Interpretability-driven Intelligent Software Reliability Engineering

HE, Shilin

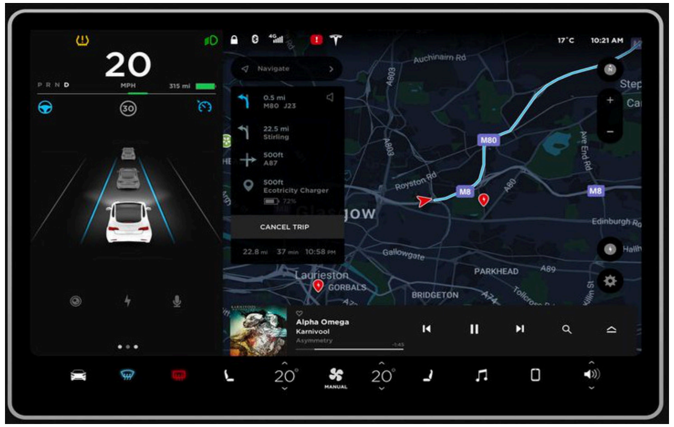
Ph.D. Oral Defense

Supervisor: Prof. Michael R. Lyu

2020/09/03

Software is Everywhere

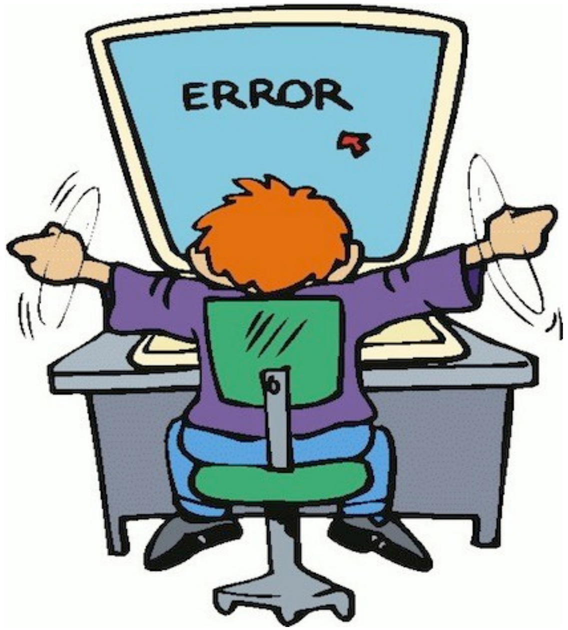
- Traditional software
- Intelligent software



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*!



Revenue Loss ← A Tiny Problem → User Dissatisfaction

Real-World Examples

- Unreliable traditional software

Microsoft news recap: **Azure outage** problems explained ...
OnMSFT (blog) - 11 Apr 2020
Sit back, grab some coffee, and enjoy the read! Microsoft explains recent **Azure outage** problems in Europe due to “constrained capacity”. Azure ...

Google **outage** hits Gmail, Snapchat and Nest
The Guardian - 8 Apr 2020
Google declared the **outage** resolved at 4:57pm BST. Big cloud providers such as Google Cloud Platform, **Amazon** Web Services (**AWS**) and ...

AWS cloud issues hit Sydney region
CRN Australia - 22 Jan 2020
#aws outage Sydney - Its been 3 hours already ...Anyone knows what's happening and recovery timeframe. Impacted ones include glue services ...
AWS suffers cloud problems in Sydney region
iTnews - 22 Jan 2020

[Statistics from: https://techcrunch.com/2017/02/28/amazon-aws-s3-outage-is-breaking-things-for-a-lot-of-websites-and-apps/]

Real-World Examples

- Unreliable intelligent software

Software reliability is a must

Software reliability engineering is challenging

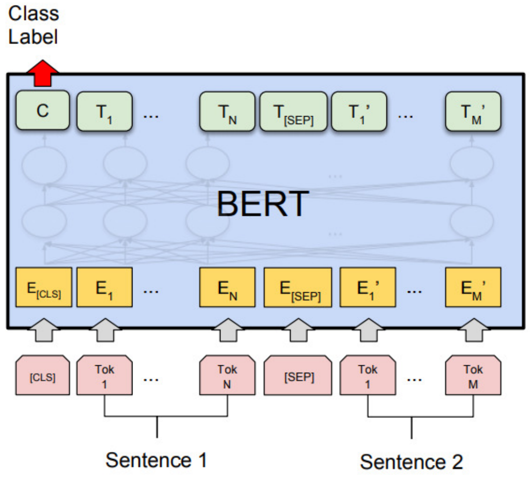
since the increasing complexity and scale of software make it ***hard to comprehend***

Software Reliability is Challenging

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

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

- Intelligent Software Complexity
  - **BERT**-large (Google): 340 million parameters
  - **T5** (Google): 11 billion parameters
  - **GPT-3** (OpenAI): 175 billion parameters

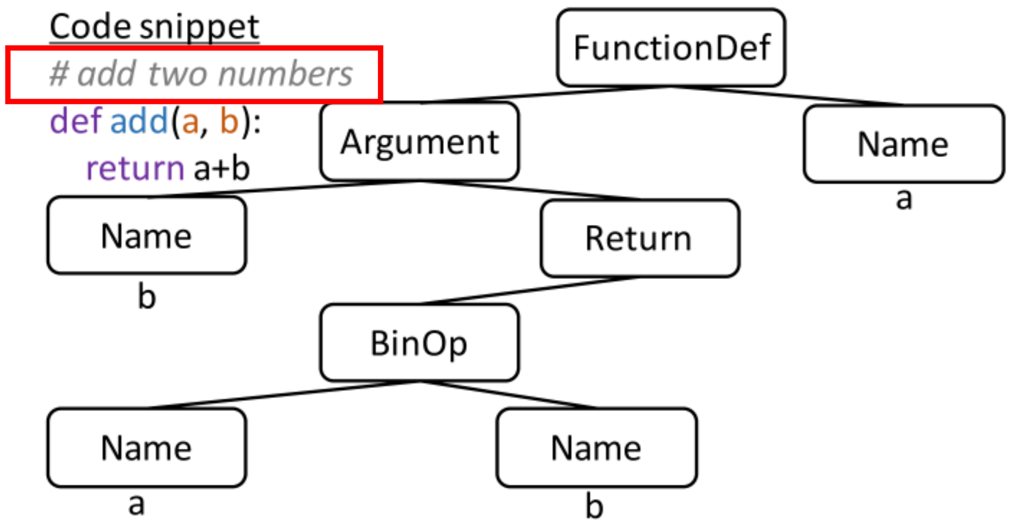


If we cannot understand the software,
how could we keep it reliable?

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*

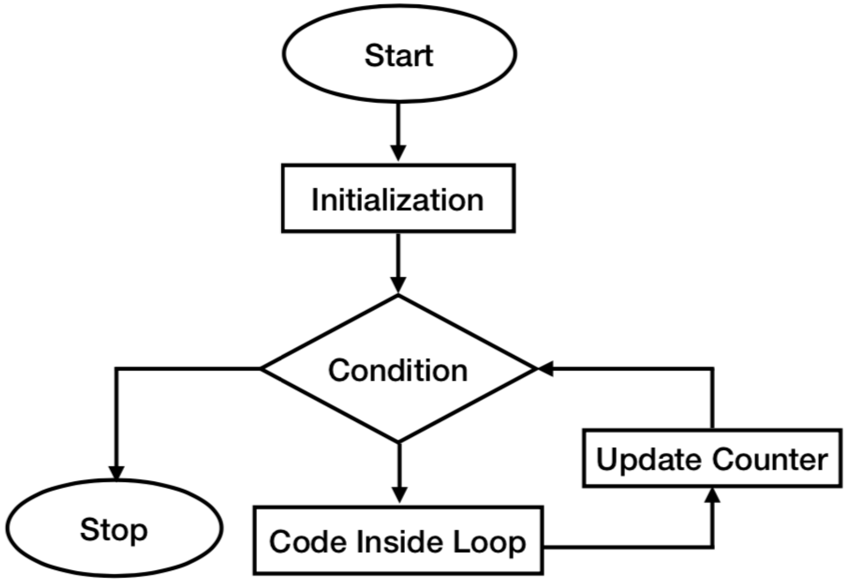


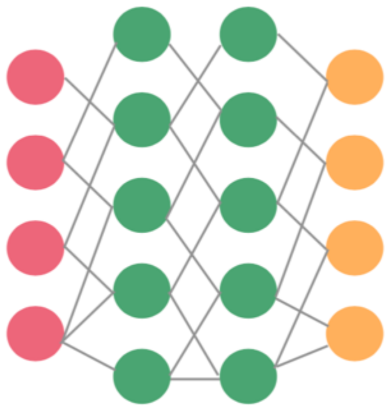
- Dynamic Program Analysis
  - *testing*
  - *program slicing*
  - ***monitoring, e.g., logs***

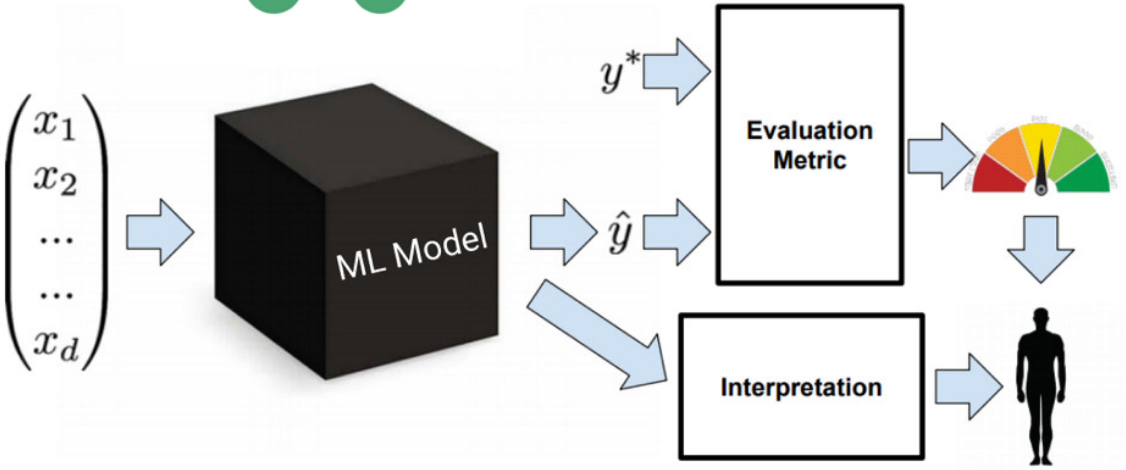
1	**2008-11-09 20:55:54** PacketResponder 0 for block blk_321 terminating
2	**2008-11-09 20:55:54** Received block blk_321 of size 67108864 from /10.251.195.70
3	**2008-11-09 20:55:54** PacketResponder 2 for block blk_321 terminating
4	**2008-11-09 20:55:54** Received block blk_321 of size 67108864 from /10.251.126.5
5	**2008-11-09 21:56:50** 10.251.126.5:50010:Got exception while serving blk_321 to /10.251.127.243
6	**2008-11-10 03:58:04** Verification succeeded for blk_321
7	**2008-11-10 10:36:37** Deleting block blk_321 file /mnt/ hadoop/dfs/data/current/subdir1/blk_321
8	**2008-11-10 10:36:50** Deleting block blk_321 file /mnt/ hadoop/dfs/data/current/subdir51/blk_321

Intelligent Software Interpretation

- A thriving research area under study

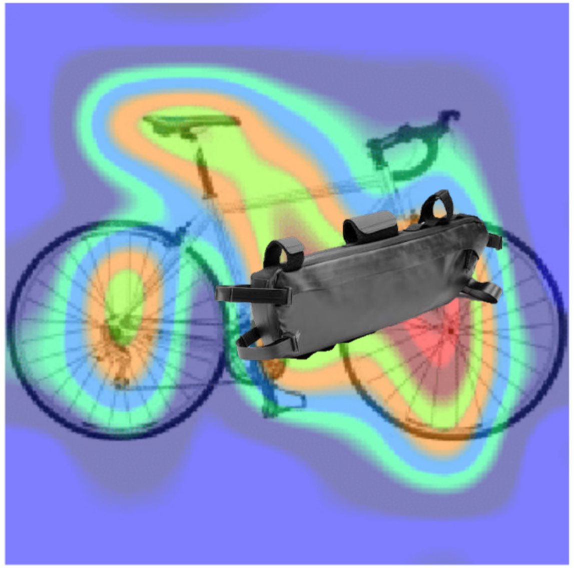




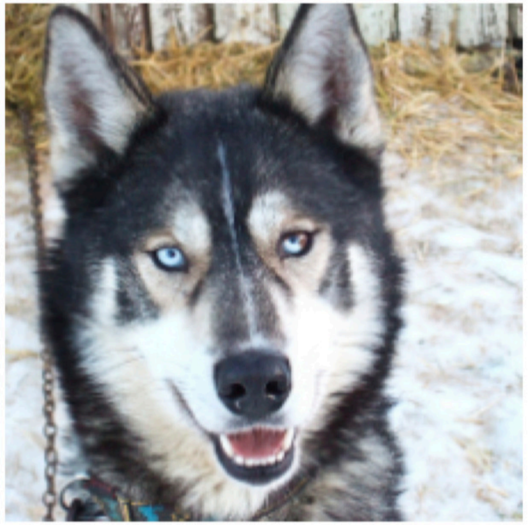


Intelligent Software Interpretation

- Interpretability helps the intelligent software reliability.
  - testing:



  - debugging



(a) Husky classified as wolf

(b) Explanation

  - robustness and safety
- interpretability ↑ reliability ↑

Thesis Contributions

Interpretability-driven SRE

Traditional Software

Intelligent Software

[Chapter 3]

Log-based Anomaly Detection

Log-based Problem Identification

Gradient-based Attribution Estimation

Phrase-table-based Knowledge Assessment

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

[ISSRE'16]

- Efficient cascading clustering algorithm
- Correlates with KPIs to identify problems

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

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

Thesis Contributions

Interpretability-driven SRE

- Traditional Software
  - Log-based Anomaly Detection
  - Log-based Problem Identification [Chapter 4]
- Intelligent Software
  - Gradient-based Attribution Estimation
  - Phrase-table-based Knowledge Assessment

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

[FSE’18]

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.

Thesis Contributions

Interpretability-driven SRE

Traditional Software

Intelligent 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

[Chapter 5]

Gradient-based Attribution Estimation

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

[EMNLP'19]

Phrase-table-based Knowledge Assessment

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

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
    - [Chapter 6] Phrase-table-based Knowledge Assessment
      - Phrase-table to globally explain model behaviors
      - Explain model learning dynamics and advanced techniques.

[EMNLP’20]*

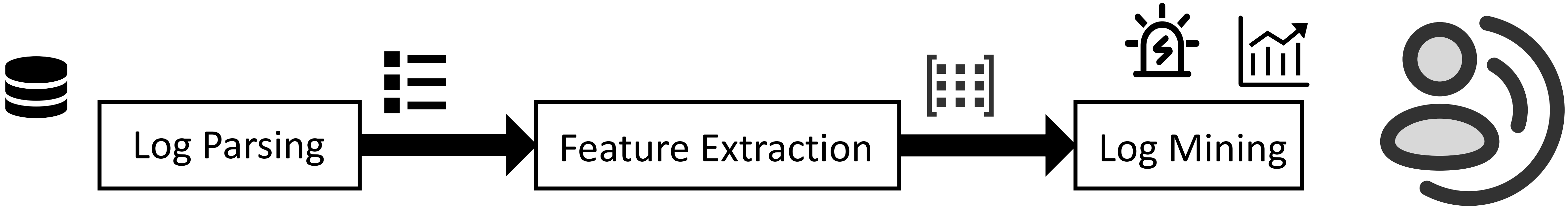
Automated Log Interpretation -- Motivation

- 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

Automated Log Interpretation

- A general framework



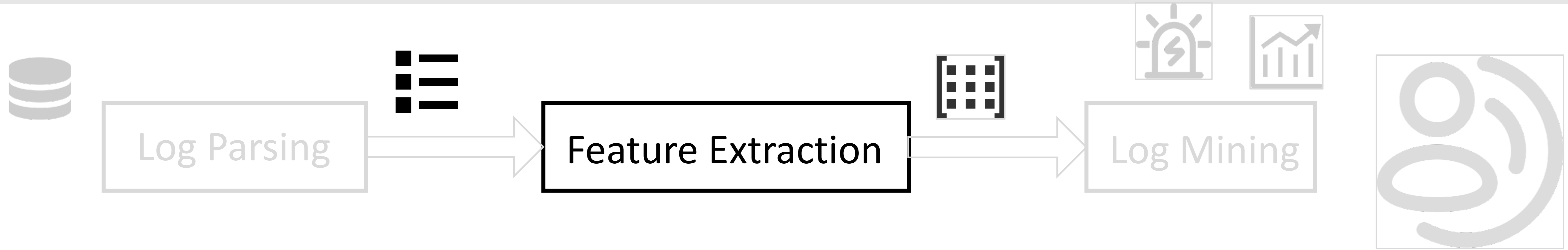
Automated Log Interpretation

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:100/BBBB/ sitedata.html)) Execution Time=334.319268903038	t_41bx0
		(Task_ID)

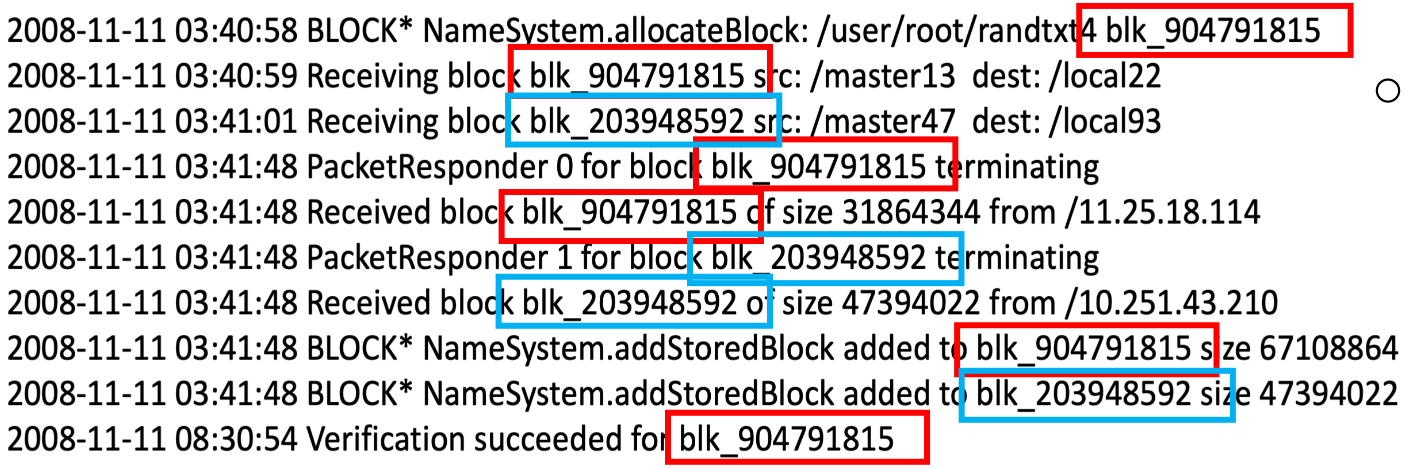
Log Parsing

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

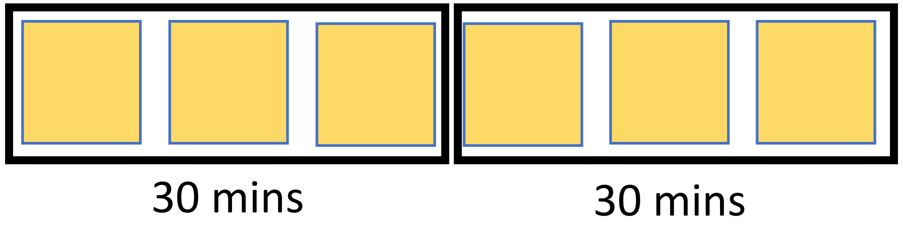


Log Sequence Grouping



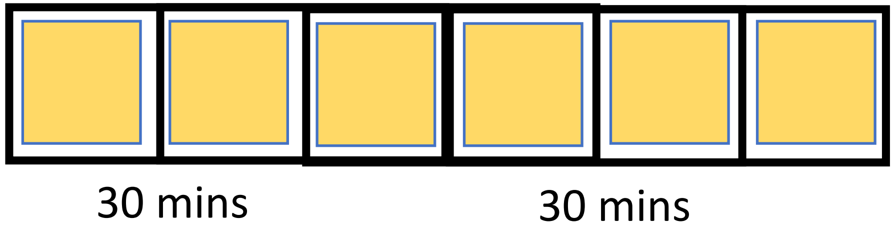
An example of HDFS logs

- ***Task identifier:***
  Job ID, Process ID, etc
- ***Time stamp:***
  1) Fixed window

