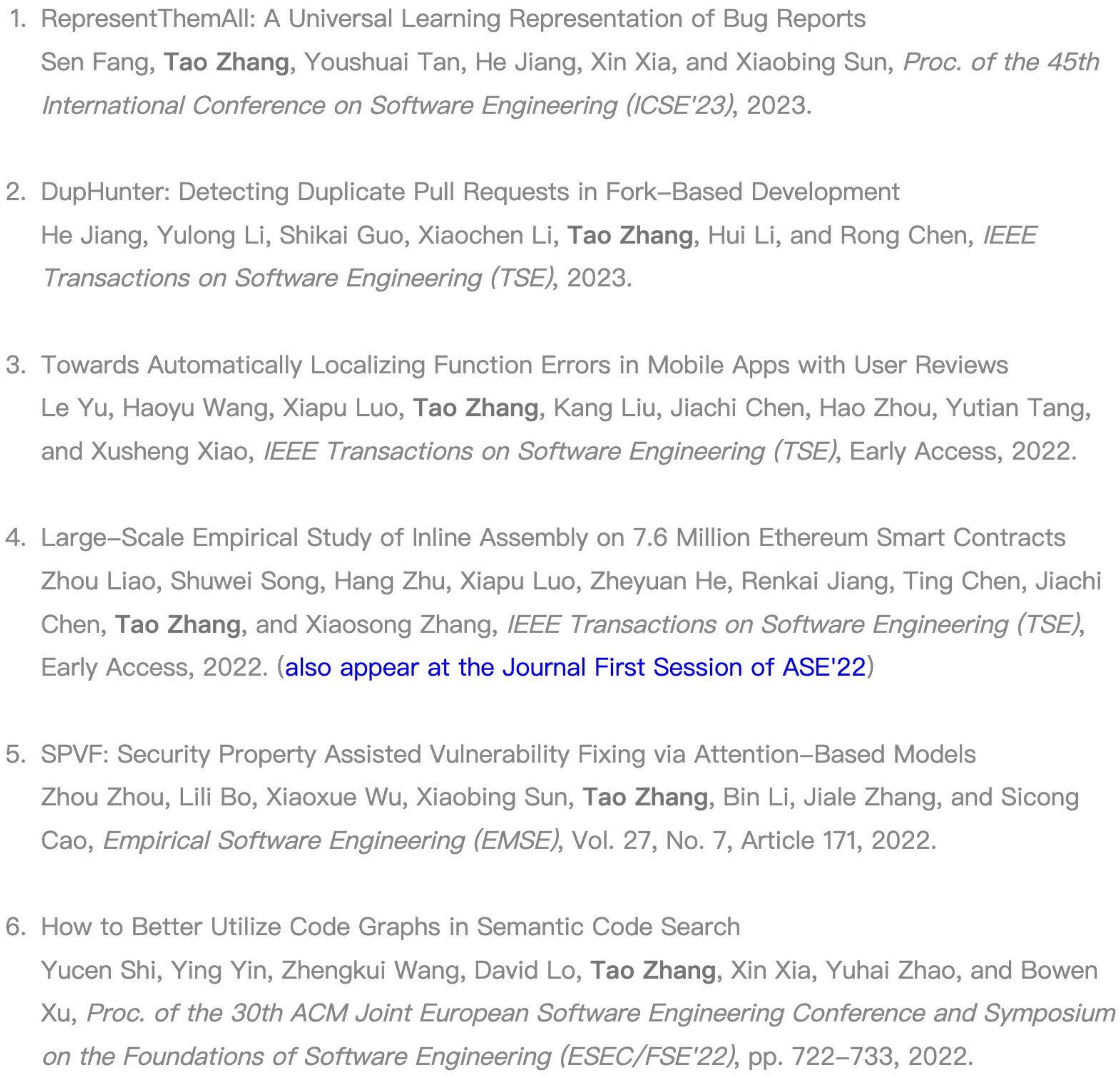
https://cszhangtao.github.io/

Tao Zhang (張濤)

Associate Professor

School of Computer Science and Engineering

Macau University of Science and Technology



1. RepresentThemAll: A Universal Learning Representation of Bug Reports
   Sen Fang, **Tao Zhang**, Youshuai Tan, He Jiang, Xin Xia, and Xiaobing Sun, *Proc. of the 45th International Conference on Software Engineering (ICSE'23)*, 2023.

2. DupHunter: Detecting Duplicate Pull Requests in Fork-Based Development
   He Jiang, Yulong Li, Shikai Guo, Xiaochen Li, **Tao Zhang**, Hui Li, and Rong Chen, *IEEE Transactions on Software Engineering (TSE)*, 2023.

3. Towards Automatically Localizing Function Errors in Mobile Apps with User Reviews
   Le Yu, Haoyu Wang, Xiapu Luo, **Tao Zhang**, Kang Liu, Jiachi Chen, Hao Zhou, Yutian Tang, and Xusheng Xiao, *IEEE Transactions on Software Engineering (TSE)*, Early Access, 2022.

4. Large-Scale Empirical Study of Inline Assembly on 7.6 Million Ethereum Smart Contracts
   Zhou Liao, Shuwei Song, Hang Zhu, Xiapu Luo, Zheyuan He, Renkai Jiang, Ting Chen, Jiachi Chen, **Tao Zhang**, and Xiaosong Zhang, *IEEE Transactions on Software Engineering (TSE)*, Early Access, 2022. (also appear at the Journal First Session of ASE'22)

5. SPVF: Security Property Assisted Vulnerability Fixing via Attention-Based Models
   Zhou Zhou, Lili Bo, Xiaoxue Wu, Xiaobing Sun, **Tao Zhang**, Bin Li, Jiale Zhang, and Sicong Cao, *Empirical Software Engineering (EMSE)*, Vol. 27, No. 7, Article 171, 2022.

6. How to Better Utilize Code Graphs in Semantic Code Search
   Yucen Shi, Ying Yin, Zhengkui Wang, David Lo, **Tao Zhang**, Xin Xia, Yuhai Zhao, and Bowen Xu, *Proc. of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE'22)*, pp. 722–733, 2022.

