University of Ottawa Université d'Ottawa

Systematic Evaluation of Deep Learning Models for Failure Prediction

Presented by: Fatemeh (Bahar) Hadadi

Fatemeh Hadadi
University of Ottawa

Joshua H. Dawes
University of Luxembourg

Donghwan Shin
University of Sheffield
(form. University of Luxembourg)

Domenico Bianculli
University of Luxembourg

Lionel Briand
University of Ottawa
University of Limerick

30 October 2024

LERO

uOttawa

uOttawa

Reliability and Availability Impact

- Ensuing **reliability and availability** of software systems is crucial.

In 2024, a **Microsoft outage** triggered by a CrowdStrike update led to crashes (BSOD) in online systems, costing US Fortune 500 companies **$5.4 billion**.

In 2023, a two-day outage at **Square**, caused by DNS issues, resulted in thousands of dollars in **lost revenue for small-business owners**.

Square

uOttawa

Log Files, Helpful Artifacts

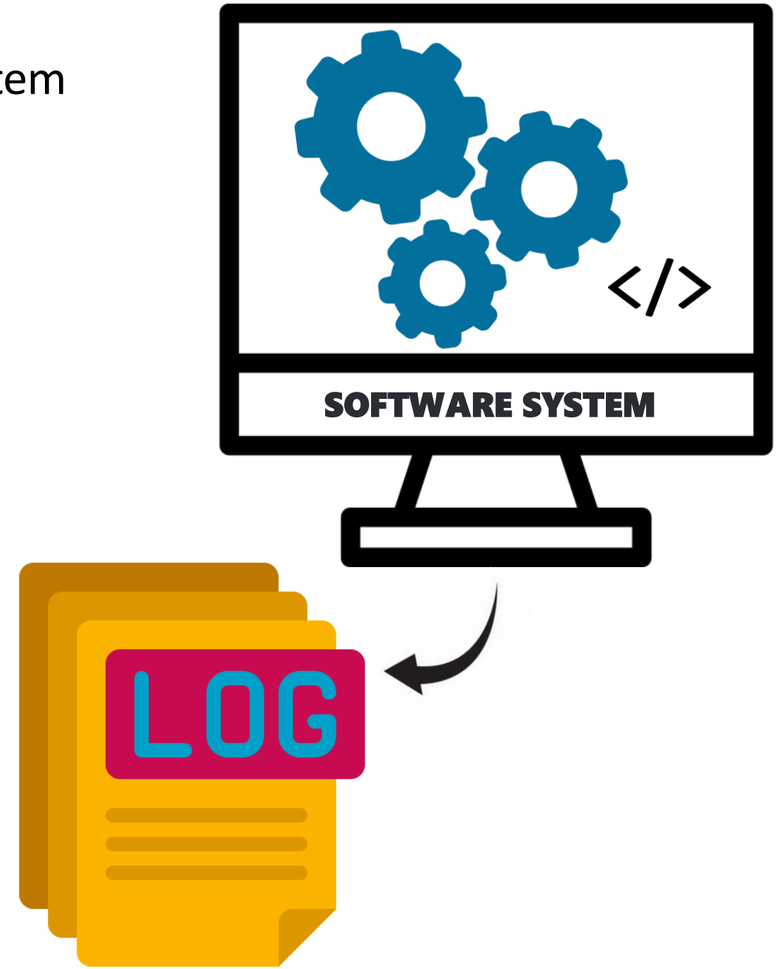
- **Log files** recorded during system execution hold information about system behavior.

Example:

```
1  public void setTemperature( Interger temperature) {
2
3    //..
4    logger.debug (“Temperature set to {}.”, temperature.intvalue());
5    if (temperature.intValue() > 50 ) {
6         logger.info(“Temperature has risen above 50 degrees.”);
7    }
8  }
```

```
1  0 [setTemperature] DEBUG Wombat - Temperature set to 61.
2  0 [setTemperature] INFO Wombat - Temperature has risen above 50 degrees.
```

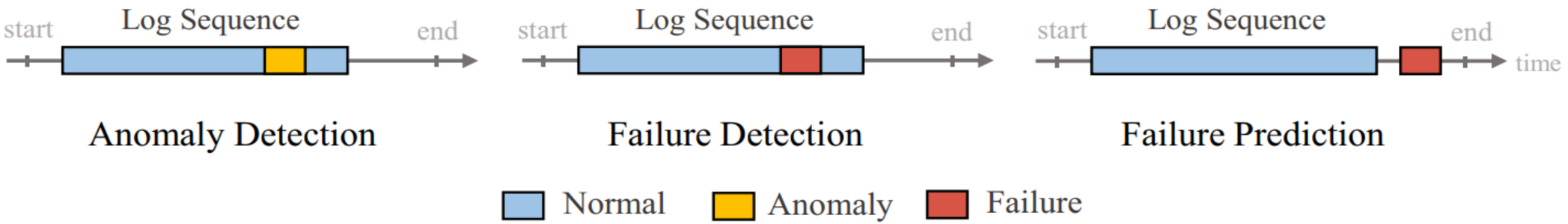
Output Log Example



uOttawa

Log-based Failure Prediction

- One important usage of log files is to **predict the occurrence of failures (Failure Prediction)**.



Main differences:

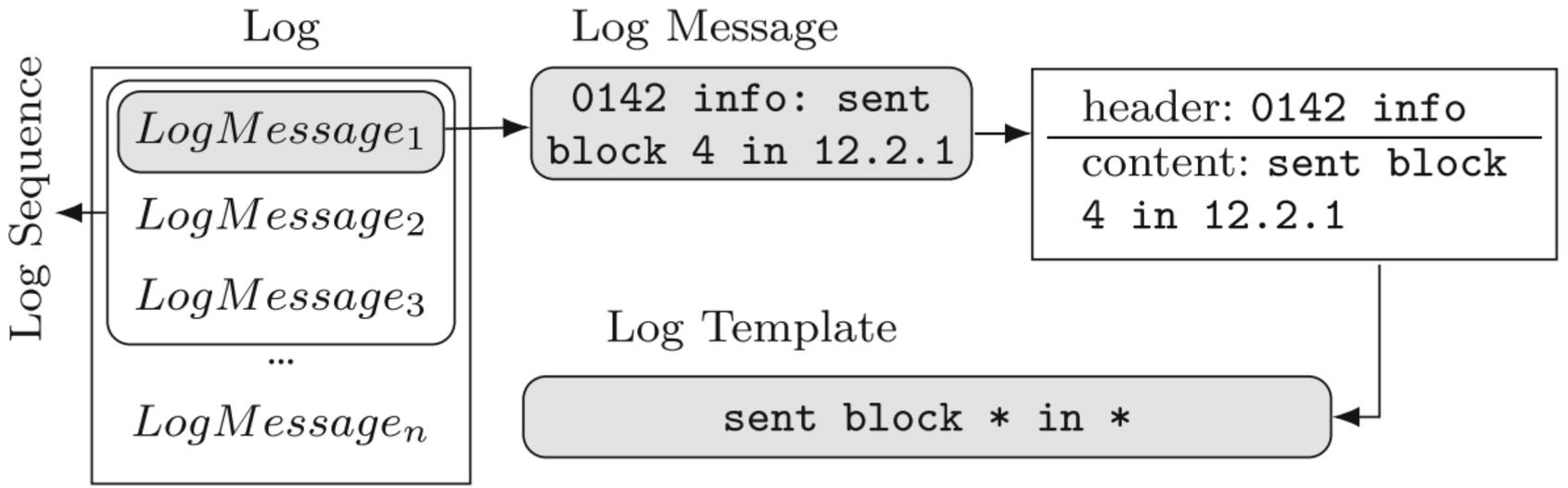
1) Mode of operation (reactive vs proactive)