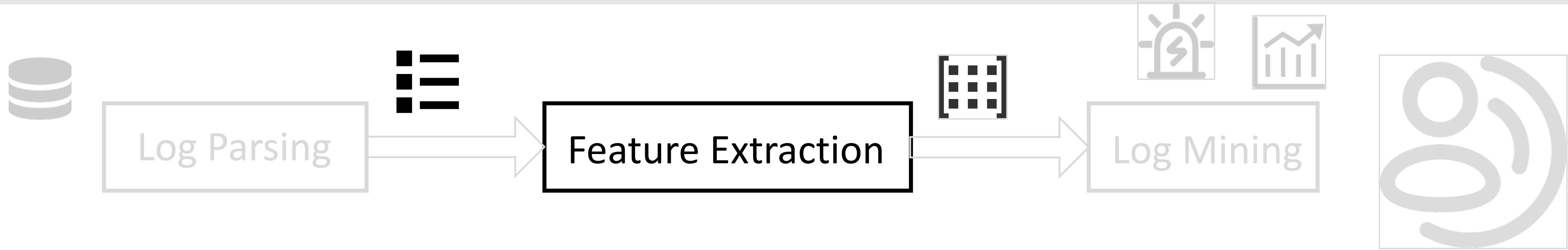


  2) Sliding window

Step: 10 min



Automated Log Interpretation



Feature Vectorization

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

- For example

$$\begin{array}{cccccc} E1 & E2 & E3 & E4 & E5 & E6 \\ \end{array}$$
$$[\,1,\ 0,\ 2,\ 3,\ 1,\ 0\,]$$

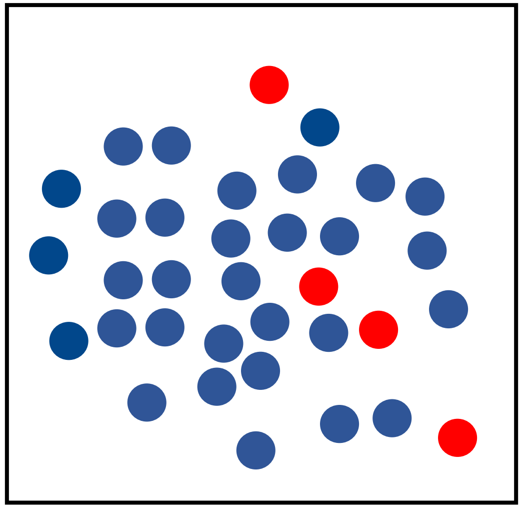
E1 occurs once

E4 occurs three times.

$$\begin{bmatrix} 1\ 0\ 1\ 2\ 1\ 0 \\ \ddots \\ 0\ 0\ 3\ 2\ 2\ 0 \end{bmatrix}$$

Automated Log Interpretation

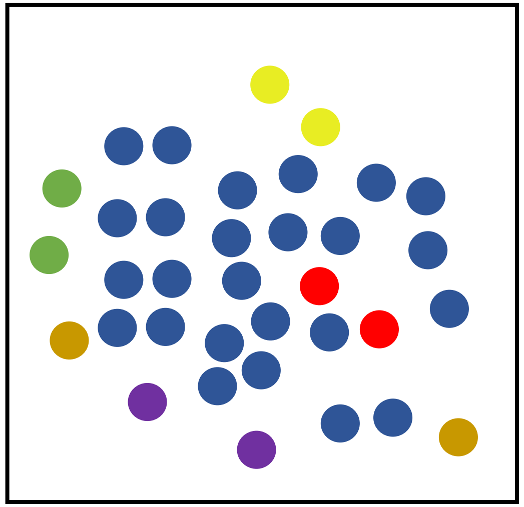
- ○ Anomaly Detection



Normal cases

Anomalies

- ○ Problem Identification



Normal cases

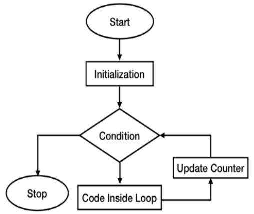
Different problem types

Outline

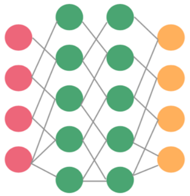
- Topic 1: Log-based Anomaly Detection
- Topic 2: Log-based Problem Identification
- Topic 3: Gradient-based Attribution Estimation
- Conclusion and Future Work

Outline

- Topic 1: Log-based Anomaly Detection



Interpretability-driven SRE



Traditional Software

Intelligent Software

[Chapter 3]

Log-based Anomaly Detection

Log-based Problem Identification

Gradient-based Attribution Estimation

Phrase-table-based Knowledge Assessment

Log-based Anomaly Detection

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

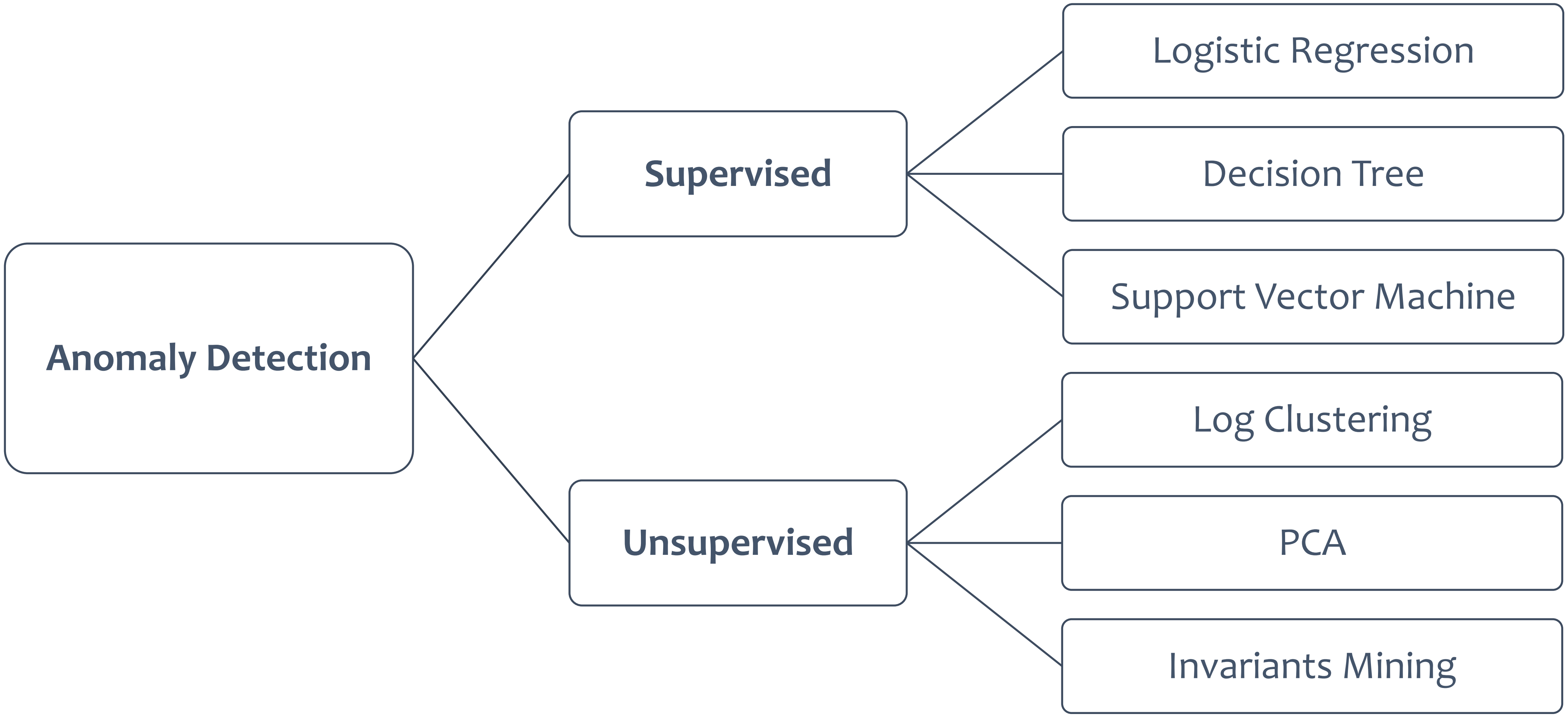
- Contribution:
  - provide the first empirical study on log-based anomaly detection methods.
  - release the toolset for public reuse.

Anomaly Detection Methods

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

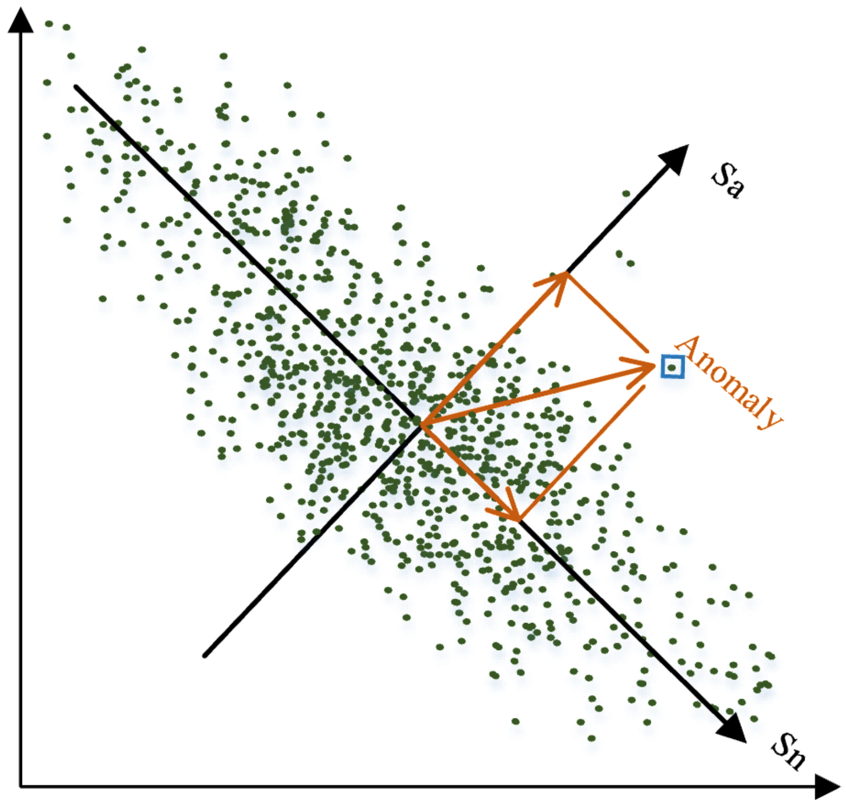
Anomaly Detection Methods

- Taxonomy



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

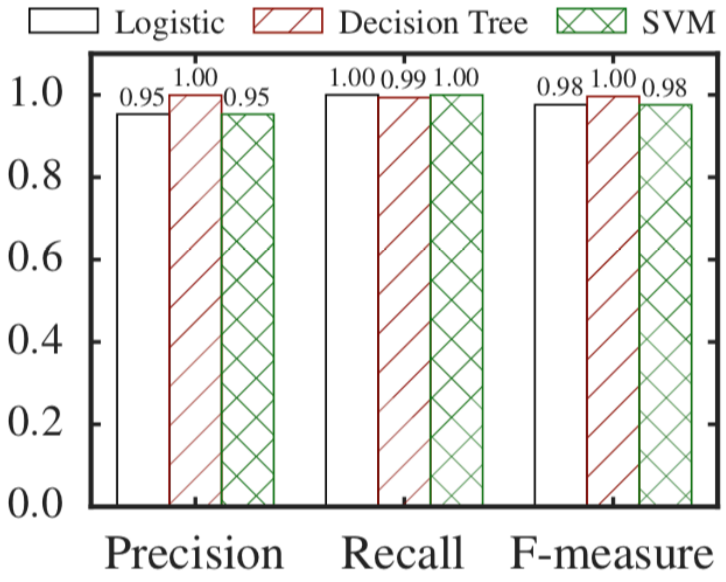
System	#Time span	#Data size	#Log messages	#Anomalies	
BGL	7 months	708 M	4,747,963	348,460	→ Time-stamp
HDFS	38.7 hours	1.55 G	11,175,629	16,838	→ Task-identifier

- Evaluation metric:

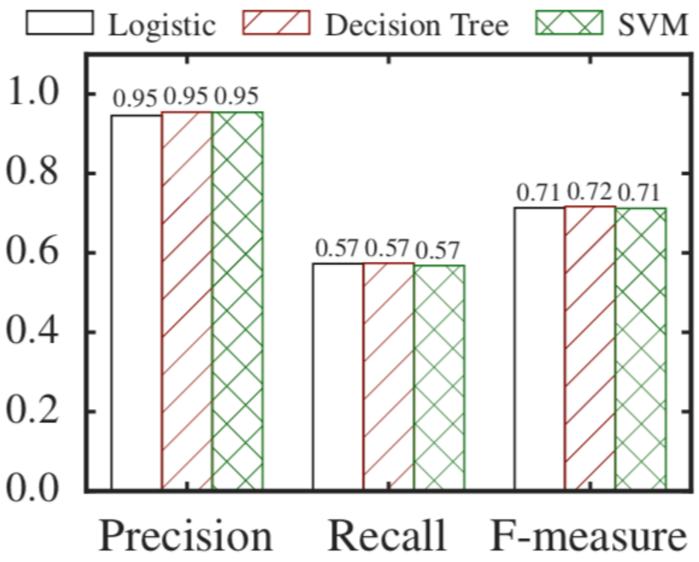
Precision / Recall / F1-Score

Experiments

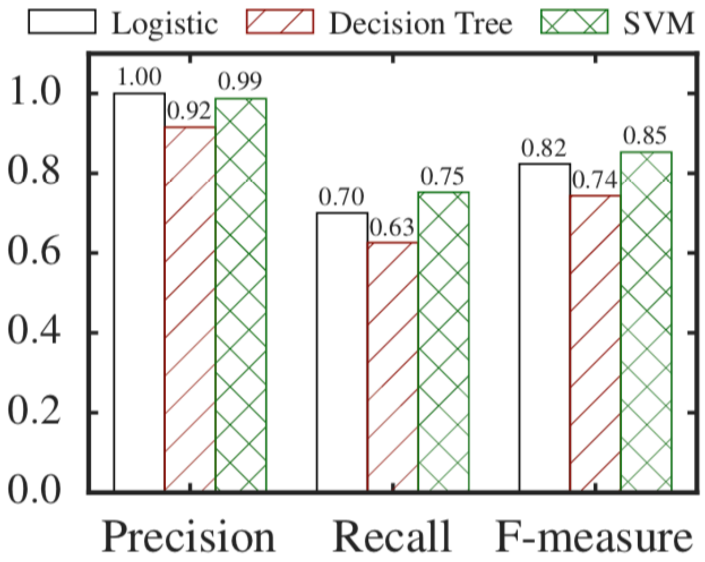
- Accuracy of supervised methods



HDFS



BGL (Fixed window)

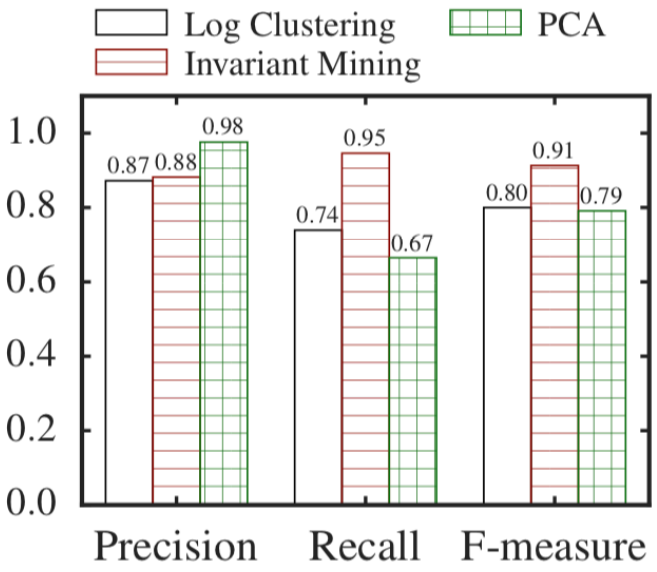


BGL (Sliding window)

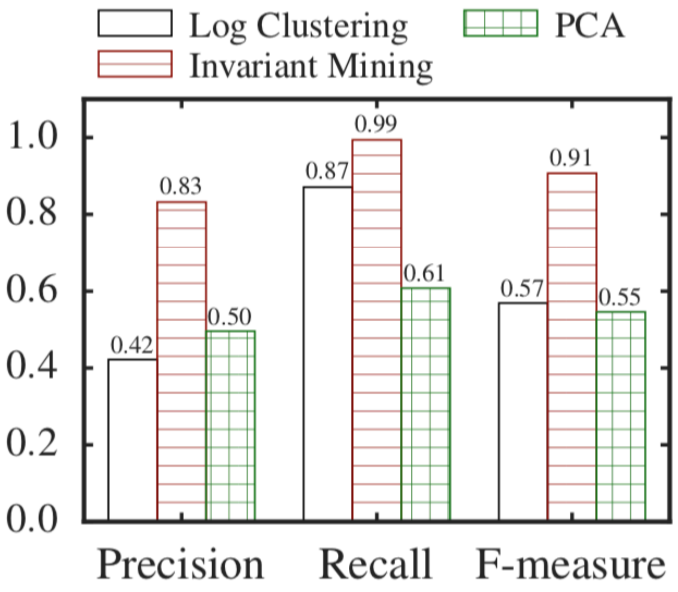
Finding 1: Supervised anomaly detection achieves **high precision,** while **recall varies.**
Finding 2: **Sliding windows** achieve higher accuracy than **fixed windows.**

