Interpretability-driven Intelligent Software Reliability Engineering

HE, Shilin

Ph.D. Oral Defense

Supervisor: Prof. Michael R. Lyu

2020/09/03
Software is Everywhere

• Traditional software

• Intelligent software

Software is Eating the World  --- Marc Andreessen, *The Wall Street Journal*
Software Reliability is Crucial

• Software reliability is important to both service providers and end users!
Real-World Examples

- Unreliable traditional software

Real-World Examples

• Unreliable intelligent software
Software reliability is a must
Software reliability engineering is **challenging**

since the increasing **complexity** and **scale** of software make it **hard to comprehend**
Software Reliability is Challenging

• Traditional Software Complexity
  • Hadoop: 4,103,332 lines of code in 14 languages

<table>
<thead>
<tr>
<th>Language</th>
<th>Code Lines</th>
<th>Comment Lines</th>
<th>Comment Ratio</th>
<th>Blank Lines</th>
<th>Total Lines</th>
<th>Total Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java</td>
<td>1,688,473</td>
<td>543,932</td>
<td>24.4%</td>
<td>287,755</td>
<td>2,520,160</td>
<td>61.4%</td>
</tr>
<tr>
<td>XML</td>
<td>1,149,831</td>
<td>31,931</td>
<td>2.7%</td>
<td>36,977</td>
<td>1,218,739</td>
<td>29.7%</td>
</tr>
<tr>
<td>C++</td>
<td>122,960</td>
<td>51,981</td>
<td>29.7%</td>
<td>25,464</td>
<td>200,405</td>
<td>4.9%</td>
</tr>
</tbody>
</table>

• Intelligent Software Complexity
  • BERT-large (Google): 340 million parameters
  • T5 (Google): 11 billion parameters
  • GPT-3 (OpenAI): 175 billion parameters
If we cannot understand the software, how could we keep it reliable?

Interpretability is the first step
Traditional Software Interpretation

- **Development Practices**
  - *source code readability*, e.g., writing code comments

- **Static Program Analysis**
  - *control-flow analysis*
  - *data-flow analysis*
  - *abstract interpretation*

- **Dynamic Program Analysis**
  - *testing*
  - *program slicing*
  - *monitoring, e.g., logs*
Intelligent Software Interpretation

• A thriving research area under study
Intelligent Software Interpretation

- Interpretability helps the intelligent software reliability.
  - testing:
  - debugging
  - robustness and safety

- interpretability ↑ reliability ↑
Thesis Contributions

Interpretability-driven SRE

Traditional Software

Log-based Anomaly Detection

The first empirical study on log anomaly detection
Release a toolkit for reuse

Intelligent Software

Log-based Problem Identification

Efficient cascading clustering algorithm
Correlates with KPIs to identify problems

Gradient-based Attribution Estimation

Gradient information to explain model predictions by word importance
Detect under-translation errors

Phrase-table-based Knowledge Assessment

Phrase-table to globally explain model behaviors
Explain model learning dynamics and advanced techniques.

[Chapter 3]
[ISSRE’16]
Thesis Contributions

Interpretability-driven SRE

Traditional Software

- Log-based Anomaly Detection
  - The first empirical study on log anomaly detection
  - Release toolkit for reuse

- Log-based Problem Identification
  - Efficient cascading clustering algorithm
  - Correlates with KPIs to identify problems

Intelligent Software

- Gradient-based Attribution Estimation
  - Gradient information to explain model predictions by word importance
  - Detect under-translation errors

- Phrase-table-based Knowledge Assessment
  - Phrase-table to globally explain model behaviors
  - Explain model learning dynamics and advanced techniques.

[Chapter 4] [FSE’18]
Thesis Contributions

Interpretability-driven SRE

Traditional Software
- Log-based Anomaly Detection
  - The first empirical study on log anomaly detection
  - Release toolkit for reuse
- Log-based Problem Identification
  - Efficient cascading clustering algorithm
  - Correlates with KPIs to identify problems

Intelligent Software
- Gradient-based Attribution Estimation
  - Gradient information to explain model predictions by word importance
  - Detect under-translation errors
- Phrase-table-based Knowledge Assessment
  - Phrase-table to globally explain model behaviors
  - Explain model learning dynamics and advanced techniques.

[Chapter 5]

[EMNLP'19]
Thesis Contributions

Interpretability-driven SRE

Traditional Software

- Log-based Anomaly Detection
  - The first empirical study on log anomaly detection
  - Release toolkit for reuse

- Log-based Problem Identification
  - Efficient cascading clustering algorithm

- Gradient-based Attribution Estimation
  - Gradient information to explain model predictions by word importance
  - Correlates with KPIs to identify problems
  - Detect under-translation errors

Intelligent Software

- Phrase-table-based Knowledge Assessment
  - Phrase-table to globally explain model behaviors
  - Explain model learning dynamics and advanced techniques.

