Use of AI for Log Analysis in CI/CD Pipelines

Bachelor Thesis - Defence

Maid Ališić

23 July 2025

University of Applied Sciences Upper Austria, Campus Hagenberg

Problem Context

Why do we care?

- CI/CD emits $\approx$ 10-20 GB build, test & deploy logs every day.

Why do we care?

- CI/CD emits $\approx$ 10-20 GB build, test & deploy logs every day.
- Manual grepping is slow $\rightarrow$ merge queue stalls, silent faults creep in.

Why do we care?

- CI/CD emits $\approx$ 10-20 GB build, test & deploy logs every day.
- Manual grepping is slow $\rightarrow$ merge queue stalls, silent faults creep in.
- Service-level objective: deliver a verdict inside ≤ 200 ms.

Why do we care?

- CI/CD emits $\approx$ 10-20 GB build, test & deploy logs every day.
- Manual grepping is slow $\rightarrow$ merge queue stalls, silent faults creep in.
- Service-level objective: deliver a verdict inside ≤ 200 ms.
- Logs can contain customer identifiers $\rightarrow$ **no SaaS export**.

Operational pain points

1. **Context sensitivity** - same token can be harmless or fatal.

Operational pain points

1. **Context sensitivity** - same token can be harmless or fatal.
2. **Concept drift** - every merge may rename tests or flags.

Operational pain points

1. **Context sensitivity** - same token can be harmless or fatal.
2. **Concept drift** - every merge may rename tests or flags.
3. **Latency pressure** - analysis must end before runner teardown.

Operational pain points

1. **Context sensitivity** - same token can be harmless or fatal.
2. **Concept drift** - every merge may rename tests or flags.
3. **Latency pressure** - analysis must end before runner teardown.
4. **Alert fatigue** - static regex rules explode over time.

Research Questions

Research questions (thesis)

$\mathbf{RQ_{main}}$ How can AI improve *accuracy & explainability* of CI/CD log analysis?

Research questions (thesis)

$\mathrm{RQ}_{\mathrm{main}}$ How can AI improve *accuracy & explainability* of CI/CD log analysis?

RQ1 To what extent can an **LLM service** (ChatGPT API) automate log interpretation?

Research questions (thesis)

$\mathbf{RQ_{main}}$ How can AI improve *accuracy & explainability* of CI/CD log analysis?

RQ1 To what extent can an **LLM service** (ChatGPT API) automate log interpretation?

RQ2 How does a **local** stack (Isolation-Forest + Random-Forest) compare with the cloud LLM?

Research questions (thesis)

RQ_{main} How can AI improve *accuracy & explainability* of CI/CD log analysis?

RQ1 To what extent can an **LLM service** (ChatGPT API) automate log interpretation?

RQ2 How does a **local** stack (Isolation-Forest + Random-Forest) compare with the cloud LLM?

RQ3 Which practical hurdles arise when embedding both approaches into existing pipelines?

Method - Feature Engineering

Vectorisation step ① - TF-IDF

1. **Normalisation:** remove timestamps, colours, IDs; lowercase.

Vectorisation step ① - TF-IDF

1. **Normalisation:** remove timestamps, colours, IDs; lowercase.
2. **Tokenise** + generate 1-2-grams.

Vectorisation step ① - TF-IDF

1. **Normalisation:** remove timestamps, colours, IDs; lowercase.
2. **Tokenise** + generate 1-2-grams.
3. Compute weights:

$$w_{t,d} = \mathrm{tf}_{t,d} \cdot \log \frac{N}{\mathrm{df}_t}$$

$w_{t,d}$ - weight of term t in document d
$\mathrm{tf}_{t,d}$ - raw term frequency in d
N - total number of documents (log lines)
df_t - number of documents that contain t

Vectorisation step ① - TF-IDF

1. **Normalisation:** remove timestamps, colours, IDs; lowercase.
2. **Tokenise** + generate 1-2-grams.
3. Compute weights:

$$w_{t,d} = \mathrm{tf}_{t,d} \cdot \log \frac{N}{\mathrm{df}_t}$$

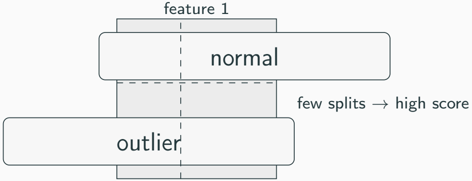
$w_{t,d}$ - weight of term t in document d
$\mathrm{tf}_{t,d}$ - raw term frequency in d
N - total number of documents (log lines)
df_t - number of documents that contain t