Using code reviews to automatically configure static analysis tools

2022 ESE

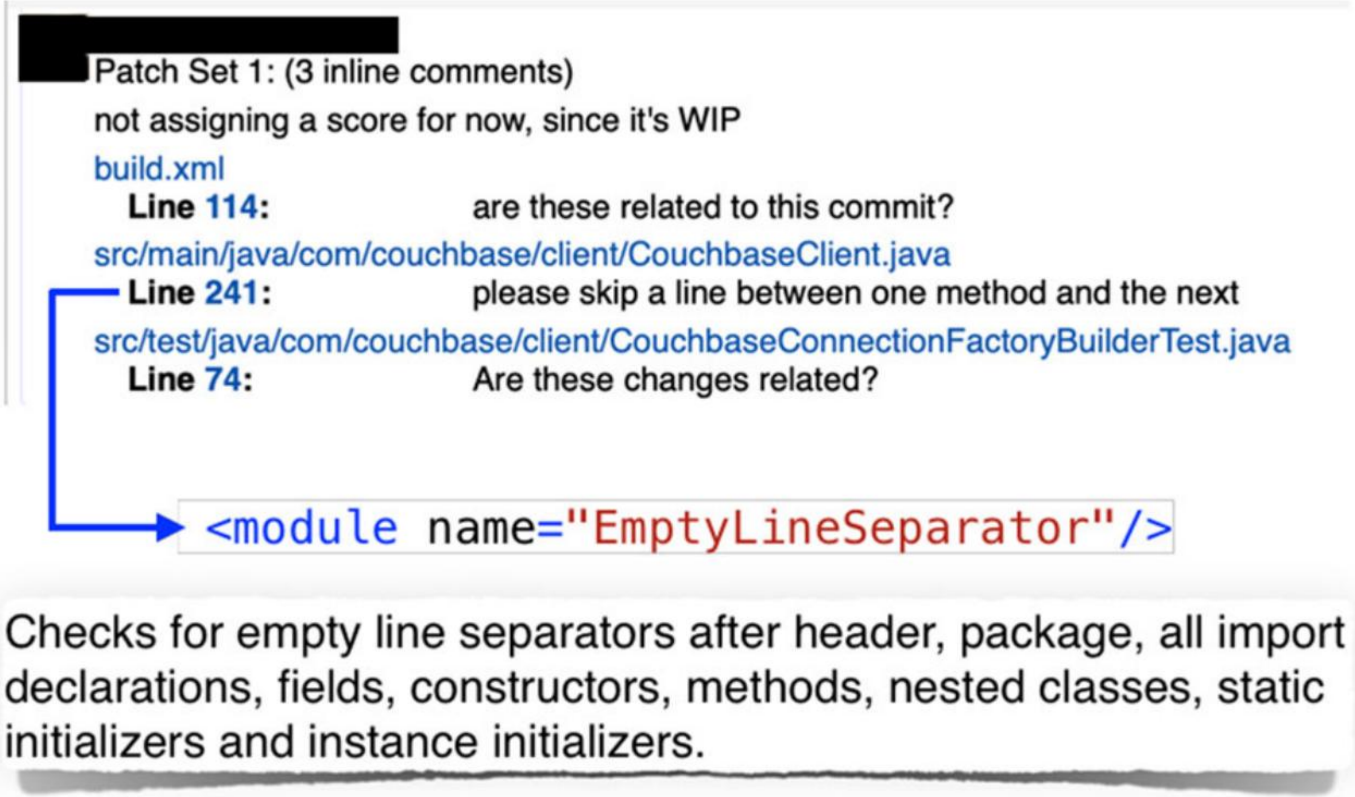
SCAT

Static Code Analysis Tools (SCATs) are, for software developers, a precious complement to more expensive quality assurance techniques, including testing and code inspection.

naming identifiers

using braces

using indentation



[redacted] Patch Set 1: (3 inline comments)

not assigning a score for now, since it's WIP

build.xml
Line 114: are these related to this commit?

src/main/java/com/couchbase/client/CouchbaseClient.java
Line 241: please skip a line between one method and the next

src/test/java/com/couchbase/client/CouchbaseConnectionFactoryBuilderTest.java
Line 74: Are these changes related?

```
<module name="EmptyLineSeparator"/>
```

Checks for empty line separators after header, package, all import declarations, fields, constructors, methods, nested classes, static initializers and instance initializers.