[Chapter 6]

EMNLP’20*
Automated Log Interpretation -- Motivation

• Manual analysis of logs is almost infeasible.
  • Logs are generated at a high rate. (10+ TB/hour)
  • Large-scale software is often implemented by hundreds of developers.
  • Manual inspection is error-prone.
Automated Log Interpretation

• A general framework

Log Parsing → Feature Extraction → Log Mining
Automated Log Interpretation

Log Parsing → Feature Extraction → Log Mining

01. Name=Request (GET:http://AAA:1000/BBBB/sitedata.html) → t_41bx0
02. Leaving Monitored Scope (EnsureListItemsData) Execution Time=52.9013 → t_51xi4
03. HTTP request URL: /14/Emails/MrX(MrX@mail.com)/1c-48f0-b29.eml → t_23hl3
04. HTTP Request method: GET → t_41bx0
05. HTTP request URL: /55/RST/UVX/ADEG/Lists/Files/docXX.doc → t_01mu1
06. Overridden HTTP request method: GET → t_41bx0
07. HTTP request URL: http://AAA:1000/BBBB/sitedata.html → t_41bx0
08. Leaving Monitored Scope (Request (POST:http://AAA:1000/BBBB/sitedata.html)) Execution Time=334.319268903038 (Task_ID) → t_41bx0

Log Parsing

E1. Name=Request (*)
E2. Leaving Monitored Scope (*) Execution Time = *
E3. HTTP Request method: *
E4. HTTP request URL: *
E5. Overridden HTTP request method: *
Automated Log Interpretation

Log Sequence Grouping

- **Task identifier:**
  Job ID, Process ID, etc

- **Time stamp:**
  1) Fixed window
  2) Sliding window

An example of HDFS logs
Automated Log Interpretation

Feature Vectorization

- Each feature denotes a log event in the log sequence.

- For example
  
  E1  E2  E3  E4  E5  E6
  [ 1, 0, 2, 3, 1, 0 ]
  E1 occurs once
  E4 occurs three times.
Automated Log Interpretation

- **Anomaly Detection**
  - Normal cases
  - Anomalies

- **Problem Identification**
  - Normal cases
  - Different problem types
Outline

• Topic 1: Log-based Anomaly Detection

• Topic 2: Log-based Problem Identification

• Topic 3: Gradient-based Attribution Estimation

• Conclusion and Future Work
Outline

• Topic 1: Log-based Anomaly Detection
Log-based Anomaly Detection

• Motivation:
  o Lack of comparison among existing anomaly detection methods.
  o The state-of-the-art anomaly detection methods are unknown.
  o No open-source tools are currently available.

• Contribution:
  o provide the first empirical study on log-based anomaly detection methods.
  o release the toolset for public reuse.
Anomaly Detection Methods

• State-of-the-art research studies (Before 2016)
  o Failure diagnosis using decision trees [ICAC’04]
  o Failure prediction in IBM bluegene/l event logs [ICDM’07]
  o Detecting largescale system problems by mining console logs [SOSP’09]
  o Mining invariants from console logs for system problem detection. [USENIX ATC’10]
  o Log clustering based problem identification for online service systems [ICSE’16]
  o ...
Anomaly Detection Methods

• Taxonomy

Anomaly Detection

Supervised
- Logistic Regression
- Decision Tree
- Support Vector Machine

Unsupervised
- Log Clustering
- PCA
- Invariants Mining
Anomaly Detection Methods

- PCA

- **Sn: Normal Space** principal components
- **Sa: Anomaly Space** remaining components

- Check whether the projected vector is far from the normal space
Experiments

• Datasets

<table>
<thead>
<tr>
<th>System</th>
<th>#Time span</th>
<th>#Data size</th>
<th>#Log messages</th>
<th>#Anomalies</th>
</tr>
</thead>
<tbody>
<tr>
<td>BGL</td>
<td>7 months</td>
<td>708 M</td>
<td>4,747,963</td>
<td>348,460</td>
</tr>
<tr>
<td>HDFS</td>
<td>38.7 hours</td>
<td>1.55 G</td>
<td>11,175,629</td>
<td>16,838</td>
</tr>
</tbody>
</table>

• Evaluation metric:

Precision / Recall / F1-Score
Experiments

• Accuracy of supervised methods

Finding 1: Supervised anomaly detection achieves high precision, while recall varies.

Finding 2: Sliding windows achieve higher accuracy than fixed windows.
Experiments

• Accuracy of unsupervised methods

Finding 3: Unsupervised methods generally achieve inferior performance against supervised methods.
Experiments

• Various hyper-parameters settings

Findings 4: The window size and step size affect both supervised and unsupervised methods a lot.
Experiments

• Efficiency

*Finding 5: Most anomaly detection methods scale linearly with log size*
Summary

• Provide an **empirical study** of **six** SOTA anomaly detection methods.

• Compare their **accuracy and efficiency** on **two representative** log datasets.

• Release an **open-source toolkit** for easy reuse and further study.
• Topic 2: Log-based Problem Identification

Interpretability-driven SRE

Traditional Software
- Log-based Anomaly Detection
- Log-based Problem Identification

Intelligent Software
- Gradient-based Attribution Estimation
- Phrase-table-based Knowledge Assessment

[Chapter 4]
• Problem type matters
• Some types of problem are more impactful, should be fixed with a higher priority.