Experiments

- Accuracy of unsupervised methods



HDFS

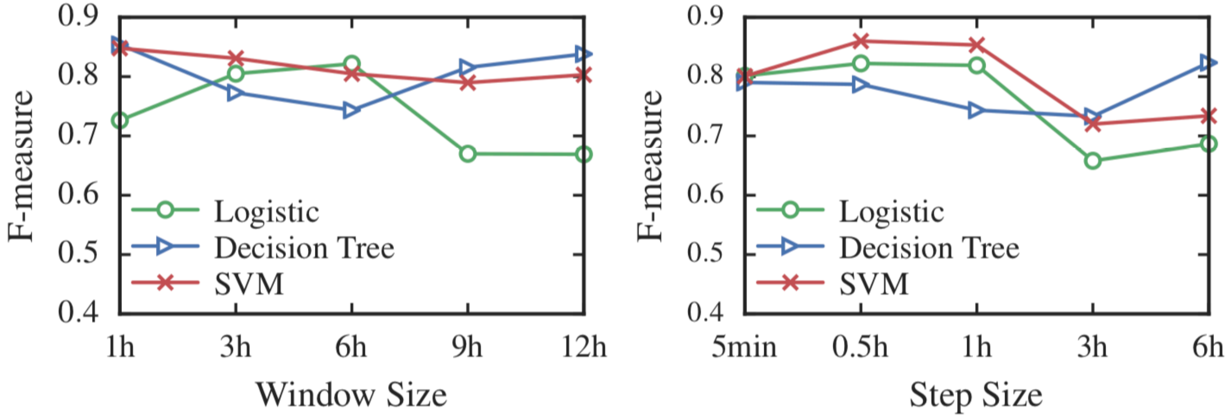


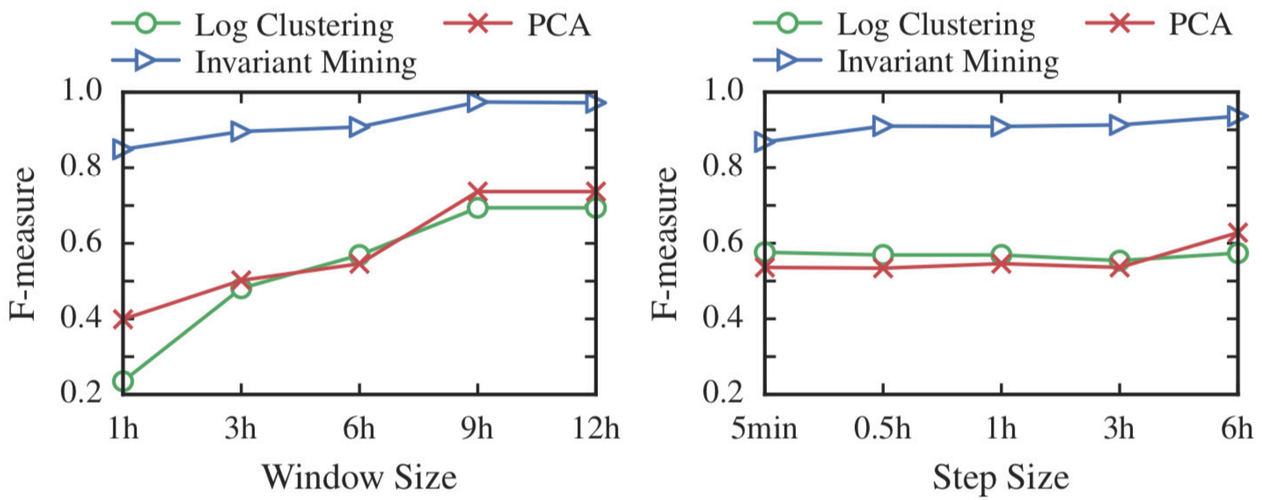
BGL (Sliding window)

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

Experiments

- Various hyper-parameters settings





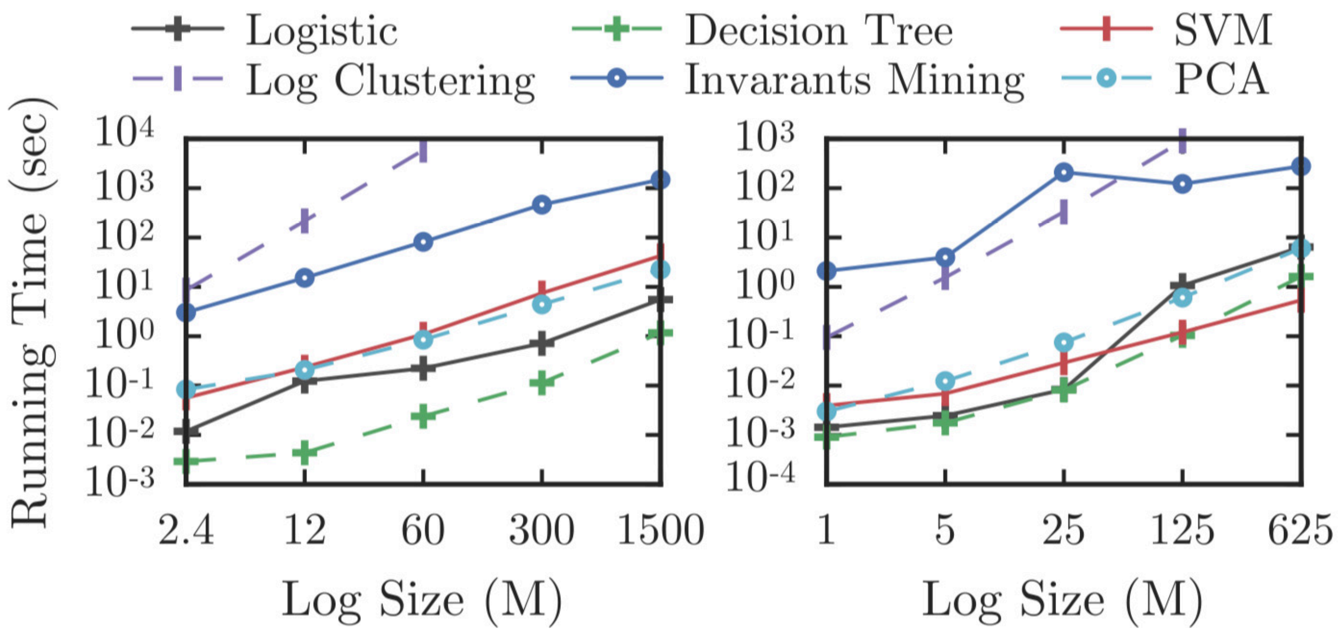
Supervised Methods

Unsupervised Methods

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

Experiments

- Efficiency

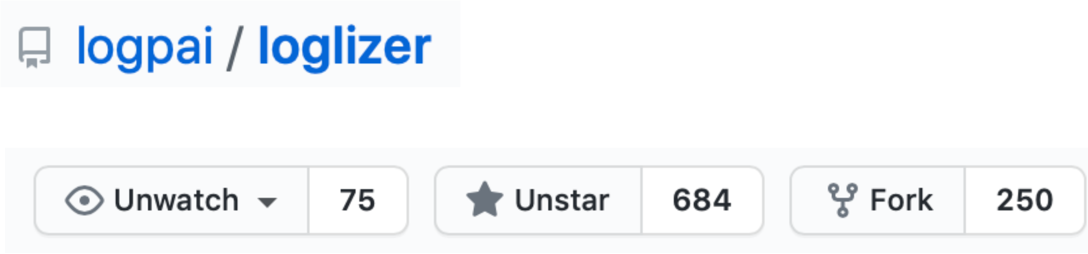


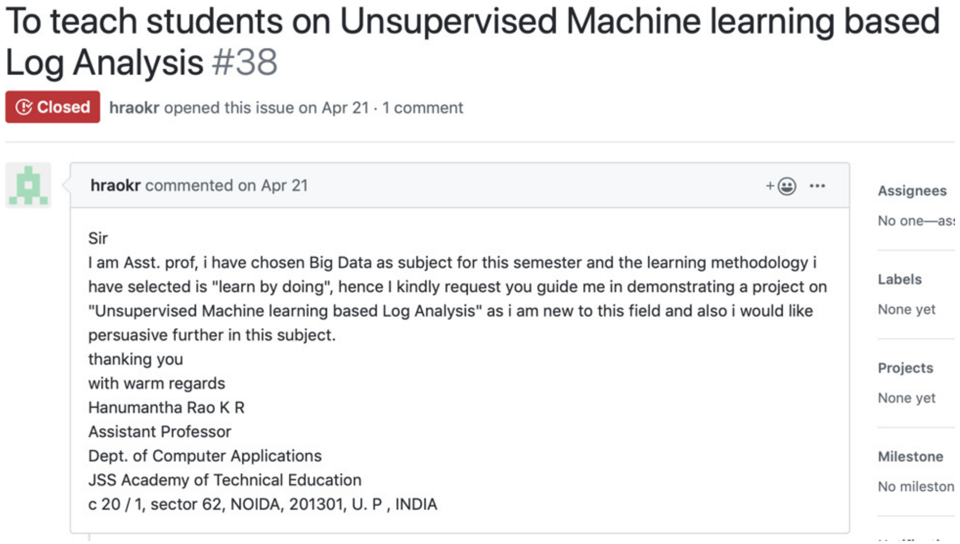
HDFS BGL (Sliding window)

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

Summary

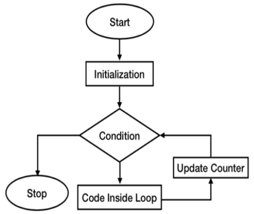
- Provide an empirical study of **six** SOTA anomaly detection methods.
- Compare their accuracy and efficiency on **two** representative log datasets.
- Release an open-source toolkit for easy reuse and further study.





Outline

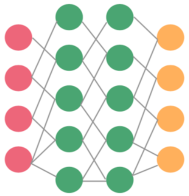
- Topic 2: Log-based Problem Identification



Interpretability-driven SRE

Traditional Software

Intelligent Software



Log-based Anomaly Detection

[Chapter 4]

Log-based Problem Identification

Gradient-based Attribution Estimation

Phrase-table-based Knowledge Assessment

Background

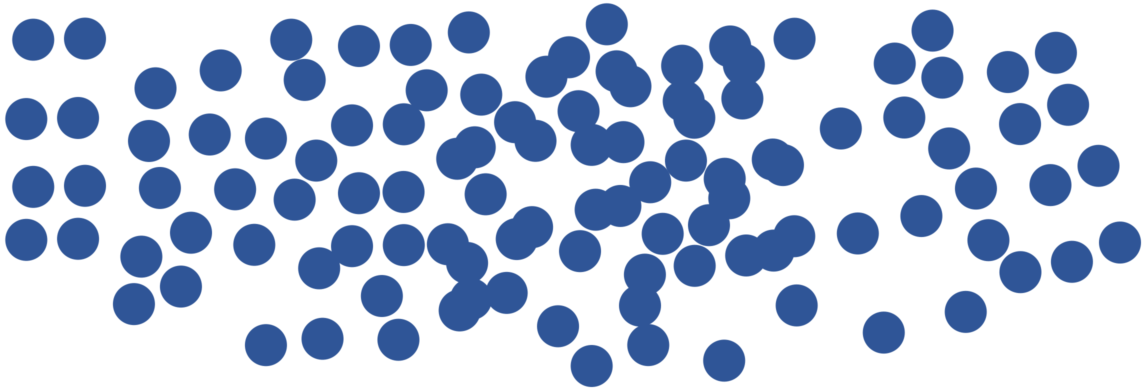
- Problem type matters
- Some types of problem are more impactful, should be fixed with a higher priority.

IMPACT / URGENCY	Widespread (Extensive)	Large (Significant)	Limited (Moderate)	Localized (Minor)
CRITICAL	CRITICAL	CRITICAL	HIGH	MEDIUM
HIGH	CRITICAL	HIGH	MEDIUM	LOW
MEDIUM	HIGH	MEDIUM	LOW	LOW
LOW	MEDIUM	LOW	LOW	LOW

Report to

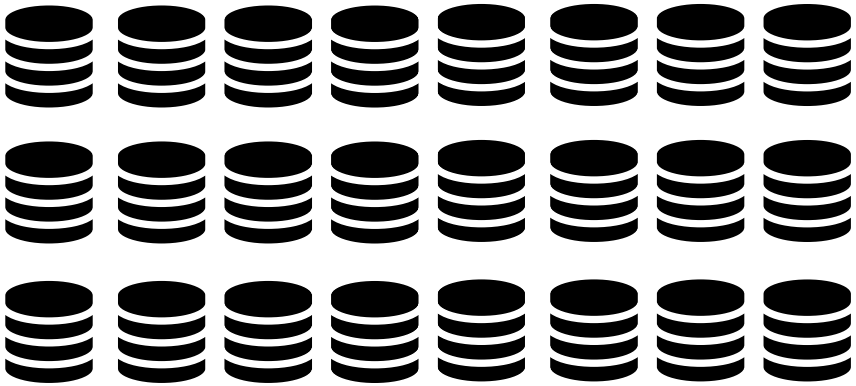
Challenges

1. Lack of labels



→ Unsupervised Methods

2. Huge log size



→ Inefficient

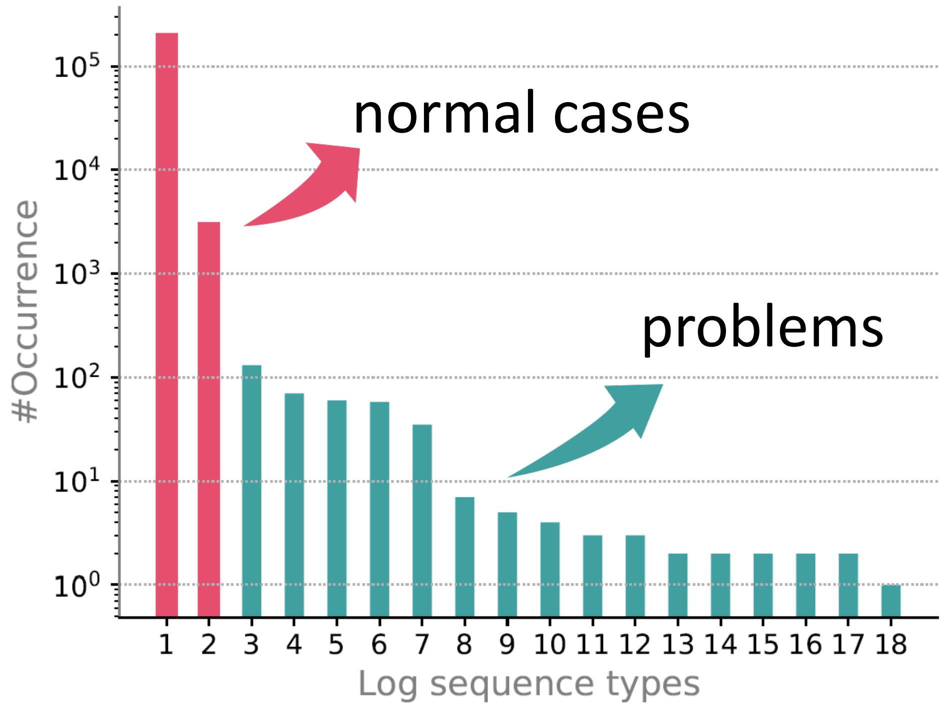
Challenges

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.

KPI

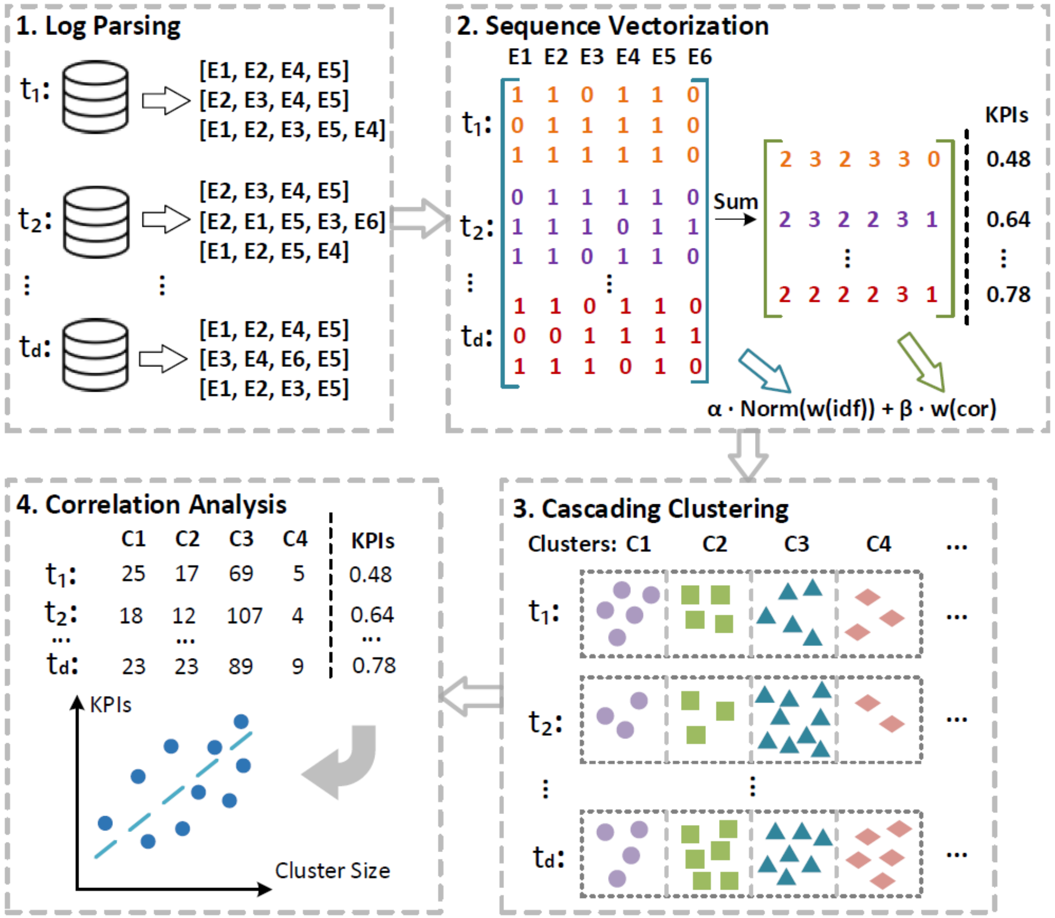
- System KPIs (Key Performance Indicators)
  - measure the system's health status in a certain time period
    - Failure Rate
    - Service Availability
    - Average Request Latency

