Duplicate bug report detection through machine learning techniques

Irving Muller Rodrigues

irving.muller-rodrigues@polymtl.ca

École Polytechnique de Montréal

Laboratoire DORSAL

Duplicate bug reports

- Duplicate bug reports describe the same bug
- Very common in Bug Tracking Systems (BTSs)
  - For instance: 12% of all reports in [1]
- Undetected duplicate bug reports
  - Waste of developer time
- Manually filtered by triage team
  - Beyond team capacity
- **Machine learning technique to help triage team**

Duplicate bug reports

- Duplicate bug reports describe the same bug
- Very common in Bug Tracking Systems (BTSs)
  - For instance: 12% of all reports in [1]
- Undetected duplicate bug reports
  - Waste of developer time
- Manually filtered by triage team
  - Beyond team capacity
- **Machine learning technique to help triage team**

Bug report deduplication

- Automatic detection of duplicate bug reports
- Challenges
  - Noise
  - Lack of information
  - Techniques do not achieve satisfactory results
- Treat as a ranking problem
  - ML system returns a list with the k-most likely duplicate reports
  - Recall Rate@k: percentage of duplicate bug reports whose their correct candidates are within the top k positions in the list
- Time window
  - Search for reports submitted x days before the new report

Bug report deduplication

- Automatic detection of duplicate bug reports
- Challenges
  - Noise
  - Lack of information
  - Techniques do not achieve satisfactory results
- Treat as a ranking problem
  - ML system returns a list with the k-most likely duplicate reports
  - Recall Rate@k: percentage of duplicate bug reports whose their correct candidates are within the top k positions in the list
- Time window
  - Search for reports submitted x days before the new report

Bug report deduplication

- Automatic detection of duplicate bug reports
- Challenges
  - Noise
  - Lack of information
  - Techniques do not achieve satisfactory results
- Treat as a ranking problem
  - ML system returns a list with the k-most likely duplicate reports
  - Recall Rate@k: percentage of duplicate bug reports whose their correct candidates are within the top k positions in the list
- Time window
  - Search for reports submitted x days before the new report

Bug report deduplication

- Automatic detection of duplicate bug reports
- Challenges
  - Noise
  - Lack of information
  - Techniques do not achieve satisfactory results
- Treat as a ranking problem
  - ML system returns a list with the k-most likely duplicate reports
  - Recall Rate@k: percentage of duplicate bug reports whose their correct candidates are within the top k positions in the list
- Time window
  - Search for reports submitted x days before the new report

Our Research

- Deep learning
  - State-of-art in many problems in NLP
  - Combine easily different data types
- Address the problem using distinct data types
  - Categorical data and Textual data
  - Stack trace

Our Research

- Deep learning
  - State-of-art in many problems in NLP
  - Combine easily different data types
- Address the problem using distinct data types
  - Categorical data and Textual data
  - Stack trace

Our Research

- Deep learning
  - State-of-art in many problems in NLP
  - Combine easily different data types
- Address the problem using distinct data types
  - Categorical data and Textual data
  - Stack trace

Our Work - Categorical and Textual data

Bug 247988 - Deleting OPEN project takes very long

Status: CLOSED WORKSFORME
Reported: 2008-09-19 11:35 EDT by Alex Bernstein — ECA
Modified: 2016-05-05 10:29 EDT (History)
Alias: None
CC List: 1 user (show)
Product: z_Archived
See Also:
Component: TPTP (show other bugs)
Version: unspecified
Hardware: PC Windows XP
Importance: P1 normal (vote)
Target Milestone: ---
Assignee: Bozier jerome — ECA

Alex Bernstein — ECA 2008-09-19 11:35:53 EDT Description

```
Build ID: 4.5.1

Steps To Reproduce:
1. Have a project with one or more large datapool files (6Mb);
2. Make sure it is the only project in the workspace (optional?)
3. Right click on the project in Test Navigator and select "Delete";
4. Do not check the "Also delete..." check box (optional);
5. debugging reveals that Datapool Proxy nodes are recreated, causing Datapools to be
loaded into memory ;


More information:
```

Our Work - Categorical and Textual data

Bug 247988 - Deleting OPEN project takes very long

Status: CLOSED WORKSFORME
Reported: 2008-09-19 11:35 EDT by Alex Bernstein
Modified: 2016-05-05 10:29 EDT (History)
Alias: None
CC List: 1 user (show)
Product: z_Archived
See Also:
Component: TPTP (show other bugs)
Version: unspecified
Hardware: PC Windows XP
Importance: P1 normal (vote)
Target Milestone: ---
Assignee: Bozier jerome

Alex Bernstein 2008-09-19 11:35:53 EDT Description

```
Build ID: 4.5.1

Steps To Reproduce:
1. Have a project with one or more large datapool files (6Mb);
2. Make sure it is the only project in the workspace (optional?)
3. Right click on the project in Test Navigator and select "Delete";
4. Do not check the "Also delete..." check box (optional);
5. debugging reveals that Datapool Proxy nodes are recreated, causing Datapools to be
loaded into memory ;


More information:
```

Our Work - Categorical and Textual data

Bug 247988 - Deleting OPEN project takes very long

Status: CLOSED WORKSFORME
Reported: 2008-09-19 11:35 EDT by Alex Bernstein
Modified: 2016-05-05 10:29 EDT (History)
Alias: None
CC List: 1 user (show)
Product: z_Archived
See Also:
Component: TPTP (show other bugs)
Version: unspecified
Hardware: PC Windows XP
Importance: P1 normal (vote)
Target Milestone: ---
Assignee: Bozier jerome