Fig. 2 Example of inline code review mapped onto a CheckStyle check

Using code reviews to automatically configure static analysis tools

2022 ESE

Overview

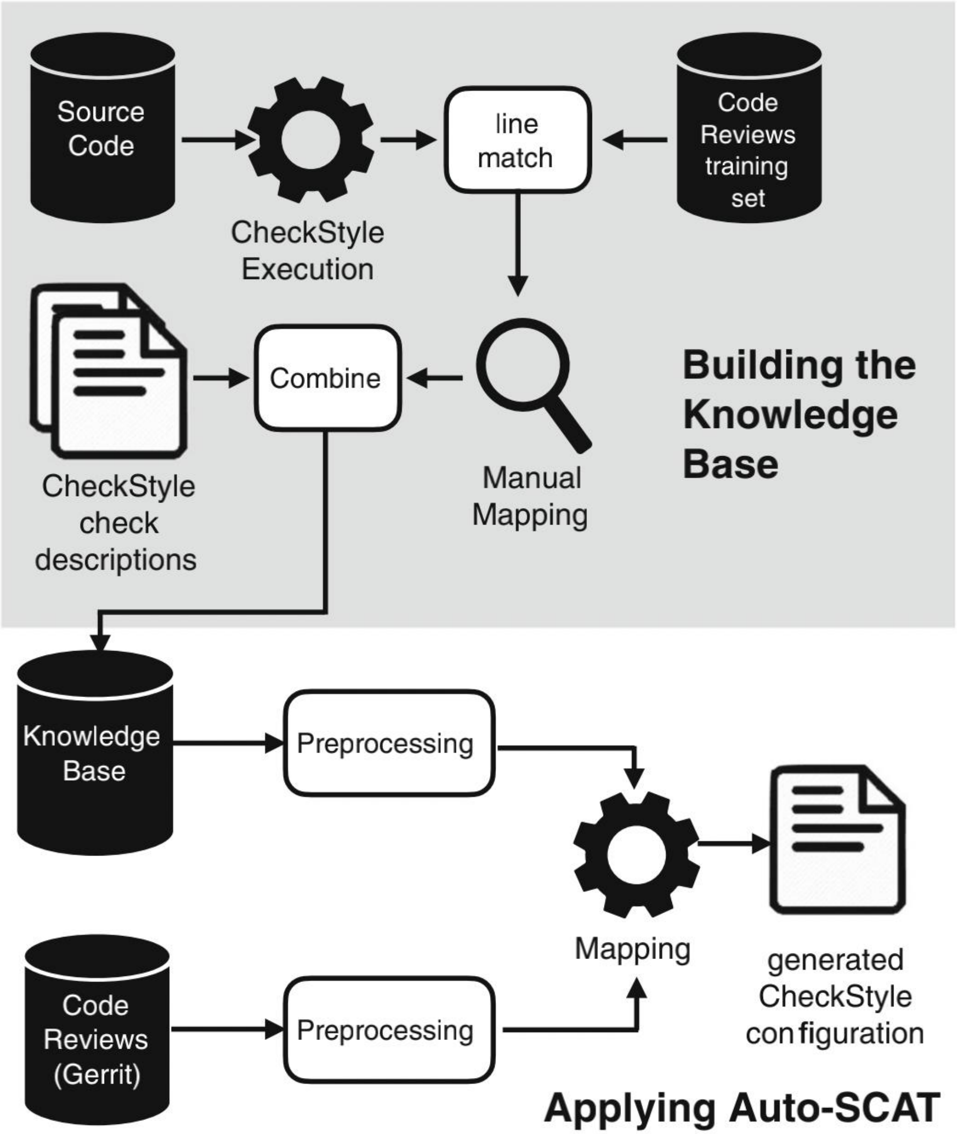


Fig. 1 Overview of Auto-SCAT

inline comments posted by developers

history

Using code reviews to automatically configure static analysis tools

2022 ESE

RQ1: *To what extent code review comments occurring on lines where CheckStyle fails are related to quality issues that can be detected by that tool?*

40% of the code review comments left on source code lines where CheckStyle raises a warning in our sample, and are related to a broad variety (45%) of CheckStyle checks

RQ2: *How accurate is Auto-SCAT?*

Auto-SCAT leverages code review comments to configure CheckStyle checks with micro- and macro-averaged precision of 75% and 32%.

RQ3: *How accurate is Auto-SCAT compared to a baseline?*

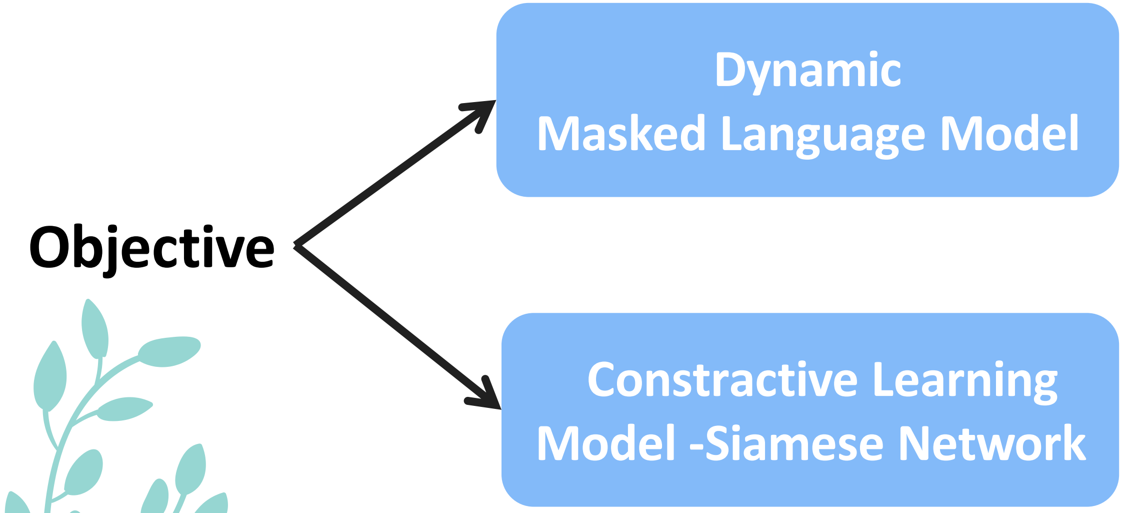
Auto-SCAT outperforms the simple use of a default CheckStyle configuration.

RepresentThemAll: A Universal Learning Representation of Bug Reports

2023 ICSE

To what extent, can we use a universal bug report representation to handle multiple ASMT-ABR?

Automated Software Maintenance Tasks associated with bug reports ***(ASMT-ABR for short)***



Objective

Dynamic Masked Language Model

Constractive Learning Model -Siamese Network

Despite the aforementioned studies having obtained early success, they also face an important challenge: existing approaches are designed for a specific ASMT-ABR and they cannot serve for multiple downstream tasks in software maintenance.

