End-to-End AutoML for Unsupervised Log Anomaly Detection

Shenglin Zhang[1,2] Yuhe Ji[1] Jiaqi Luan[1] Xiaohui Nie[3]

Zi'ang Chen[1] Minghua Ma[4] Yongqian Sun[1,5] Dan Pei[6]

[1]Nankai University

[2]Haihe Laboratory of Information Technology Application Innovation

[3]Computer Network Information Center,Chinese Academy of Sciences.

[4]Microsoft

[5]Tianjin Key Laboratory of Software Experience and Human Computer Interaction

[6]Tsinghua University

Background

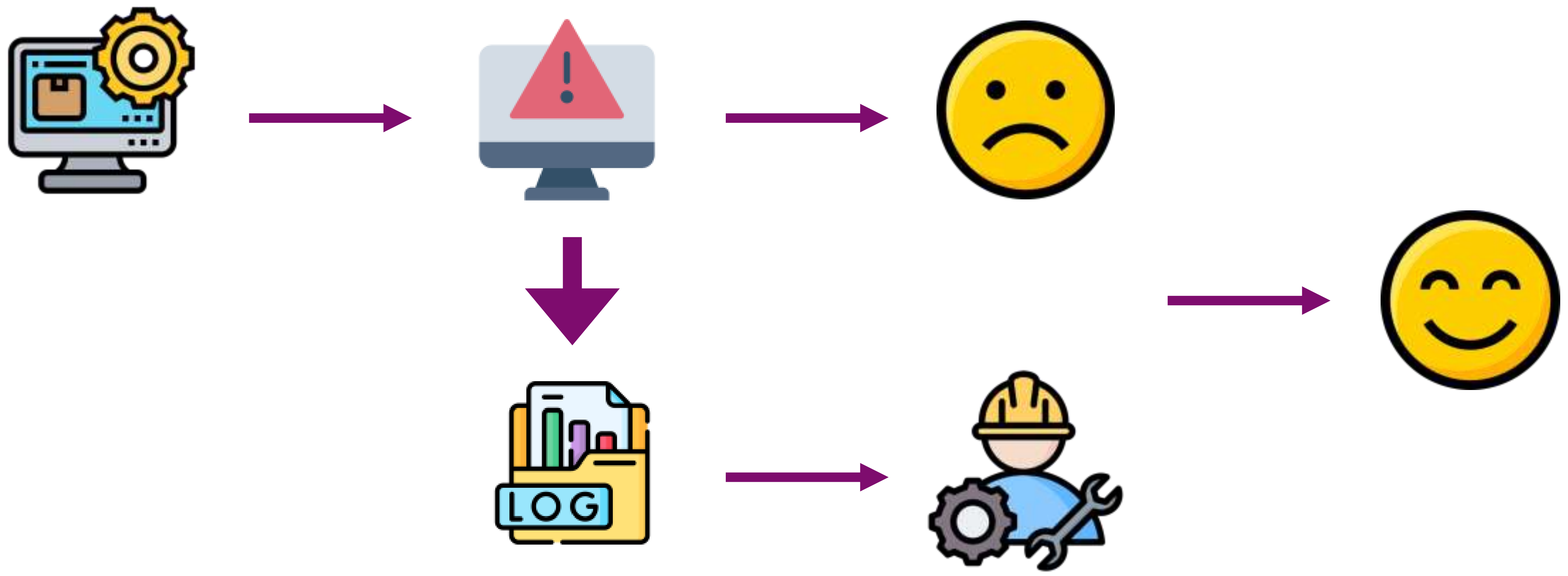
Design

Evaluation

Conclusion

Background

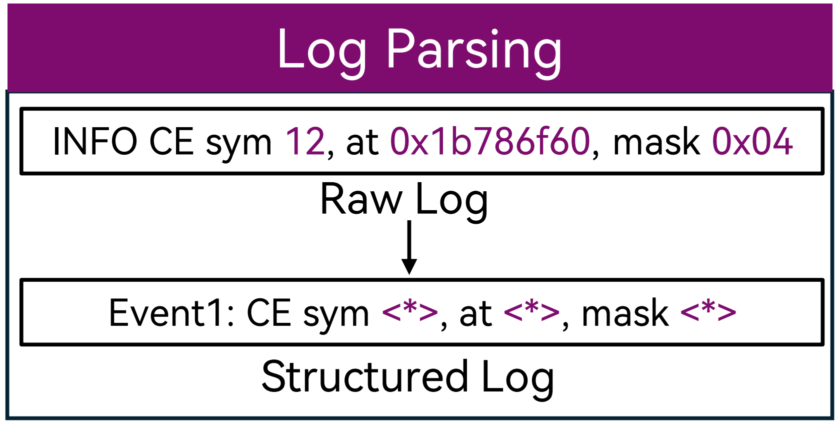
System stability is crucial for modern software systems.
Logs play a key role in maintaining system stability.



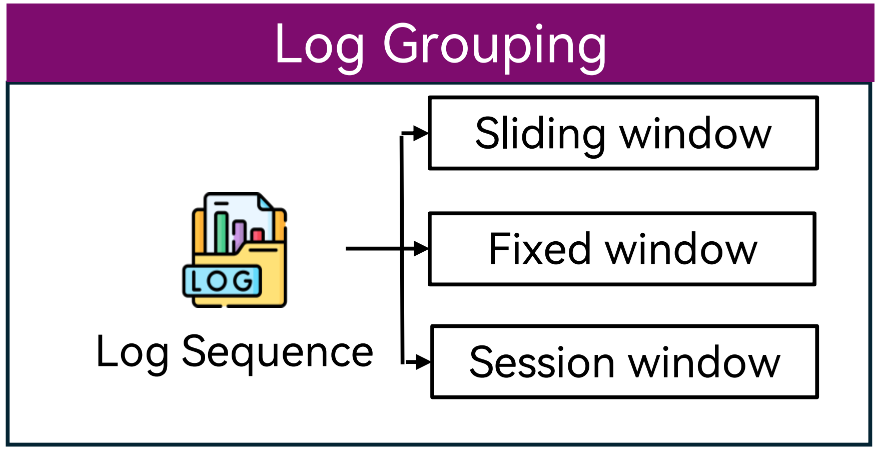
Background

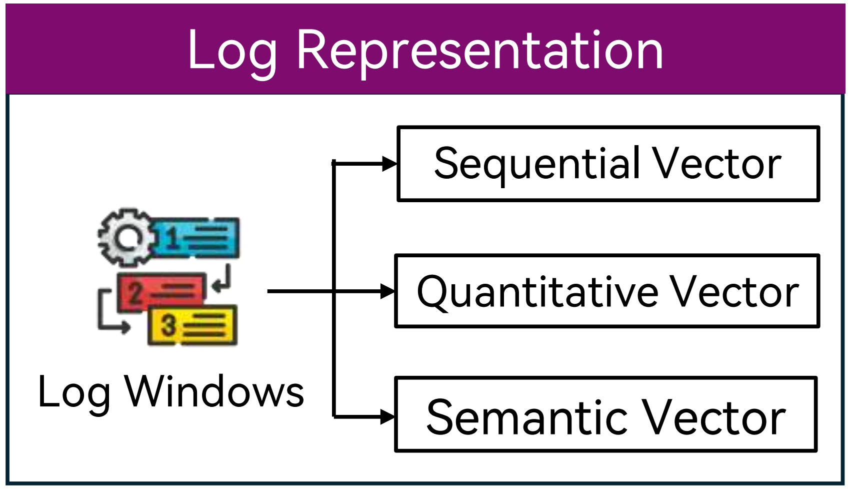
Log anomaly detection includes four steps:

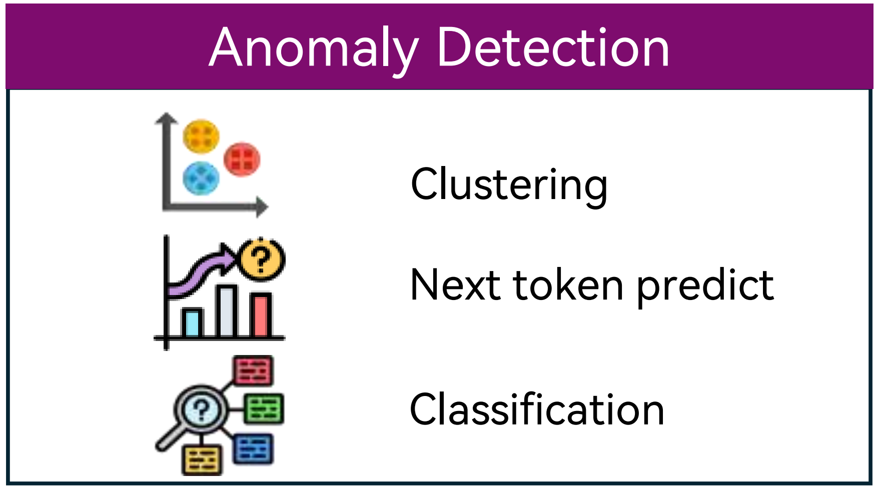
Log Parsing Log Grouping Log representation Anomaly Detection



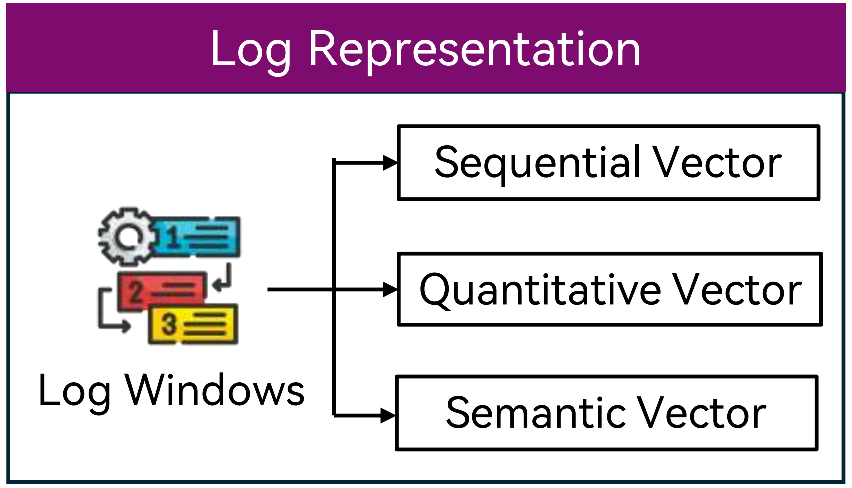
```
INFO CE sym 12, at 0x1b786f60, mask 0x04
Event1: CE sym <*>, at <*>, mask <*>
```