Alex Bernstein 2008-09-19 11:35:53 EDT Description

```
Build ID: 4.5.1

Steps To Reproduce:
1. Have a project with one or more large datapool files (6Mb);
2. Make sure it is the only project in the workspace (optional?)
3. Right click on the project in Test Navigator and select "Delete";
4. Do not check the "Also delete..." check box (optional);
5. debugging reveals that Datapool Proxy nodes are recreated, causing Datapools to be
loaded into memory ;


More information:
```

Our Work - Categorical and Textual data

Bug 247988 - Deleting OPEN project takes very long

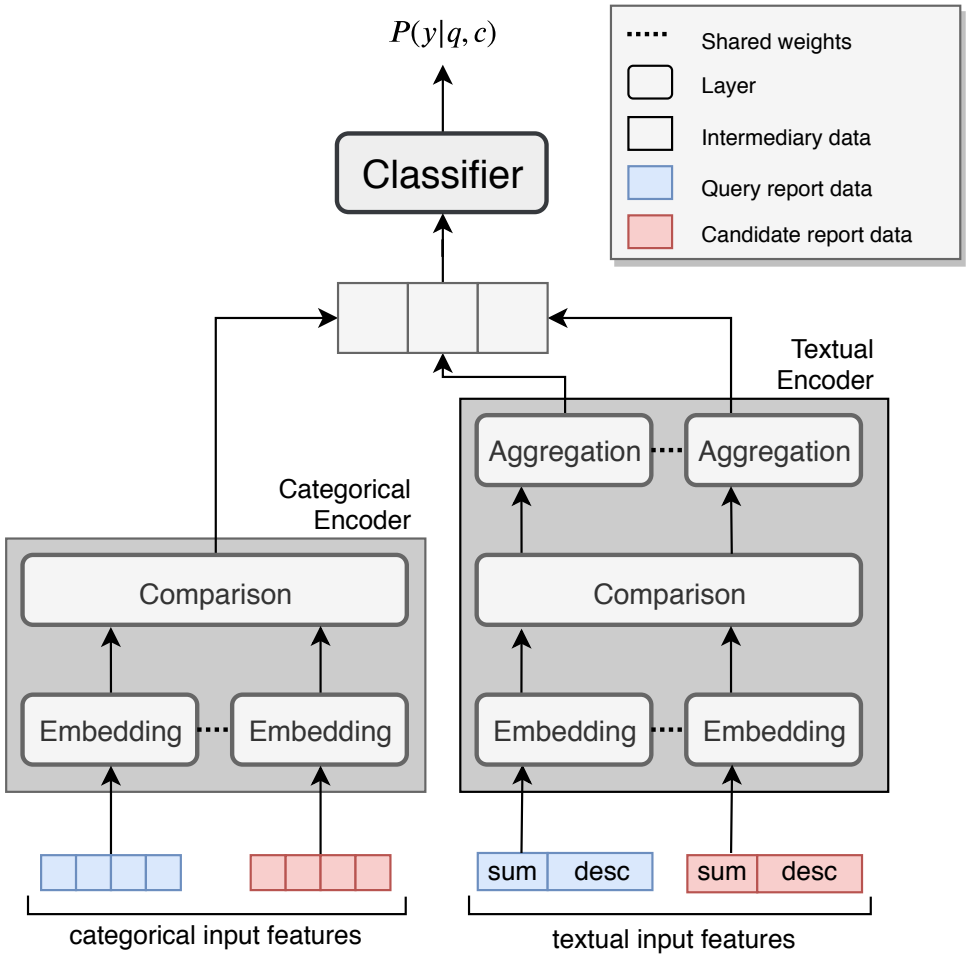
Status: CLOSED WORKSFORME
Alias: None
Product: z_Archived
Component: TPTP (show other bugs)
Version: unspecified
Hardware: PC Windows XP
Importance: P1 normal (vote)
Target Milestone: ---
Assignee: Bozier jerome ECA

Reported: 2008-09-19 11:35 EDT by Alex Bernstein ECA
Modified: 2016-05-05 10:29 EDT (History)
CC List: 1 user (show)
See Also:

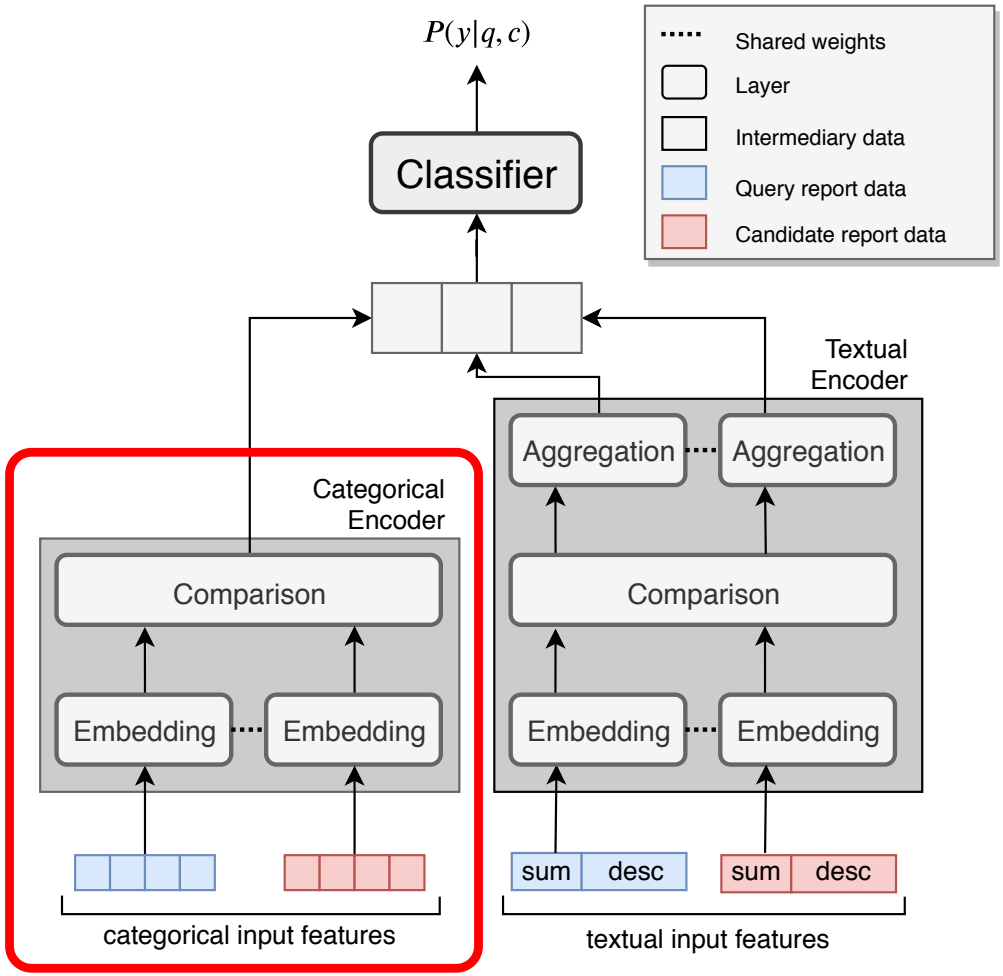
Alex Bernstein ECA 2008-09-19 11:35:53 EDT Description

```
Build ID: 4.5.1

Steps To Reproduce:
1. Have a project with one or more large datapool files (6Mb);
2. Make sure it is the only project in the workspace (optional?)
3. Right click on the project in Test Navigator and select "Delete";
4. Do not check the "Also delete..." check box (optional);
5. debugging reveals that Datapool Proxy nodes are recreated, causing Datapools to be
loaded into memory ;


More information:
```