2) input data

uOttawa

Background on Logs

- **log** *:* a sequence of log messages generated by logging statements
- **log sequence** : a fragment of a log
- **log template** : an abstraction of a log message content (also called event template or log key)



For convenience, we abstract log sequences by replacing the log messages with their log templates.

uOttawa

DL networks & Log-based Failure Prediction

- Recent studies explored several automated tasks including **Deep Learning (DL)** networks.
- DL networks are based on Neural Networks.
- **There is lack of a study evaluating all the main types of DL networks**.

Recurrent Neural Network (RNN)

Convolutional Neural Network (CNN)

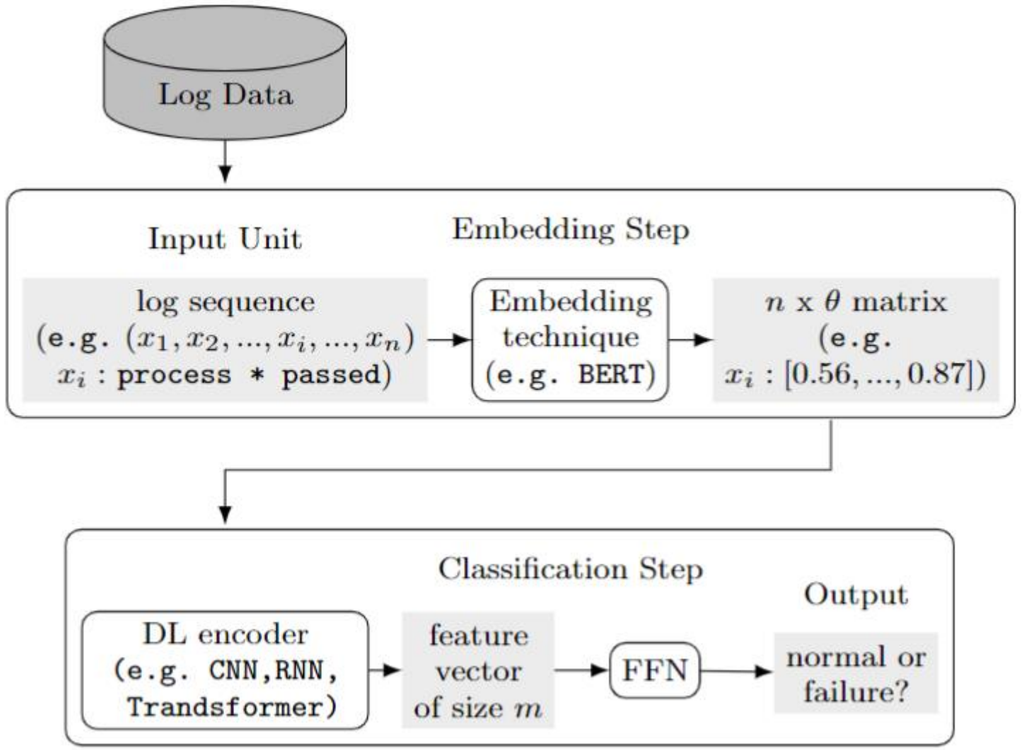
transformer

Three Main Type of DL Network

uOttawa

Modular Architecture

A generic architecture to study different DL methods (as encoders) next to embedding strategies for input log sequences.



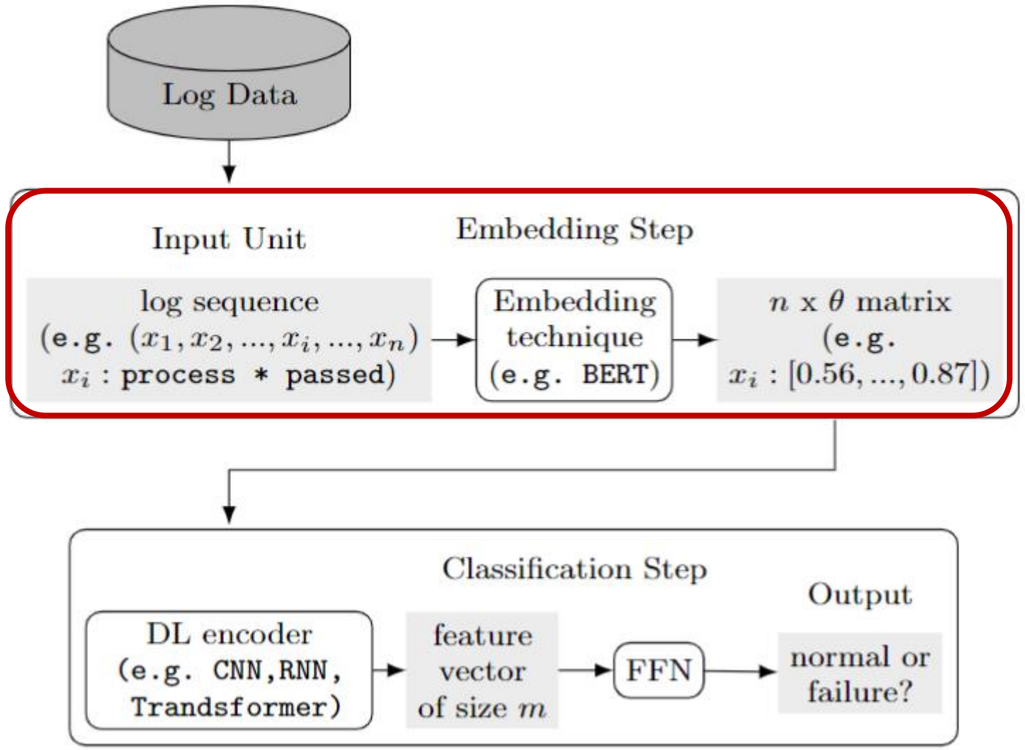
uOttawa

Modular Architecture

A generic architecture to study different DL methods (as encoders) next to embedding strategies for input log sequences.

Embedding techniques:

- **Logkey2vec (Template ID-based Strategy)**
- **BERT (Semantic-based Strategy)**
- **FastText + TF-IDF (Hybrid Strategy)**



uOttawa

Modular Architecture

A generic architecture to study different DL methods (as encoders) next to embedding strategies for input log sequences.

