Using Neural Networks for Stacktrace Deduplication

Adem Aber Aouni

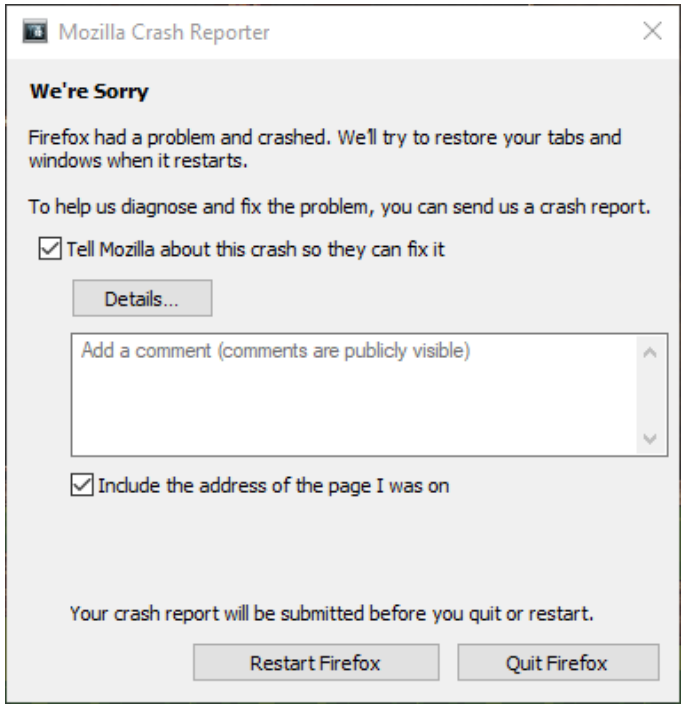
adem.aber-aouni@polymtl.ca

École Polytechnique de Montréal

DORSAL Laboratory

Software Bugs

- It is almost impossible to ship a bug-less software.
- Software may crash due to bugs.
- Error reporting systems were created to gather crash reports anonymously.



Mozilla Crash Reporter [1]

Duplicate Bug reports

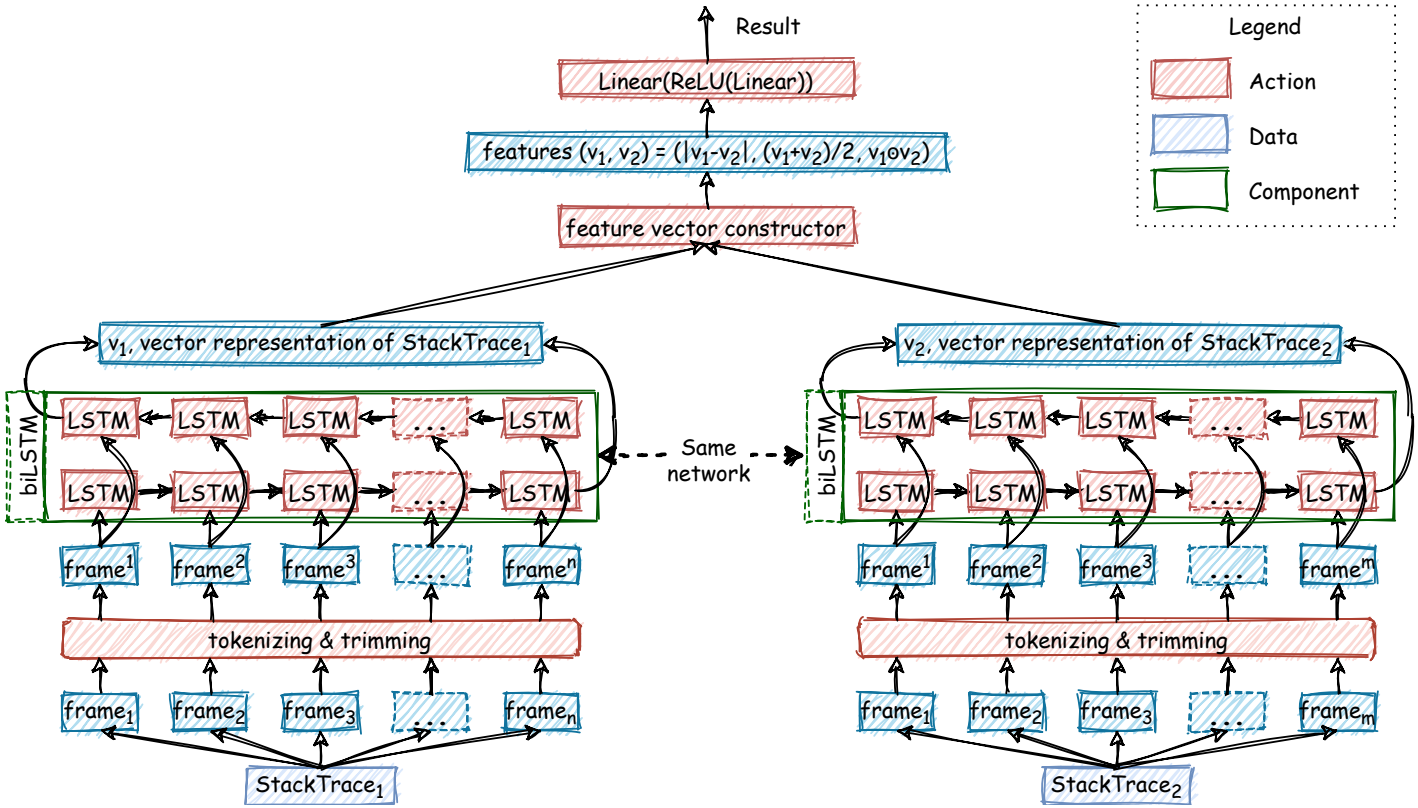
- Popular applications receive a high volume of bug reports.
- ex. Mozilla Core:
  - Average of 3 361.15 bug reports submitted per month [5].
  - 24.70% of the reports are duplicates [5].
- It takes, on average, 17 days less to fix bugs with crash reports grouped together [6].