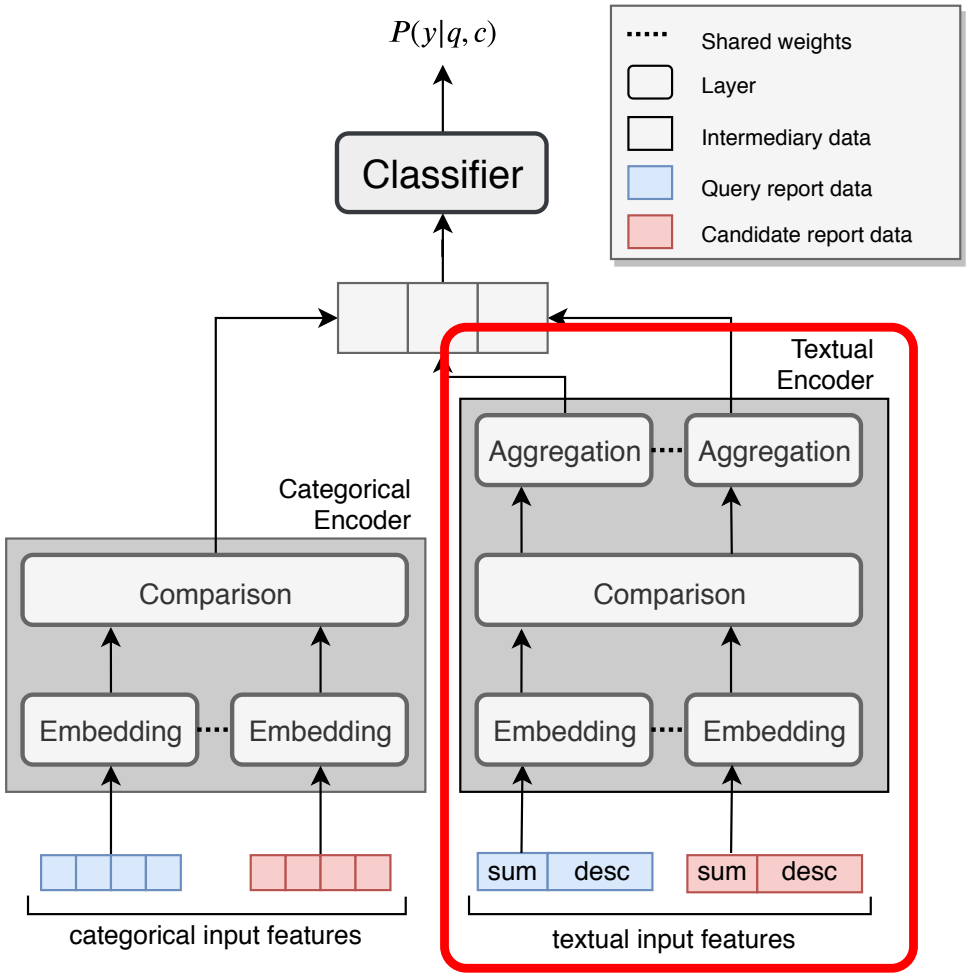
Our Work - Categorical and Textual data



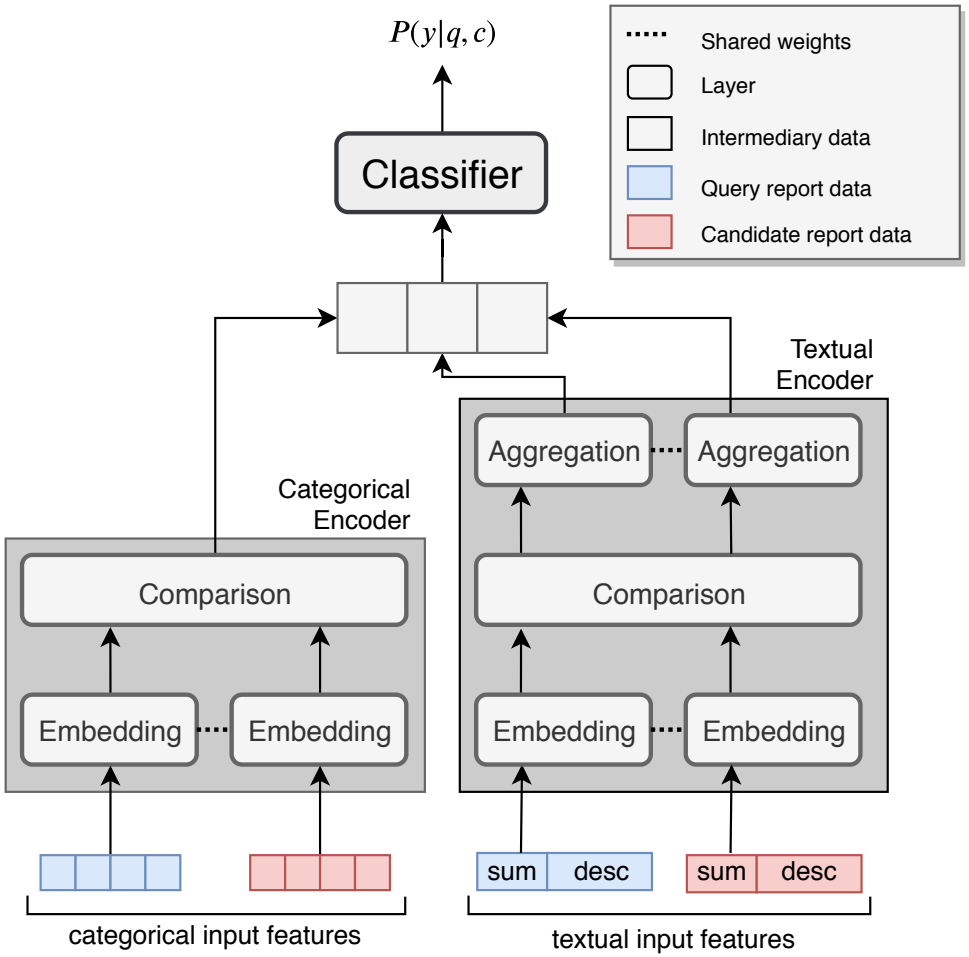
Our Work - Categorical and Textual data



Our Work - Categorical and Textual data



Our Work - Categorical and Textual data