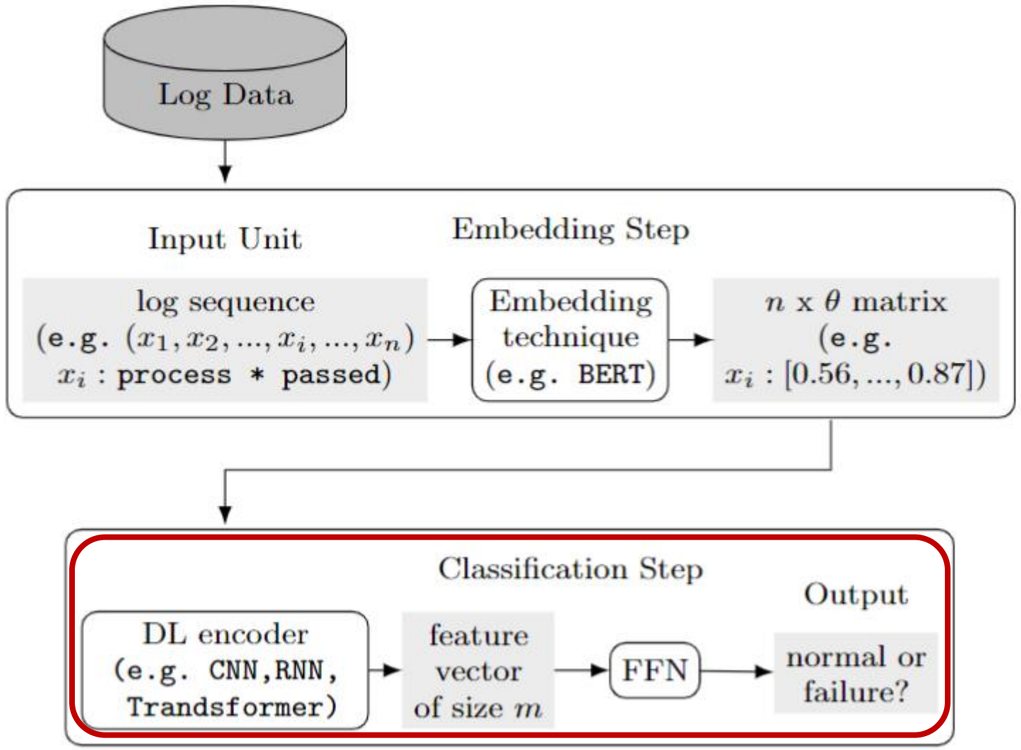
Embedding techniques:

- **Logkey2vec (Template ID-based Strategy)**
- **BERT (Semantic-based Strategy)**
- **FastText + TF-IDF (Hybrid Strategy)e)**

DL encoder (trainable):

- **LSTM**
- **BiLSTM**
- **CNN**
- **Transformer**

Total of 12 configurations



uOttawa

Research Questions

RQ1. What is the impact of **different DL encoders** on failure prediction accuracy?

RQ2. What is the impact of **different log sequence embedding strategies** on failure prediction accuracy?

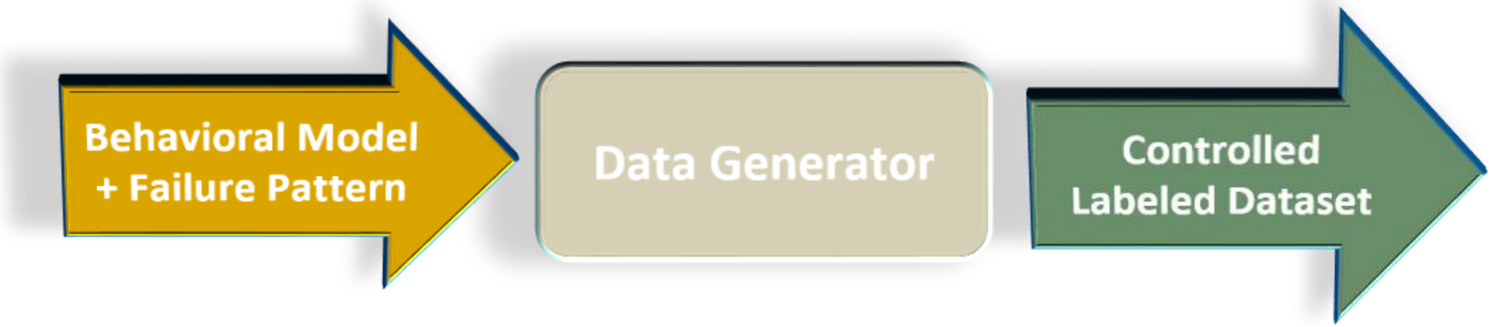
RQ3. How do DL-based failure predictors fare **compared to traditional ML-based** ones in terms of failure prediction accuracy?

RQ4. What is the impact of **different dataset characteristics** on failure prediction accuracy?

RQ5. How does the accuracy of failure prediction on synthesised datasets **compare to that of real-world datasets**?

uOttawa

Synthetic Data Generation



Controlled Parameters: Dataset Size, Percentage of Failure, etc.

- There are **limited available datasets**.
- There is **lack of a method** for generating datasets in our desired way
- An approach which can **control output labels** is required.
- We designed a **comprehensive and controllable** data generator:

uOttawa

Synthetic Data Generation

- Behavioral Model is in shape of a **Finite Automaton** accepting log sequences of systems.
- Language of failure patterns as **RegEx** is a subset of behavioral model's language.
- We use a **filtered random walk** on FSM for generating **normal** log sequences.
- **Failure** log sequences are randomly generated sequences directly from the regex of **failure patterns**.

Algorithm 1: *generateNormalSequence*

Input: $\mathcal{M}$: *behaviour model*, $mlsl : int$
Output: $sequence : \langle s_1, s_2, \ldots, s_n \rangle \in \mathcal{L}(\mathcal{M})$

```
1  sequence : list ← filteredRandomWalk(M, mlsl)
2  while sequence ∈ ⋃_{fp_i ∈ failurePatterns(M)} L(fp_i) do
3      sequence ← filteredRandomWalk(M, mlsl)
4  return sequence
```

Algorithm 2: *filteredRandomWalk*

Input: $\mathcal{M} = \langle Q, A, q_0, \Sigma, \delta \rangle$: *behaviour model*, $mlsl : int$
Output: $sequence : \langle s_1, s_2, \ldots, s_n \rangle \in \mathcal{L}(\mathcal{M})$

```
1  sValue : int ← calculateSValues(M)
2  sequence : list ← ⟨⟩
3  maximumWalk : int ← mlsl
4  currentState : state ← q_0
5  while currentState ∉ A do
6      options : set ← ∅
7      transitions : set ← {⟨currentState, s, q⟩ : δ(currentState, s) = q}
8      for ⟨q, s, q'⟩ ∈ transitions do
9          if sValue(q') < maximumWalk then
10             options ← options ∪ {⟨q, s, q'⟩}
11     ⟨q, s, q'⟩ ← random choice from options
12     sequence ← sequence + ⟨s⟩
13     currentState ← q'
14     maximumWalk ← maximumWalk − 1
15 return sequence
```