· **Complexity** · **Cost** · **Compatibility**

RepresentThemAll: A Universal Learning Representation of Bug Reports

2023 ICSE

RTA - Framework

- **Severity，Priority，Duplicate**
- **Bug Report Summary**

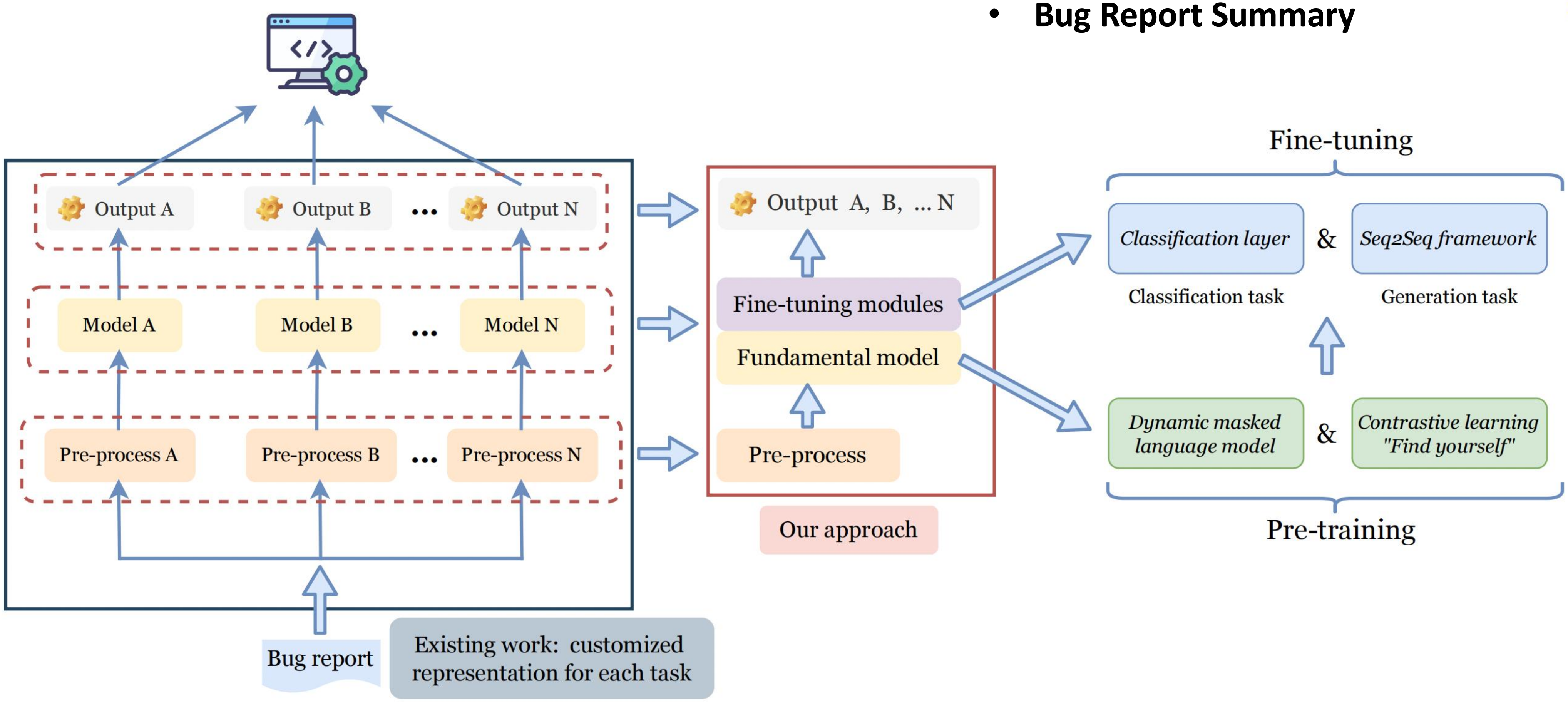


Fig. 1. The comparison of automating the process of software maintenance between existing approaches (left part) and our approach (right part).

RepresentThemAll: A Universal Learning Representation of Bug Reports

2023 ICSE

反思

- **Style of Writing** 包装
- **Summarize** 提纲挈领
- **github** 写的很规范
- 思维和表达

落脚点

- 增强提取语义信息,-通用的**br**语义抽取模型
- 用一个通用流程去解决**multi tasks**

However, training multiple models to automate multiple downstream tasks in software maintenance may face three issues: **complexity, cost, and compatibility,** due to the customization, disparity, and uniqueness of these automated approaches.

RepresentThemAll: A Universal Learning Representation of Bug Reports

2023 ICSE

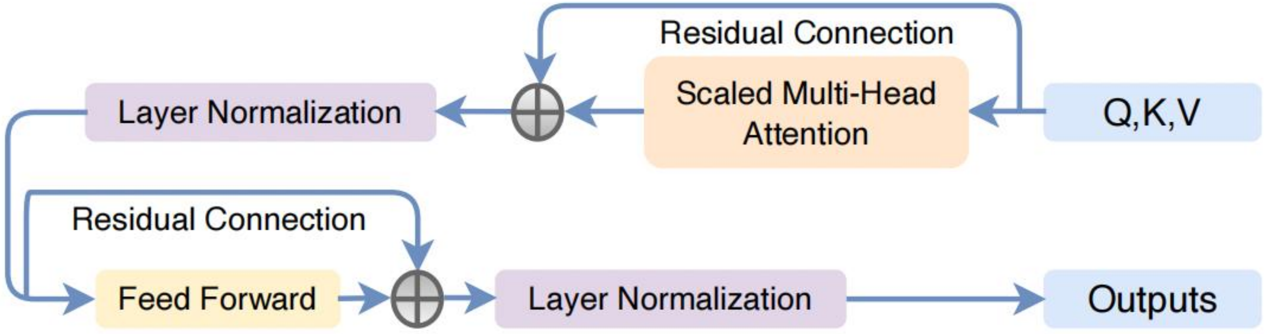


Fig. 3. Transformer encoder layer.