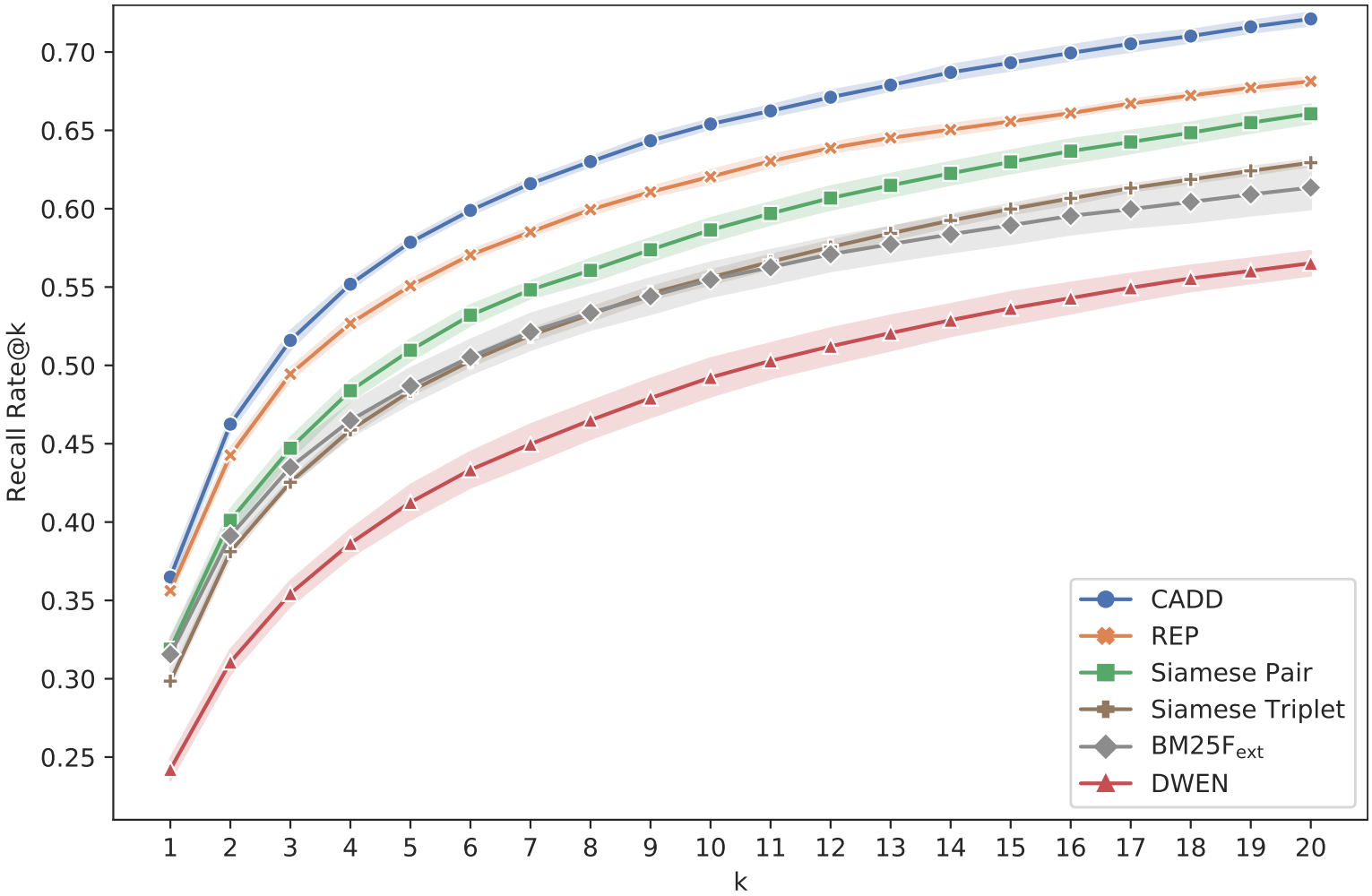
Experiments

- Four BTSs
  - Eclipse, Firefox, OpenOffice and Netbeans
- Our method (CADD) is compared to:
  - BM25F [2]
  - REP [2]
  - DWEN [3]
  - Siamese Pairs [4]
  - Siamese Triplets [4]