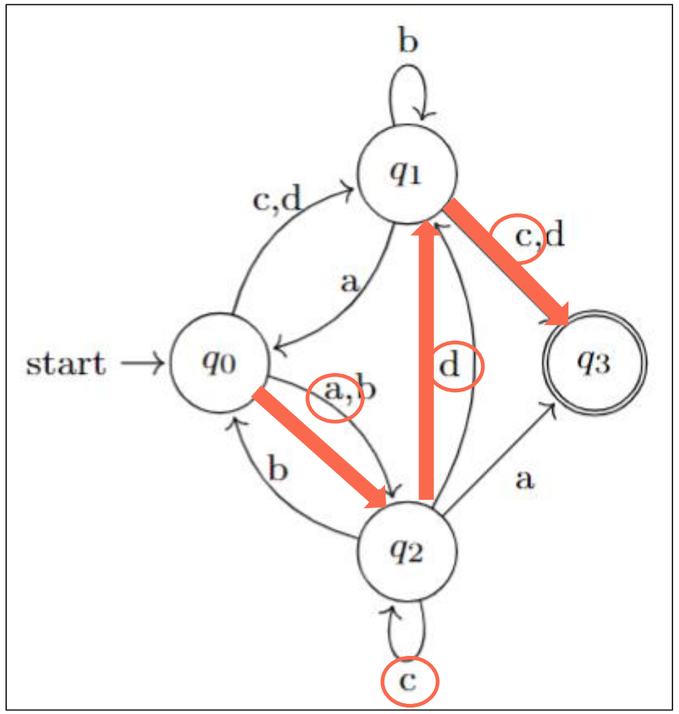
uOttawa

Data Generation: Example

Log Messages: **a**, **b**, **c**, **d**

System States: **q_0**, **q_1**, **q_2**, **q_3**

Terminal State: **q_3**



Behavioral Model Example

- **Possible path:**

  a -> c -> d -> c

- **Log Sequence:**

  (a, c, d, c)

uOttawa

Experiment Setting

- **Synthetic Dataset characteristics**

Dataset Size, 6 levels: 200, 500, 1000, 5000, 10000, 50000
Failure Percentage, 6 levels, 5%, 10%, 20%, 30%, 40%, 50%
Log Sequence Length (LSL), 5 levels (on their maximum), 20, 50, 100, 500, 1000
Failure Pattern, 2 types, Type-F and Type-I (Finite vs Infinite)

Total of 360 (6x6x5x2) combinations

- **Behavioural Models**

Model	#Templates	Avg Template Length	#States	#Transitions	#States-NSCC
$\mathcal{M}_1$	70	54	154	195	5
$\mathcal{M}_2$	16	51	91	189	72
$\mathcal{M}_3$	115	39	350	486	331

Accurate models of real-world systems are inferred from their execution logs using the state-of-the-art model inference tool, PRINS.

- Total of 1080 (3x360) datasets

uOttawa

Experiment Setting

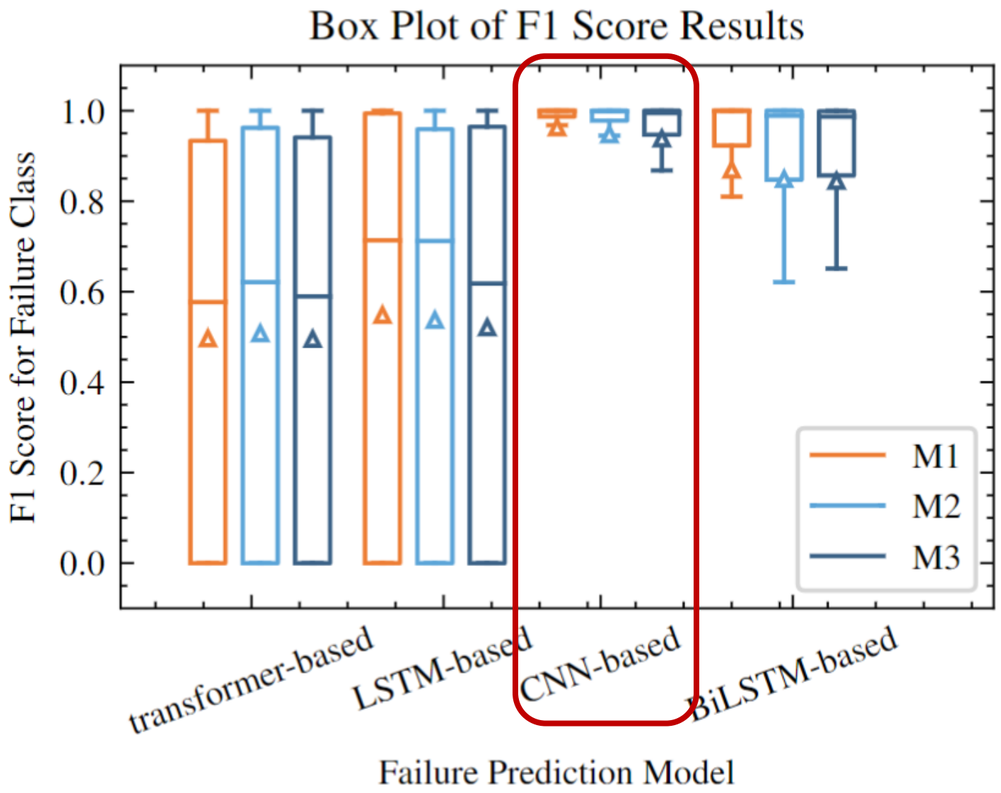
❑ Real-world Dataset characteristics

OpenStack dataset is **originally for failure/anomaly detection**. We further processed it for failure prediction by **retaining log messages up before the first failure message.**

#Logs	#Failures Log Sequences	Failure Percentage	#Unique Log Templates	Log Sequence Length		
				avg	min	max
876	188	21.46%	468	228	4	462

uOttawa

RQ1: DL Encoders



- The best performance in terms of F1 score for all behavior models: **CNN-based model**
- The large variance for LSTM-based and transformer-based models suggests that **these models are very sensitive to the dataset characteristics**.

The CNN-based encoder significantly outperforms the other encoders **regardless of dataset characteristics and log sequence embedding strategies.**