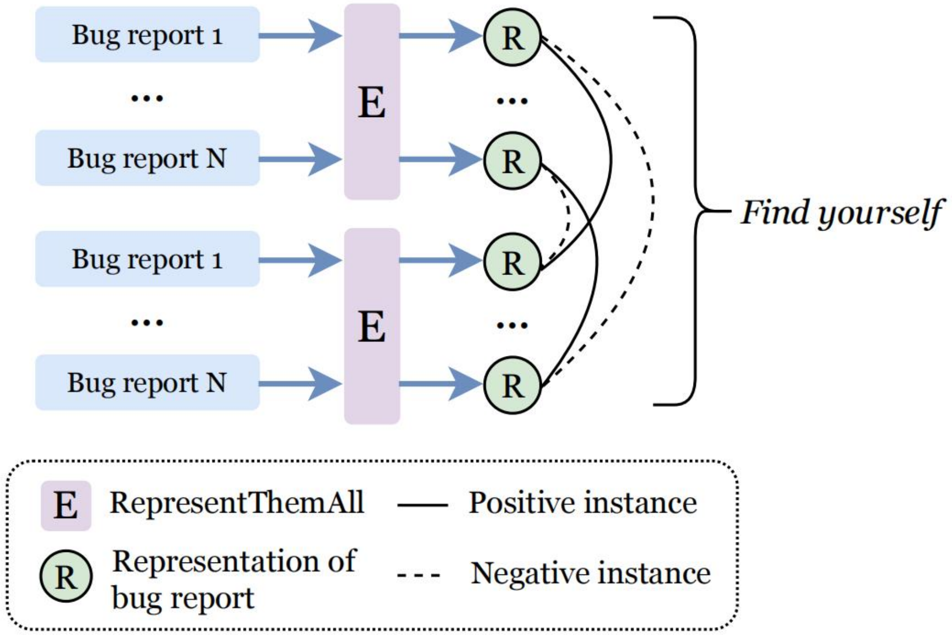


Fig. 5. Siamese `RepresentThemAll` network with shared parameters predicts the input bug report itself from in-batch negative bug reports. N is the batch size.

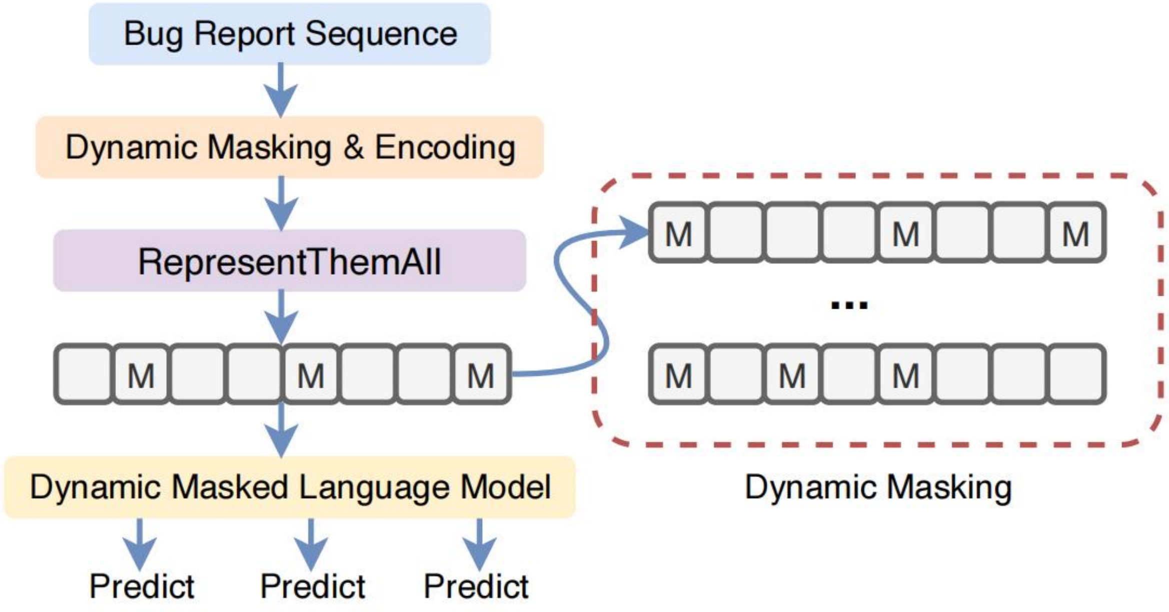
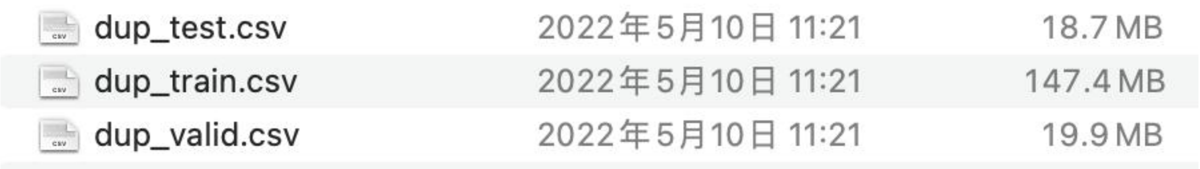


Fig. 4. `RepresentThemAll` predicts masked tokens according to their context. “M” denotes the masked token. Due to the dynamic masking, masked tokens in the bug report sequence are different at each epoch.

RepresentThemAll: A Universal Learning Representation of Bug Reports

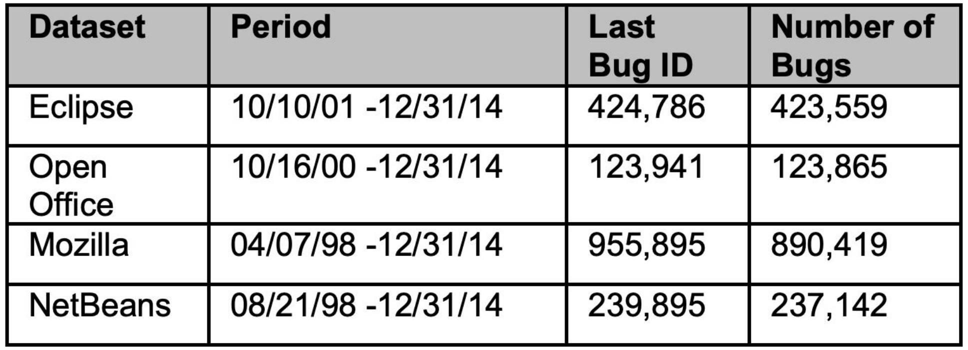
2023 ICSE

Specifically, we have collected 275,639 bug reports from Feb. 2000 to Sep. 2020.



dup_test.csv	2022年5月10日 11:21	18.7 MB
dup_train.csv	2022年5月10日 11:21	147.4 MB
dup_valid.csv	2022年5月10日 11:21	19.9 MB

Table 1. Datasets and number of bugs



Dataset	Period	Last Bug ID	Number of Bugs
Eclipse	10/10/01 -12/31/14	424,786	423,559
Open Office	10/16/00 -12/31/14	123,941	123,865
Mozilla	04/07/98 -12/31/14	955,895	890,419
NetBeans	08/21/98 -12/31/14	239,895	237,142