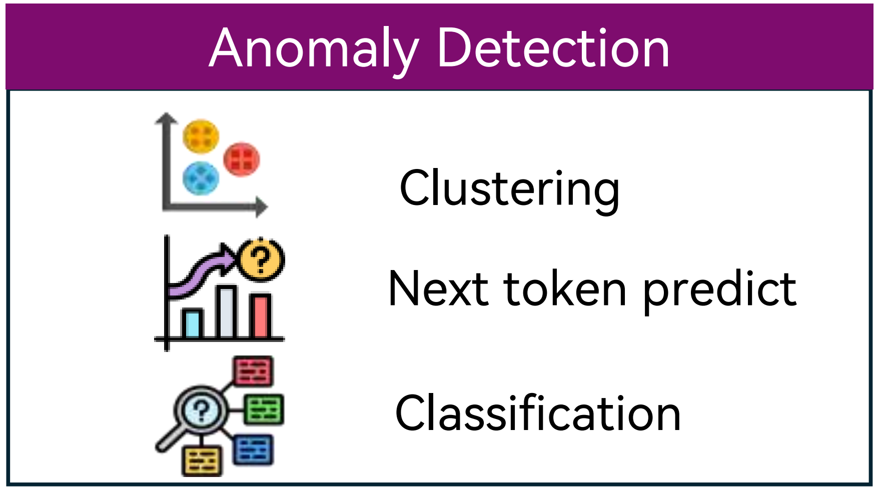


Background



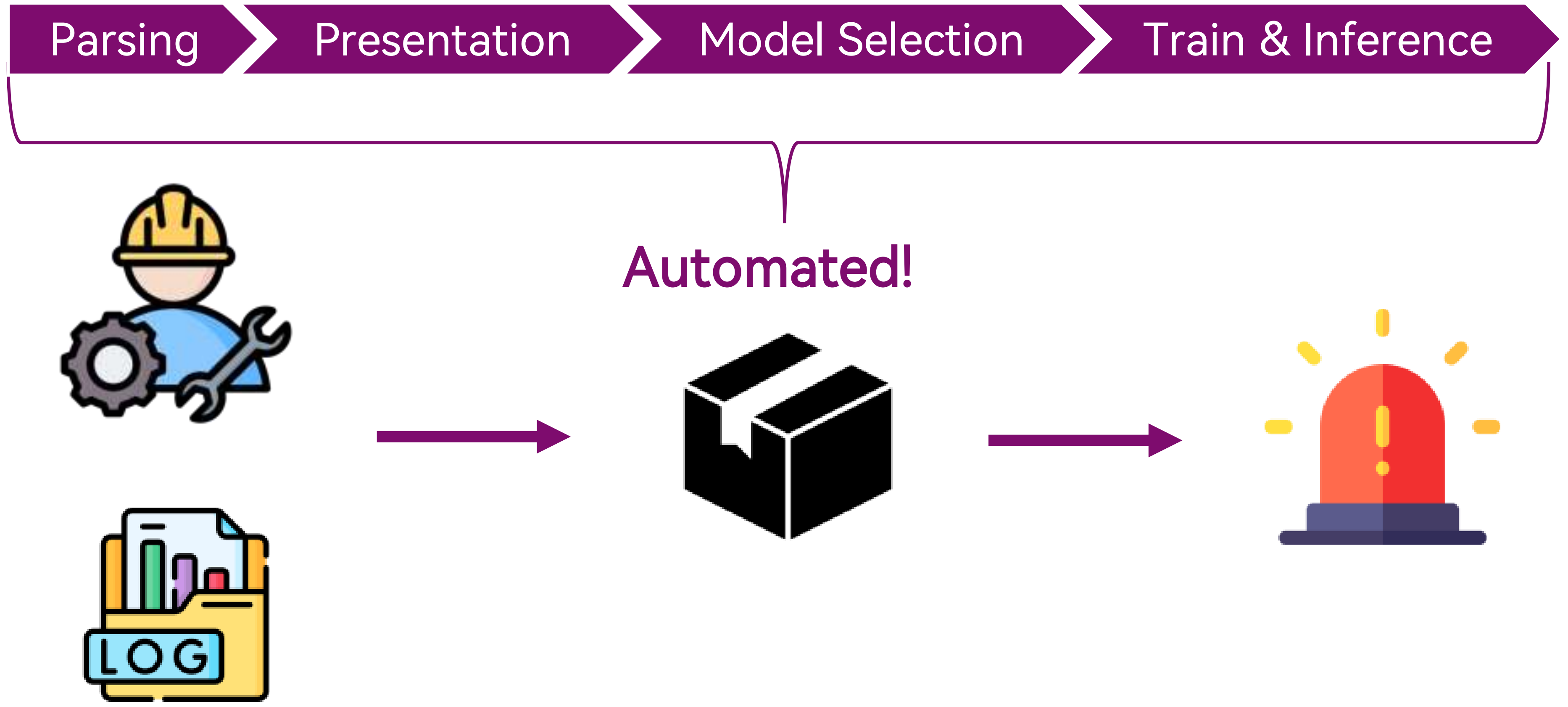
Which type of feature performs better in anomaly detection on the specified log dataset: sequence, quantity, semantics, or their combination?

How should existing anomaly detection methods be selected and deployed? How should their hyperparameters be determined?



Background

Our goal:



Background

Challenges:

Diversified datasets present challenges to feature engineering.

- The quality of template extraction and the chosen log representation methods significantly impact the final anomaly detection results.
- The significant differences between datasets lead to the need for deep manual involvement in the feature engineering process.

Background

Challenges:

Massive hyperparameter combinations and unlabeled data present challenges to model selection and evaluation.

- Each model has numerous hyperparameters, making model selection highly challenging.
- Model evaluation needs labeled data, making performance assessment difficult with unlabeled data.