Experiments

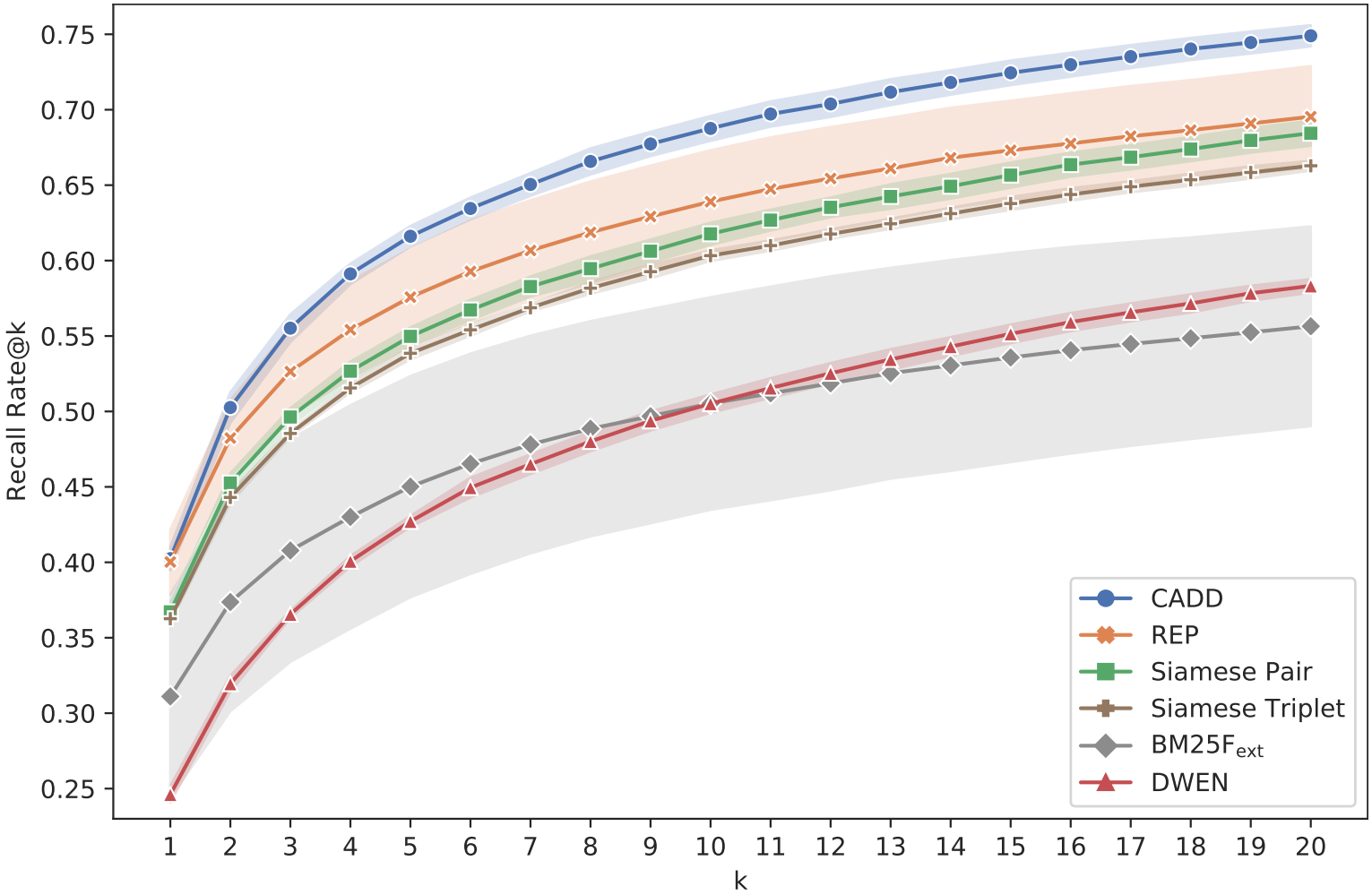
- Four BTSs
  - Eclipse, Firefox, OpenOffice and Netbeans
- Our method (CADD) is compared to:
  - BM25F [2]
  - REP [2]
  - DWEN [3]
  - Siamese Pairs [4]
  - Siamese Triplets [4]