- periodically collected

Time interval	1h	1h	1h	1h
Failure rate	0.48	0.23	0.14	0.53

Method

Log3C: **C**ascading **C**lustering and **C**orrelation Analysis



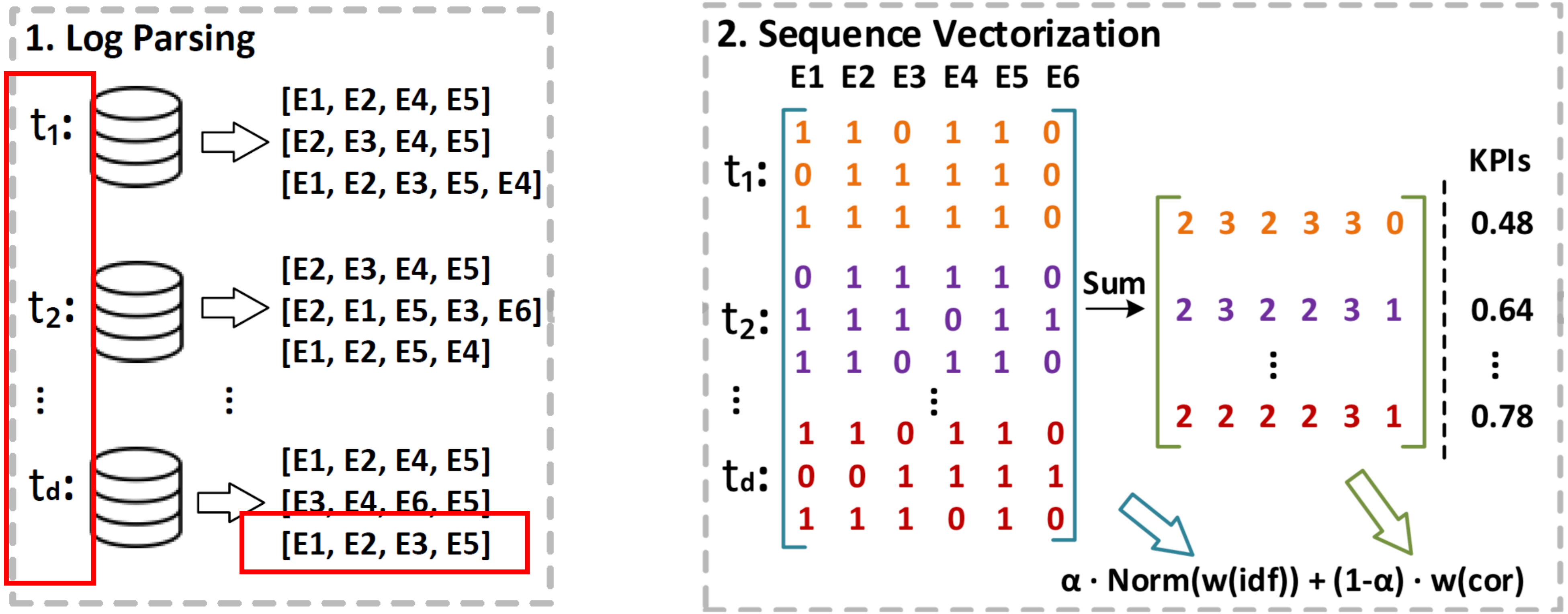
Framework of Log3C

Input: Raw logs, KPIs

Output: Clusters of impactful problems

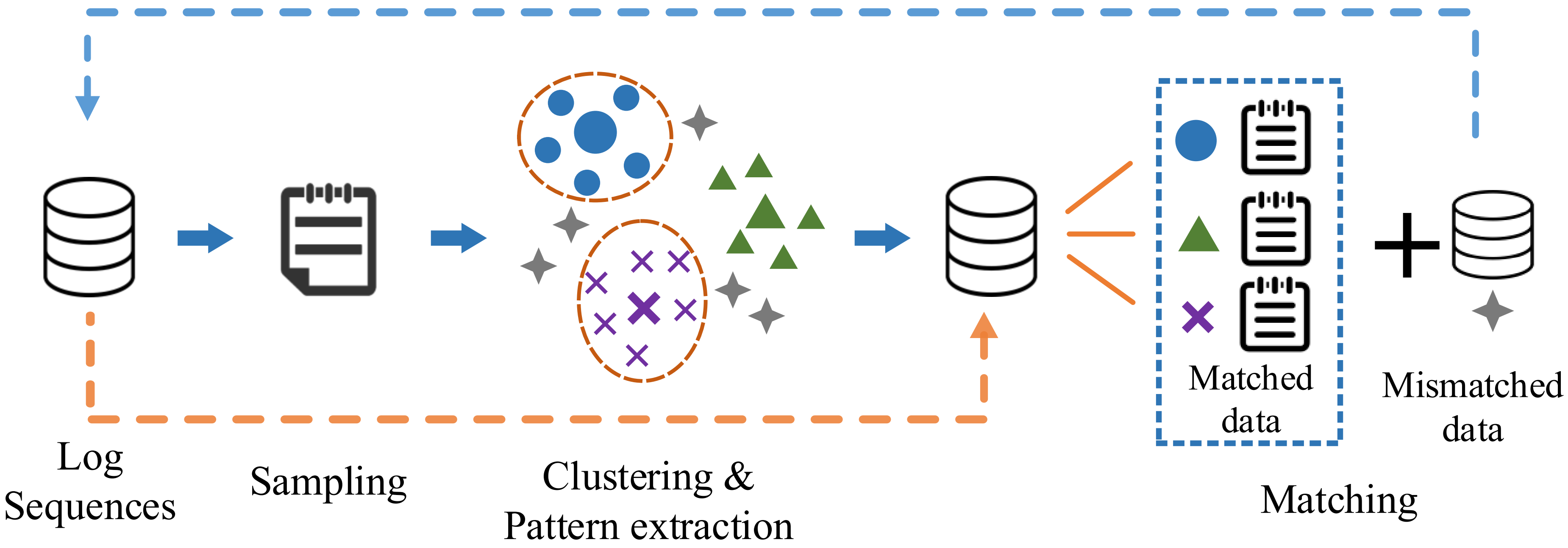
Parsing and Vectorization

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



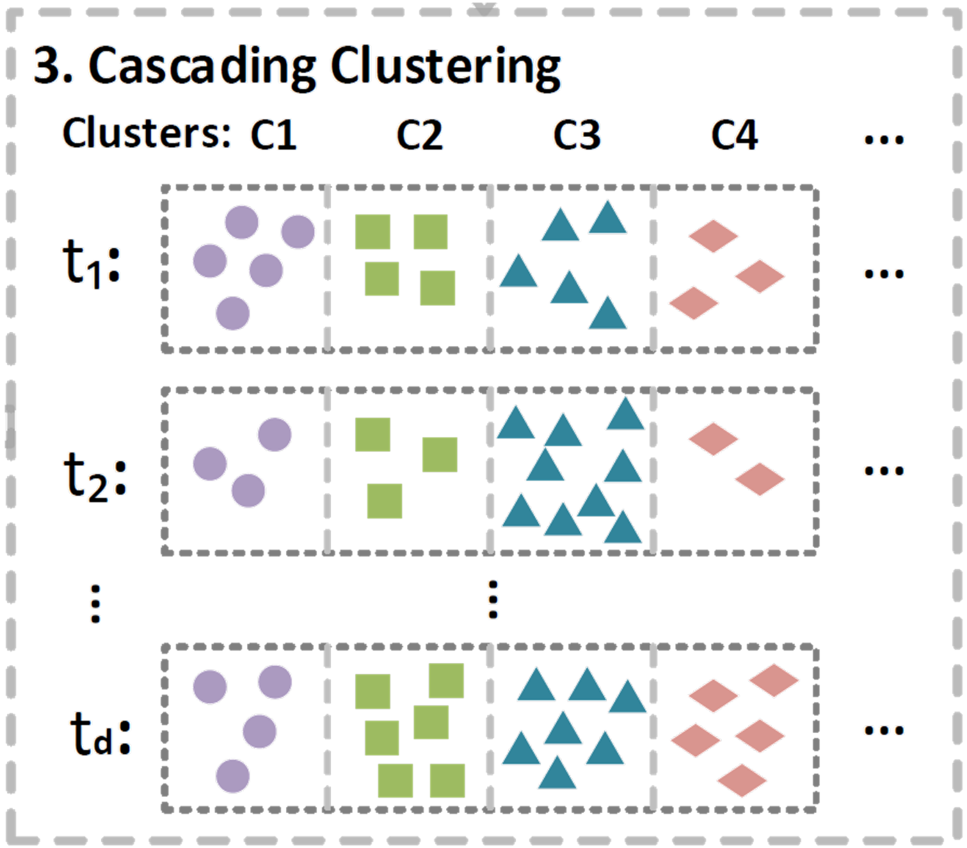
Cascading Clustering

Traditional clustering methods are infeasible.



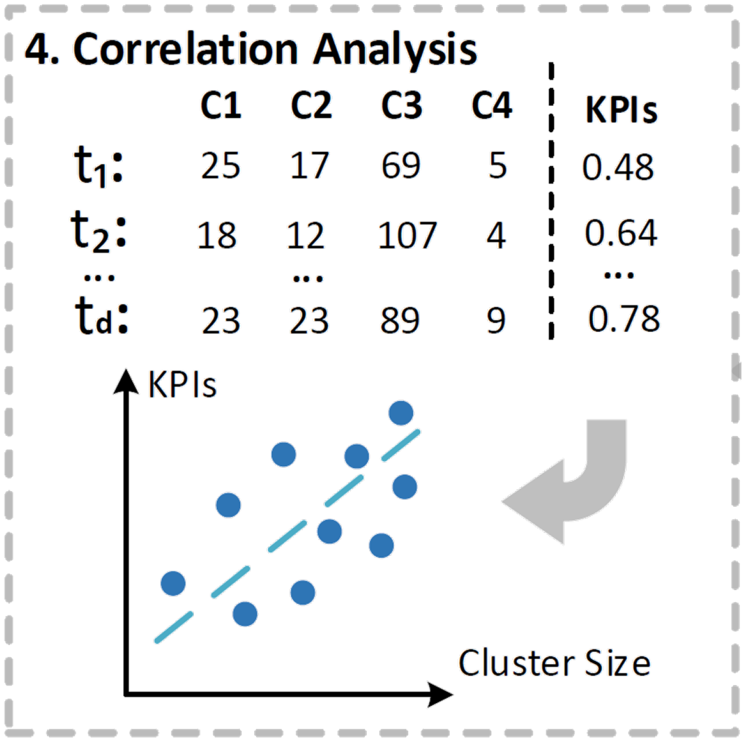
Cascading Clustering

- Group log sequences with cascading clustering in each time interval



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

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

- Manual labelling
  1. Problem or not?
  2. Problem type?

Experiments

- Effectiveness Evaluation:
  - Problem Detection (Binary Classification)

    Precision / Recall / F1-Measure

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

  - Problem Identification (Clustering)

    Normalized Mutual Information (NMI) ~ between [0, 1]

$$NMI(Y, C) = \frac{2 \times I(Y;C)}{[H(Y) + H(C)]}$$

Y = class labels

C = cluster labels

H(.) = Entropy

I(Y;C) = Mutual Information b/w Y and C

- Efficiency Evaluation:
  - Clustering Time (in seconds)

Experiments

- Accuracy of Problem Detection:

Data 1	Precision	Recall	F1-measure
PCA	0.465	0.946	0.623
Invariants Mining	0.604	1	0.753
Log3C	**0.900**	0.920	**0.910**
Data 2	Precision	Recall	F1-measure
PCA	0.142	0.834	0.242
Invariants Mining	0.160	0.847	0.269
Log3C	**0.897**	0.826	**0.860**
Data 3	Precision	Recall	F1-measure
PCA	0.207	0.922	0.338
Invariants Mining	0.168	0.704	0.271
Log3C	**0.834**	0.903	**0.868**

Experiments

- Accuracy of Problem Identification (NMI):

Data 1	Size	10k	50k	100k	200k
	Log3C-SC	0.659	0.706	0.781	0.822
	Log3C	**0.720**	**0.740**	**0.798**	**0.834**
Data 2	Size	10k	50k	100k	200k
	Log3C-SC	0.610	0.549	0.600	0.650
	Log3C	**0.624**	0.514	**0.663**	**0.715**
Data 3	Size	10k	50k	100k	180k
	Log3C-SC	0.601	0.404	0.792	0.828
	Log3C	**0.680**	**0.453**	**0.837**	**0.910**

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

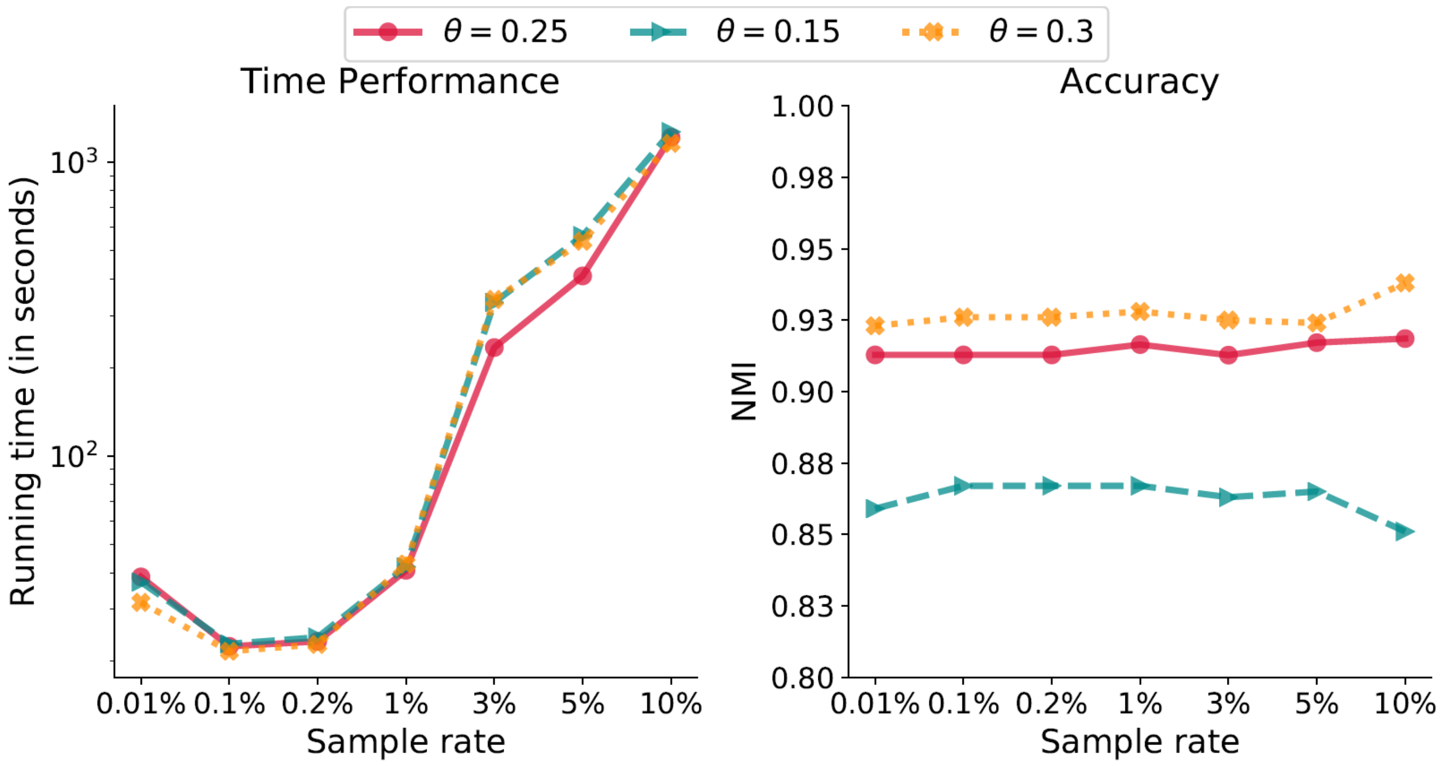
Experiments

- Efficiency of Cascading Clustering (seconds):

Data 1	Size	10k	50k	100k	200k
	SC	127.6	2319.2	9662.3	38415.5
	CC	1.0	4.3	9.2	20.7
Data 2	Size	10k	50k	100k	200k
	SC	80.6	2469.1	8641.2	38614.0
	CC	0.7	3.8	9.5	18.9
Data 3	Size	10k	50k	100k	180k
	SC	81.5	2417.2	8761.2	33728.3
	CC	0.8	4.0	8.8	18.3

Experiments

- Cascading clustering under various configurations



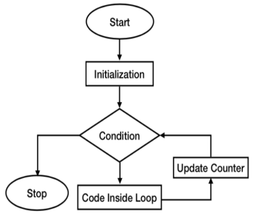
Decreasing sample rate does not sacrifice the accuracy while greatly reducing the time

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

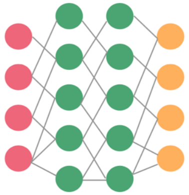
- Topic 3: Gradient-based Attribution Estimation



Interpretability-driven SRE

Traditional Software

Intelligent Software



Log-based Anomaly Detection

Log-based Problem Identification

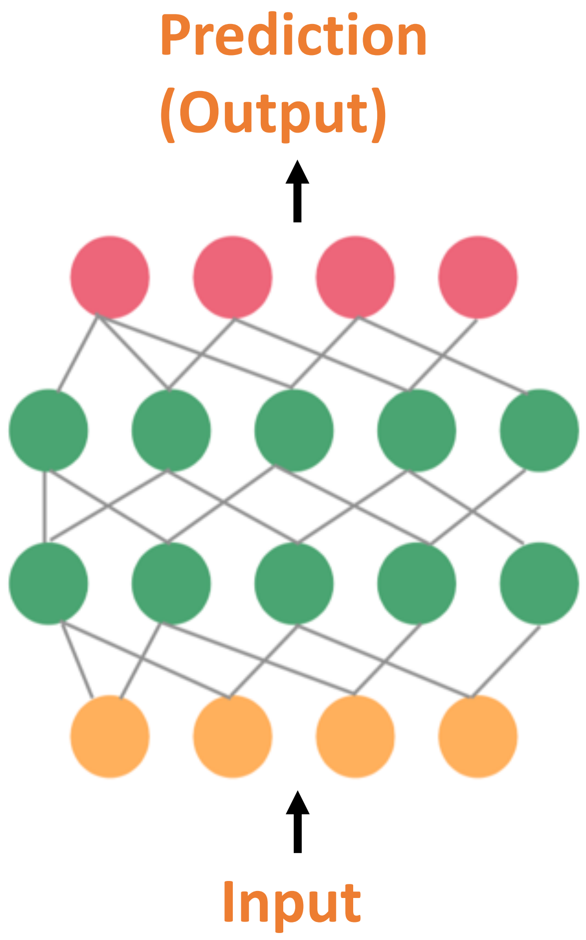
[Chapter 5]

Gradient-based Attribution Estimation

Phrase-table-based Knowledge Assessment

Background

- What is the "Log" in intelligent software?
  - Parameters?   Millions, Billions
  - Architecture?   CNN, RNN
  - **Gradient Information**



Background

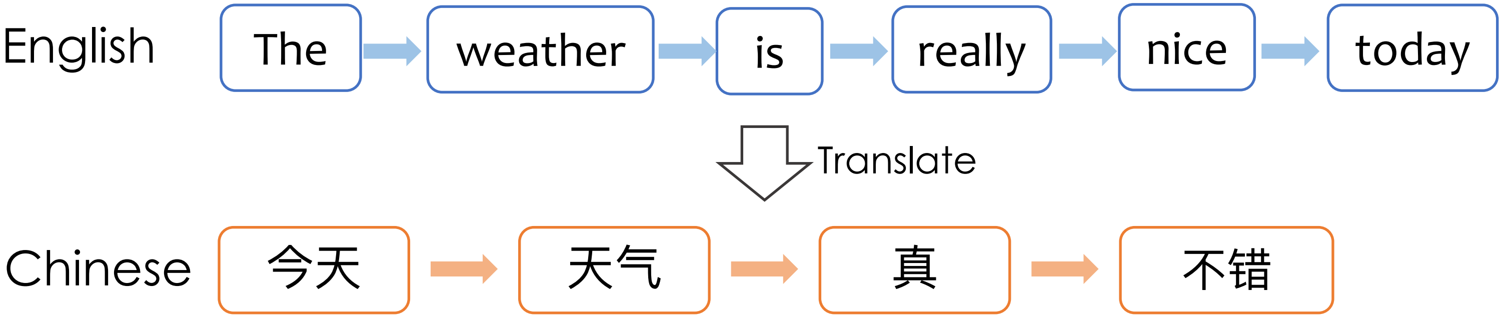
- Neural Machine Translation (NMT) as the intelligent software

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

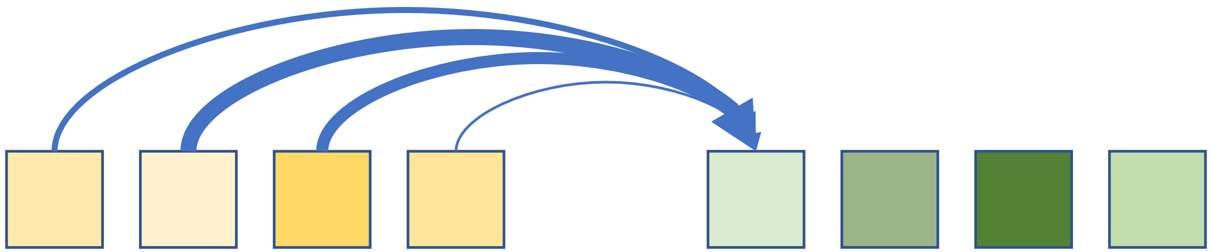
▲ Facebook's machine translation mix-up sees man questioned over innocuous post confused with attack threat. Photograph: Thibault Camus/AP

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?
  - Input-output correspondence



- **Word Importance:** the importance of each *input word* to the *output sentence*.