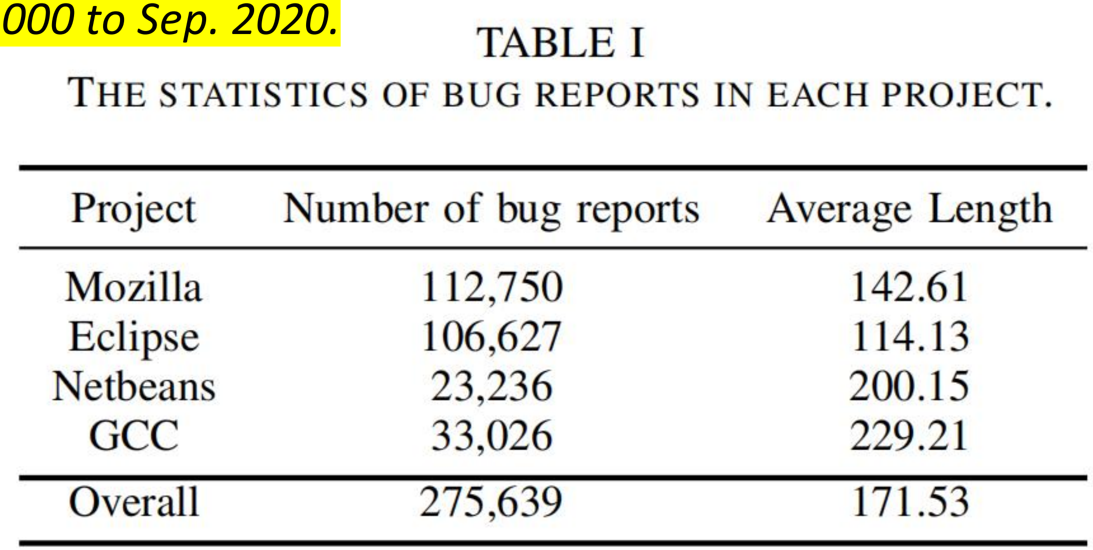


TABLE I

THE STATISTICS OF BUG REPORTS IN EACH PROJECT.

Project	Number of bug reports	Average Length
Mozilla	112,750	142.61
Eclipse	106,627	114.13
Netbeans	23,236	200.15
GCC	33,026	229.21
Overall	275,639	171.53

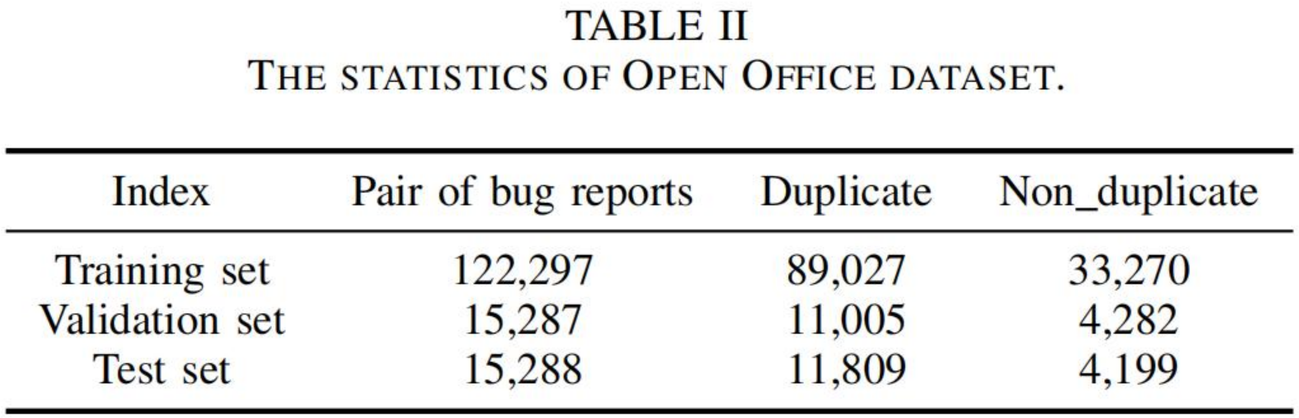


TABLE II

THE STATISTICS OF OPEN OFFICE DATASET.

Index	Pair of bug reports	Duplicate	Non_duplicate
Training set	122,297	89,027	33,270
Validation set	15,287	11,005	4,282
Test set	15,288	11,809	4,199

dup是分类！数据量也不多！

RepresentThemAll: A Universal Learning Representation of Bug Reports

2023 ICSE

RQs

- *RQ1： Retrieval Effectiveness*
- *RQ2：Effectiveness Comparison of Different PLMs*
- *RQ3：Ablation Study*