Experimental Results - Eclipse



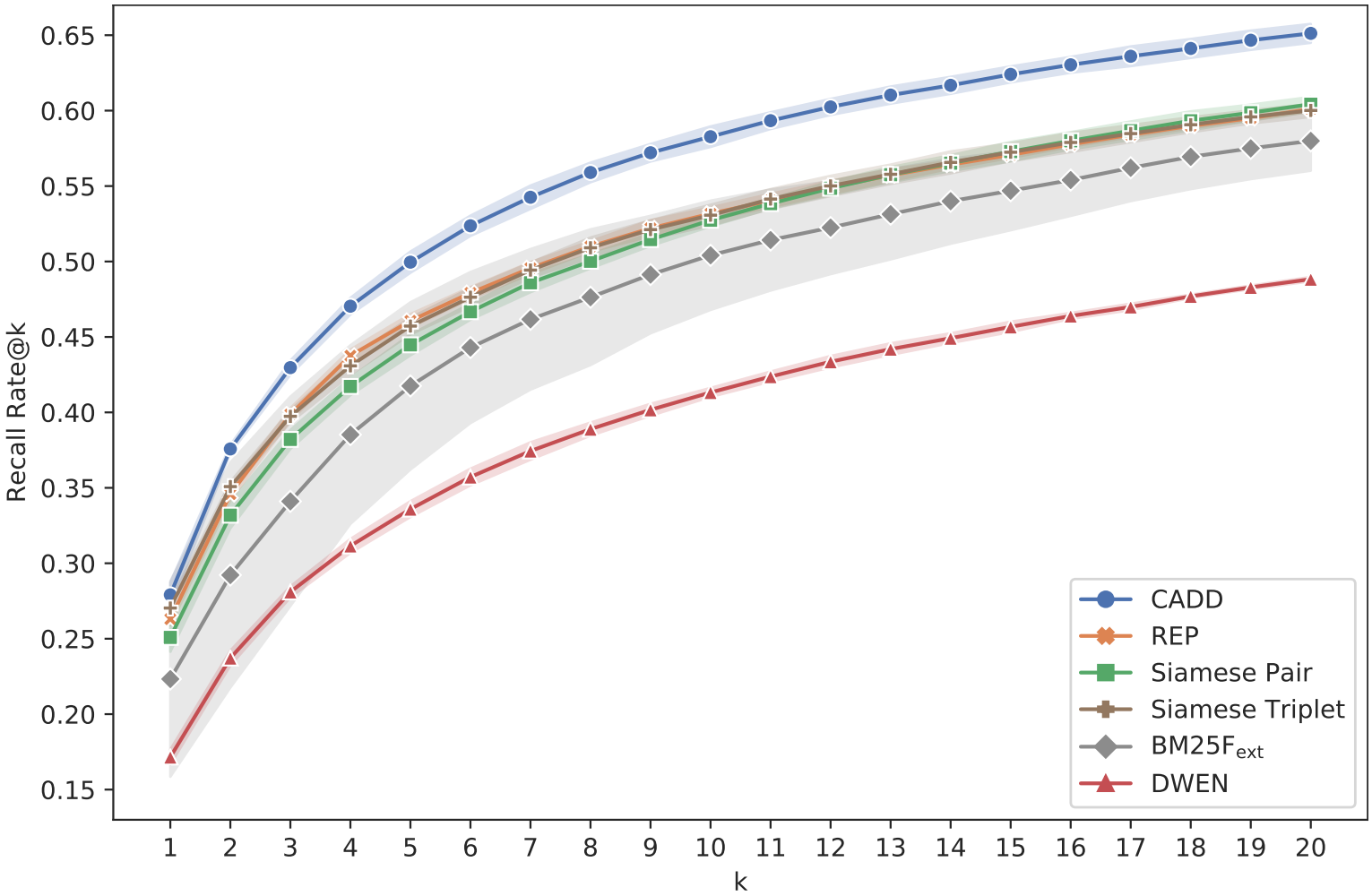
Recall Rate @k in test datasets of **Eclipse**

Experimental Results - Netbeans



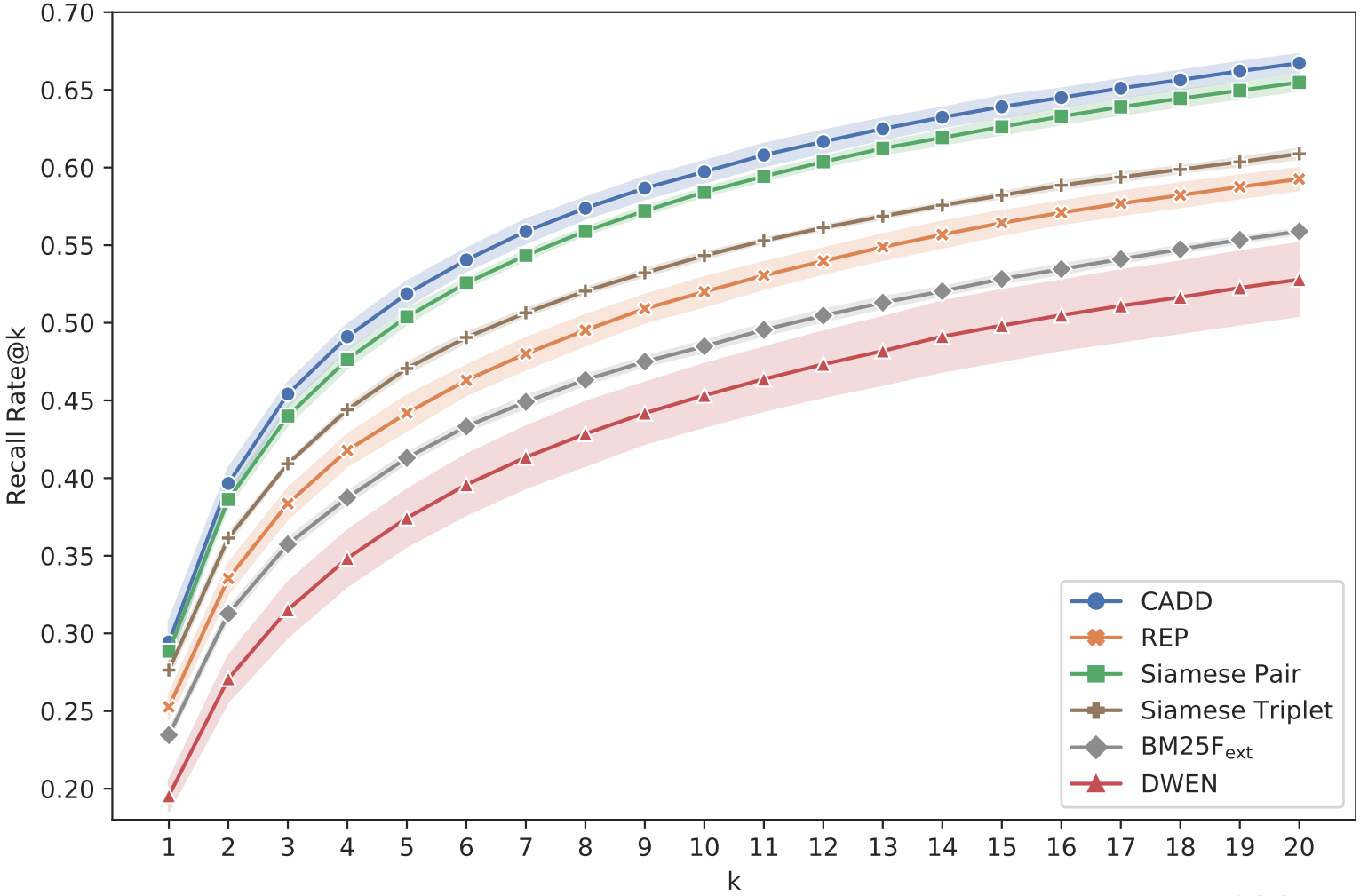
Recall Rate @k in test datasets of **Netbeans**