Methods used

- Stacktrace Alignment Based [2–4; 13–15; 17].
  - Finds a candidate stacktrace that requires the least amount of edits to turn into a query stacktrace.
- TF-IDF and Graph Based [7; 8; 10; 11; 16; 18].
  - Uses term frequency and function call interaction to find best candidate.

S3M: Siamese Stack (Trace) Similarity Measure [9]

- Siamese network and function name trimming.



S3M Architecture [9]

S3M trimming

- Function names trimmed :
  - trim = 0 : com.company.Class1.method2
  - trim = 1 : com.company.Class1
  - trim = 2 : com.company
  - trim = 3 : com

Training

- Training is done by ranking the similarity between good and bad candidates in regards to the query stack trace.
- Good candidates are picked at random from stack traces in the same group.
- Bad candidates are picked from the 50 most similar stacktraces not in the group based on TF-IDF [12].

Lack of ground truth

- Programmers use their knowledge of the code base to group stack together.
- We only have access to the grouped stacks.
- We don't know why a specific report is in a certain group.
- We can only provide a approximate sense of direction for training.

Similarity must be a complex function

- Embedding networks generated through training become intricate.
- Computing similarity between embeddings cannot be done using simple distance functions (cosine or euclidean distance).
- Blocks the use of embedding databases (eg. FAISS) and embedding space search algorithms.

Separation between training and use case

- The model training aims to rank better the all good reports compared to bad reports.
- Real world usage relies on the best suggestion.
- The separation between the goals leads to worst recall rates as training progresses.

Evolving Dataset

- After a set number of epochs, bad candidates are picked from the top 50 wrong predictions using the model.
- This mitigates the overfitting of the model and gives better results.

Better recursion removal

- Multi-pass recursion removal.
- Can remove nested recursions.
- Can simplify AAABCBCBBBBC $\rightarrow$ ABC.

Comparison Against State of the Art

- Better results than state of the art.

S3M Best Mrr

0.62

S3M Best rr@1

0.53

S3M Best rr@5

0.72

S3M Best rr@5

0.76

Best Mrr
S3M autosync trim=1

0.7645

(0.7198, 0.7645)

Best rr@1
S3M autosync trim=1

0.7185

(0.666, 0.7185)

Best rr@5
S3M autosync trim=2

0.8338

(0.7934, 0.8338)

Best rr@10
S3M autosync trim=2

0.863

(0.8194, 0.863)

Next Steps

- Explore non-supervised machine learning methods.
- Mix clustering methods with neural networks.

Questions

?

[1] Mozilla Crash Reporter | Firefox Help.

[2] K Bartz, Jw Stokes, Jc Platt, Ryan Kivett, Ryan Kivett Jack W. Stokes Kevin Bartz, Gretchen Loihle Silviu Calinoiu David Grant John C. Platt, K Bartz, Jw Stokes, Jc Platt, and Ryan Kivett. 2008. Finding Similar Failures Using Callstack Similarity. *SysML*.