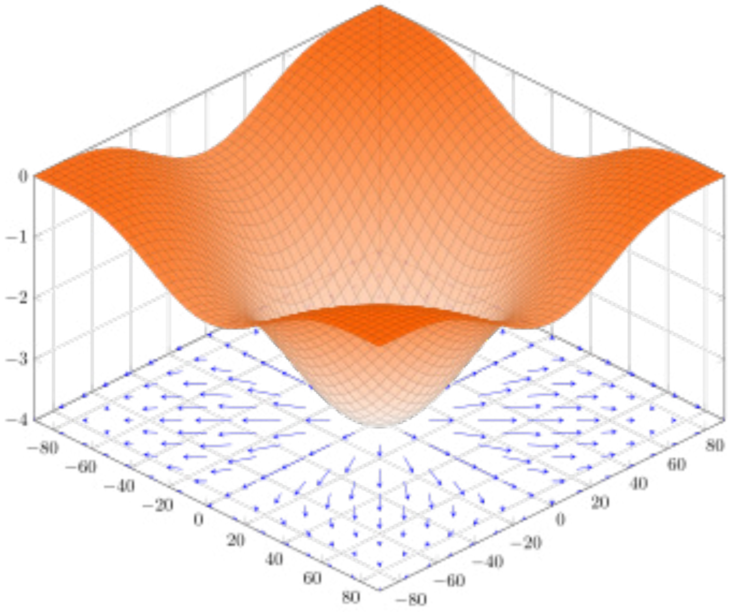
  - Also applicable in the adversarial attack and defense.

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

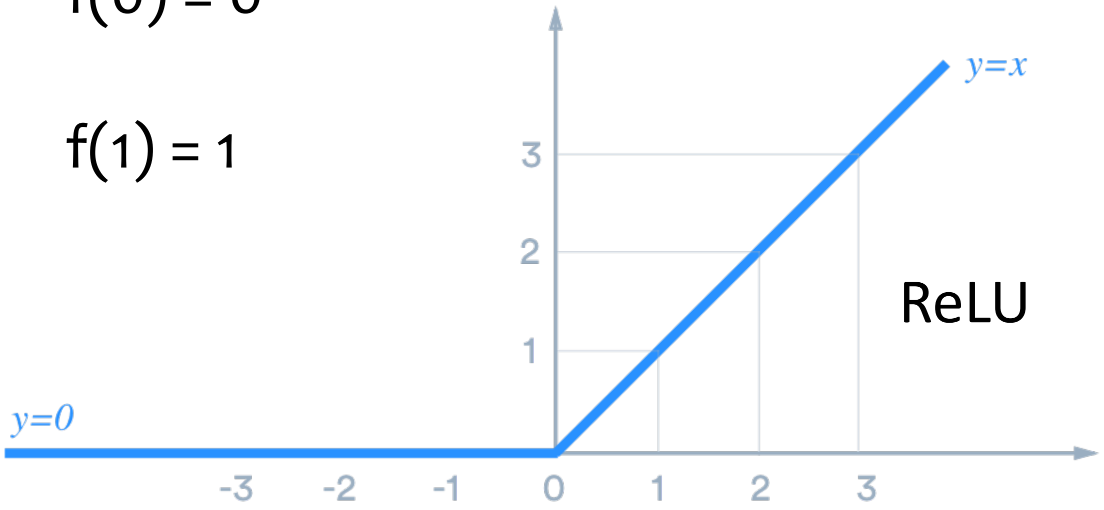


$f(x) = 1 - \text{ReLU}(1-x)$

Gradient Saturation

$f(0) = 0$

$f(1) = 1$



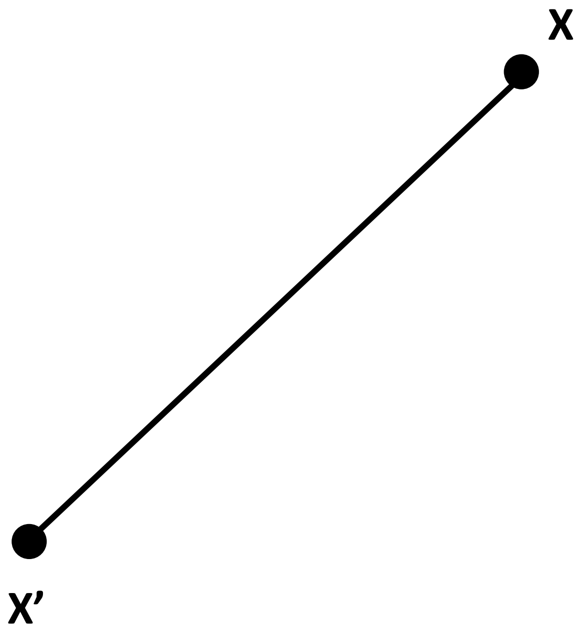
gradient is 0 since f is flat when x = 1

Method

Integrated Gradients

- Intuition: find a baseline input **x'** to calculate the *relative* feature importance in **x**

$$IG_m^n(\mathbf{x}) = (\mathbf{x}_m - \mathbf{x}'_m) \int_{\alpha=0}^{1} \frac{\partial F(\mathbf{x}' + \alpha(\mathbf{x} - \mathbf{x}'))_n}{\partial \mathbf{x}_m} d\alpha$$



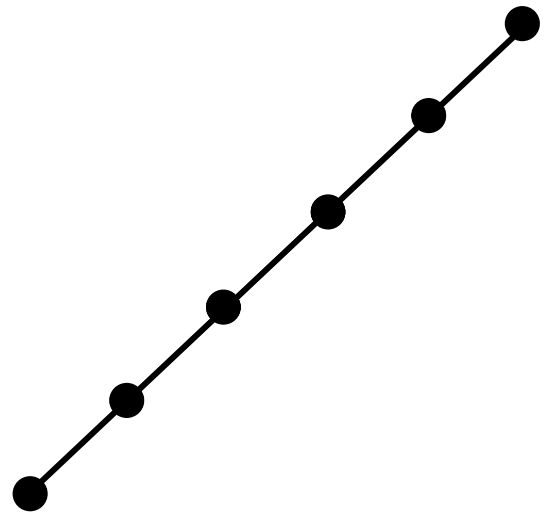
- F : the model, e.g., Transformer, RNNSearch
- m: the m-th word in the input sentence
- n: the n-th word in the output sentence
- alpha: interpolation ratio

Method

- Integrated Gradients with approximation

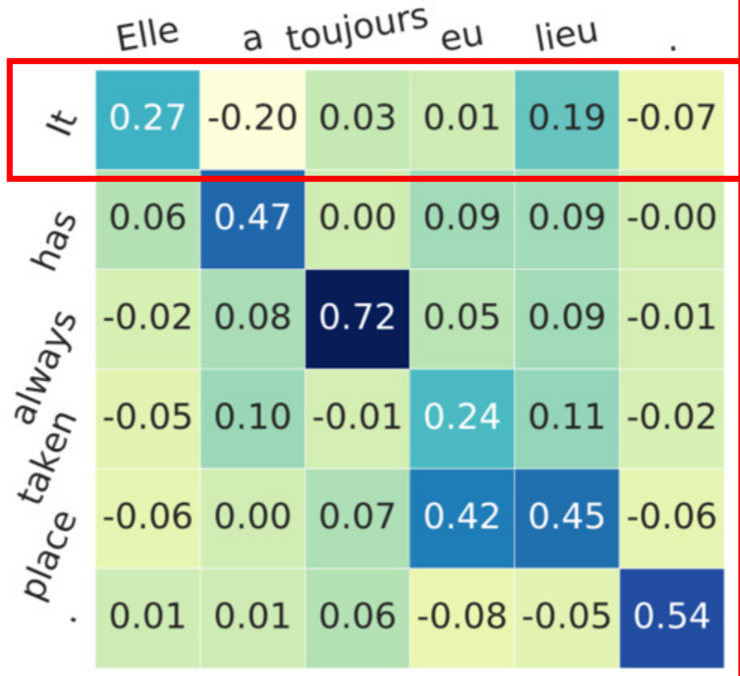
$$IG_m^n(\mathbf{x}) = \frac{(\mathbf{x}_m - \mathbf{x}'_m)}{S} \sum_{k=0}^{S} \frac{\partial F(\mathbf{x}' + \frac{k}{S}(\mathbf{x} - \mathbf{x}'))_n}{\partial \mathbf{x}_m}$$

- S: Total interpolation steps
- k: the k-th interpolation step



Method

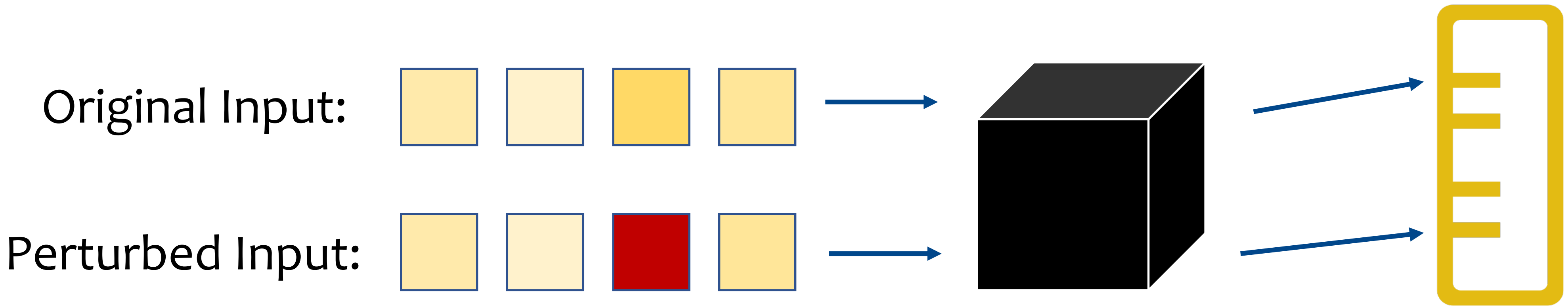
- Word Importance:
  - Step 1: Estimate the integrated gradient of each word pair;
  - Step 2: Sum the contribution of an input word to all output words;
  - Step 3: Normalize with the Softmax function.



(b)

Evaluation Metric

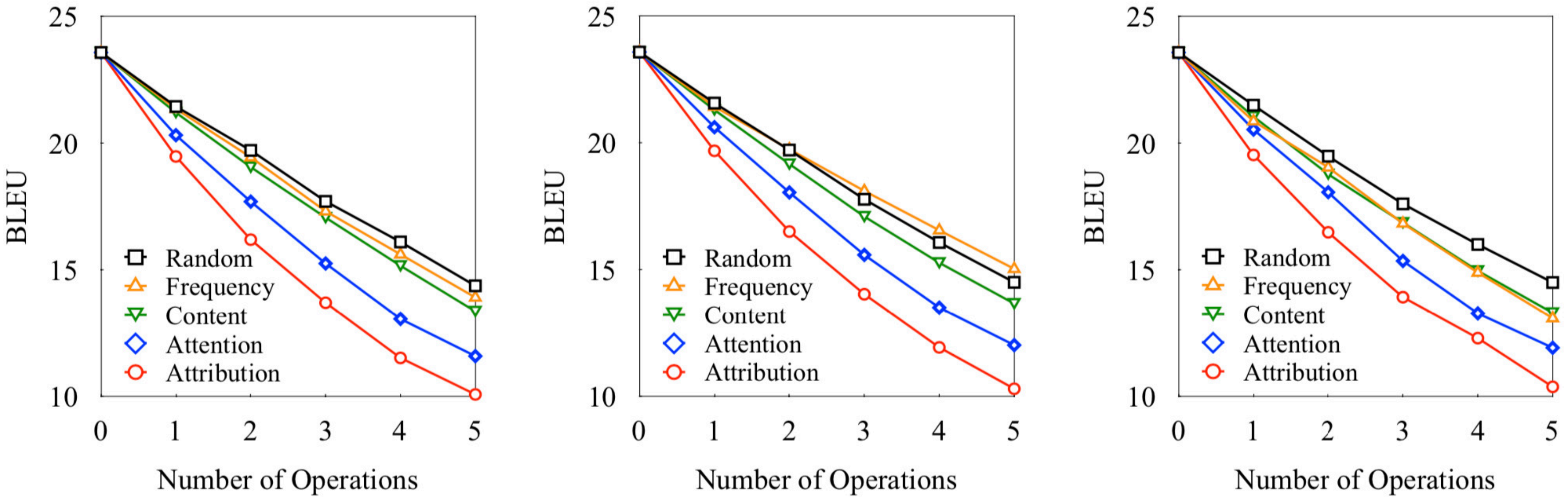
- Translation performance when perturbing the most important words



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

Experiments

- Effectiveness of different word importance estimation methods.

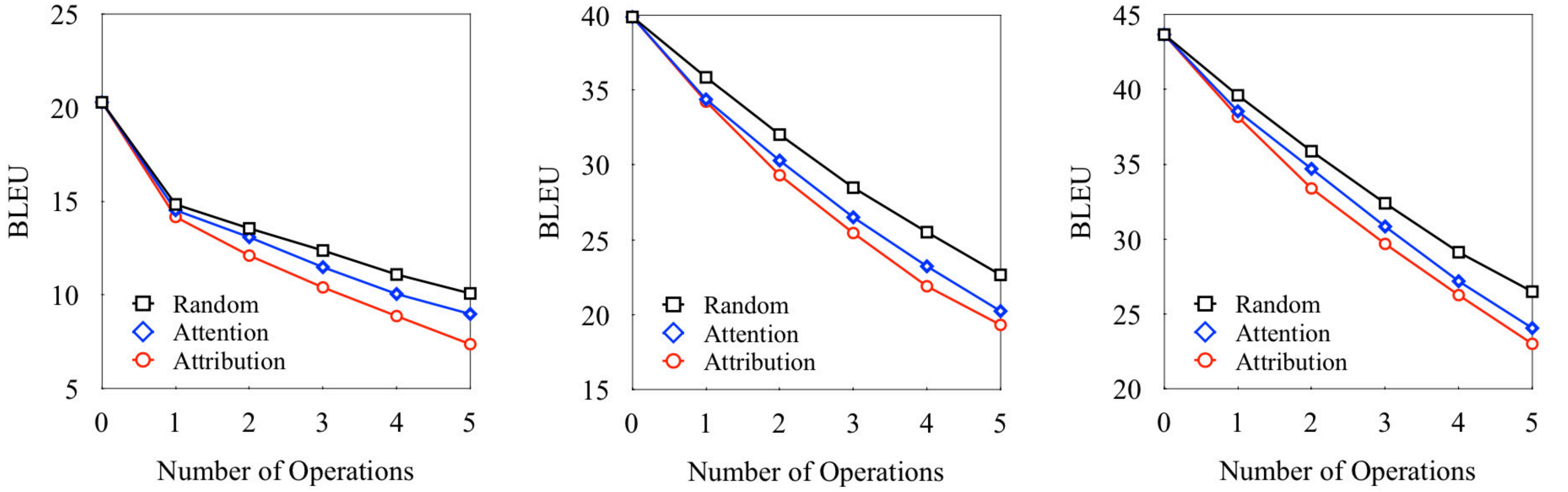


(a) **Deletion** (b) **Mask** (c) **Grammatical Replacement**

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.

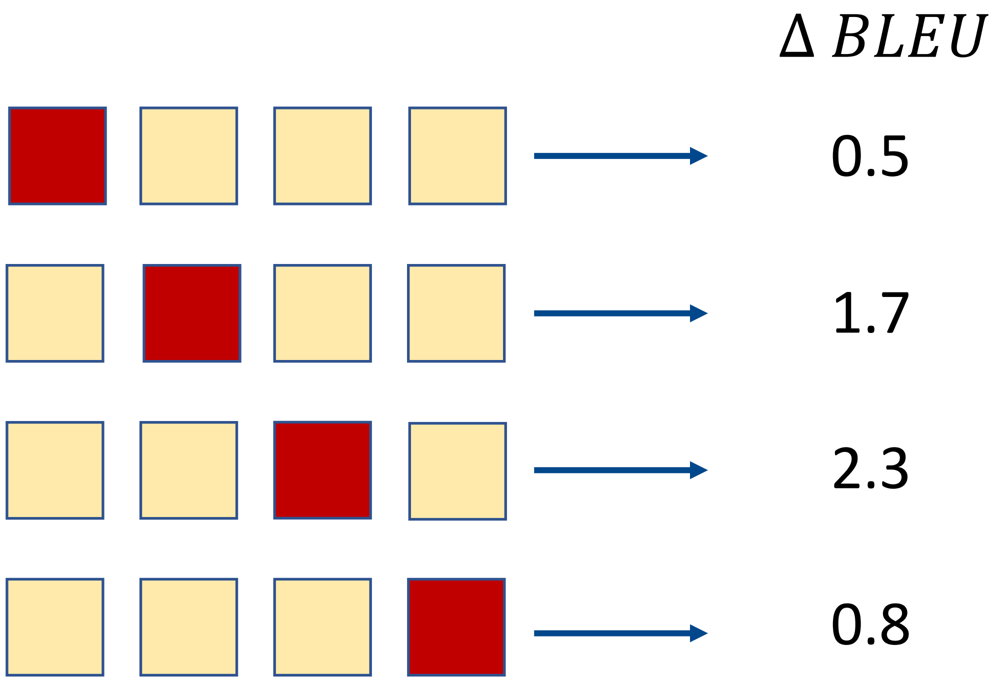


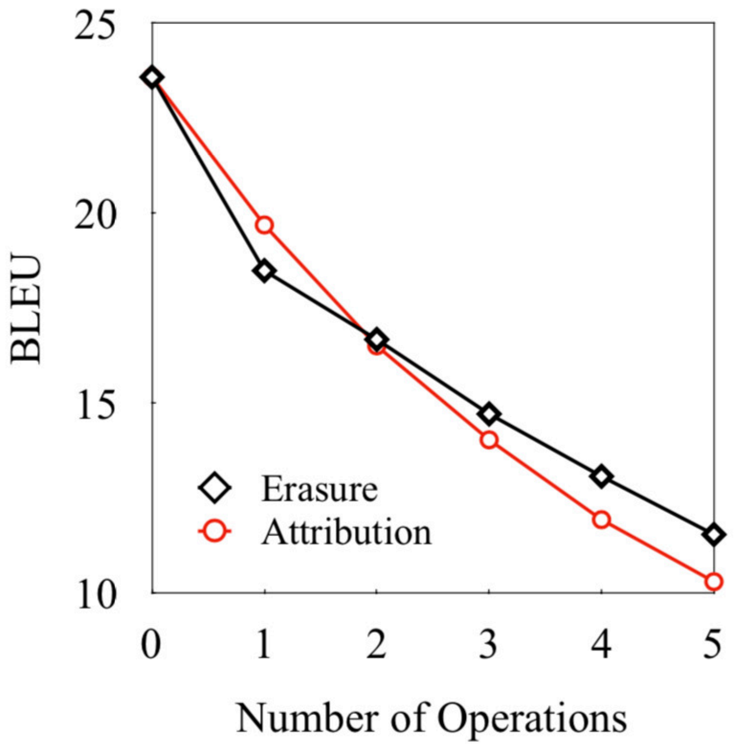
(a) **RNN-Search Model** (b) **English⇒French** (c) **English⇒Japanese**