4. Result: 50 k-dimensional sparse matrix; $\approx$ 100 000 lines /s on a single core.

Method - Algorithm ②

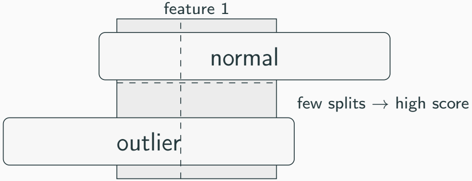
How Isolation-Forest detects outliers

- Randomly split feature space into binary trees.



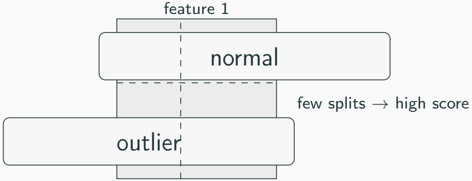
How Isolation-Forest detects outliers

- Randomly split feature space into binary trees.
- Rare / unusual points are isolated *early*. Path length $h(x)$ $\rightarrow$ anomaly score $s(x) = 2^{-h(x)/c(n)}$.



How Isolation-Forest detects outliers

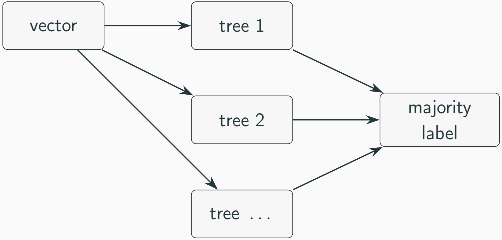
- Randomly split feature space into binary trees.
- Rare / unusual points are isolated *early*. Path length $h(x)$ $\rightarrow$ anomaly score $s(x) = 2^{-h(x)/c(n)}$.
- CPU-friendly: 30 µs per line, no GPU.



Method - Algorithm ③

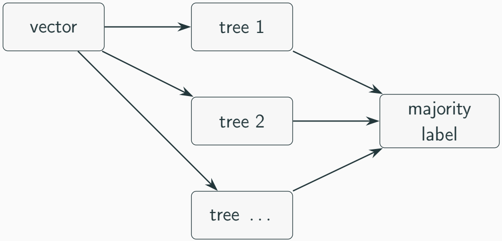
Why add a Random-Forest classifier?

1. Isolation-Forest says *"something odd"*. Operators still ask *"why?"*



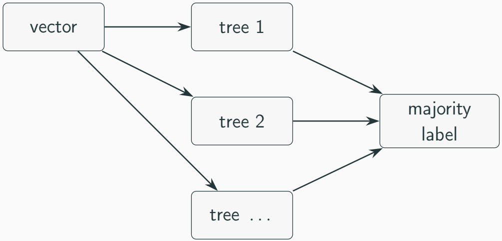
Why add a Random-Forest classifier?

1. Isolation-Forest says *"something odd"*. Operators still ask *"why?"*
2. Random-Forest majority vote assigns one of seven error categories.



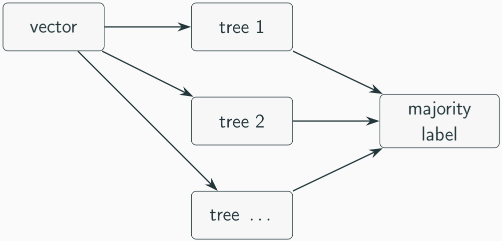
Why add a Random-Forest classifier?

1. Isolation-Forest says *"something odd"*. Operators still ask *"why?"*
2. Random-Forest majority vote assigns one of seven error categories.
3. Per-token feature importance allows future visual highlights.