Background

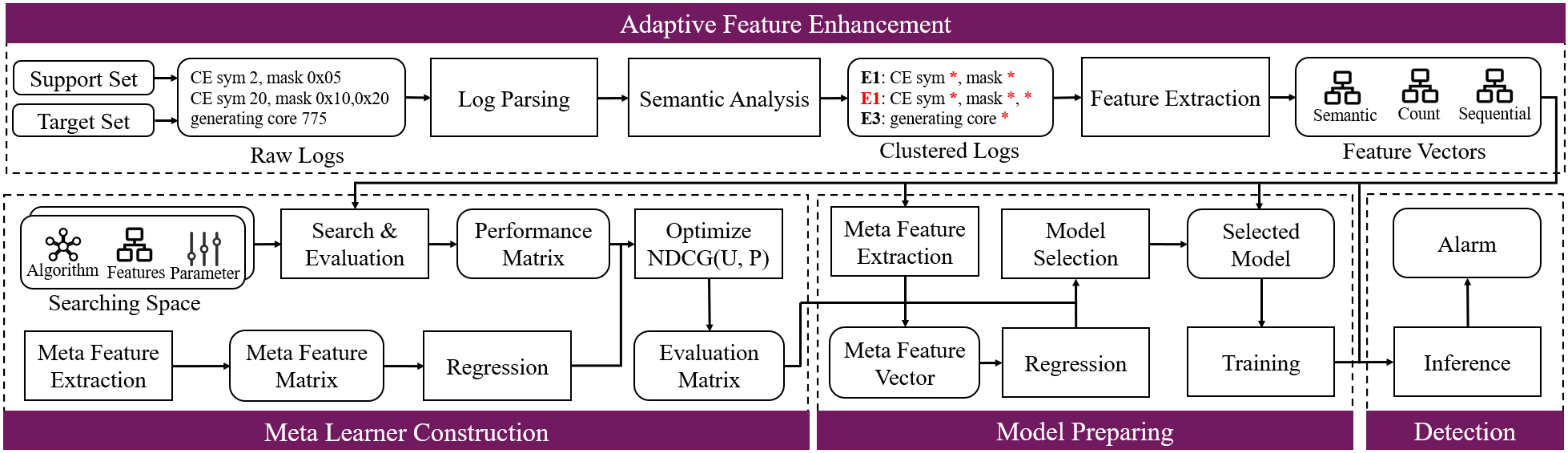
Design

Evaluation

Conclusion

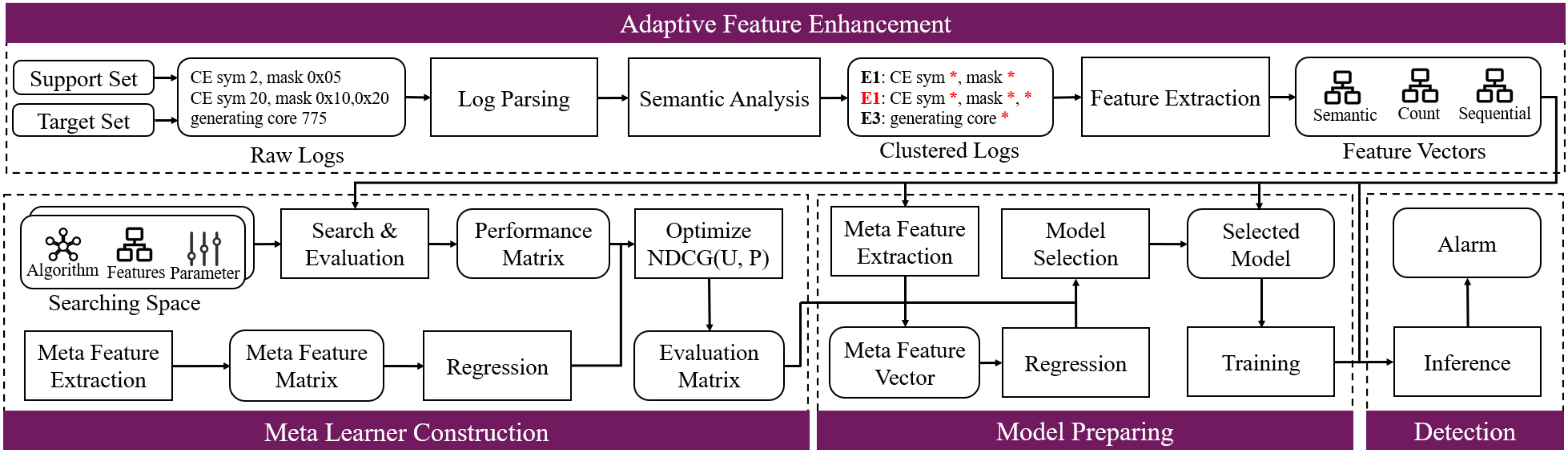
Design

Overview of LogCraft



Design

Core idea of LogCraft

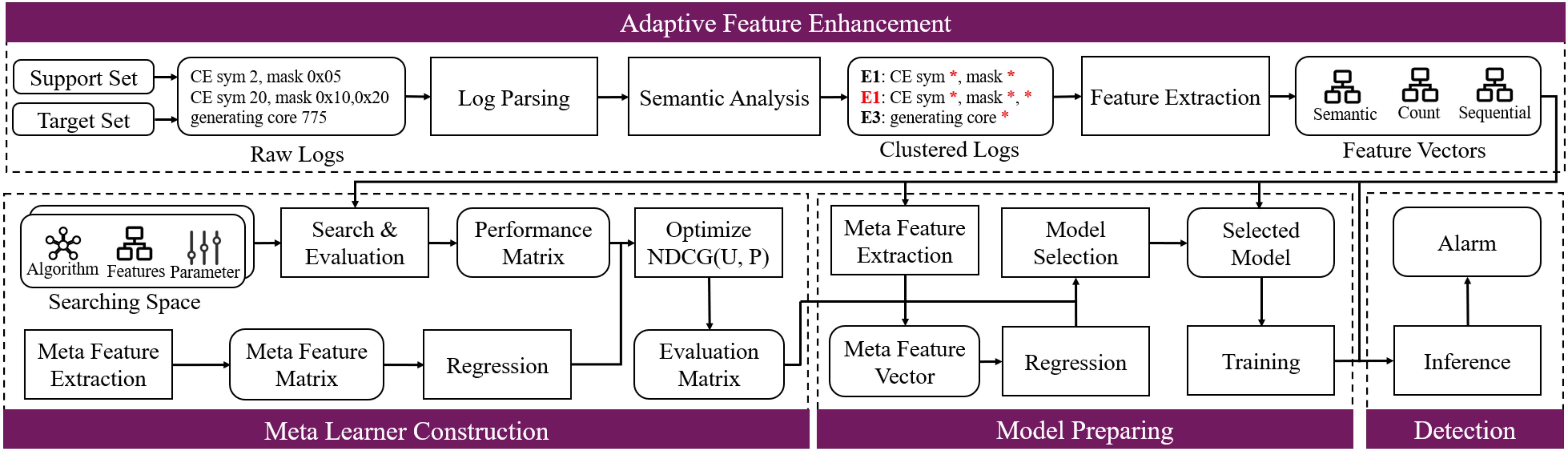


To address Challenge 1:

- Combine semantic analysis and clustering techniques to improve template parsing accuracy.
- Use feature combinations as one of the candidate set features.

Design

Core idea of LogCraft



To address Challenge 2:

- Design a meta-feature extractor and combine it with collaborative filtering techniques to recommend models from the candidate set for unlabeled datasets.

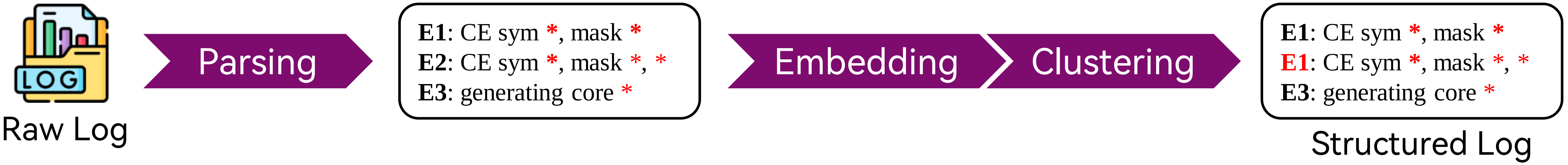
Adaptive Feature Enhancement

Traditional Log Parsing :

- Inconsistent variable lengths
- Lack of filtering rules
- Neglect of semantic information

I need to write appropriate regular expressions for each type of log and make multiple attempts to achieve reasonable results!

Or rather, through automated sentence vector encoding and clustering?



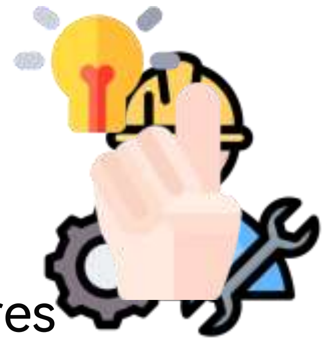
E1: CE sym *, mask *
E2: CE sym *, mask *, *
E3: generating core *

E1: CE sym *, mask *
E1: CE sym *, mask *, *
E3: generating core *

Adaptive Feature Enhancement

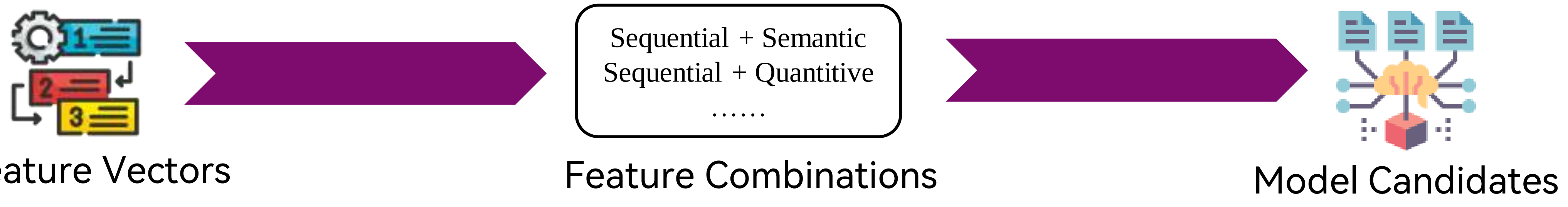
Log Engineering:

- Differences between datasets
- Difficult to identify key anomaly features



How should I choose the features for model training when the key characteristics that determine anomalies differ across various logs?

Incorporate combinations of features as part of the model's hyperparameters and include them in the candidate set!



Meta Learner Construction

To select a well-performing model for unlabeled log datasets, the results of the models on existing datasets are used for recommendation.