[3] Mark Brodie, Sheng Ma, Guy Lohman, Tanveer Syeda-Mahmood, Laurent Mignet, Natwar Modani, Mark Wilding, Jon Champlin, and Peter Sohn. 2005. Quickly finding known software problems via automated symptom matching. *Proceedings - Second International Conference on Autonomic Computing, ICAC 2005*, 2005:101—110.

[4] Yingnong Dang, Rongxin Wu, Hongyu Zhang, Dongmei Zhang, and Peter Nobel. 2012. ReBucket: A Method for Clustering Duplicate Crash Reports Based on Call Stack Similarity. *2012 34th International Conference on Software Engineering (ICSE)*, 1:1084–1093.

[5] Jennifer L. Davidson, Nitin Mohan, and Carlos Jensen. 2011. Coping with duplicate bug reports in free/open source software projects. In *2011 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 101–108.

[6] Tejinder Dhaliwal, Foutse Khomh, and Ying Zou. 2011. Classifying Field Crash Reports for Fixing Bugs: A Case Study of Mozilla Firefox. *2011 27th IEEE International Conference on Software Maintenance (ICSM)*, pages 333–342.

[7] Neda Ebrahimi and Abdelwahab Hamou-Lhadj. 2015. 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*, CASCON '15, page 201–210. IBM Corp.

[8] Neda Ebrahimi, Md. Shariful Islam, Abdelwahab Hamou-Lhadj, and Mohammad Hamdaqa. 2016. 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*, CASCON '16, page 75–84, USA. IBM Corp.

[9] Aleksandr Khvorov, Roman Vasiliev, George Chernishev, Irving Muller Rodrigues, Dmitrij Koznov, and Nikita Povarov. 2021. S3M: Siamese Stack (Trace) Similarity Measure. *arXiv*.

[10] Sunghun Kim, Thomas Zimmermann, and Nachiappan Nagappan. 2011. Crash graphs: An aggregated view of multiple crashes to improve crash triage. *Proceedings of the International Conference on Dependable Systems and Networks*, pages 486—493.

[11] Neda Ebrahimi Koopaei, Abdelaziz Trabelsi, Md. Shariful Islam, Abdelwahab Hamou-Lhadj, and Kobra Khanmohammadi. 2019. An hmm-based approach for automatic detection and classification of duplicate bug reports. *Information and Software Technology*, 113:98–109.

[12] Johannes Lerch and Mira Mezini. 2013. Finding duplicates of your yet unwritten bug report. *Proceedings of the European Conference on Software Maintenance and Reengineering, CSMR*, pages 69—78.

[13] Natwar Modani, Rajeev Gupta, Guy Lohman, Tanveer Syeda-Mahmood, and Laurent Mignet. 2007. Automatically Identifying Known Software Problems. *2007 IEEE 23rd International Conference on Data Engineering Workshop*, pages 433–441.

[14] Akira Moroo, Akiko Aizawa, and Takayuki Hamamoto. 2017. Reranking-based crash report deduplication. *Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE*, pages 507—510.

[15] Irving Muller Rodrigues, Aleksandr Khvorov, Daniel Aloise, Roman Vasiliev, Dmitrij Koznov, Eraldo Rezende Fernandes, George Chernishev, Dmitry Luciv, and Nikita Povarov. 2021.

Tracesim: An alignment method for computing stack trace similarity. *Empirical Software Engineering manuscript*.

[16] Korosh Koochekian Sabor, Abdelwahab Hamou-Lhadj, and Alf Larsson. 2017. DURFEX: A feature extraction technique for efficient detection of duplicate bug reports. *Proceedings - 2017 IEEE International Conference on Software Quality, Reliability and Security, QRS 2017*, pages 240—250.

[17] Roman Vasiliev, Dmitrij Koznov, George Chernishev, Aleksandr Khvorov, Dmitry Luciv, and Nikita Povarov. 2020. TraceSim: A Method for Calculating Stack Trace Similarity. *arXiv*.

[18] Rongxin Wu, Hongyu Zhang, Shing Chi Cheung, and Sunghun Kim. 2014. Crashlocator: Locating crashing faults based on crash stacks. *2014 International Symposium on Software Testing and Analysis, ISSTA 2014 - Proceedings*, pages 204—214.