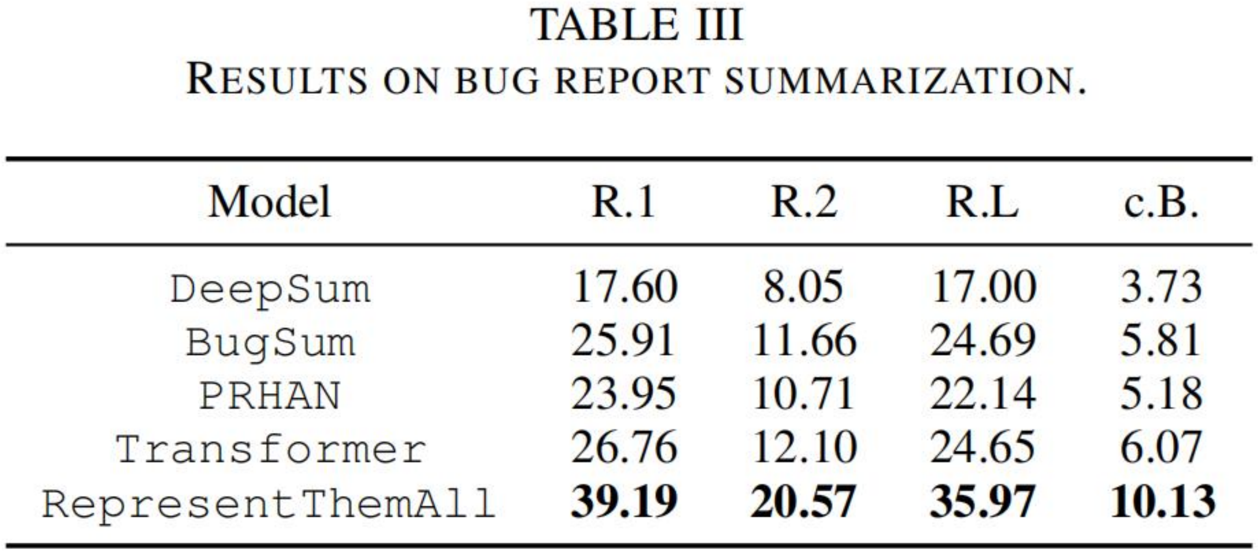


TABLE III
RESULTS ON BUG REPORT SUMMARIZATION.

Model	R.1	R.2	R.L	c.B.
DeepSum	17.60	8.05	17.00	3.73
BugSum	25.91	11.66	24.69	5.81
PRHAN	23.95	10.71	22.14	5.18
Transformer	26.76	12.10	24.65	6.07
RepresentThemAll	**39.19**	**20.57**	**35.97**	**10.13**

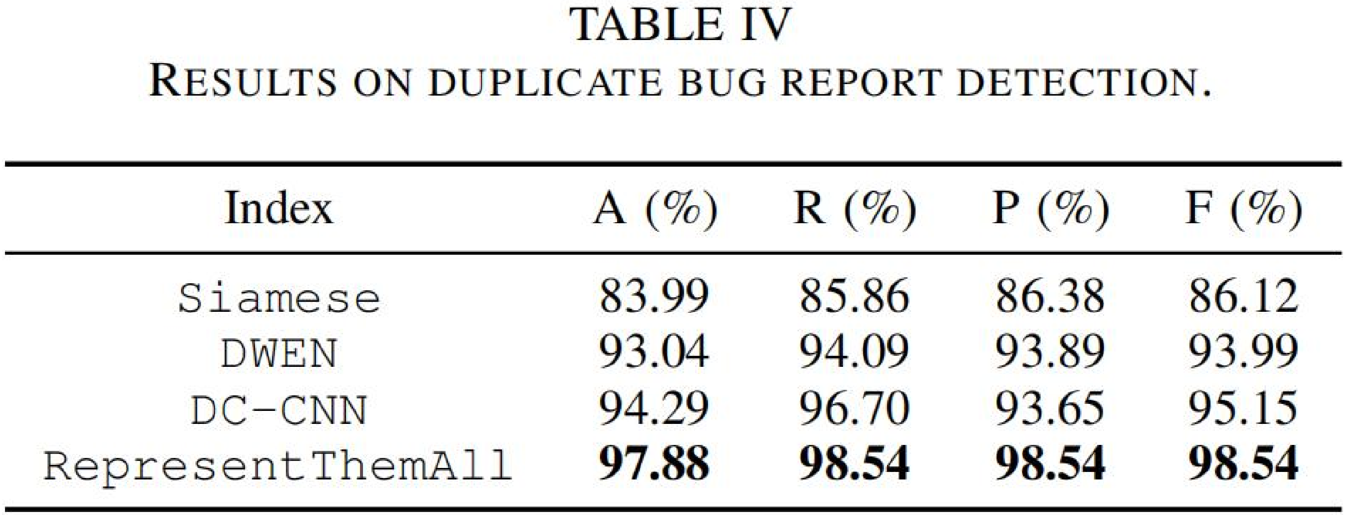
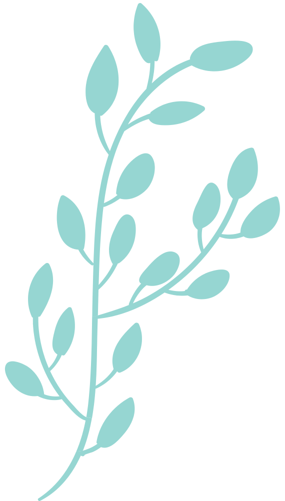


TABLE IV
RESULTS ON DUPLICATE BUG REPORT DETECTION.

Index	A (%)	R (%)	P (%)	F (%)
Siamese	83.99	85.86	86.38	86.12
DWEN	93.04	94.09	93.89	93.99
DC-CNN	94.29	96.70	93.65	95.15
RepresentThemAll	**97.88**	**98.54**	**98.54**	**98.54**