![Like Grid](image.png)
Challenges

1. Lack of labels

2. Huge log size

Unsupervised Methods

Inefficient
Challenges

3. Highly imbalanced log distribution
   - High service availability in cloud-based online service systems

99.999%
Challenges

3. Highly imbalanced log distribution
   ○ problems occasionally happen, demonstrating a long-tail distribution.

4. Problem impact
   ○ difficult to quantitatively identify the impact of a problem.
• System KPIs (Key Performance Indicators)
  o measure the system‘s health status in a certain time period
    ▪ Failure Rate
    ▪ Service Availability
    ▪ Average Request Latency

• periodically collected

<table>
<thead>
<tr>
<th>Time interval</th>
<th>Failure rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1h</td>
<td>0.48</td>
</tr>
<tr>
<td>1h</td>
<td>0.23</td>
</tr>
<tr>
<td>1h</td>
<td>0.14</td>
</tr>
<tr>
<td>1h</td>
<td>0.53</td>
</tr>
</tbody>
</table>
Log3C: **Cascading Clustering and Correlation Analysis**

**Input:** Raw logs, KPIs

**Output:** Clusters of impactful problems

Framework of Log3C
Parsing and Vectorization

- Logs are parsed into log events with log parsing.
- Different log events play different roles in problem identification.
  - IDF weighting
  - Importance weighting

1. Log Parsing

$$t_1: [E_1, E_2, E_4, E_5]$$
$$[E_2, E_3, E_4, E_5]$$
$$[E_1, E_2, E_3, E_5, E_4]$$

$$t_2: [E_2, E_3, E_4, E_5]$$
$$[E_2, E_1, E_5, E_3, E_6]$$
$$[E_1, E_2, E_5, E_4]$$

$$\vdots$$

$$t_d: [E_1, E_2, E_4, E_5]$$
$$[E_3, E_4, E_6, E_5]$$
$$[E_1, E_2, E_3, E_5]$$

2. Sequence Vectorization

<table>
<thead>
<tr>
<th>E1</th>
<th>E2</th>
<th>E3</th>
<th>E4</th>
<th>E5</th>
<th>E6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

$$\text{KPIs}$$

$$\text{Sum}$$

$$\alpha \cdot \text{Norm}(w(idf)) + (1-\alpha) \cdot w(cor)$$
Cascading Clustering

Traditional clustering methods are infeasible.

Log Sequences → Sampling → Clustering & Pattern extraction → Matching

Matched data → Mismatched data
Cascading Clustering

• Group log sequences with cascading clustering in each time interval

![Cascading Clustering Diagram]
Correlation Analysis

• Impactful problems: Can lead to the degradation of KPI.

• Goal: Identify clusters that are highly correlated with KPI’s changes.

1. Correlate cluster sizes—KPI values with the Multivariate Linear Regression (MLR)
2. T-statistic hypothesis test
Experiments

• Datasets: Real-world data from the service system X

<table>
<thead>
<tr>
<th>Data</th>
<th>Snapshot starts</th>
<th>#Log Seq (Size)</th>
<th>#Events</th>
<th>#Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data 1</td>
<td>Sept 5th 10:50</td>
<td>359,843 (722MB)</td>
<td>365</td>
<td>16</td>
</tr>
<tr>
<td>Data 2</td>
<td>Oct 5th 04:30</td>
<td>472,399 (996MB)</td>
<td>526</td>
<td>21</td>
</tr>
<tr>
<td>Data 3</td>
<td>Nov 5th 18:50</td>
<td>184,751 (407MB)</td>
<td>409</td>
<td>14</td>
</tr>
</tbody>
</table>

• Manual labelling
  1. Problem or not?
  2. Problem type?
Experiments

• Effectiveness Evaluation:
  o Problem Detection (Binary Classification)
    Precision / Recall / F1-Measure
    \[
    \text{Precision} = \frac{TP}{TP + FP} \quad \text{Recall} = \frac{TP}{TP + FN}
    \]
  
  o Problem Identification (Clustering)
    Normalized Mutual Information (NMI) \( \sim \) between \([0, 1]\)
    \[
    NMI(Y, C) = \frac{2 \times I(Y; C)}{H(Y) + H(C)}
    \]
    Y = class labels \quad H(.) = \text{Entropy}
    C = cluster labels \quad I(Y;C) = \text{Mutual Information b/w Y and C}

• Efficiency Evaluation:
  o Clustering Time (in seconds)
Experiments

• Accuracy of Problem Detection:

<table>
<thead>
<tr>
<th>Data 1</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>0.465</td>
<td>0.946</td>
<td>0.623</td>
</tr>
<tr>
<td>Invariants Mining</td>
<td>0.604</td>
<td>1</td>
<td>0.753</td>
</tr>
<tr>
<td>Log3C</td>
<td><strong>0.900</strong></td>
<td>0.920</td>
<td><strong>0.910</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data 2</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>0.142</td>
<td>0.834</td>
<td>0.242</td>
</tr>
<tr>
<td>Invariants Mining</td>
<td>0.160</td>
<td>0.847</td>
<td>0.269</td>
</tr>
<tr>
<td>Log3C</td>
<td><strong>0.897</strong></td>
<td>0.826</td>
<td><strong>0.860</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data 3</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>0.207</td>
<td>0.922</td>
<td>0.338</td>
</tr>
<tr>
<td>Invariants Mining</td>
<td>0.168</td>
<td>0.704</td>
<td>0.271</td>
</tr>
<tr>
<td>Log3C</td>
<td><strong>0.834</strong></td>
<td>0.903</td>
<td><strong>0.868</strong></td>
</tr>
</tbody>
</table>
Experiments

• Accuracy of **Problem Identification** (NMI):

<table>
<thead>
<tr>
<th></th>
<th>Size</th>
<th>10k</th>
<th>50k</th>
<th>100k</th>
<th>200k</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data 1</strong></td>
<td>Log3C-SC</td>
<td>0.659</td>
<td>0.706</td>
<td>0.781</td>
<td>0.822</td>
</tr>
<tr>
<td></td>
<td>Log3C</td>
<td>0.720</td>
<td>0.740</td>
<td>0.798</td>
<td>0.834</td>
</tr>
<tr>
<td><strong>Data 2</strong></td>
<td>Log3C-SC</td>
<td>0.610</td>
<td>0.549</td>
<td>0.600</td>
<td>0.650</td>
</tr>
<tr>
<td></td>
<td>Log3C</td>
<td>0.624</td>
<td>0.514</td>
<td>0.663</td>
<td>0.715</td>
</tr>
<tr>
<td><strong>Data 3</strong></td>
<td>Log3C-SC</td>
<td>0.601</td>
<td>0.404</td>
<td>0.792</td>
<td>0.828</td>
</tr>
<tr>
<td></td>
<td>Log3C</td>
<td>0.680</td>
<td>0.453</td>
<td>0.837</td>
<td>0.910</td>
</tr>
</tbody>
</table>

Log3C-SC is the comparison method, which replaces the **Cascading Clustering** with the **standard clustering** (HAC)
Experiments

- Efficiency of Cascading Clustering (seconds):

<table>
<thead>
<tr>
<th></th>
<th>Size</th>
<th>10k</th>
<th>50k</th>
<th>100k</th>
<th>200k</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data 1</td>
<td>SC</td>
<td>127.6</td>
<td>2319.2</td>
<td>9662.3</td>
<td>38415.5</td>
</tr>
<tr>
<td></td>
<td>CC</td>
<td>1.0</td>
<td>4.3</td>
<td>9.2</td>
<td>20.7</td>
</tr>
<tr>
<td>Data 2</td>
<td>SC</td>
<td>80.6</td>
<td>2469.1</td>
<td>8641.2</td>
<td>38614.0</td>
</tr>
<tr>
<td></td>
<td>CC</td>
<td>0.7</td>
<td>3.8</td>
<td>9.5</td>
<td>18.9</td>
</tr>
<tr>
<td>Data 3</td>
<td>SC</td>
<td>81.5</td>
<td>2417.2</td>
<td>8761.2</td>
<td>33728.3</td>
</tr>
<tr>
<td></td>
<td>CC</td>
<td>0.8</td>
<td>4.0</td>
<td>8.8</td>
<td>18.3</td>
</tr>
</tbody>
</table>
Experiments

• Cascading clustering under various configurations

![Graph showing time performance and accuracy vs. sample rate]

Decreasing sample rate does not sacrifice the accuracy while greatly reducing the time
Summary

• Propose Cascading Clustering, an efficient clustering method.

• Propose the Log3C framework, leverage the KPI information as the supervision.

• Experiments on real-world datasets confirm its effectiveness and efficiency.

• Deployed to the actual maintenance of Microsoft products.
Outline

• **Topic 3: Gradient-based Attribution Estimation**
What is the “Log” in intelligent software?

- Parameters? Millions, Billions
- Architecture? CNN, RNN
- Gradient Information
• Neural Machine Translation (NMT) as the intelligent software

Facebook translates 'good morning' into 'attack them', leading to arrest

Palestinian man questioned by Israeli police after embarrassing mistranslation of caption under photo of him leaning against bulldozer

Facebook has apologised after an error in its machine-translation service saw Israeli police arrest a Palestinian man for posting “good morning” on his social media profile.
Background

• How to “interpret” the intelligent software?
  o Input-output correspondence

• **Word Importance**: the importance of each *input word* to the *output sentence*.
  