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

Experiments

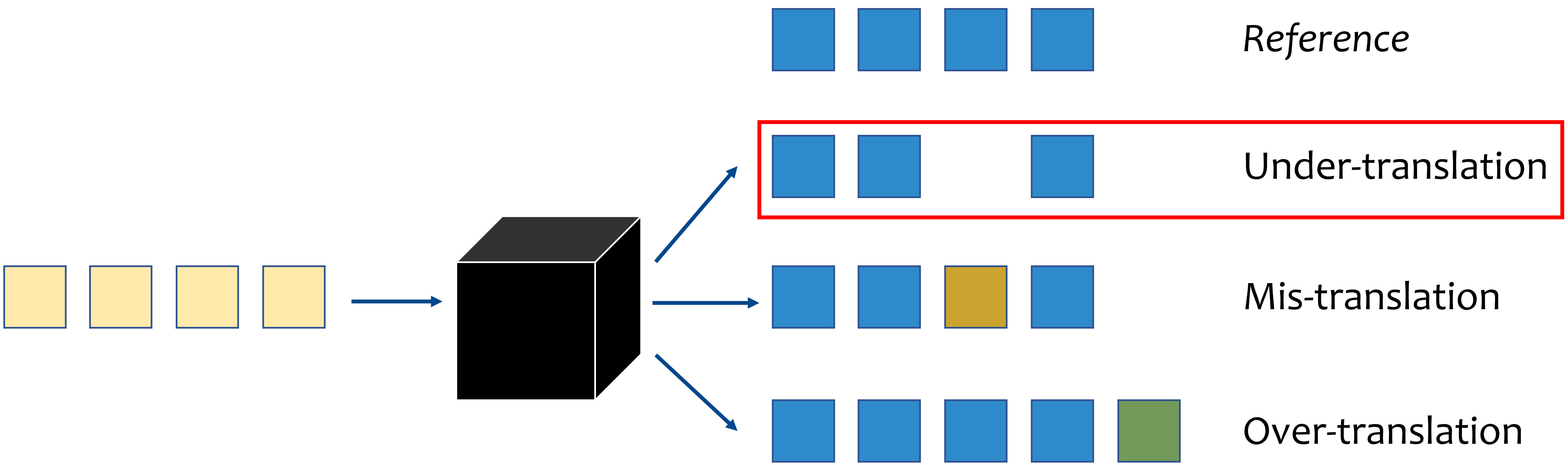
- Comparison with the supervised erasure method.
- Erasure:
  - Estimate the word importance by perturbing each word one by one and calculate the performance drop





Experiments

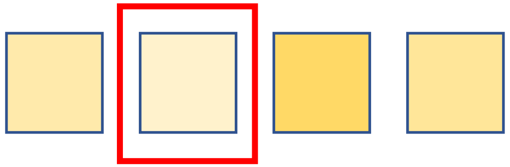
- Machine translation problems



Experiments

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

Original Input:



Method	Top 5%	Top 10%	Top 15%
Attention	0.058	0.077	0.119
Erasure	0.154	0.170	0.192
Attribution	**0.248**	**0.316**	**0.342**

F1-measure

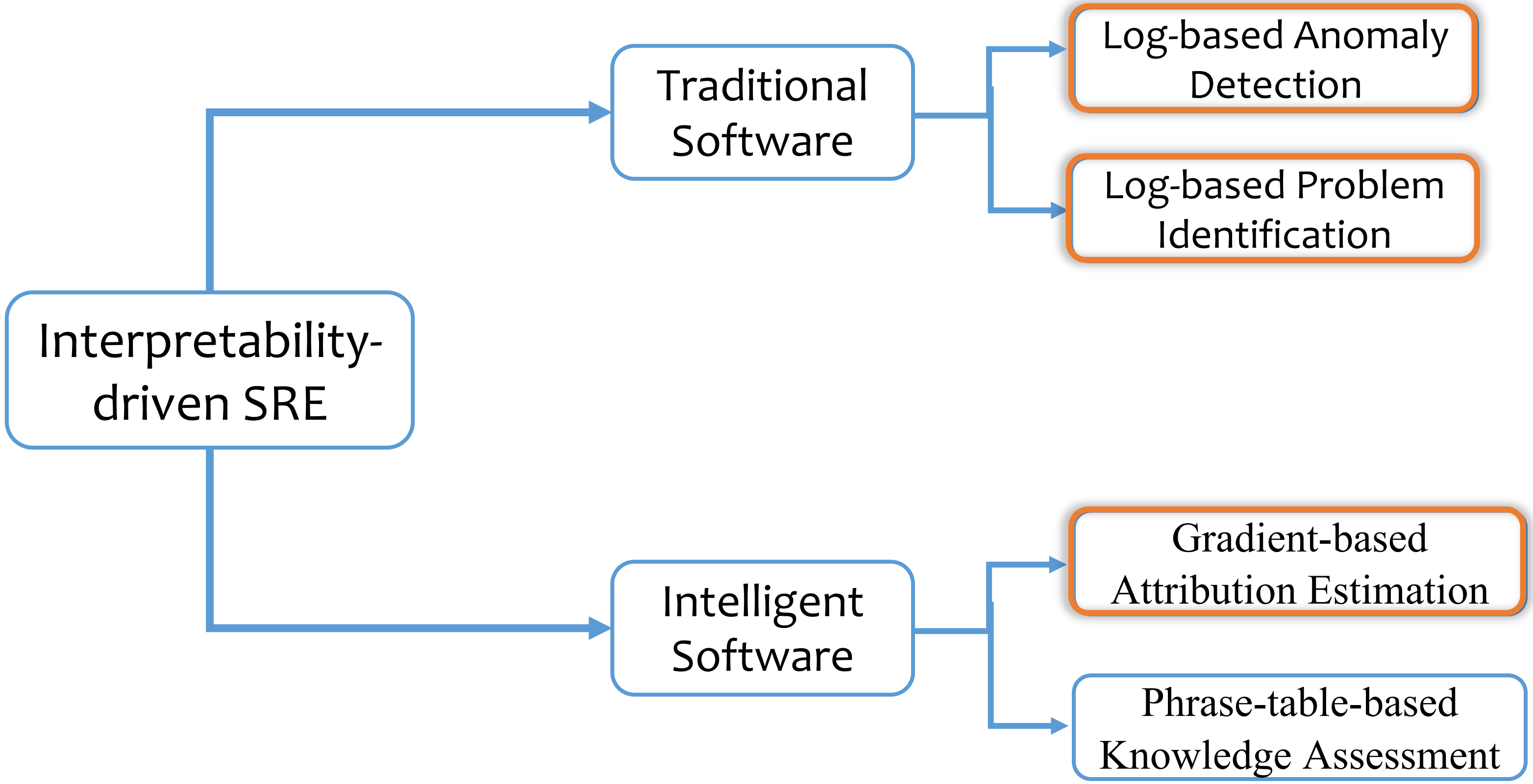
Summary

- 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



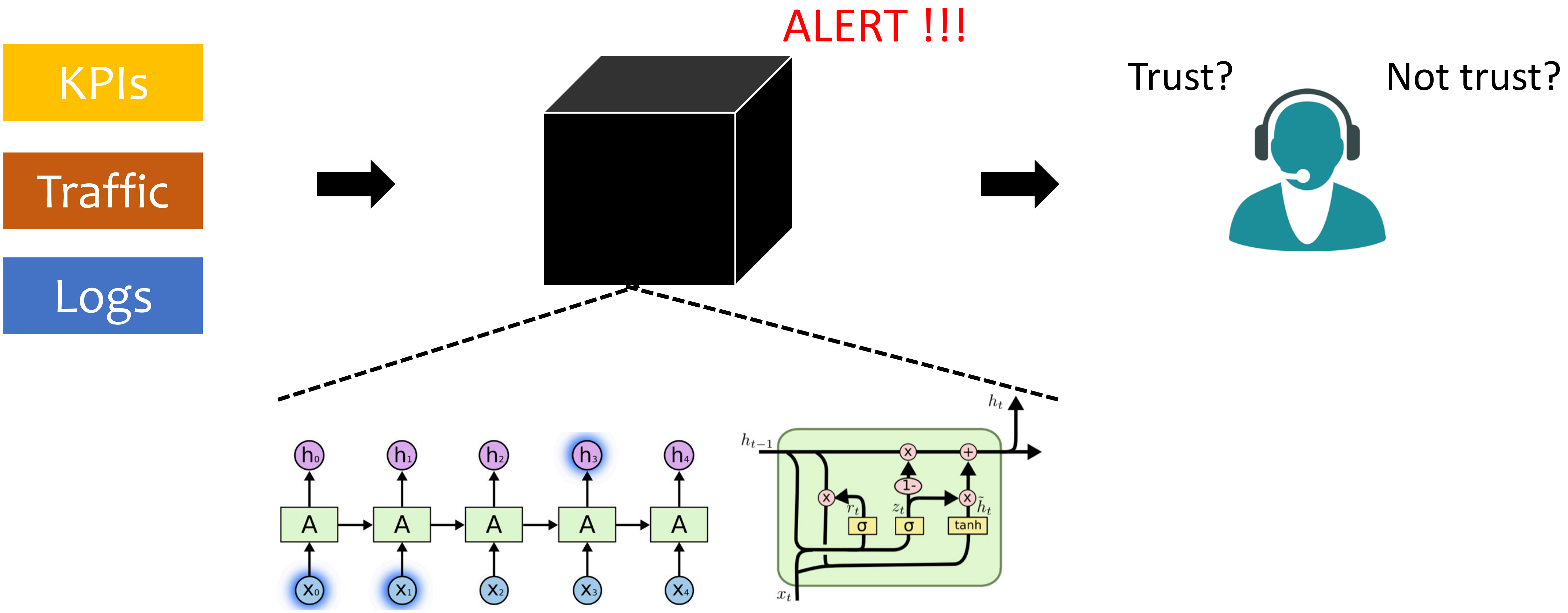
- The first empirical study
- Release a toolkit for reuse

- Highly imbalanced data w/o labels
- Cascading clustering and Correlation with KPI

- Gradient information for word importance
- Detect translation errors

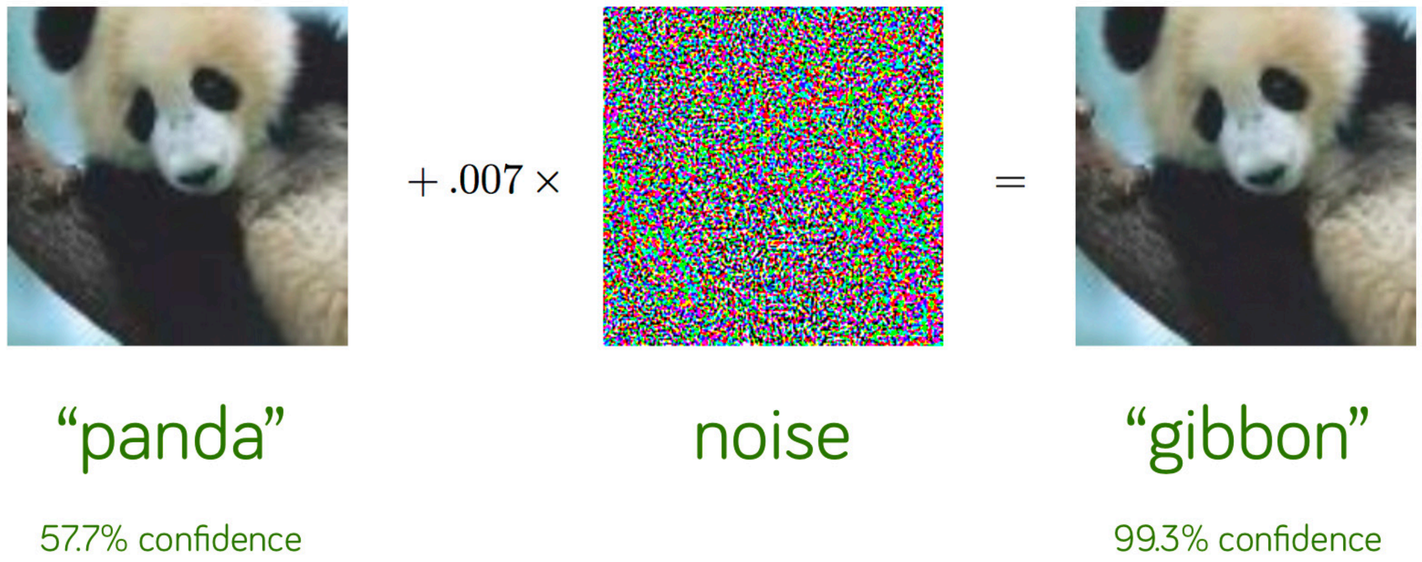
Future Work

- Interpretable automated log analysis



Future Work

- Robustness of Intelligent Software



Publications

[1] **Shilin He**, Xing Wang, Shuming Shi, Michael R. Lyu, Zhaopeng Tu. *Assessing the Bilingual Knowledge Learned by Neural Machine Translation Models.* (EMNLP 2020) *

[2] **Shilin He**, Yongchang Hao, Xing Wang, Shuming Shi, Michael R. Lyu, Zhaopeng Tu. *Multi-Task Learning with Auxiliary Autoregressive Decoder for Non-Autoregressive Machine Translation.* (EMNLP 2020) *

[3] **Shilin He**, Jieming Zhu, Pinjia He, Michael R. Lyu. *Loghub: A Large Collection of System Log Datasets towards Automated Log Analytics* (Arxiv 2020)

[4] **Shilin He**, Zhaopeng Tu, Xing Wang, Longyue Wang, Michael R. Lyu, Shuming Shi. *Towards Understanding Neural Machine Translation with Word Importance.* (EMNLP 2019)

[5] **Shilin He**, Qingwei Lin, Jianguang Lou, Hongyu Zhang, Michael R. Lyu, Dongmei Zhang. *Identifying Impactful Service System Problems via Log Analysis.* (ESEC/FSE 2018)

[6] **Shilin He**, Jieming Zhu, Pinjia He, Michael R. Lyu. *Experience Report: System Log Analysis for Anomaly Detection.* (ISSRE2016)

* denotes in submission

Publications

[7] Jinyang Liu, Jieming Zhu, **Shilin He**, Pinjia He, Zibin Zheng, Michael R. Lyu. *Logzip: Extracting Hidden Structures via Iterative Clustering for Execution Log Compression.* (ASE 2019)

[8] Jieming Zhu, **Shilin He**, Jinyang Liu, Pinjia He, Qi Xie, Zibin Zheng, Michael R. Lyu. *Tools and Benchmarks for Automated Log Parsing*. (ICSE 2019)

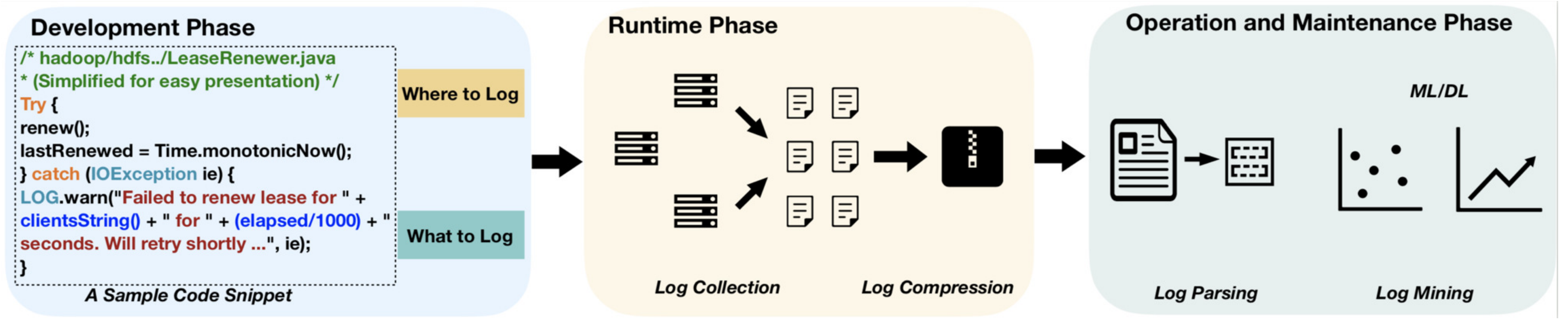
[9] Pinjia He, Zhuangbin Chen, **Shilin He**, Michael R. Lyu. *Characterizing the Natural Language Descriptions in Software Logging Statements*. (ASE 2018)

[10] Pinjia He, Jieming Zhu, **Shilin He**, Jian Li, Michael R. Lyu. *Towards Automated Log Parsing for Large-Scale Log Data Analysis*. IEEE Transactions on Dependable and Secure Computing (TDSC 2017)

[11] Pinjia He, Jieming Zhu, **Shilin He**, Jian Li, Michael R. Lyu. *An Evaluation Study on Log Parsing and Its Use in Log Mining*. (DSN 2016)

Intelligent Log Analysis

- LogPAI (Log analytics power by AI)



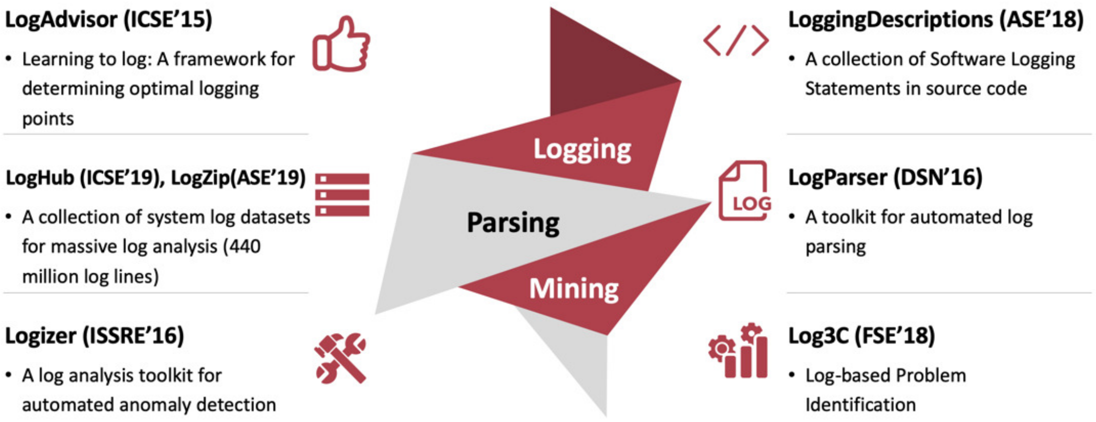
LoggingStatements	Loghub	Logzip	Logparser	Loglizer & Log3C
[ASE' 2018]	[Arxiv' 2020]	[ASE' 2019]	[ICSE' 2019] [TDSC' 2017] [DSN' 2016]	[ISSRE' 2016] [FSE' 2018]

Open-Source Projects

- LogPAI on GitHub

LOGPAI

— Log Analytics Powered by AI



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

14,596	16,846
views	downloads

Thanks!