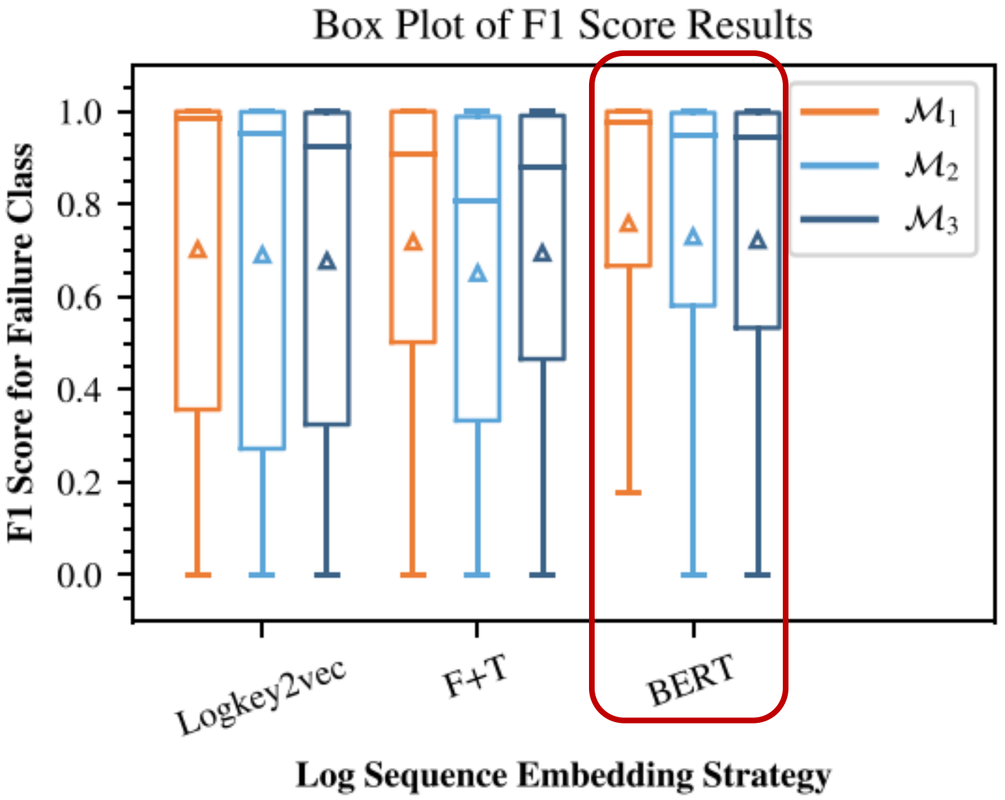
uOttawa

RQ2: Embedding Strategies

- The **BERT** outperforms in all behaviour model datasets in terms of mean values and smaller IQRs.
- The **semantic-aware** embedding of BERT fares better than the ones relying on template IDs.

Table 7 Friedman test results (p-values). A level of significance $\alpha = 0.01$ is used. In case of a significant difference, the best strategy is denoted as l (Logkey2vec), f (F+T), and b (BERT)

DL encoder	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	All
CNN	$\ll 0.001$ (B)	$\ll 0.001$ (L)	$\ll 0.001$ (L)	$\ll 0.001$ (L)
BiLSTM	$\ll 0.001$ (B)	$\ll 0.001$ (B)	0.001 (B)	$\ll 0.001$ (B)
transformer	0.068	$\ll 0.001$ (F, B)	$\ll 0.001$ (F, B)	$\ll 0.001$ (F, B)
LSTM	$\ll 0.001$ (B)	$\ll 0.001$ (L, B)	$\ll 0.001$ (L)	$\ll 0.001$ (B)
All	$\ll 0.001$ (l)	$\ll 0.001$ (L, B)	$\ll 0.001$ (B)	$\ll 0.001$ (B)

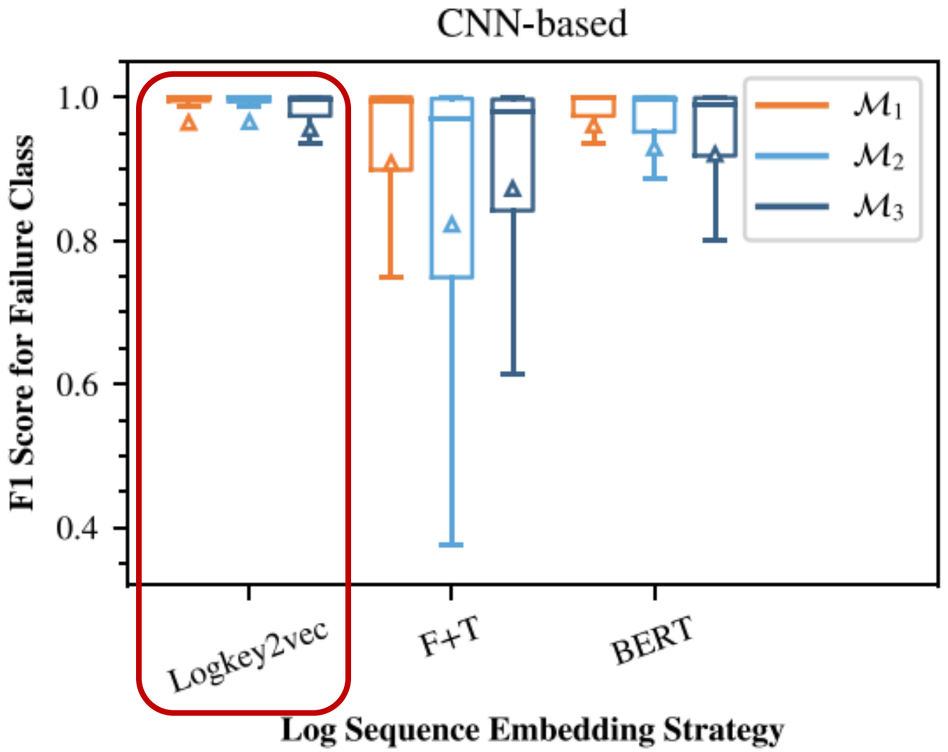


uOttawa

RQ2: Embedding Strategies

- BERT is statistically better than or equal to Logkey2vec for each encoder **except the CNN-based encoder**.
- Both BERT and Logkey2vec, in fact, achieved **high accuracy** for CNN-based.
- We suspect that characteristics of Logkey2vec **(e.g. trainability)** play a positive role in combination with the CNN-based encoder.

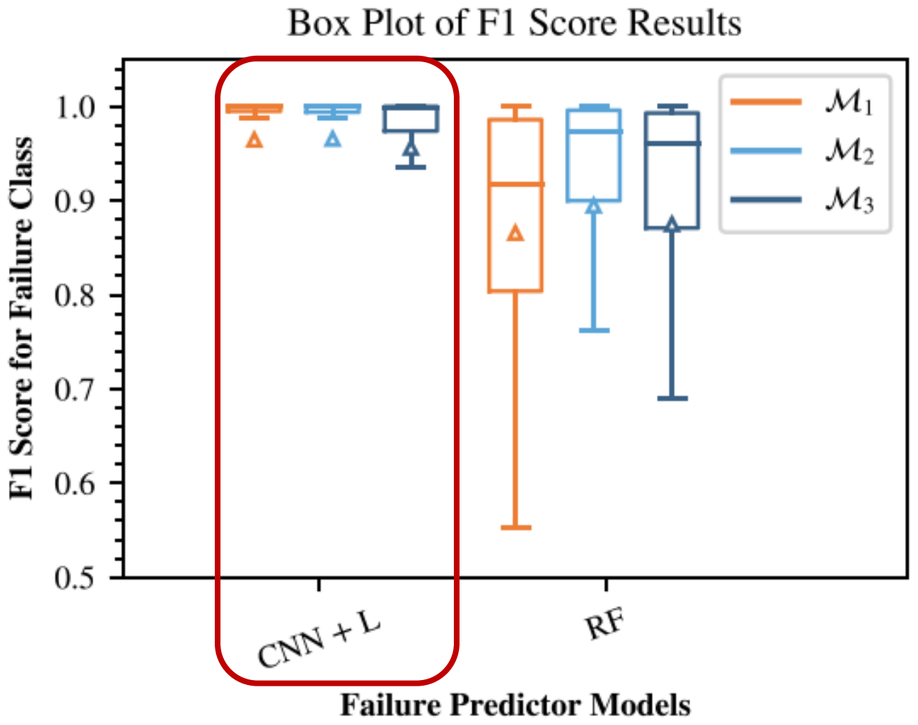
The performance of the log sequence embedding strategies varies **depending on the DL encoders used**.



uOttawa

RQ3: Traditional ML

- RF relies on aggregating decisions from multiple trees which can limit its ability to capture intricate, non-linear patterns of failures.
- RF embedding strategy: **TF-IDF** (Template ID-based Strategy) .