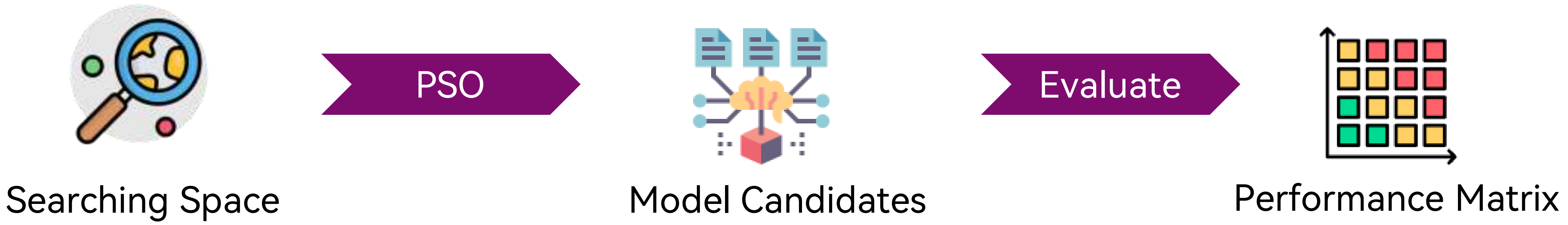
Two factors are essential:

Performance Evaluation

Similarity Measurement

Meta Learner Construction

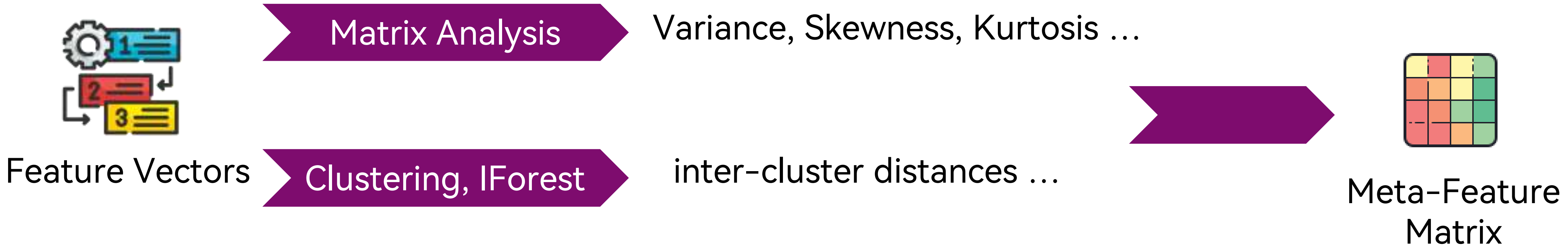
We selected four base algorithms and combined them with different hyperparameters to establish a candidate set containing tens of millions of models.



Next, we ran the particle swarm optimization algorithm on four labeled datasets, retaining potential candidates and recording their performance as Performance Matrix.

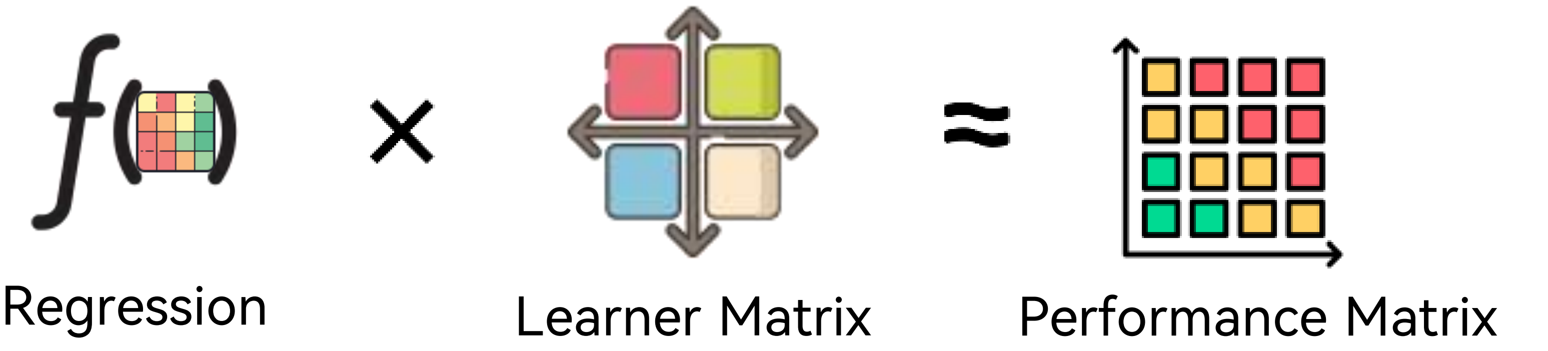
Meta Learner Construction

We creatively designed a meta-feature extractor for log data, which extracts fixed-length vectors from log datasets as their representation.



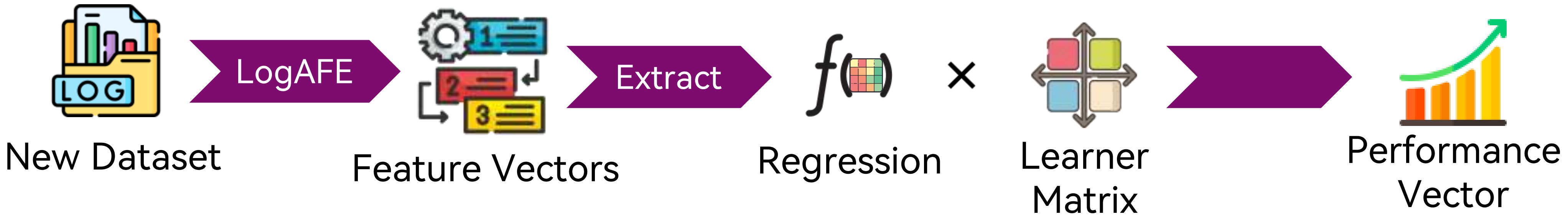
The meta-feature extractor is designed to extract two types of meta-features from datasets: statistical meta-features and model-based meta-features.

Meta Learner Construction



The goal of the meta-learner is to learn a regression function and a new matrix (referred to as the Learner Matrix), such that the dot product of the meta-feature matrix, after a linear transformation, with this matrix approximates the Performance Matrix.

Model Preparing



When new log data arrives, it is structured and associated with meta-feature extraction. The resulting meta-feature vector undergoes a linear transformation and is multiplied by the Performance Matrix, resulting in the Performance Vector. Each column of the vector represents the estimated performance of the corresponding model on that dataset.