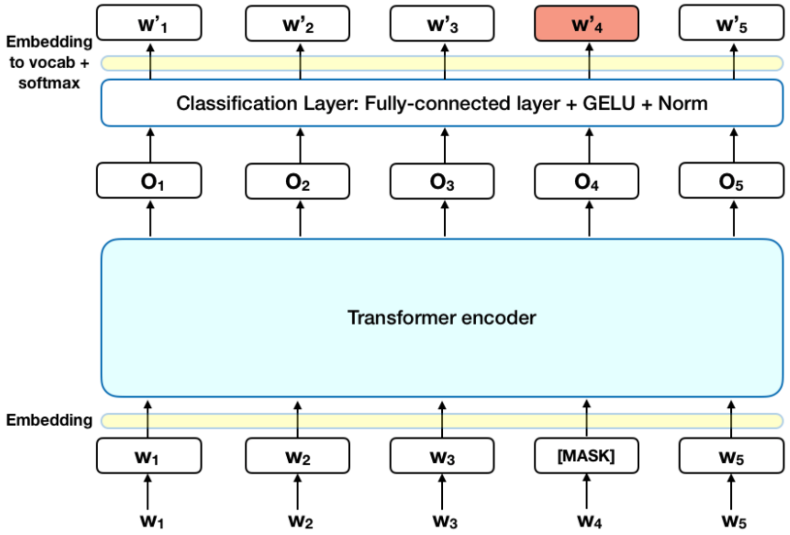
Back up slides

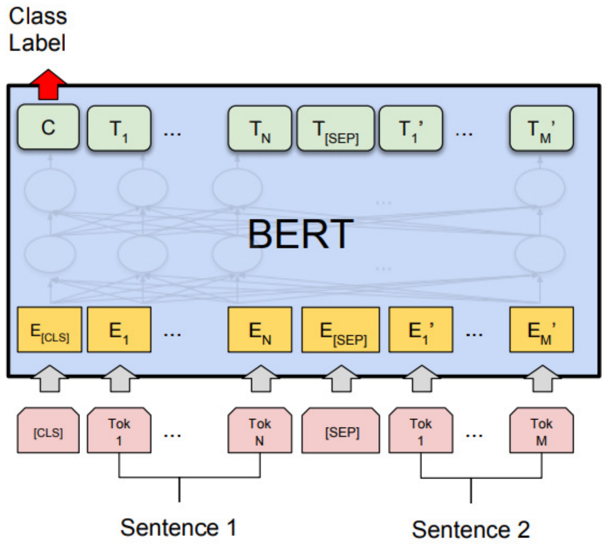
Software Reliability is Challenging

- Intelligent Software Complexity
  - **BERT** (Google):

    base: 110 million parameters with 12 layers and 12 attention heads

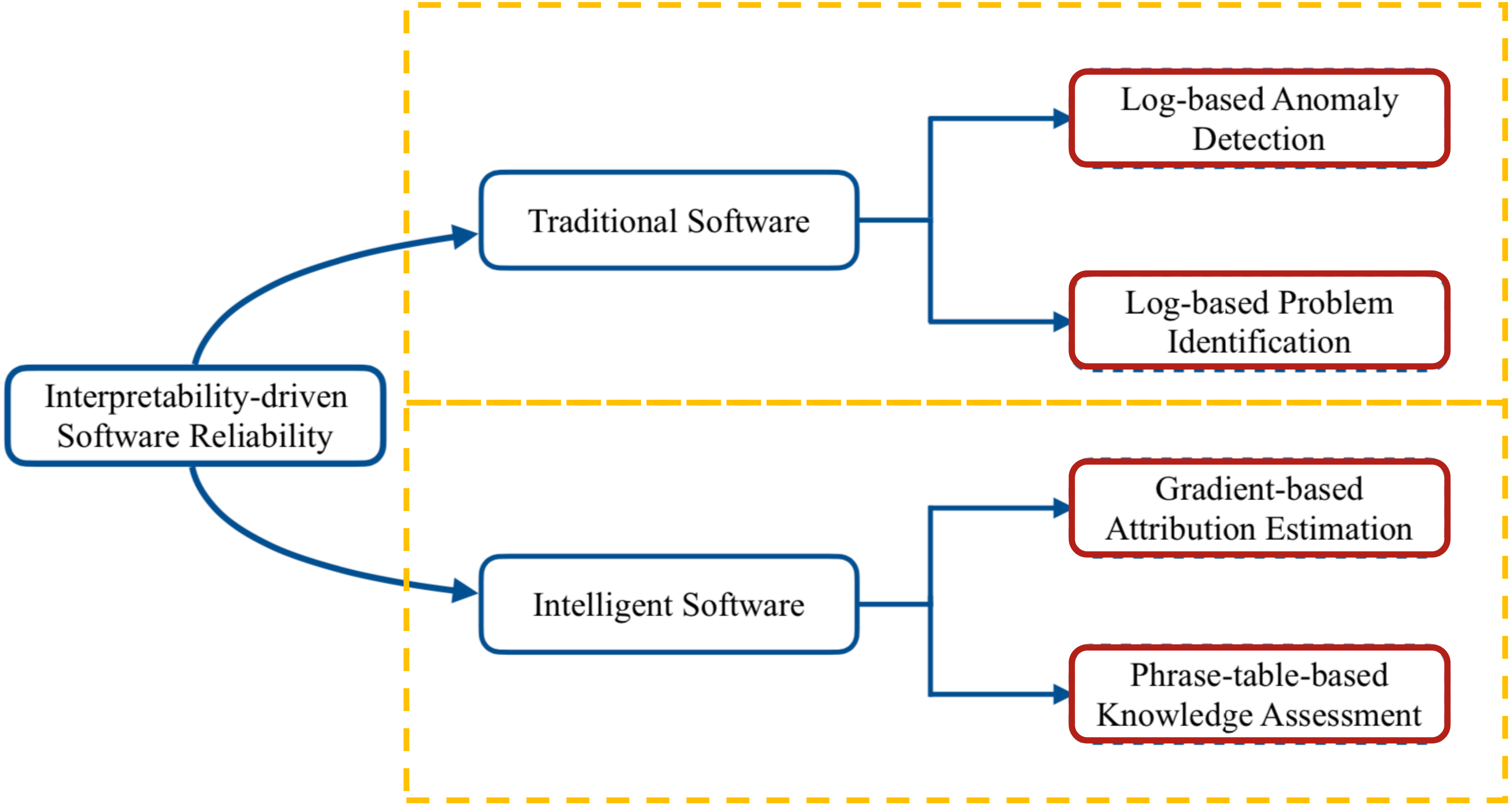
    large: 340 million parameters with 24 layers and 16 attention heads





  - **T5** (Google): 11 billion parameters
  - **GPT-3** (OpenAI): 175 billion parameters

An Overview



Interpretability-driven Software Reliability
Traditional Software
Intelligent Software
Log-based Anomaly Detection
Log-based Problem Identification
Gradient-based Attribution Estimation
Phrase-table-based Knowledge Assessment