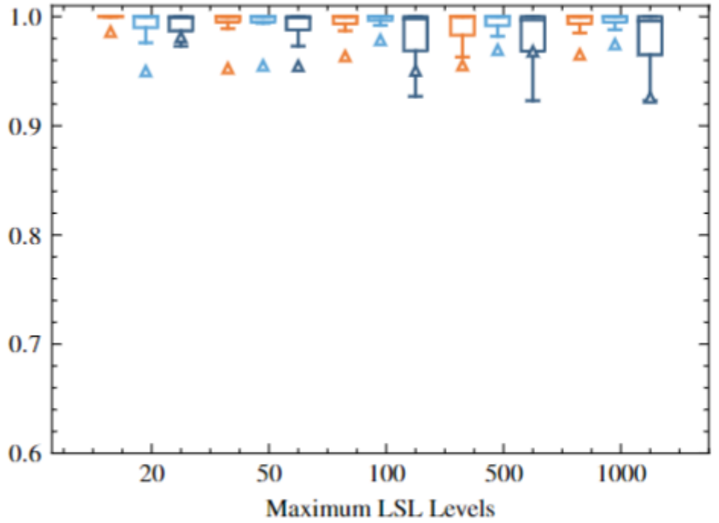


Using the best configuration of the DL-based failure predictor results in **significantly higher accuracy and robustness** compared to the top traditional ML classifiers.

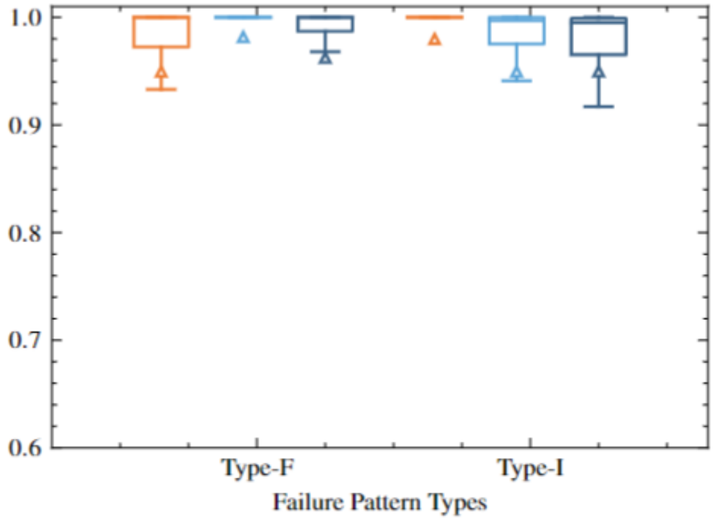
uOttawa

RQ4: Dataset Characteristics

- The impact of log sequence length is **relatively small**.
- We suspect that its impact could be significant **for much longer log sequences**.



- For failure pattern types, there is **no consistent trend** across models M1, M2, and M3.
- We suspect that failure pattern types are not defined in a way that is conducive to explaining variations in accuracy (different hypotheses will have to be tested in future work).

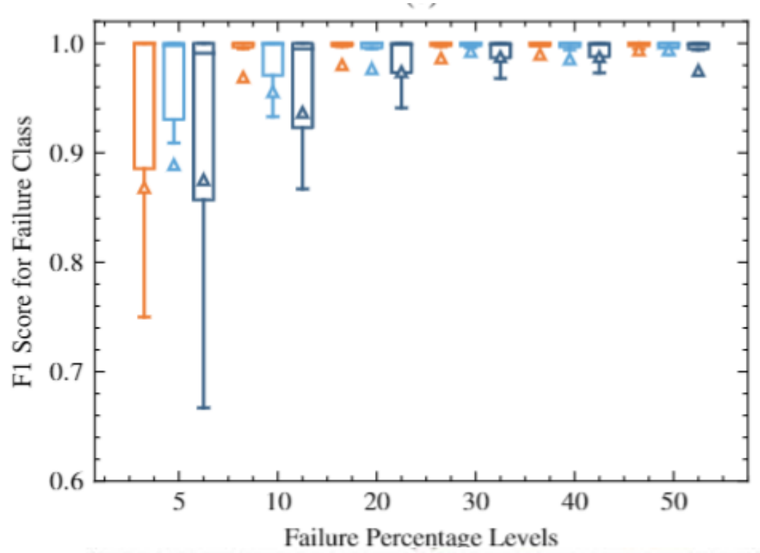


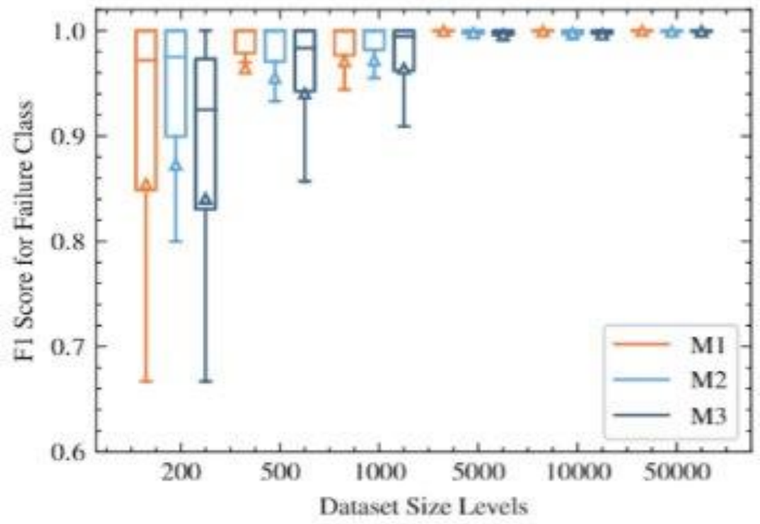
uOttawa

RQ4: Dataset Characteristics

- The F1 score increases as the failure percentage increase.
- Accuracy decreases with smaller datasets,

Dataset size, followed by **failure percentage,** plays an important role in the accuracy of DL-based failure predictors while LSL is important only for some configurations.