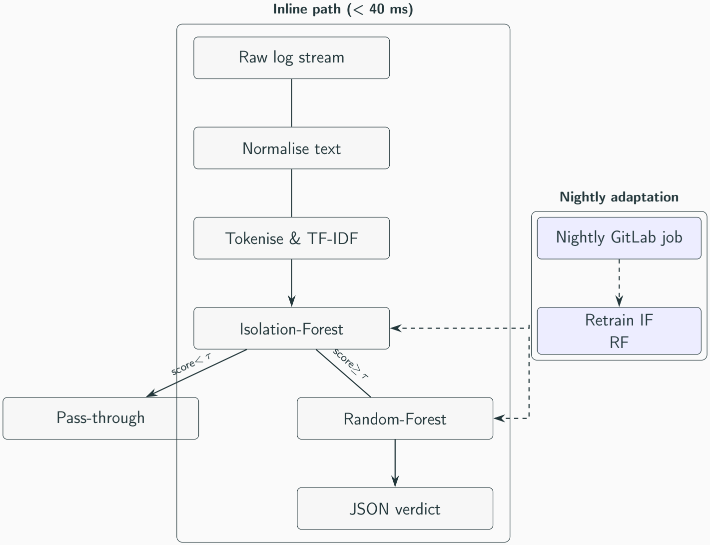
Why add a Random-Forest classifier?

1. Isolation-Forest says *"something odd"*. Operators still ask *"why?"*
2. Random-Forest majority vote assigns one of seven error categories.
3. Per-token feature importance allows future visual highlights.
4. Nightly training < 90 s (400 trees, depth 30).



Architecture

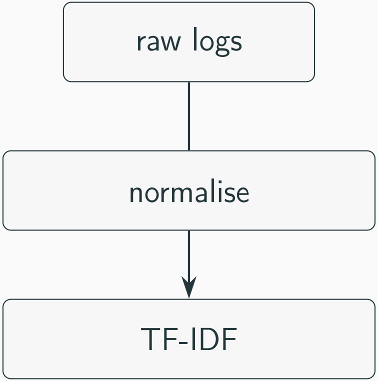
End-to-end pipeline ($<$ 40 ms inline)



Data & Experimental Design

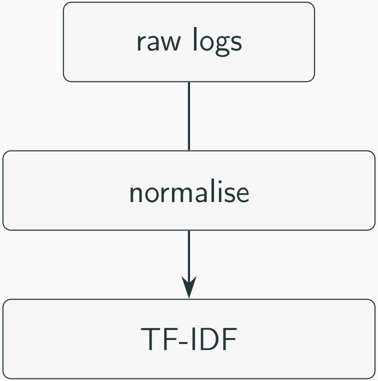
Datasets & metrics

- 117 k macOS system log lines



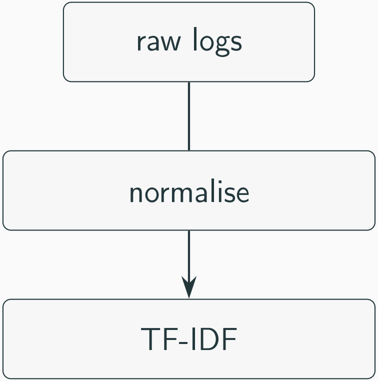
Datasets & metrics

- 117 k macOS system log lines
- 655 k OpenSSH authentication lines



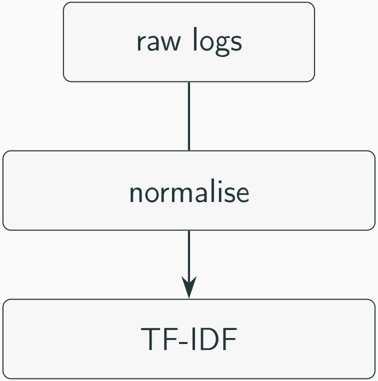
Datasets & metrics

- 117 k macOS system log lines
- 655 k OpenSSH authentication lines
- 504 ground-truth anomalies



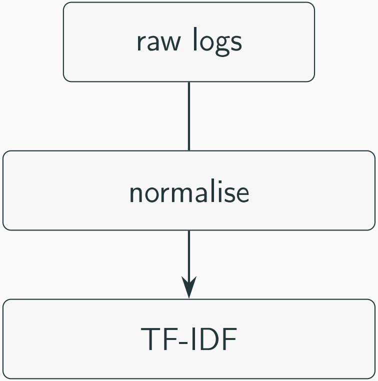
Datasets & metrics

- 117 k macOS system log lines
- 655 k OpenSSH authentication lines
- 504 ground-truth anomalies
- Split 70 / 15 / 15 % (train / validation / test)