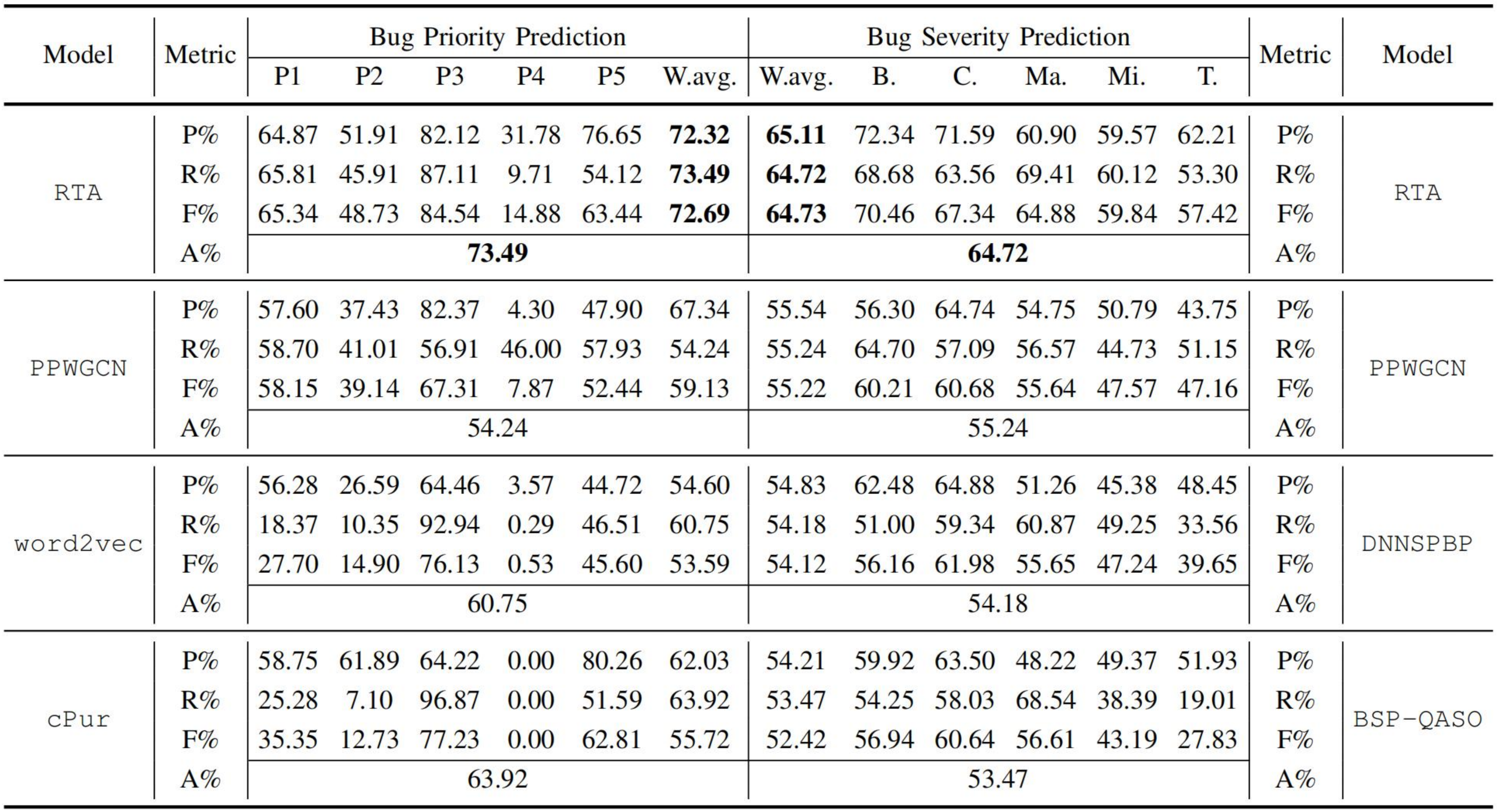


RepresentThemAll: A Universal Learning Representation of Bug Reports

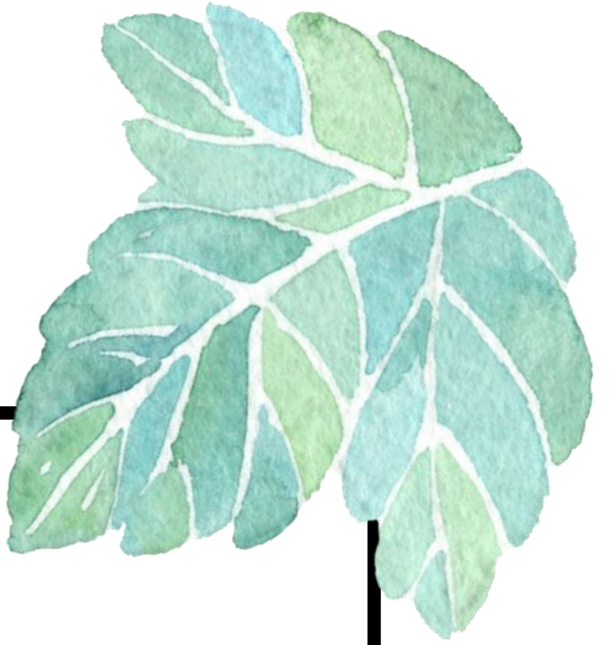
2023 ICSE

TABLE V

Results on bug priority & severity prediction. We denote "RepresentThemAll" as "RTA".



Model	Metric	Bug Priority Prediction						Bug Severity Prediction						Metric	Model
		P1	P2	P3	P4	P5	W.avg.	W.avg.	B.	C.	Ma.	Mi.	T.		
RTA	P%	64.87	51.91	82.12	31.78	76.65	**72.32**	**65.11**	72.34	71.59	60.90	59.57	62.21	P%	RTA
	R%	65.81	45.91	87.11	9.71	54.12	**73.49**	**64.72**	68.68	63.56	69.41	60.12	53.30	R%	
	F%	65.34	48.73	84.54	14.88	63.44	**72.69**	**64.73**	70.46	67.34	64.88	59.84	57.42	F%	
	A%	**73.49**						**64.72**						A%	
PPWGCN	P%	57.60	37.43	82.37	4.30	47.90	67.34	55.54	56.30	64.74	54.75	50.79	43.75	P%	PPWGCN
	R%	58.70	41.01	56.91	46.00	57.93	54.24	55.24	64.70	57.09	56.57	44.73	51.15	R%	
	F%	58.15	39.14	67.31	7.87	52.44	59.13	55.22	60.21	60.68	55.64	47.57	47.16	F%	
	A%	54.24						55.24						A%	
word2vec	P%	56.28	26.59	64.46	3.57	44.72	54.60	54.83	62.48	64.88	51.26	45.38	48.45	P%	DNNSPBP
	R%	18.37	10.35	92.94	0.29	46.51	60.75	54.18	51.00	59.34	60.87	49.25	33.56	R%	
	F%	27.70	14.90	76.13	0.53	45.60	53.59	54.12	56.16	61.98	55.65	47.24	39.65	F%	
	A%	60.75						54.18						A%	
cPur	P%	58.75	61.89	64.22	0.00	80.26	62.03	54.21	59.92	63.50	48.22	49.37	51.93	P%	BSP-QASO
	R%	25.28	7.10	96.87	0.00	51.59	63.92	53.47	54.25	58.03	68.54	38.39	19.01	R%	
	F%	35.35	12.73	77.23	0.00	62.81	55.72	52.42	56.94	60.64	56.61	43.19	27.83	F%	
	A%	63.92						53.47						A%	



谢谢聆听

演讲完毕