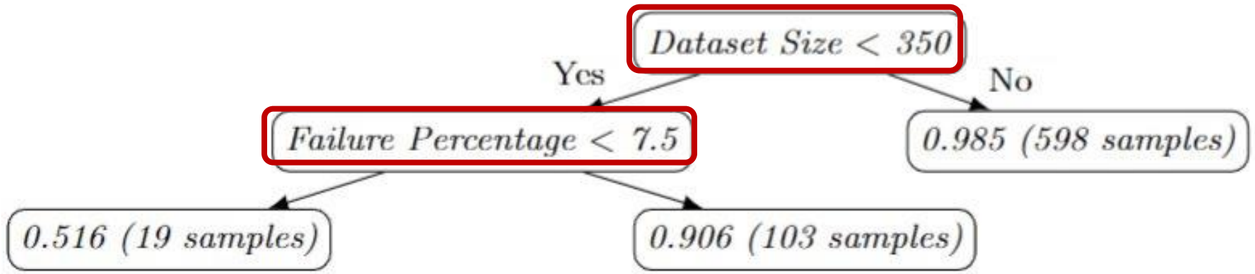


uOttawa

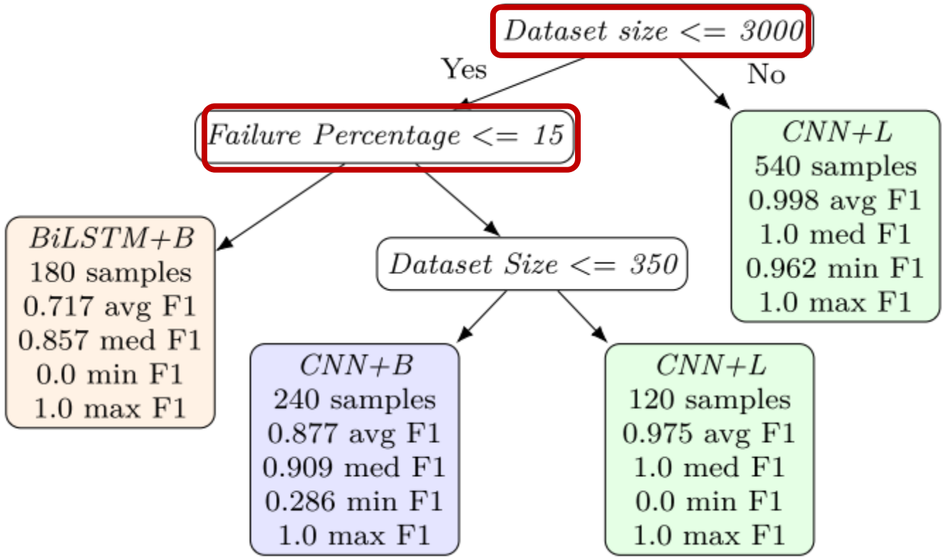
RQ4: Guidelines

Regression and decision trees to explain F1 score variations:

When **dataset size** is **above 350** or **failure percentage** is **above 7.5%,** failure predictors are **very accurate** (F1-score > 0.95) and robust (IQR < 0.01)



Regression Tree of CNN + Logkey2vec



Decision Tree of Top Configurations

uOttawa

RQ5: Real-world Dataset

- Synthetic datasets with close characteristics to OpenStack_FP are selected for comparison.

Dataset	DS	MLSL	PF	CNN + L		
				P	R	F1
Synthesised$_{\mathcal{M}_1}$	1000	500	20	0.987	1.000	0.993
Synthesised$_{\mathcal{M}_2}$	1000	500	20	0.932	0.965	0.9480
Synthesised$_{\mathcal{M}_3}$	1000	500	20	0.947	0.988	0.967
average				0.955	0.984	0.969
OpenStack_FP	876	468	21.46	0.974	0.974	0.974

- The Wilcoxon test among results shows **p-values above 0.05** for precision, recall, and F1 score.

There is no significant difference between the accuracy results obtained **on comparable synthesised datasets and a real-world one (OpenStack_FP)** when using the best configuration for failure prediction (CNN-based encoder with Logkey2vec).

uOttawa

Empirical Result Summary

- The **outperforming DL encoder** regardless of embedding and dataset characteristics
- The **best embedding strategies** for each of the DL encoders
- We identified **the best configuration**
- Characteristics which have a **direct impact on model accuracy**
- **Comprehensive and robust guidelines** for future application
- A **publicly available replication package,** including the modular architecture

Replication Package

uOttawa

Future Directions

- Using **real-world log data** to further investigate **the effect of other factors**, such as log parsing techniques, on model accuracy.
- Including **time-aware evaluation metrics**, such as lead time, to assess the accuracy of these models at predicting failures early on.

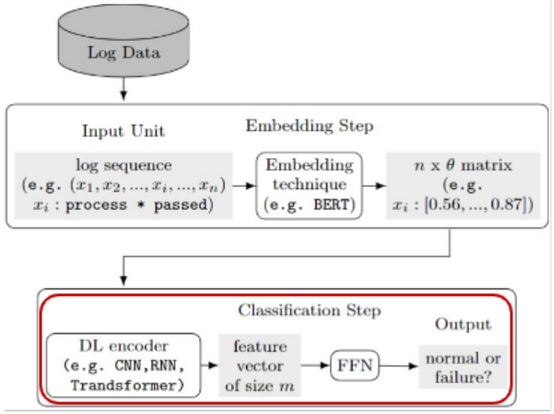
Modular Architecture

A generic architecture to study different DL methods (as encoders) next to embedding strategies for input log sequences.

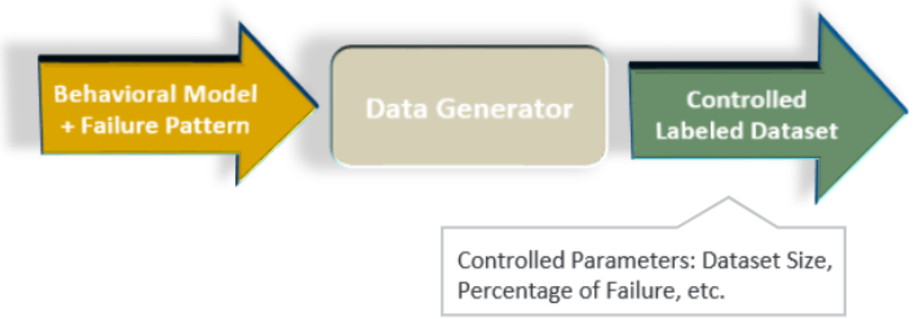
Embedding techniques:
- **Logkey2vec (Template ID-based Strategy)**
- **BERT (Semantic-based Strategy)**
- **FastText + TF-IDF (Hybrid Strategy)e)**

DL encoder (trainable):
- **LSTM**
- **BiLSTM**
- **CNN**
- **Transformer**

Total of 12 configurations



Synthetic Data Generation



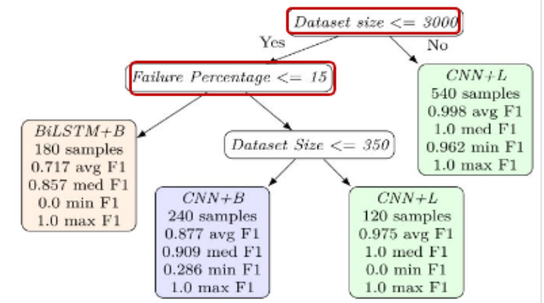
- There are **limited available datasets**.
- There is **lack of a method** for generating datasets in our desired way
- An approach which can **control output labels** is required.
- We designed a **comprehensive and controllable** data generator:

RQ4: Guidelines

Regression and decision trees to explain F1 score variations:

When **dataset size** is **above 350** or **failure percentage** is **above 7.5%,** failure predictors are **very accurate** (F1-score > 0.95) and robust (IQR < 0.01)