Datasets & metrics

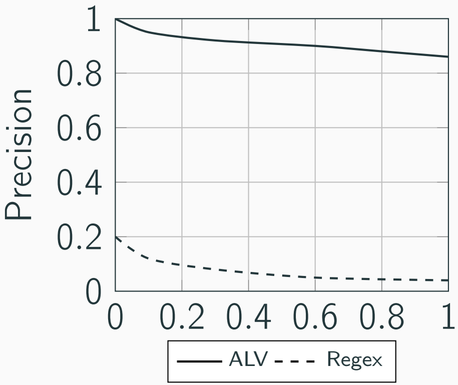
- 117 k macOS system log lines
- 655 k OpenSSH authentication lines
- 504 ground-truth anomalies
- Split 70 / 15 / 15 % (train / validation / test)
- Metrics: Macro-F_1, Area Under the Precision-Recall Curve (AUPRC), 99.9 % latency percentile



Results

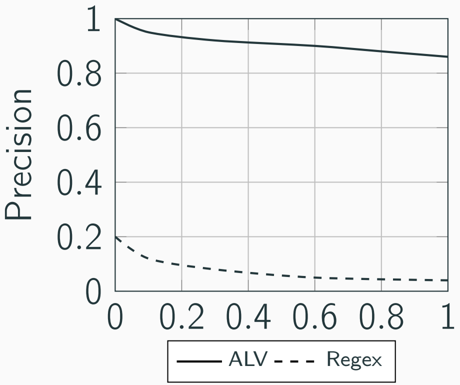
Headline numbers

- Detection (Isolation-Forest): $\mathbf{F_1} = \mathbf{0.89}$
  AUPRC = 0.94



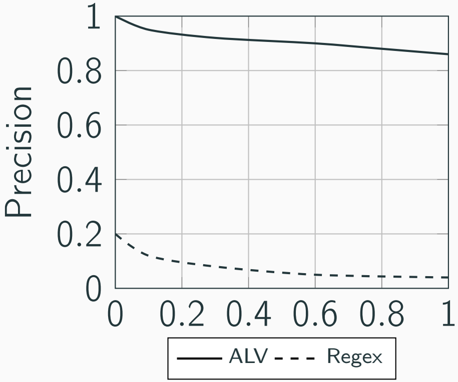
Headline numbers

- Detection (Isolation-Forest): $\mathbf{F_1} =$ **0.89**
  AUPRC = 0.94
- Diagnosis (Random-Forest): Macro-F_1 = **0.99**,
  Matthews Correlation Coefficient = 0.98



Headline numbers

- Detection (Isolation-Forest): $\mathbf{F_1} =$ **0.89**
  AUPRC = 0.94
- Diagnosis (Random-Forest): Macro-F_1 = **0.99**,
  Matthews Correlation Coefficient = 0.98
- Latency: 37 ms (p99.9); throughput 45 k lines /s



Impact

Organisation-level benefits

- Minutes $\rightarrow$ Milliseconds: failures surface during the same pipeline run.

Organisation-level benefits

- Minutes $\rightarrow$ Milliseconds: failures surface during the same pipeline run.
- **Zero token fees**: 2300 Lines of Code + FastAPI; vendor neutral.

Organisation-level benefits

- Minutes $\rightarrow$ Milliseconds: failures surface during the same pipeline run.
- **Zero token fees**: 2300 Lines of Code + FastAPI; vendor neutral.
- Data never leaves the VPN - compliant with GDPR & NDAs.

Organisation-level benefits

- Minutes $\rightarrow$ Milliseconds: failures surface during the same pipeline run.
- **Zero token fees**: 2300 Lines of Code + FastAPI; vendor neutral.
- Data never leaves the VPN - compliant with GDPR & NDAs.
- Deterministic labels enable auditable root-cause trails.

Limitations & Roadmap

Limitations & next steps

- Only seven error classes - grow ontology as label volume rises.

Limitations & next steps

- Only seven error classes - grow ontology as label volume rises.
- First Drain template pass is manual (cold-start issue).

Limitations & next steps

- Only seven error classes - grow ontology as label volume rises.
- First Drain template pass is manual (cold-start issue).
- Explainability: token-level *SHapley Additive ExPlanations* and *Integrated Gradients* still on the to-do list.

Limitations & next steps

- Only seven error classes - grow ontology as label volume rises.
- First Drain template pass is manual (cold-start issue).
- Explainability: token-level *SHapley Additive ExPlanations* and *Integrated Gradients* still on the to-do list.
- Roadmap: threshold-free Bayesian change-point detection, distilled LogBERT encoder, federated nightly retrain.

Wrap-up

Take-away

Light-weight, on-prem ML matches AIOps SaaS

without latency, cost or data-protection pain.

Questions welcome - thank you!