Experimental Results - Open Office



Recall Rate @k in test datasets of **Open Office**

Experimental Results - Mozilla



Recall Rate @k in test datasets of **Mozilla**

CADD

- Superior performance compared to previous methods
  - Reduce triage team overload
  - Help with more difficult instances
- Scalable
  - GPUs
- Easily applied to other BTSs
  - No need for feature engineering
- CADD is limited by report qualities

CADD

- Superior performance compared to previous methods
  - Reduce triage team overload
  - Help with more difficult instances
- Scalable
  - GPUs
- Easily applied to other BTSs
  - No need for feature engineering
- CADD is limited by report qualities

CADD

- Superior performance compared to previous methods
  - Reduce triage team overload
  - Help with more difficult instances
- Scalable
  - GPUs
- Easily applied to other BTSs
  - No need for feature engineering
- CADD is limited by report qualities

CADD

- Superior performance compared to previous methods
  - Reduce triage team overload
  - Help with more difficult instances
- Scalable
  - GPUs
- Easily applied to other BTSs
  - No need for feature engineering
- CADD is limited by report qualities

Textual Data Disadvantage

- Heavily dependent on users expertise
  - Vague and Ambiguous
  - Different technical background $\rightarrow$ different terminologies
- Limited information about the system execution
  - User can only describe exterior system behaviors
- Alternatives
  - Stack Traces
  - Tracing data

Textual Data Disadvantage

- Heavily dependent on users expertise
  - Vague and Ambiguous
  - Different technical background $\rightarrow$ different terminologies
- Limited information about the system execution
  - User can only describe exterior system behaviors
- Alternatives
  - Stack Traces
  - Tracing data

Textual Data Disadvantage

- Heavily dependent on users expertise
  - Vague and Ambiguous
  - Different technical background $\rightarrow$ different terminologies
- Limited information about the system execution
  - User can only describe exterior system behaviors
- Alternatives
  - Stack Traces
  - Tracing data

Textual Data Disadvantage

- Heavily dependent on users expertise
  - Vague and Ambiguous
  - Different technical background $\rightarrow$ different terminologies
- Limited information about the system execution
  - User can only describe exterior system behaviors
- Alternatives
  - Stack Traces
  - Tracing data

Our Work - Categorical and Textual data

Bug 15247

Position	Function call
1	localstore.FileSystemResourceManager.read
2	resources.File.getContents
3	resources.File.getContents
4	core.util.SyncFileWriter.readLines
5	core.util.SyncFileWriter.readAllResourceSync
6	EclipseSynchronizer.cacheResourceSyncForChildren
7	EclipseSynchronizer.getResourceSync
8	EclipsePhantomSynchronizer.getResourceSync

Bug 51547

Position	Function call
1	localstore.FileSystemResourceManager.read
2	internal.resources.File.getContents
3	internal.resources.File.getContents
4	core.util.SyncFileWriter.readFirstLine
5	core.util.SyncFileWriter.readFolderSync
6	EclipseSynchronizer.cacheFolderSync
7	EclipseSynchronizer.getFolderSync
8	EclipseFolder.getFolderSyncInfo

Our Work - Categorical and Textual data

Bug 15247

Position	Function call
1	localstore.FileSystemResourceManager.read
2	resources.File.getContents
3	resources.File.getContents
4	core.util.SyncFileWriter.readLines
5	core.util.SyncFileWriter.readAllResourceSync
6	EclipseSynchronizer.cacheResourceSyncForChildren
7	EclipseSynchronizer.getResourceSync
8	EclipsePhantomSynchronizer.getResourceSync

Bug 51547

Position	Function call
1	localstore.FileSystemResourceManager.read
2	internal.resources.File.getContents
3	internal.resources.File.getContents
4	core.util.SyncFileWriter.readFirstLine
5	core.util.SyncFileWriter.readFolderSync
6	EclipseSynchronizer.cacheFolderSync
7	EclipseSynchronizer.getFolderSync
8	EclipseFolder.getFolderSyncInfo