Regression Tree of CNN + Logkey2vec



Decision Tree of Top Configurations

Empirical Result Summary

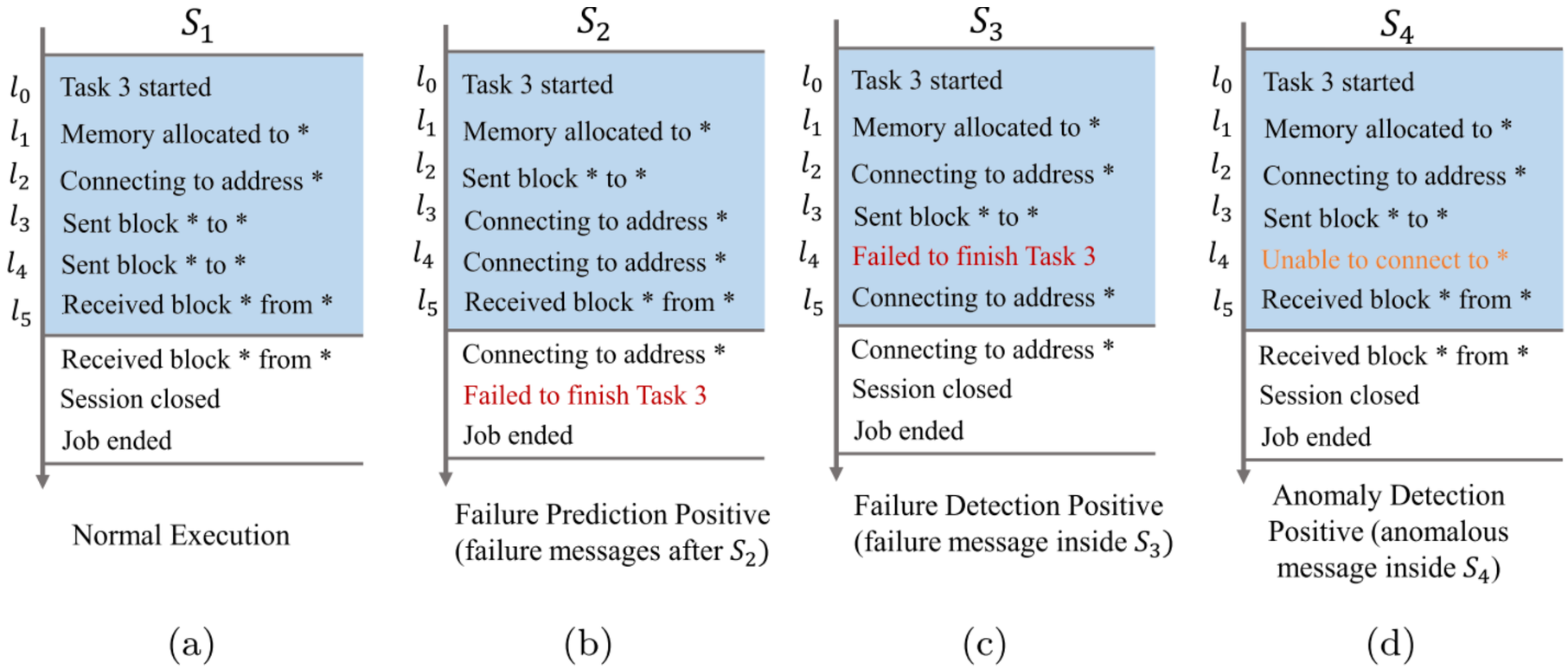
- The **outperforming DL encoder** regardless of embedding and dataset characteristics
- The **best embedding strategies** for each of the DL encoders
- We identified **the best configuration**
- Characteristics which have a **direct impact on model accuracy**
- **Comprehensive and robust guidelines** for future application
- A **publicly available replication package,** including the modular architecture

Replication Package

uOttawa

Log-based Failure Prediction

Example



uOttawa

Experiment Setting: fine-tuning

Fine Tuning

360 different dataset characteristics

Hyperparameter	Condition	Dataset Size					
		200	500	1000	5000	10000	50000
Batch Size	Default	10	15	20	30	150	300
	PF $\leq$ 30	10	15	30	60	300	600
	MLSL $\geq$ 500*	5	5	5	5	5	5
Number of Epochs	Default	20	20	20	20	20	20
	MLSL $\geq$ 500	20	20	10	10	5	5

* This condition has higher priority than the other

FP: Failure Percentage
MLSL: maximum log sequence length

uOttawa

Setting of Data Generation (Behavioural Models)

- Using state-of-the-art model inference tools

⇨ Accurate models from real-world system

From the potential models, main criteria for choosing are:

- ☑ Enable to generate logs within the LSL boundaries
- ☑ Inference time less than 1 hour

Three behavioral models were chosen from NGLCLient, HDFS, and Linux log data:

Model	#Templates	Avg Template Length	#States	#Transitions	#States-NSCC
$\mathcal{M}_1$	70	54	154	195	5
$\mathcal{M}_2$	16	51	91	189	72
$\mathcal{M}_3$	115	39	350	486	331

* number of states in the largest strongly connected component; the higher, the more complex

uOttawa

Setting of Data Generation (Behavioural Models)

- Lack of a way of automatically generating failure patterns from behavioral model
  ⇨ Had to manually create failure patterns

Goals:

- ☑ Unbiased
- ☑ Satisfy the dataset characteristics (especially the LSL)
- ☑ Within the time limit (1 hour)

Model	Type	Length	#Alphabet	#Operators	Star Depth
$\mathcal{M}_1$	F	(14, 34, 35)	(7, 17, 30)	(5, 8, 1)	-
	I	(17, 25, 41)	(16, 15, 16)	(1, 7, 8)	(1, 2, 2)
$\mathcal{M}_2$	F	(20, 27, 32)	(11, 9, 11)	(3, 6, 7)	-
	I	(31, 8, 39)	(5, 5, 12)	(10, 1, 9)	(1, 1, 2)
$\mathcal{M}_3$	F	(134, 36, 48)	(77, 16, 14)	(16, 10, 5)	-
	I	(44, 30, 124)	(11, 16, 78)	(12, 7, 13)	(1, 2, 1)

Steps for creating failure patterns:

❑ *Step 1.* Random choice of the failure pattern's parameters (alphabet size of a regular expression and the number of operators)

❑ *Step 2.* Manually writing a failure pattern satisfying our goals and the above parameters

❑ *Step 3.* Repeat Steps 1 and 2 ten times and randomly select three out of them

uOttawa

Synthetic Data Generation: Compliance to Requirements

R1 - Allow datasets' characteristics to be controlled.

R2 - Be able to generate realistic datasets.

R3 - Be able to generate datasets corresponding to a diverse set of systems.

R4 - Avoid bias in the log sequences that make up the generated datasets.

R1 is met because we have two procedures for generating failure log sequences and normal log sequences. By having these procedures, we can precisely control the number of each type of log sequence in our dataset.

R2 and R3 Existing tools allow one to infer behaviour models of real-world systems from collections of these systems' logs (in a process called model inference).

R3 A result of meeting **R2** is that one can easily infer behaviour models for multiple systems, provided the logs of those systems are accessible.

R4 is met because of the randomisation used in our data generation algorithm.