A System-Wide Debugging Assistant Powered by Natural Language Processing

Pradeep Dogga*  Karthik Narasimhan†  Anirudh Sivaraman‡  Ravi Netravali*
Distributed Systems are complex
Debugging is hard - abstraction gap

Application is not loading some content!
Painful debugging process

Developer

Application is not loading some content!

Is it a bug or feature request?

Which team is relevant for this?

Find root-cause

Preliminary Diagnosis
Painful debugging process – Finding root cause

Developer

Application is not loading some content!

Corrupt key-value store?

Check logs from API calls to key-value store

Wrong hypothesis!

Query Generation

Active Debugging

Routing loop at switch

Check traffic logs from that switch

Correct hypothesis! (Identified a loop)

Largely manual and error-prone
Painful debugging process – Generate Fix

Developer

Change switch configuration file

Verify application behavior

Fix
Systems debugging tools

Application Logs
Systems debugging tools

**Marple** (SIGCOMM 17)

Network Metrics
Systems debugging tools

Canopy (SOSP 17)
Pivot Tracing (SOSP 15)
Debugging remains difficult

• Still manual and error-prone:
  • Which tool?
  • When?
  • How?

• Debugging intuitions are hard-won!
Can we use a data-driven approach to automate steps in end-to-end debugging?
Large amounts of debugging data

Distributed tracing at Pinterest with new open source tools

To monitor its thousands of services, Facebook captures about billion traces a day (about ~100TB collected), a dynamic sampling of the total number of interactions per day — @Facebook’s Haozhe Gao and Joe O’Neill #QConNYC

Two big classes of data:

**Quantitative/Structured**
- Logs from tools
- Performance metrics
- Source code

**Unstructured/Natural Language**
- User Issues
- Documentation and comments
- Past bug reports
Related Work

• **Program Analysis and Synthesis:**
  • NLP for code generation, Deep API learning (FSE 16)

• **Program Debugging:**
  • Net2Text: English queries => SQL queries (NSDI 18)

• **Big Code:**
  • Initiative to perform statistical program analysis on large amounts of code

Limitations:
  • Only ingest data from a single subsystem
  • Assume a single-step prediction
A System-Wide Debugging Assistant Powered by Natural Language Processing

Issues/Bug Reports \rightarrow \text{Code/Configuration Files} \rightarrow \text{NL Debugging Assistant}

Suggestion:
• Label
• Folder/Module
• Use tcpdump
• Issue query X with Marple

Feedback

System-wide concern

End Host Logs

Network Metrics

Application Logs
Automating steps in end-to-end debugging

1. Preliminary Diagnosis
2. Generating Debugging queries
3. Active Debugging
4. Fix!
Preliminary Diagnosis

- **Automate**: Label assignment and Module prediction
- **Category**: Text classification and document retrieval

- **Challenge**: Learn joint representations of data from both unstructured text and structured source code.
Label Prediction – Preliminary Evaluation

- 165966 labeled issues from the top 98 open-source Github repositories (based on stars)
- **Bag-of-words** representation of issue text

```
Menu panel not being closed when not detached
```

<table>
<thead>
<tr>
<th></th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Menu</td>
<td></td>
</tr>
<tr>
<td>panel</td>
<td></td>
</tr>
<tr>
<td>not</td>
<td>2</td>
</tr>
<tr>
<td>tool</td>
<td>0</td>
</tr>
<tr>
<td>closed</td>
<td>1</td>
</tr>
<tr>
<td>css</td>
<td>0</td>
</tr>
<tr>
<td>being</td>
<td>1</td>
</tr>
<tr>
<td>detached</td>
<td>1</td>
</tr>
<tr>
<td>when</td>
<td>1</td>
</tr>
</tbody>
</table>

`FFN`
Results

Prediction performance

- Precision
- Recall
- F1-score

Label Prediction
Source Code Folder Prediction – Preliminary Evaluation

- 240138 issues with corresponding fixes from Github repositories

Menu panel not being closed when not detached

Fix in: /src/lib/menu

FFN

Relevance Score
Results

![Prediction performance chart]

Folder prediction
Automating steps in end-to-end debugging
Generating debugging queries

- **Automate**: Query generation for use with debugging tools
- **Category**: Language generation

Application loading contents slowly → Debugging Assistant

System logs → Issue debugging query:

'Stream = filter(T, (switch == 2) );
R = map(stream, [qin], [qin]);'

Developer: Found large queue depths due to a flow!

- **Challenge**: Understand system logs, source code semantics and language syntax
Template-based query prediction

- A platform to let users interact with the system and collect data for query generation.
- Network debugging tool for performance queries (Marple)

Distributed reddit setup

Pick a fault from:
- Shut down Cassandra host
- Create congestion on reddit-switch link with other traffic
- ...

Fault Injector
Template-based query prediction

Distributed reddit setup

stream = filter(T, (switch == 4));
R = map(stream, [qin], [qin]);

Queue depths
Template-based query prediction

- Predict the correct template and switch to diagnose the root-cause
- Collected issue reports using the testbed from one user for faults injected using fault injector.

Application loading content slowly

Template1 Switch 10

 FFN

Relevance Score
Results

Prediction performance

- Precision
- Recall
- F1-score

Query generation
Automating steps in end-to-end debugging

1. Preliminary Diagnosis
2. Generating Debugging queries
3. Active Debugging
4. Fix!

Developer
Active (interactive) debugging

- **Automate**: Iterative query generation by incorporating feedback
- **Category**: Sequential decision making

![Diagram](attachment:image.png)
Active (interactive) debugging

- **Automate**: Iterative query generation by incorporating feedback
- **Category**: Sequential decision making

Done: Found an issue in routing!

- **Challenge**: Developer-assistant interface to leverage developer’s experience
Challenges & Future Work

• Need to determine optimal model to leverage information from text and traces to generate queries syntactically
• Data collection, training time – need to develop novel systems and algorithmic techniques
• End-to-end evaluation – Evaluate impact of the assistant in the debugging experience with real issues.
• Developer study on systems with reasonable complexity
Conclusion

• Our work paints a vision for an end-to-end debugging assistant which can:
  • Process natural language inputs
  • Various system logs
  • Leverage multiple domain specific debugging tools
  • Automate the three steps in debugging
Thank you!

Contact: dogga@cs.ucla.edu
http://web.cs.ucla.edu/~dogga