Intelligent Log Analysis

- Log Generation

Source Code Snippet

```
/* 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

[1] *2015-10-18 18:05:48,680* WARN [LeaseRenewer:service@clusters:9000] org.apache.hadoop.hdfs.LeaseRenewer: Failed to renew lease for [DFSClient_NONMAPREDUCE_1537864556_1] for 51 seconds. Will retry shortly …

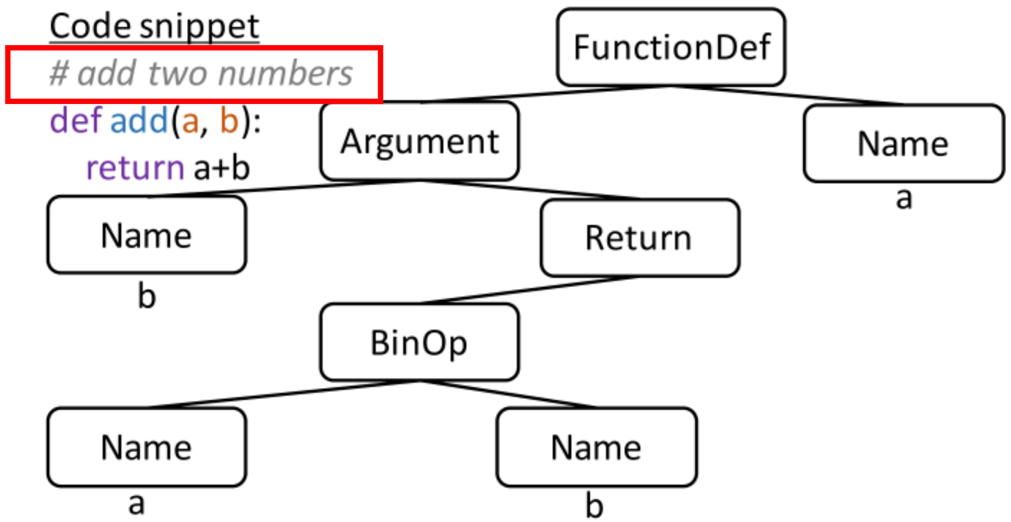
[2] *2015-10-18 18:05:51,180* WARN [LeaseRenewer:service@clusters:9000] org.apache.hadoop.hdfs.LeaseRenewer: Failed to renew lease for [DFSClient_NONMAPREDUCE_-274751412_1] for 79 seconds.

[3] *2015-10-18 21:51:51,181* WARN [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



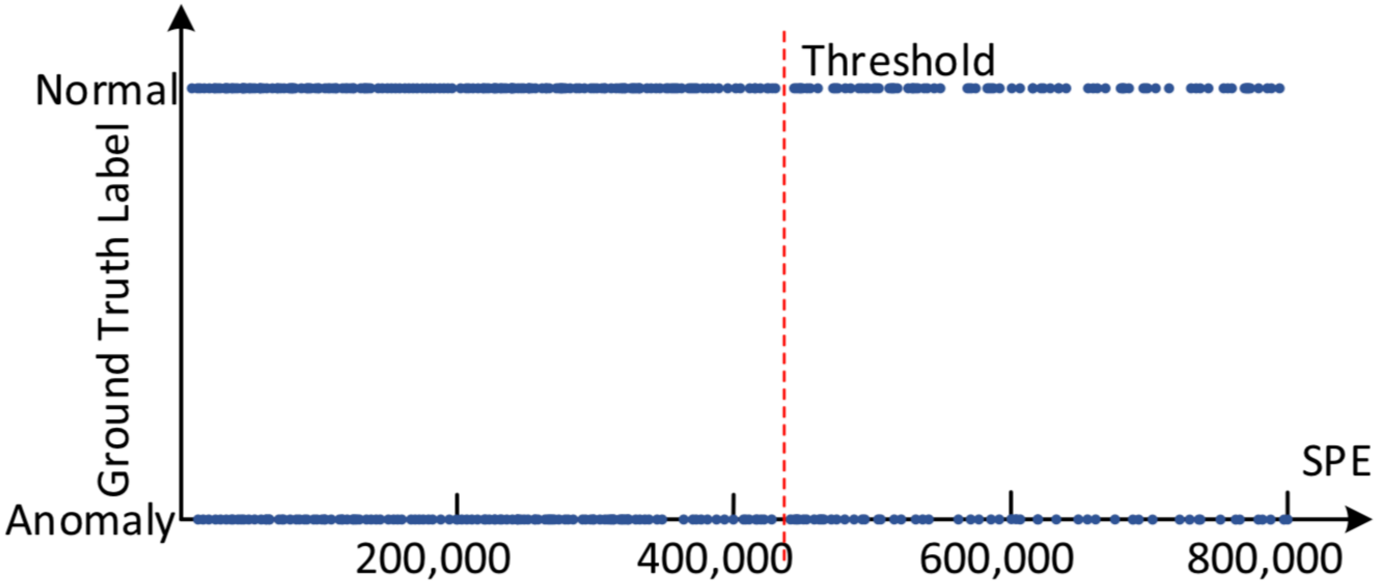
- sometimes referred as "Program Analysis", "Program Comprehension", "Program Understanding"

Background

- Interpretability is approached from the following aspects:
  - Input-Output Attribution
  - Internal Representations
  - Data Point Attribution

Experiments

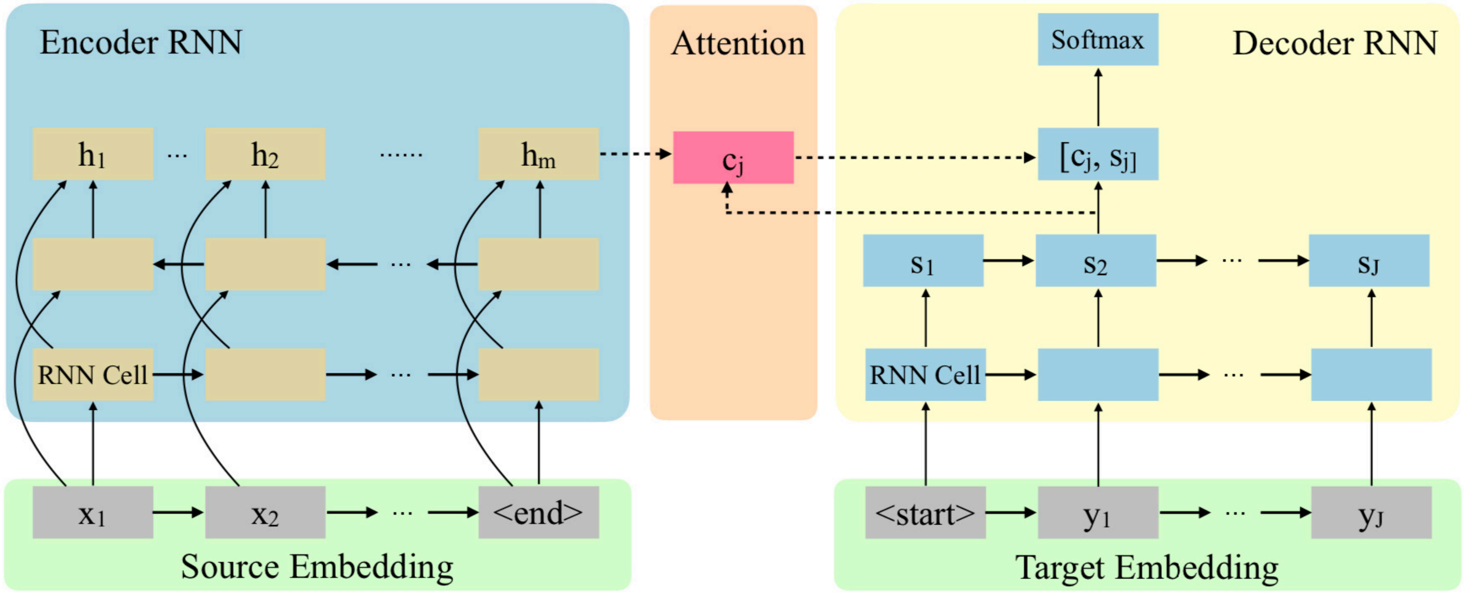
- Why PCA does not perform well on BGL?

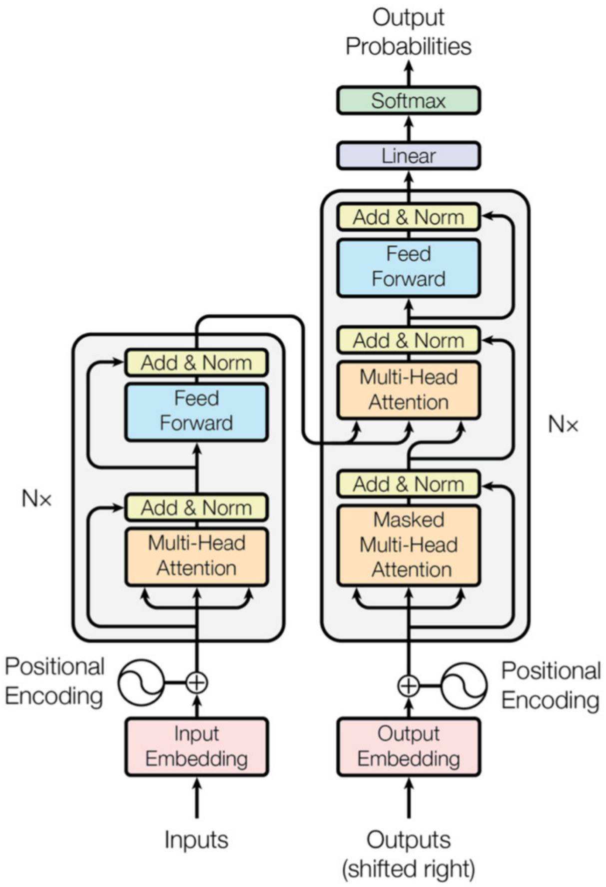


The BGL data distribution after PCA projection, normal cases and anomalies are not separable

Background

- NMT model structures





Experiments

- Linguistic Analysis on important words
  - POS Tag

Type		Chinese⇒English			English⇒French			English⇒Japanese		
		Count	Attri.	△	Count	Attri.	△	Count	Attri.	△
Content	Noun	0.383	0.407	*+6.27%*	0.341	0.355	*+4.11%*	0.365	0.336	*-7.95%*
	Verb	0.165	0.160	*-3.03%*	0.146	0.131	*-10.27%*	0.127	0.123	*-3.15%*
	Adj.	0.032	0.029	*-9.38%*	0.076	0.072	*-5.26%*	0.094	0.088	*-6.38%*
	Total	0.579	0.595	*+2.76%*	0.563	0.558	*-0.89%*	0.587	0.547	*-6.81%*
Content-Free	Prep.	0.056	0.051	*-8.93%*	0.120	0.132	*+10.00%*	0.129	0.151	*+17.05%*
	Dete.	0.043	0.043	*0.00%*	0.102	0.101	*-0.98%*	0.112	0.103	*-8.04%*
	Punc.	0.137	0.131	*-4.38%*	0.100	0.091	*-9.00%*	0.096	0.120	*+25.47%*
	Others	0.186	0.179	*-3.76%*	0.115	0.118	*+2.61%*	0.076	0.079	*+3.95%*
	Total	0.421	0.405	*-3.80%*	0.437	0.442	*+1.14%*	0.413	0.453	*+9.69%*

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

Experiments

- Linguistic Analysis on important words
  - Fertility: word alignment

Fertility	Chinese⇒English			English⇒French			English⇒Japanese		
	Count	Attri.	$\triangle$	Count	Attri.	$\triangle$	Count	Attri.	$\triangle$
≥ 2	0.087	0.146	*+67.82%*	0.126	0.138	*+9.52%*	0.117	0.143	*+22.22%*
1	0.621	0.622	*+0.16%*	0.672	0.670	*-0.30%*	0.570	0.565	*-0.88%*
$(0, 1)$	0.115	0.081	*-29.57%*	0.116	0.113	*-2.59%*	0.059	0.055	*-6.78%*
0	0.176	0.150	*-14.77%*	0.086	0.079	*-8.14%*	0.254	0.237	*-6.69%*

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

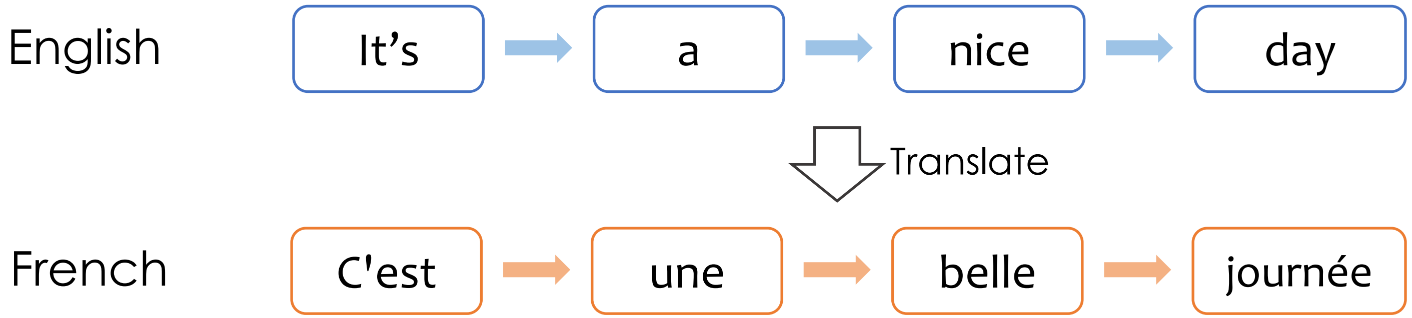
- NMT evolution path

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

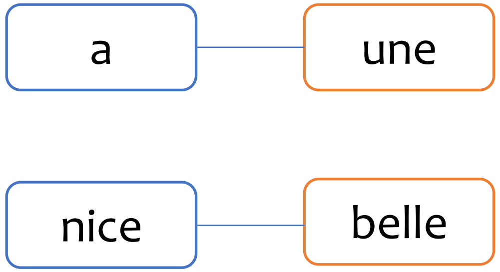
1. The input-output attribution provides local explanations ***only***



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:

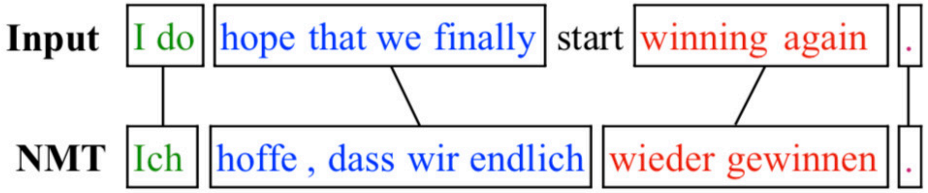


- Bilingual knowledge is at the core of adequacy modelling, a major weakness of NMT models

- 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

Source	***Target***
I do	Ich
I do hope that	hoffe ich , dass
hope that we finally	hoffe , dass wir endlich
winning again	wieder gewinnen
winning again	gewinnen einer
.	.
...	

(b) Phrase table extracted from the NMT model

Method

- Phrase table extraction

Algorithm 1 Constructing Phrase Table

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

procedure PHRASETABLE
    EXTRACTION
    ESTIMATION
procedure EXTRACTION
    $\widehat{\mathcal{R}} \leftarrow$ extract candidates from $\{(\mathbf{x}, \mathbf{y}), \mathbf{a}\}$
    for each $r \in \widehat{\mathcal{R}}$ **do** $\triangleright$ *priors of NMT predictions*
        if r is consistent with $\mathbf{m}$ **then**
            $\mathcal{R}$.append(r)
procedure ESTIMATION
    standard procedure

Method

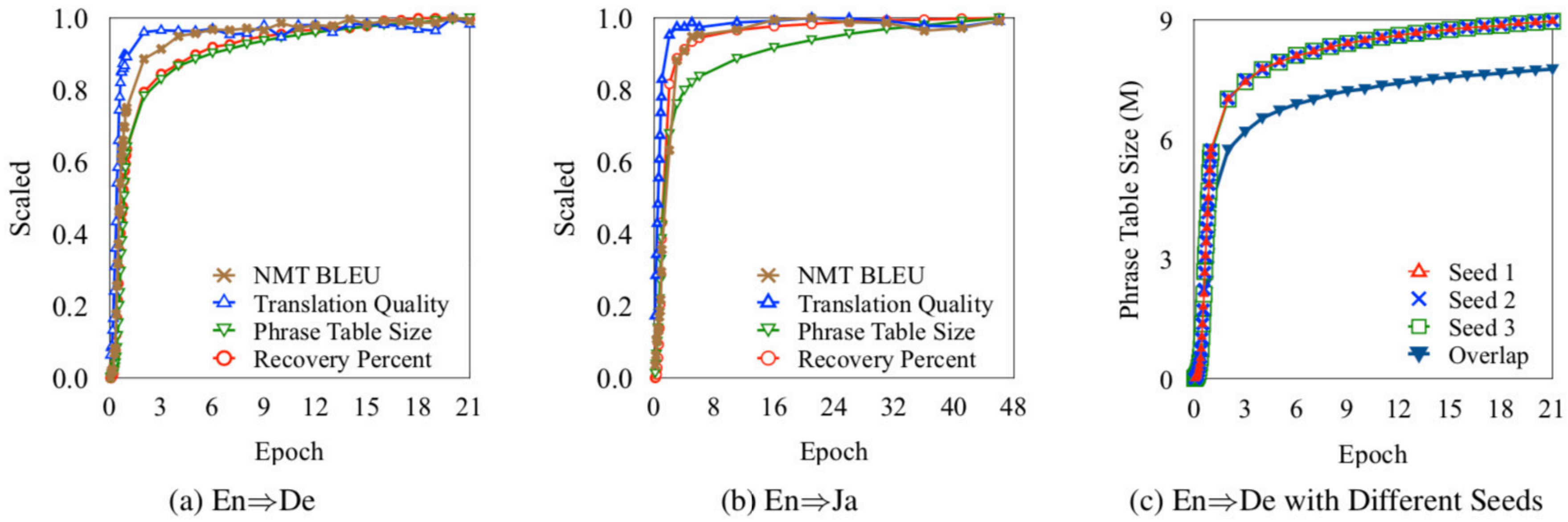
- Implementation
  1. Force-decode the training examples

$$m_j = \begin{cases} 1, & \text{if } y_j = \underset{y'_j \in V}{\operatorname{argmax}} P(y'_j | \mathbf{y}_{<j}, \mathbf{x}) \\ 0, & \text{otherwise} \end{cases}$$

  2. Build masked training data, \$MASK\$
  3. Extract the phrase table
  4. Remove phrase pairs that contain the \$MASK\$

Experiments

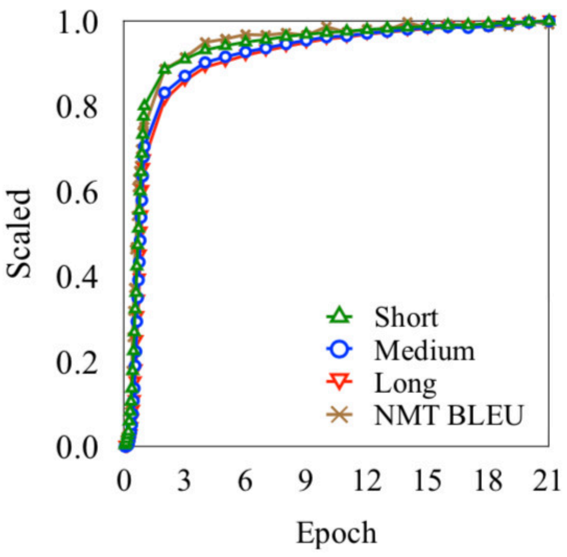
- RQ1: Is phrase table a reasonable bilingual knowledge representation?
- Evaluation metric for phrase table
  - *Phrase Table Size*
  - *Recovery Percent*
  - *Translation Quality*



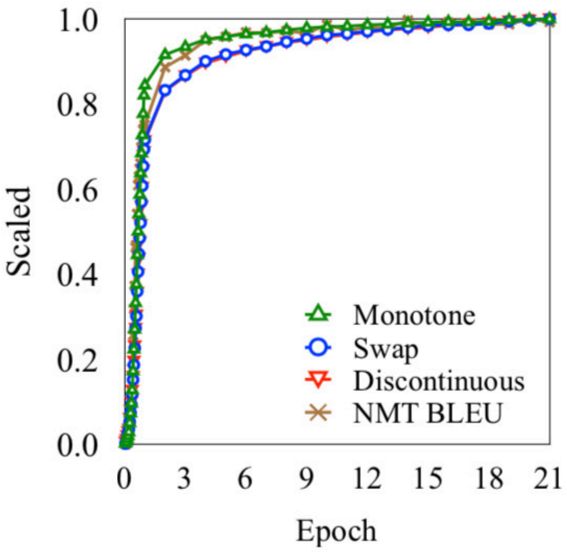
The extracted phrase table correlates well with the NMT performance, consistent across *language pairs, random seeds and model structures*.

Experiments

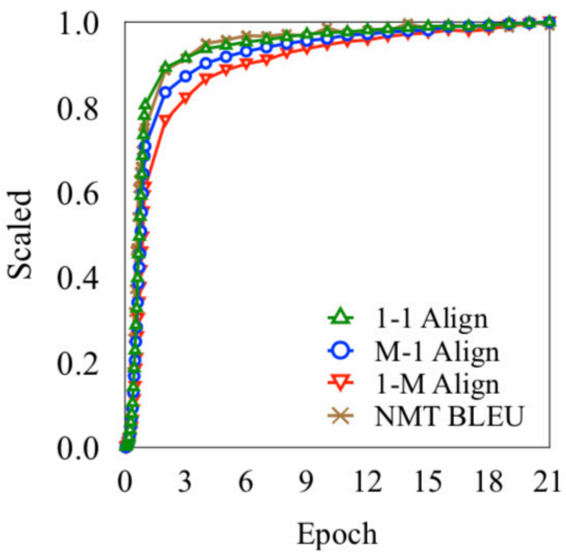
- RQ2: How do NMT models learn the bilingual knowledge during training?
- Different types of phrase pairs with increasing complexity
  - *Phrase Length*
  - *Reordering Type*
  - *Word Fertility*



(a) Phrase Length



(b) Reordering Type

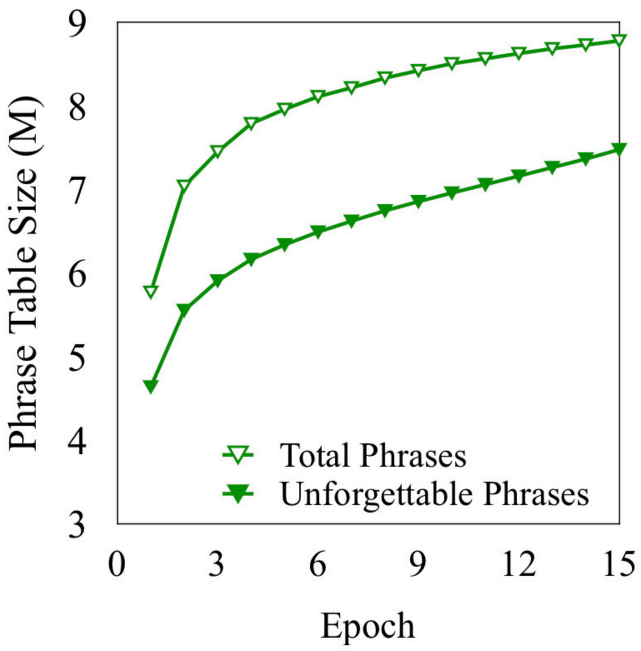


(c) Word Fertility

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?

Phrase Table	Shared		Non-Shared		All	
	Size	*BLEU*	*Size*	*BLEU*	*Size*	*BLEU*
Full	9.0M	17.32	8.5M	4.50	17.5M	17.91
NMT	9.0M	17.90	0M	0	9.0M	17.90

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

Model	NMT		Phrase Table	
	#Para	*BLEU*	*Size*	*BLEU*
SMALL	38M	25.45	7.7M	17.35
BASE	98M	27.11	9.0M	17.90
BIG	284M	28.40	9.2M	17.89

Model	Shared		Non-Shared	
	Size	*BLEU*	*Size*	*BLEU*
SMALL	7.0M	17.53	0.7M	2.37
BASE	7.0M	17.49	2.0M	3.57
BIG	7.0M	17.29	2.2M	3.47

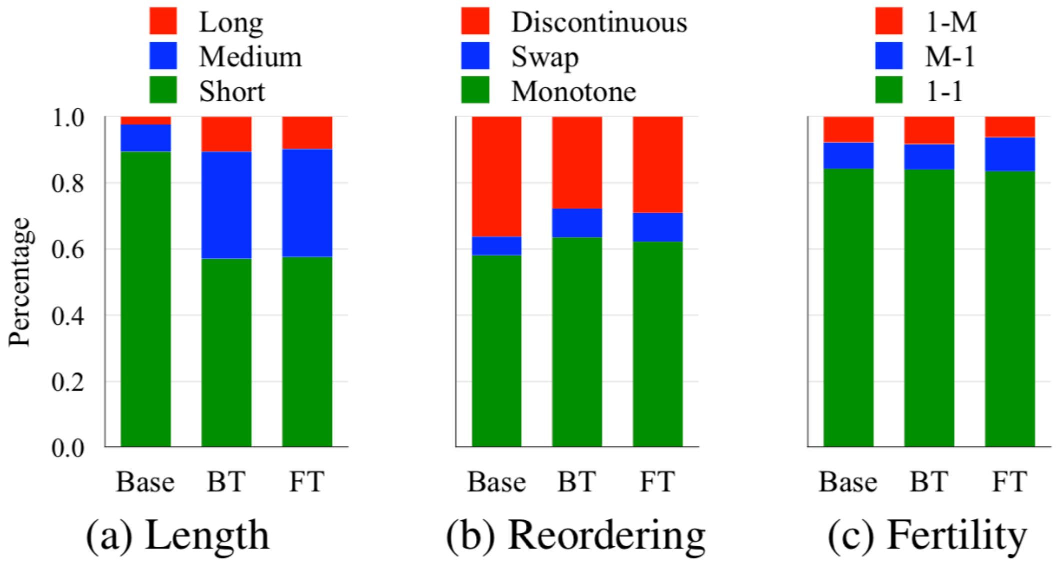
Increasing the model capacity does not increase the bilingual knowledge

Experiments

- Revisit recent advances
  - Data augmentation

Model	NMT		Phrase Table	
	#Para	*BLEU*	*Size*	*BLEU*
BASE	98M	27.11	9.0M	17.90
+ BT	98M	29.75	20.9M	19.26
+ FT	98M	28.43	28.0M	19.33

Model	Shared		Non-Shared	
	Size	*BLEU*	*Size*	*BLEU*
BASE	8.3M	17.67	0.7M	1.78
+ BT	8.3M	18.61	12.6M	10.45
BASE	8.4M	17.83	0.5M	1.21
+ FT	8.4M	18.30	19.6M	11.25



Experiments

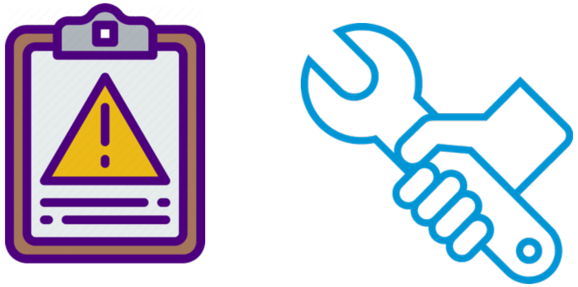
- Revisit recent advances
  - Domain Adaptation

Fine Tune	NMT		Phrase Table	
	# Para.	*BLEU*	*Size*	*BLEU*
×	98M	15.78	168K	16.08
✓	98M	31.26	316K	18.50

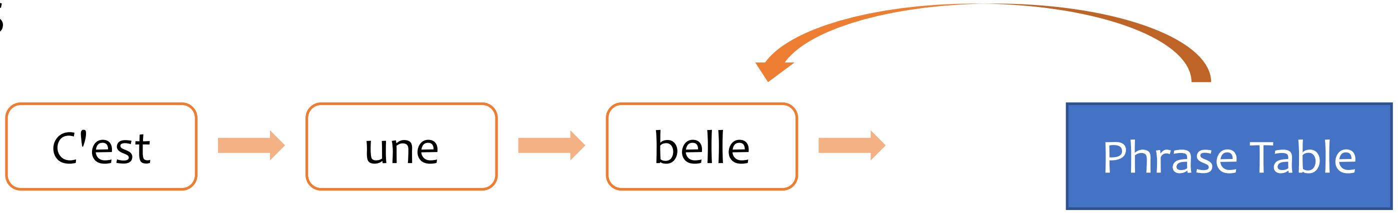
Fine Tune	Shared		Non-Shared	
	Size	*BLEU*	*Size*	*BLEU*
×	0.16M	15.95	0.01M	1.65
✓	0.16M	16.92	0.16M	6.95

Discussion

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



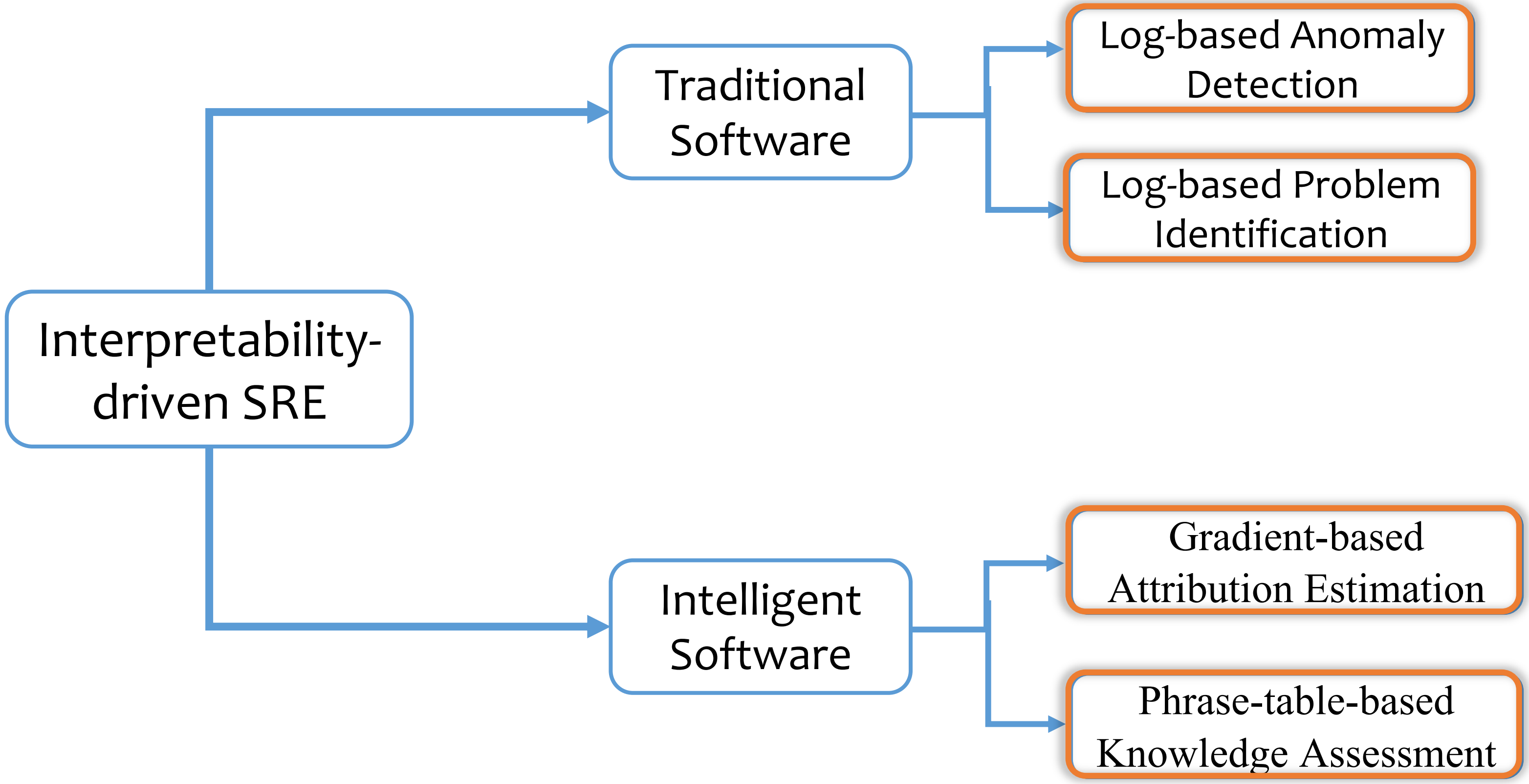
  - *Curriculum learning*: dynamically assigns more weights to unlearned instances
  - *Phrase memory:* stores unlearned phrases in NMT to query when generating translations



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