  • Also applicable in the adversarial attack and defense.
Challenges

1. Traditional methods on interpreting NMT:
   - Attention: attention is not explanation [Jain et al. 2019]
   - Erasure: it requires the reference [Li et al. 2016]
   - Causality: it requires a Variational Auto Encoder model and ensembles the attention. [Alvarez-Melis et al. 2017]
Challenges

2. The basic gradient information does not apply to deep neural networks

\[ f(x) = 1 - \text{ReLU}(1-x) \]

- \( f(0) = 0 \)
- \( f(1) = 1 \)

Gradient saturation

The gradient is 0 since \( f \) is flat when \( x = 1 \)

ReLU
Integrated Gradients

• Intuition: find a baseline input $x'$ to calculate the \textit{relative} feature importance in $x$

$$IG^m_m(x) = (x_m - x'_m) \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha(x - x'))_n}{\partial x_m} \, d\alpha$$

• $F$: the model, e.g., Transformer, RNNSearch
• $m$: the $m$-th word in the input sentence
• $n$: the $n$-th word in the output sentence
• $alpha$: interpolation ratio
Method

• Integrated Gradients with approximation

\[
IG_m^n(x) = \frac{(x_m - x'_m)}{S} \sum_{k=0}^{S} \frac{\partial F(x' + \frac{k}{S}(x - x'))}{\partial x_m}
\]

• S: Total interpolation steps
• k: the k-th interpolation step
• **Word Importance:**
  - Step 1: Estimate the integrated gradient of each word pair;
  - Step 2: Sum the contribution of an input word to all output words;
  - Step 3: Normalize with the Softmax function.
Evaluation Metric

- Translation performance when perturbing the most important words

Original Input: \[\text{Yellow, Gray, Yellow, Yellow}\]

Perturbed Input: \[\text{Yellow, Gray, Red, Yellow}\]

- Perturbation Types:
  - Deletion
  - Mask
  - Grammatical Replacement
Experiments

• Effectiveness of different word importance estimation methods.

Finding 1: Important words are more influential on translation performance than the others.
Finding 2: The gradient-based method is superior to comparative methods (e.g., Attention) in estimating word importance.
Experiments

- Further experiments on model structures, language pairs, and directions.

Finding 3: The proposed method is consistently effective against model structures, language pairs and translation directions.
Experiments

• Comparison with the supervised erasure method.

• Erasure:
  • Estimate the word importance by perturbing each word one by one and calculate the performance drop

\[ \Delta \text{BLEU} \]

\[
\begin{align*}
0.5 & \quad 1.7 & \quad 2.3 & \quad 0.8
\end{align*}
\]
Experiments

• Machine translation problems

Reference

Under-translation

Mis-translation

Over-translation
Experiments

- Detecting under-translation errors without reference
  - a straightforward method: words with the least word importance (top N%)

Original Input:

<table>
<thead>
<tr>
<th>Method</th>
<th>Top 5%</th>
<th>Top 10%</th>
<th>Top 15%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attention</td>
<td>0.058</td>
<td>0.077</td>
<td>0.119</td>
</tr>
<tr>
<td>Erasure</td>
<td>0.154</td>
<td>0.170</td>
<td>0.192</td>
</tr>
<tr>
<td>Attribution</td>
<td><strong>0.248</strong></td>
<td><strong>0.316</strong></td>
<td><strong>0.342</strong></td>
</tr>
</tbody>
</table>

F1-measure
Summary

• We approach understanding NMT by investigating the word importance via a gradient-based method.

• Empirical results show that the proposed method is superior to baseline methods.

• Our study suggests the possibility of detecting the under-translation error via a gradient-based method.
Outline

- Topic 1: Log-based Anomaly Detection
- Topic 2: Log-based Problem Identification
- Topic 3: Gradient-based Attribution Estimation
- Conclusion and Future Work
Conclusion

- The first empirical study
- Release a toolkit for reuse
- Highly imbalanced data w/o labels
- Cascading clustering and Correlation with KPI
- Gradient information for word importance
- Detect translation errors
Future Work

• Interpretable automated log analysis

KPIs

Trust?

Traffic

Not trust?

Logs

ALERT !!!!
Future Work

• Robustness of Intelligent Software

"panda" + .007 × noise = "gibbon"

57.7% confidence

99.3% confidence


* denotes in submission
Publications


Intelligent Log Analysis

• LogPAI (Log analytics power by AI)

Development Phase

```
/* hadoop/dfs../LeaseRenewer.java
 * (Simplified for easy presentation) */
Try {
    renew;
    lastRenewed = Time.monotonicNow();
} catch (IOException ie) {
    LOG.warn("Failed to renew lease for " +
            clientsString()) + " for " + (elapsed/1000) + " seconds. Will retry shortly ...", ie);
}
```

A Sample Code Snippet

LoggingStatements
[ASE’ 2018]

Loghub
[Arxiv’ 2020]

Logzip
[ASE’ 2019]

Logparser
[ICSE’ 2019]
[TDSC’ 2017]
[DSN’ 2016]

Loglizer & Log3C
[ISSRE’ 2016]
[FSE’ 2018]
Open-Source Projects

• LogPAI on GitHub

- LogAnalytics Powered by AI

LogAdvisor (ICSE'15)
- Learning to log: A framework for determining optimal logging points

LogHub (ICSE'19), LogZip(ASE'19)
- A collection of system log datasets for massive log analysis (440 million log lines)

Logizer (ISSRE'16)
- A log analysis toolkit for automated anomaly detection

LoggingDescriptions (ASE'18)
- A collection of Software Logging Statements in source code

LogParser (DSN'16)
- A toolkit for automated log parsing

Log3C (FSE'18)
- Log-based Problem Identification

- 2000+ stars
- 800+ forks
- Release a large dataset (77GB log)