Background

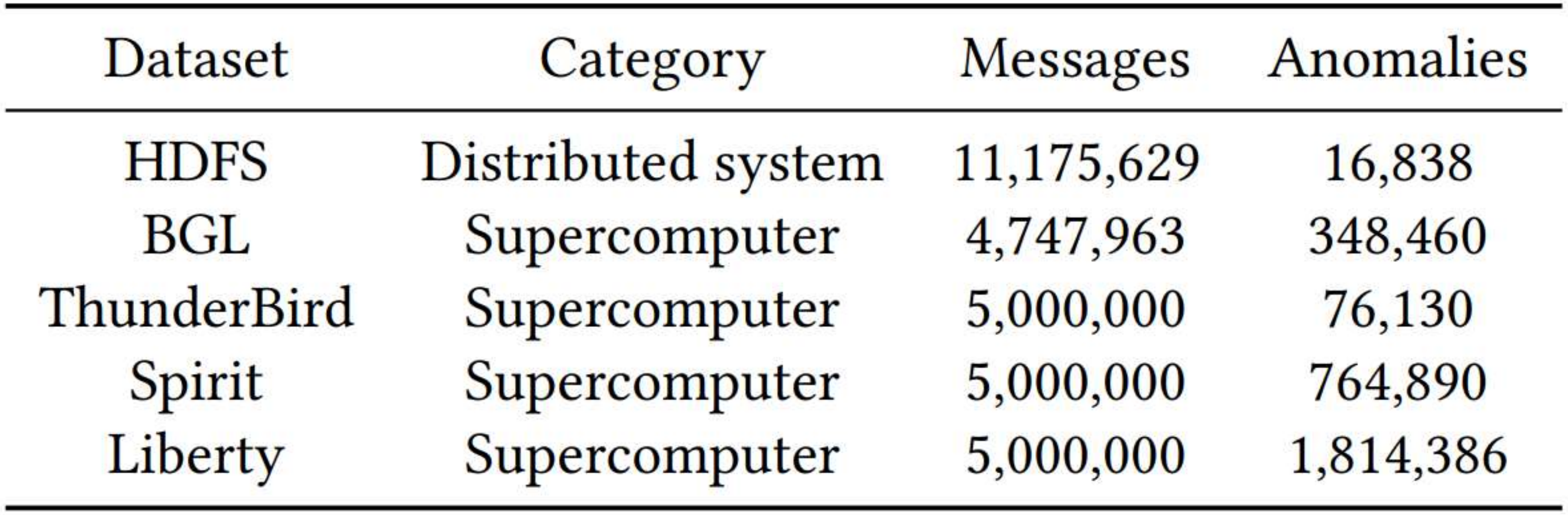
Design

Evaluation

Conclusion

Dataset

Detailed information of the datasets



Dataset	Category	Messages	Anomalies
HDFS	Distributed system	11,175,629	16,838
BGL	Supercomputer	4,747,963	348,460
ThunderBird	Supercomputer	5,000,000	76,130
Spirit	Supercomputer	5,000,000	764,890
Liberty	Supercomputer	5,000,000	1,814,386

Research Questions

RQ1: How effective is LogCraft in unsupervised log anomaly detection?

RQ2: How effective are the main components of LogCraft?

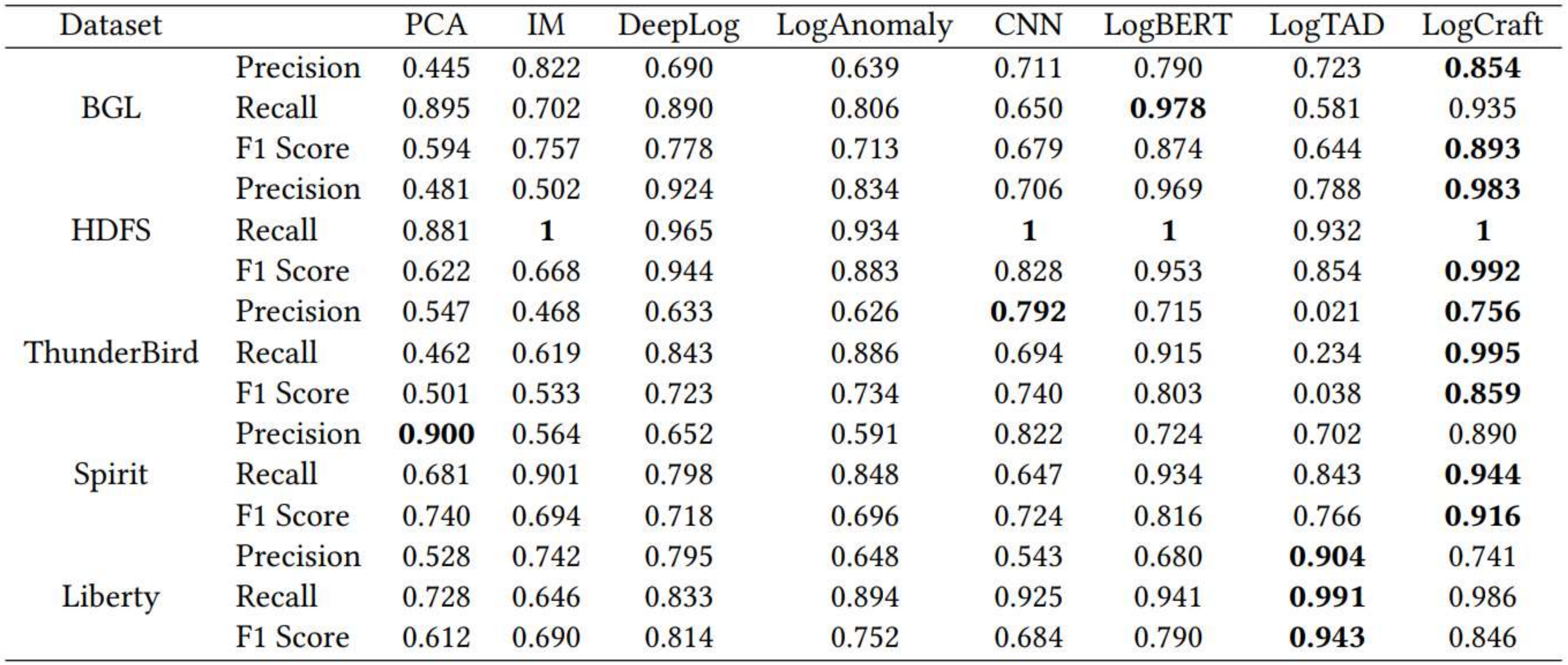
RQ3: How do hyperparameter settings affect the performance of LogCraft?

Research Questions

RQ1: How effective is LogCraft in unsupervised log anomaly detection?

RQ2:

RQ3:

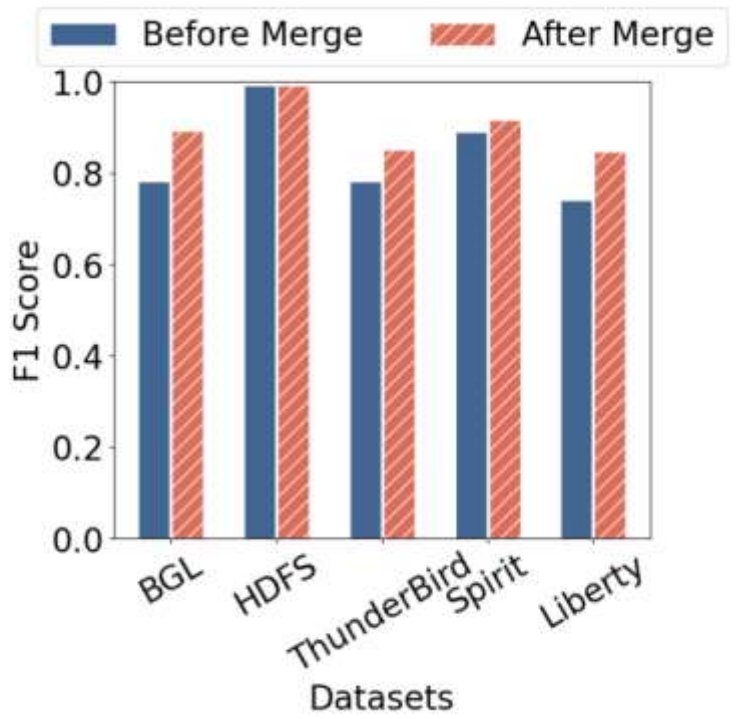


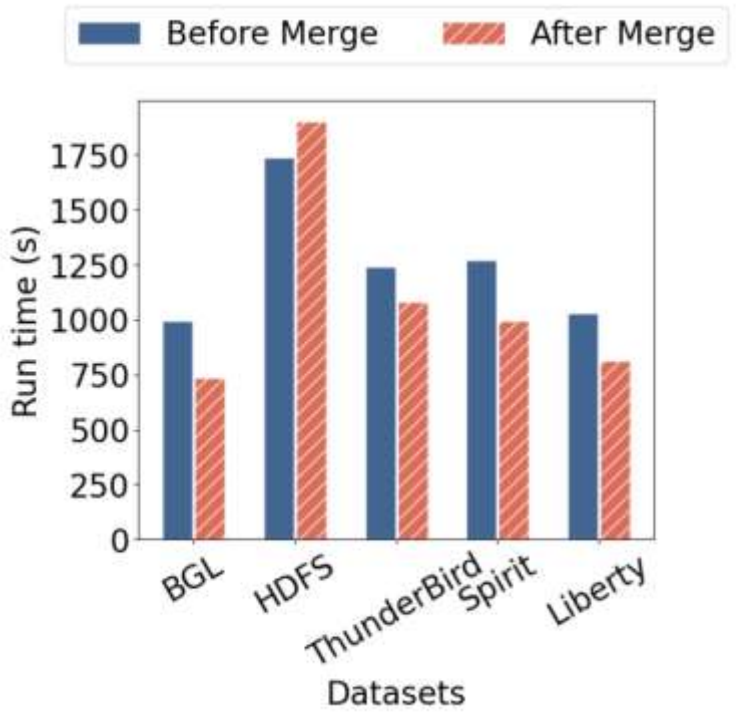
Dataset		PCA	IM	DeepLog	LogAnomaly	CNN	LogBERT	LogTAD	LogCraft
BGL	Precision	0.445	0.822	0.690	0.639	0.711	0.790	0.723	**0.854**
	Recall	0.895	0.702	0.890	0.806	0.650	**0.978**	0.581	0.935
	F1 Score	0.594	0.757	0.778	0.713	0.679	0.874	0.644	**0.893**
HDFS	Precision	0.481	0.502	0.924	0.834	0.706	0.969	0.788	**0.983**
	Recall	0.881	**1**	0.965	0.934	**1**	**1**	0.932	**1**
	F1 Score	0.622	0.668	0.944	0.883	0.828	0.953	0.854	**0.992**
ThunderBird	Precision	0.547	0.468	0.633	0.626	**0.792**	0.715	0.021	**0.756**
	Recall	0.462	0.619	0.843	0.886	0.694	0.915	0.234	**0.995**
	F1 Score	0.501	0.533	0.723	0.734	0.740	0.803	0.038	**0.859**
Spirit	Precision	**0.900**	0.564	0.652	0.591	0.822	0.724	0.702	0.890
	Recall	0.681	0.901	0.798	0.848	0.647	0.934	0.843	**0.944**
	F1 Score	0.740	0.694	0.718	0.696	0.724	0.816	0.766	**0.916**
Liberty	Precision	0.528	0.742	0.795	0.648	0.543	0.680	**0.904**	0.741
	Recall	0.728	0.646	0.833	0.894	0.925	0.941	**0.991**	0.986
	F1 Score	0.612	0.690	0.814	0.752	0.684	0.790	**0.943**	0.846

We compared the performance of LogCraft with six unsupervised log anomaly detection baselines: PCA, IM, DeepLog, LogAnomaly, CNN, and LogBERT.

Research Questions

RQ2: How effective are the main components of LogCraft?





RQ1:

RQ3:

g

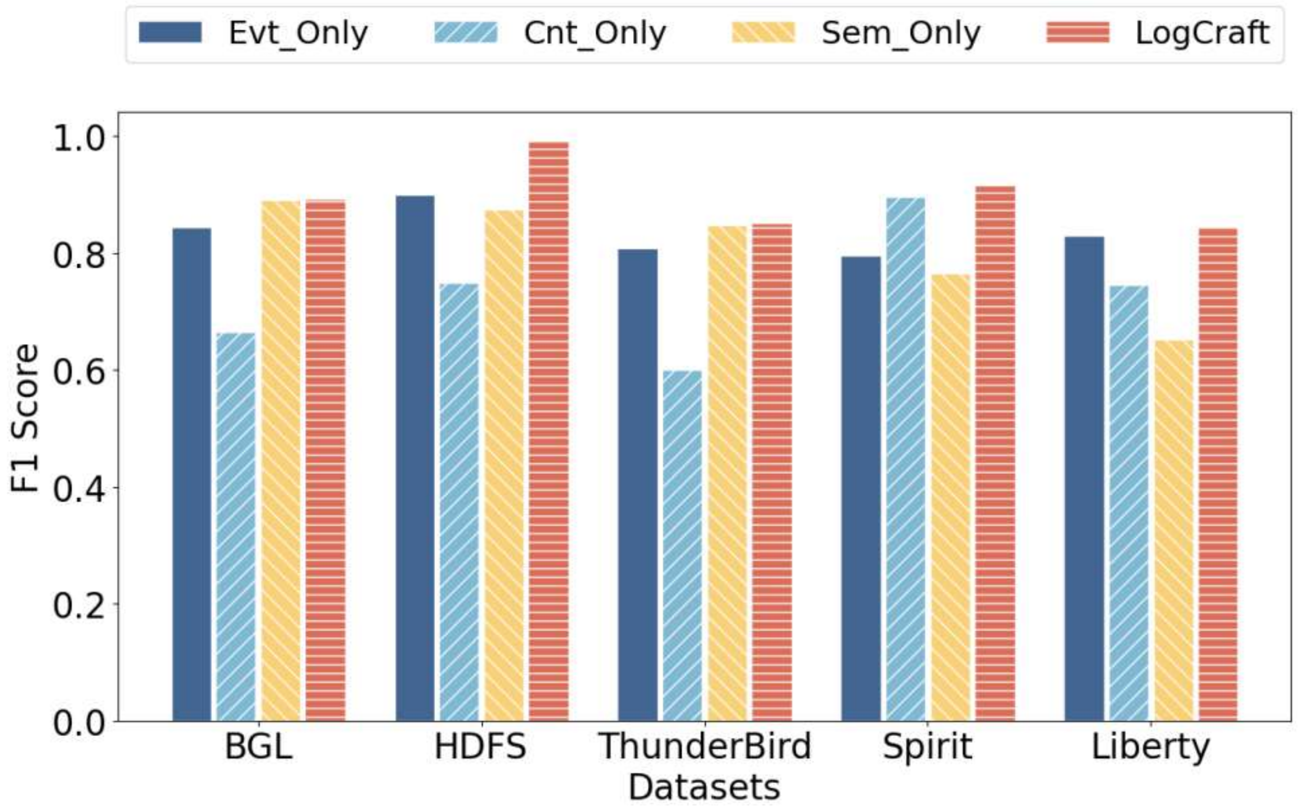
After template merging, the model's performance improved across all datasets, and the detection speed increased on four datasets.