Bug deduplication - Stack Trace

Bug 493220

Position	Function call
1	java.util.ArrayList.rangeCheck
2	java.util.ArrayList.get
3	formatter.TokenManager.get
4	formatter.TokenManager.findFirstTokenInLine
5	formatter.TokenManager.findFirstTokenInLine
6	formatter.TextEditsBuilder.bufferIndent
7	formatter.TextEditsBuilder.bufferLineSeparator
8	formatter.TextEditsBuilder.bufferWhitespaceBefore

Bug 488642

Position	Function call
1	formatter.TextEditsBuilder.appendIndentationString
2	formatter.TextEditsBuilder.bufferIndent
3	formatter.TextEditsBuilder.bufferWhitespaceBefore
4	formatter.TextEditsBuilder.token
5	formatter.TokenTraverser.traverse
6	formatter.TokenManager.traverse
7	formatter.DefaultCodeFormatter.format

Bug deduplication - Stack Trace

Bug 493220

Position	Function call
1	java.util.ArrayList.rangeCheck
2	java.util.ArrayList.get
3	formatter.TokenManager.get
4	formatter.TokenManager.findFirstTokenInLine
5	formatter.TokenManager.findFirstTokenInLine
6	formatter.TextEditsBuilder.bufferIndent
7	formatter.TextEditsBuilder.bufferLineSeparator
8	formatter.TextEditsBuilder.bufferWhitespaceBefore

Bug 488642

Position	Function call
1	formatter.TextEditsBuilder.appendIndentationString
2	formatter.TextEditsBuilder.bufferIndent
3	formatter.TextEditsBuilder.bufferWhitespaceBefore
4	formatter.TextEditsBuilder.token
5	formatter.TokenTraverser.traverse
6	formatter.TokenManager.traverse
7	formatter.DefaultCodeFormatter.format

Bug deduplication - Stack Trace

- Studies[5, 6, 7, 8] compare frame positions and do not consider textual similarity
  - Use Package+Function as signature
- *Campbell et. al, 2016*[9] compare only the textual similarity of the stack traces
  - Tokenization: punctuation or camel case
  - TF-IDF

Bug deduplication - Stack Trace

- Studies[5, 6, 7, 8] compare frame positions and do not consider textual similarity
  - Use Package+Function as signature
- *Campbell et. al, 2016*[9] compare only the textual similarity of the stack traces
  - Tokenization: punctuation or camel case
  - TF-IDF

Our solution

- Deep learning method
  - Leverage the data structure
    - Position
    - Package and function
    - Arguments
  - Textual Similarity
    - Assumption: the signature similarity is related to the function behavior
  - Attention mechanism + data reduction

Our solution

- Deep learning method
  - Leverage the data structure
    - Position
    - Package and function
    - Arguments
  - Textual Similarity
    - Assumption: the signature similarity is related to the function behavior
  - Attention mechanism + data reduction

Our solution

- Deep learning method
  - Leverage the data structure
    - Position
    - Package and function
    - Arguments
  - Textual Similarity
    - Assumption: the signature similarity is related to the function behavior
  - Attention mechanism + data reduction

Our solution

- Deep learning method
  - Leverage the data structure
    - Position
    - Package and function
    - Arguments
  - Textual Similarity
    - Assumption: the signature similarity is related to the function behavior
  - Attention mechanism + data reduction

Future Work

- Implement and test our solution for bug deduplication using stack traces
  - Compare with previous methods
  - How can we integrate stack traces and other data types (categorical data)?
- Study how to address the problem with tracing data
  - Generate data
  - Can we use the same method developed by stack traces?

Thank You for your attention!

S. Banerjee, Z. Syed, J. Helmick, M. V. Culp, K. J. Ryan, and B. Cukic, "Automated triaging of very large bug repositories," *Information & Software Technology*, vol. 89, pp. 1–13, 2017.

C. Sun, D. Lo, S.-C. Khoo, and J. Jiang, "Towards more accurate retrieval of duplicate bug reports," in *Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering*, ser. ASE '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 253–262. [Online]. Available: http://dx.doi.org/10.1109/ASE.2011.6100061

A. Budhiraja, K. Dutta, R. Reddy, and M. Shrivastava, "Poster: Dwen: Deep word embedding network for duplicate bug report detection in software repositories," in *2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion)*, May 2018, pp. 193–194.

J. Deshmukh, A. K. M, S. Podder, S. Sengupta, and N. Dubash, "Towards accurate duplicate bug retrieval using deep learning techniques," in *2017 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, Sep. 2017, pp. 115–124.

Y. Dang, R. Wu, H. Zhang, D. Zhang, and P. Nobel, "Rebucket: A method for clustering duplicate crash reports based on call stack similarity," in *2012 34th International Conference on Software Engineering (ICSE)*, June 2012, pp. 1084–1093.

N. E. Koopaei, M. S. Islam, A. Hamou-Lhadj, and M. Hamdaqa, "An effective method for detecting duplicate crash reports using crash traces and hidden markov models," in *Proceedings of the 26th Annual International Conference on Computer Science and Software Engineering*, ser. CASCON '16. Riverton, NJ, USA: IBM Corp., 2016, pp. 75–84.

[Online]. Available: http://dl.acm.org/citation.cfm?id=3049877.3049885

N. E. Koopaei and A. Hamou-Lhadj, "Crashautomata: An approach for the detection of duplicate crash reports based on generalizable automata," in *Proceedings of the 25th Annual International Conference on Computer Science and Software Engineering*, ser. CASCON '15. Riverton, NJ, USA: IBM Corp., 2015, pp. 201–210. [Online]. Available: http://dl.acm.org/citation.cfm?id=2886444.2886474

Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2014, pp. 1746–1751. [Online]. Available: http://aclweb.org/anthology/D14-1181


J. C. Campbell, E. A. Santos, and A. Hindle, "The unreasonable effectiveness of traditional information retrieval in crash report deduplication," in *2016 IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR)*, May 2016, pp. 269–280.