Downloads:

14,596 views 16,846 downloads
Thanks!
Back up slides
Software Reliability is Challenging

• Intelligent Software Complexity
  • BERT (Google):
    base: 110 million parameters with 12 layers and 12 attention heads
    large: 340 million parameters with 24 layers and 16 attention heads
  • T5 (Google): 11 billion parameters
  • GPT-3 (OpenAI): 175 billion parameters
An Overview

Interpretability-driven Software Reliability

Traditional Software
- Log-based Anomaly Detection
- Log-based Problem Identification

Intelligent Software
- Gradient-based Attribution Estimation
- Phrase-table-based Knowledge Assessment
Intelligent Log Analysis

• Log Generation

**Source Code Snippet**

```java
/* hadoop/hdfs.
LeaseRenewer.java
*(Simplified for easy presentation)*
*/
Try
{
    renew();
    lastRenewed =
    Time.monotonicNow();
} catch (IOException ie)
{
    LOG.warn("Failed to renew
lease for " + clientsString() + " for "+
(elapsed/1000) + " seconds. Will
retry shortly ...", ie);
}
```

**Log Messages**

[LeaseRenewer:service@clusters:9000]
org.apache.hadoop.hdfs.LeaseRenewer: Failed to renew lease for
[DFSClient_NONMAPREDUCE_1537864556_1] for 51 seconds. Will retry shortly …

[LeaseRenewer:service@clusters:9000]
org.apache.hadoop.hdfs.LeaseRenewer: Failed to renew lease for
[DFSClient_NONMAPREDUCE_-274751412_1] for 79 seconds.

[LeaseRenewer:service@clusters:9000]
org.apache.hadoop.hdfs.LeaseRenewer: Failed to renew lease for
[DFSClient_NONMAPREDUCE_-1547462655_1] for 785 seconds. Will retry shortly …
Interpretability

- Interpretability is the degree to which a human can understand the cause of a decision.

- Human-understandable insights:
  - visual explanations
  - natural language explanations
  - domain specific explanations

- Sometimes referred as “Program Analysis”, “Program Comprehension”, “Program Understanding”
• Interpretability is approached from the following aspects:
  
  • Input-Output Attribution
  
  • Internal Representations
  
  • Data Point Attribution
• Why PCA does not perform well on BGL?

The BGL data distribution after PCA projection, normal cases and anomalies are not separable.
Background

• NMT model structures
Experiments

- Linguistic Analysis on important words
  - POS Tag

<table>
<thead>
<tr>
<th>Type</th>
<th>Chinese→English</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Count</td>
<td>Attri.</td>
<td>Δ</td>
<td>Count</td>
<td>Attri.</td>
<td>Δ</td>
<td>Count</td>
</tr>
<tr>
<td>Noun</td>
<td>0.383</td>
<td>0.407</td>
<td>+6.27%</td>
<td>0.341</td>
<td>0.355</td>
<td>+4.11%</td>
<td>0.365</td>
</tr>
<tr>
<td>Verb</td>
<td>0.165</td>
<td>0.160</td>
<td>-3.03%</td>
<td>0.146</td>
<td>0.131</td>
<td>-10.27%</td>
<td>0.127</td>
</tr>
<tr>
<td>Adj.</td>
<td>0.032</td>
<td>0.029</td>
<td>-9.38%</td>
<td>0.076</td>
<td>0.072</td>
<td>-5.26%</td>
<td>0.094</td>
</tr>
<tr>
<td>Total</td>
<td>0.579</td>
<td>0.595</td>
<td>+2.76%</td>
<td>0.563</td>
<td>0.558</td>
<td>-0.89%</td>
<td>0.587</td>
</tr>
<tr>
<td>Prep.</td>
<td>0.056</td>
<td>0.051</td>
<td>-8.93%</td>
<td>0.120</td>
<td>0.132</td>
<td>+10.00%</td>
<td>0.129</td>
</tr>
<tr>
<td>Dete.</td>
<td>0.043</td>
<td>0.043</td>
<td>0.00%</td>
<td>0.102</td>
<td>0.101</td>
<td>-0.98%</td>
<td>0.112</td>
</tr>
<tr>
<td>Punc.</td>
<td>0.137</td>
<td>0.131</td>
<td>-4.38%</td>
<td>0.100</td>
<td>0.091</td>
<td>-9.00%</td>
<td>0.096</td>
</tr>
<tr>
<td>Others</td>
<td>0.186</td>
<td>0.179</td>
<td>-3.76%</td>
<td>0.115</td>
<td>0.118</td>
<td>+2.61%</td>
<td>0.076</td>
</tr>
<tr>
<td>Total</td>
<td>0.421</td>
<td>0.405</td>
<td>-3.80%</td>
<td>0.437</td>
<td>0.442</td>
<td>+1.14%</td>
<td>0.413</td>
</tr>
</tbody>
</table>

Finding 4: Certain syntactic categories have higher importance while the categories vary across language pairs.
Experiments