Research Questions

RQ2: How effective are the main components of LogCraft?

RQ1:

RQ3:

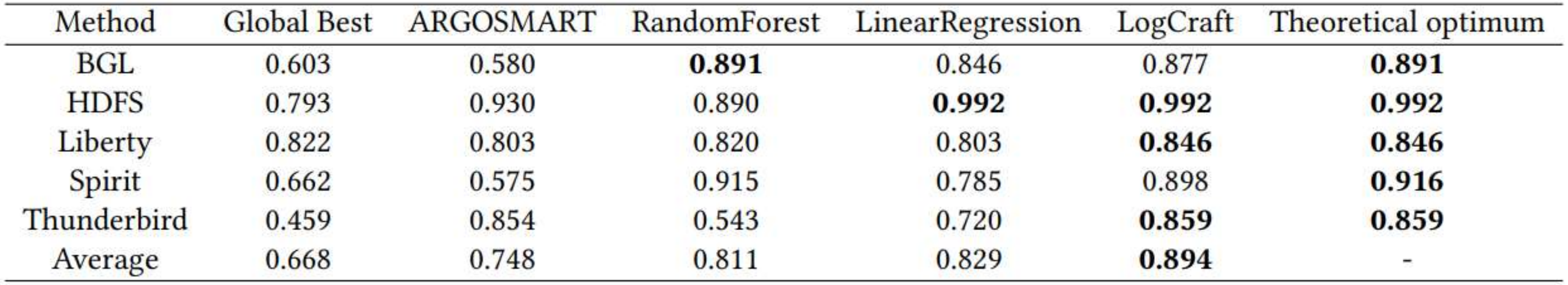


After considering feature combinations, LogCraft's performance either matched or exceeded that of the model which only considered single features across all datasets.

Research Questions

RQ2: How effective are the main components of LogCraft?

Table 8: F1 scores of the models with different recommendation algorithms



Method	Global Best	ARGOSMART	RandomForest	LinearRegression	LogCraft	Theoretical optimum
BGL	0.603	0.580	**0.891**	0.846	0.877	**0.891**
HDFS	0.793	0.930	0.890	**0.992**	**0.992**	**0.992**
Liberty	0.822	0.803	0.820	0.803	**0.846**	**0.846**
Spirit	0.662	0.575	0.915	0.785	0.898	**0.916**
Thunderbird	0.459	0.854	0.543	0.720	**0.859**	**0.859**
Average	0.668	0.748	0.811	0.829	**0.894**	-

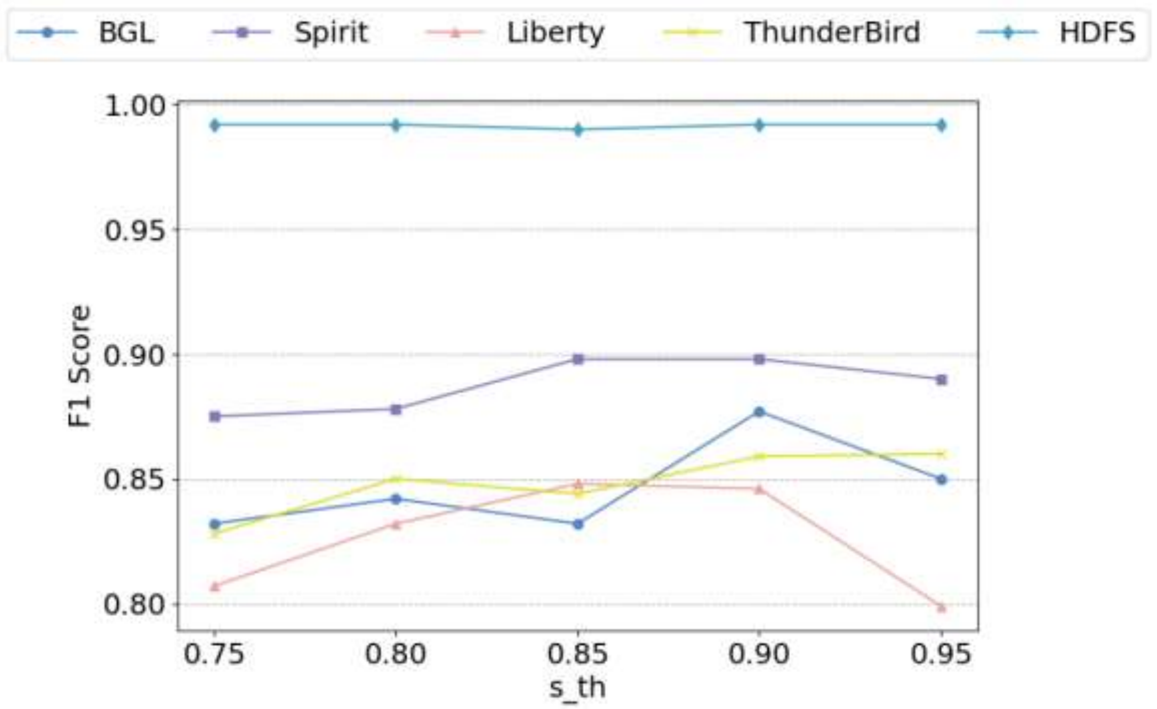
LogCraft outperformed other recommendation algorithms on four datasets and achieved the theoretical optimal value on three datasets.

Research Questions

RQ3: How do hyperparameter settings affect the performance of LogCraft?

RQ1:

RQ2:



LogCraft is designed as a black box, automatically selecting parameters. However, the operator can still influence the model's performance by modifying the similarity threshold (s_th) for template merging. We evaluated LogCraft's performance under different hyperparameter settings.

Background

Design

Evaluation

Conclusion

Conclusion

Log Data Representation Matters.

Meta Feature Potentials Log Analytics.

Conclusion

- This paper introduces LogCraft: an end-to-end unsupervised log anomaly detection framework based on AutoML
- LogCraft shown good performance on five public datasets with an average F1 score of 0.899, surpassing existing unsupervised detection algorithms.
- LogCraft represents the initial effort to derive fixed-dimensional vectors as latent feature representations from an entire log dataset.
- The meta-feature extractor we propose also demonstrates significant promise for assessing dataset similarity and advancing research in the field of log analytics.

Thanks