• Linguistic Analysis on important words
  • Fertility: word alignment

<table>
<thead>
<tr>
<th>Fertility</th>
<th>Chinese→English</th>
<th></th>
<th>English→French</th>
<th></th>
<th>English→Japanese</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Count</td>
<td>Attri.</td>
<td>Δ</td>
<td>Count</td>
<td>Attri.</td>
<td>Δ</td>
</tr>
<tr>
<td>≥ 2</td>
<td>0.087</td>
<td>0.146</td>
<td>+67.82%</td>
<td>0.126</td>
<td>0.138</td>
<td>+9.52%</td>
</tr>
<tr>
<td>1</td>
<td>0.621</td>
<td>0.622</td>
<td>+0.16%</td>
<td>0.672</td>
<td>0.670</td>
<td>-0.30%</td>
</tr>
<tr>
<td>(0, 1)</td>
<td>0.115</td>
<td>0.081</td>
<td>-29.57%</td>
<td>0.116</td>
<td>0.113</td>
<td>-2.59%</td>
</tr>
<tr>
<td>0</td>
<td>0.176</td>
<td>0.150</td>
<td>-14.77%</td>
<td>0.086</td>
<td>0.079</td>
<td>-8.14%</td>
</tr>
</tbody>
</table>

Finding 5: Words of high fertility are always important.
Outline

• Topic 1: Log-based Anomaly Detection

• Topic 2: Log-based Problem Identification

• Topic 3: Gradient-based Attribution Estimation

• Topic 4: Phrase-table-based Knowledge Assessment

• Conclusion and Future Work
Motivations

• NMT evolution path

- Rule-based Machine Translation
- Statistical Machine Translation
- Neural Machine Translation

• Essential translation knowledge should be the same
  • bilingual lexicons (translation model)
  • grammar (reordering and language models)
Motivations

1. The input-output attribution provides local explanations only

   ![Diagram of English to French translation]

   - English: It’s → a → nice → day
   - French: C'est → une → belle → journée

2. There is no previous work on the knowledge assessment in NMT
   - How to represent the knowledge?
   - How to quantitatively assess the knowledge?
Method

• Bilingual knowledge:

- a → une
- nice → belle

• Bilingual knowledge is at the core of adequacy modelling, a major weakness of NMT models

• We propose to assess the *bilingual knowledge* with the statistical translation model, also known as the *phrase table*. 
An Example

• Phrase table extracted from the NMT model

(a) Output of an English ⇒ German NMT model

<table>
<thead>
<tr>
<th>Source</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>I do</td>
<td>Ich</td>
</tr>
<tr>
<td>I do hope that we finally</td>
<td>hoffe ich, dass wir endlich</td>
</tr>
<tr>
<td>winning again</td>
<td>wieder gewinnen</td>
</tr>
<tr>
<td>winning again</td>
<td>gewinnen einer</td>
</tr>
</tbody>
</table>

(b) Phrase table extracted from the NMT model
Method

• Phrase table extraction

Algorithm 1 Constructing Phrase Table

Input: training example \((x, y)\), alignment \(a\), mask \(m\)
Output: phrase set \(\mathcal{R}\)

1: procedure PHRASE TABLE
2: \hspace{1cm} EXTRACTION
3: \hspace{1cm} ESTIMATION
4: procedure EXTRACTION
5: \hspace{1cm} \(\hat{\mathcal{R}} \leftarrow \) extract candidates from \(\{(x, y), a\}\)
6: \hspace{1cm} for each \(r \in \hat{\mathcal{R}}\) do \(\triangleright \) priors of NMT predictions
7: \hspace{1cm} \hspace{1cm} if \(r\) is consistent with \(m\) then
8: \hspace{1cm} \hspace{1cm} \hspace{1cm} \(\mathcal{R}.\text{append}(r)\)
9: \hspace{1cm} \hspace{1cm} procedure ESTIMATION
10: \hspace{1cm} \hspace{1cm} standard procedure
Method

• Implementation
  1. Force-decode the training examples

\[
m_j = \begin{cases} 
1, & \text{if } y_j = \arg\max_{y'_j \in V} P(y'_j | y_{<j}, x) \\
0, & \text{otherwise}
\end{cases}
\]

  2. Build masked training data, $\text{MASK}$

  3. Extract the phrase table

  4. Remove phrase pairs that contain the $\text{MASK}$
Experiments

• RQ1: Is phrase table a reasonable bilingual knowledge representation?
• Evaluation metric for phrase table
  ○ Phrase Table Size
  ○ Recovery Percent
  ○ Translation Quality

The extracted phrase table correlates well with the NMT performance, consistent across language pairs, random seeds and model structures.
Experiments

- **RQ2:** How do NMT models learn the bilingual knowledge during training?
- Different types of phrase pairs with increasing complexity
  - *Phrase Length*
  - *Reordering Type*
  - *Word Fertility*

NMT models tend to learn simple patterns first and complex patterns later.
Experiments

• RQ3: Are the phrase pairs never forgotten once learnt?

Forgetting dynamics occur in the learning of bilingual knowledge.
Experiments

• RQ4: Does the trained NMT model sufficiently exploit the bilingual knowledge embedded in the training examples?

<table>
<thead>
<tr>
<th>Phrase Table</th>
<th>Shared</th>
<th>Non-Shared</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>BLEU</td>
<td>Size</td>
<td>BLEU</td>
</tr>
<tr>
<td>Full</td>
<td>9.0M</td>
<td>8.5M</td>
<td>17.5M</td>
</tr>
<tr>
<td>NMT</td>
<td>9.0M</td>
<td>0M</td>
<td>9.0M</td>
</tr>
</tbody>
</table>

NMT models distill the bilingual knowledge by discarding those low-quality phrase pairs.
Experiments

• Revisit recent advances
  • Model capacity
    Increasing the model capacity does not increase the bilingual knowledge

• Data Augmentation
  Data Augmentation induces new knowledge and enhance existing knowledge over the baseline

• Domain Adaptation
  Domain Adaptation learns more and better bilingual knowledge from the in-domain data while forgetting partial out-of-domain knowledge
## Experiments

- Revisit recent advances
  - Model capacity

### Table: Model Capacity Comparison

<table>
<thead>
<tr>
<th>Model</th>
<th>NMT</th>
<th>Phrase Table</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#Para</td>
<td>BLEU</td>
</tr>
<tr>
<td>SMALL</td>
<td>38M</td>
<td>25.45</td>
</tr>
<tr>
<td>BASE</td>
<td>98M</td>
<td>27.11</td>
</tr>
<tr>
<td>BIG</td>
<td>284M</td>
<td>28.40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>Shared</th>
<th>Non-Shared</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Size</td>
<td>BLEU</td>
</tr>
<tr>
<td>SMALL</td>
<td>7.0M</td>
<td>17.53</td>
</tr>
<tr>
<td>BASE</td>
<td>7.0M</td>
<td>17.49</td>
</tr>
<tr>
<td>BIG</td>
<td>7.0M</td>
<td>17.29</td>
</tr>
</tbody>
</table>

Increasing the model capacity does not increase the bilingual knowledge.
Experiments

- Revisit recent advances
  - Data augmentation

<table>
<thead>
<tr>
<th>Model</th>
<th>NMT</th>
<th>Phrase Table</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#Para</td>
<td>BLEU</td>
</tr>
<tr>
<td>BASE</td>
<td>98M</td>
<td>27.11</td>
</tr>
<tr>
<td>+ BT</td>
<td>98M</td>
<td>29.75</td>
</tr>
<tr>
<td>+ FT</td>
<td>98M</td>
<td>28.43</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>Shared</th>
<th>Non-Shared</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Size</td>
<td>BLEU</td>
</tr>
<tr>
<td>BASE</td>
<td>8.3M</td>
<td>17.67</td>
</tr>
<tr>
<td>+ BT</td>
<td>8.3M</td>
<td>18.61</td>
</tr>
<tr>
<td>BASE</td>
<td>8.4M</td>
<td>17.83</td>
</tr>
<tr>
<td>+ FT</td>
<td>8.4M</td>
<td>18.30</td>
</tr>
</tbody>
</table>

(a) Length (b) Reordering (c) Fertility
Experiments

- Revisit recent advances
  - Domain Adaptation

<table>
<thead>
<tr>
<th>Fine Tune</th>
<th>NMT # Para.</th>
<th>BLEU</th>
<th>Phrase Table Size</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>×</td>
<td>98M</td>
<td>15.78</td>
<td>168K</td>
<td>16.08</td>
</tr>
<tr>
<td>✓</td>
<td>98M</td>
<td>31.26</td>
<td>316K</td>
<td>18.50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fine Tune</th>
<th>Shared Size</th>
<th>BLEU</th>
<th>Non-Shared Size</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>×</td>
<td>0.16M</td>
<td>15.95</td>
<td>0.01M</td>
<td>1.65</td>
</tr>
<tr>
<td>✓</td>
<td>0.16M</td>
<td>16.92</td>
<td>0.16M</td>
<td>6.95</td>
</tr>
</tbody>
</table>
Discussion

• Potential applications:
  • Error diagnosis: debugs mistaken predictions by tracing associated phrase pairs

• Curriculum learning: dynamically assigns more weights to unlearned instances

• Phrase memory: stores unlearned phrases in NMT to query when generating translations

C'est → une → belle → Phrase Table
Summary

• We interpret NMT models by assessing the bilingual knowledge with the phrase table.

• Extensive experiments show that the phrase table is reasonable and consistent.

• Equipped with the interpretable phrase table, we obtain several interesting findings.
Conclusion

- Experience report
- Release toolkit for reuse
- Highly imbalanced data w/o labels
- Cascading clustering and Correlation with KPI
- Gradient information for word importance
- Detect translation errors
- Phrase-table to globally explain model behaviors
- Explain recent model improvements
Thesis Contributions

Interpretability-driven Software Reliability

- Traditional Software
  - Log-based Anomaly Detection
  - Log-based Problem Identification

- Intelligent Software
  - Gradient-based Attribution Estimation
  - Phrase-table-based Knowledge Assessment