

# CONFUSE: CONFIDENCE-GATED MULTIMODAL FUSION FOR ANOMALY DETECTION AND ROOT-CAUSE ATTRIBUTION IN MICROSERVICE SYSTEMS

<sup>1</sup> Kadiyam Baladithya, <sup>2</sup> V.V.S. Sasank

<sup>1</sup> MTech Student, Department of Computer Science and Engineering, KL University, Vaddeswaram, AP, [kadiyambalu@gmail.com](mailto:kadiyambalu@gmail.com)

<sup>2</sup> Assistant Professor, Department of Computer Science and Engineering, KL University, Vaddeswaram, AP, [vvssasank@kluniversity.in](mailto:vvssasank@kluniversity.in)

## Abstract

A Microservice failures rarely occur through a singular observable signal. Microservices failures typically feature increasing CPU usage in one container, an out-of-memory log message in another, and dropping spans in yet another one. It is common practice to inspect metrics, logs, and traces together. Nevertheless, most published detectors treat the three types of streams as equal in credibility at any point in time, which is wrong. Prometheus scrapes can be subject to jitter. Loki ships can crash. A delay fault will prevent Jaeger from receiving the full trace. When a model combines point embeddings with fixed weights, the faulty stream will displace the fused vector from the decision boundary leading to either delayed or erroneous alarms. The research proposes a ConFuse a detector that encodes the individual modalities as Gaussian latent variables instead of point estimates, and controls fusion based on inverse predictive variance. Temporal convolutional network with gated recurrent head produces metric uncertainty estimates. The template extracted from logs using Drain algorithm, embedded using frozen sentence transformer, and aggregated with attention mechanism produces the estimate of log uncertainty. Graph attention network using the dependency graph of the services predicts the trace uncertainty. Four losses are minimized simultaneously: anomaly classification, supervised contrastive alignment of three latent spaces, an evidence lower bound for the probabilistic encoders, and a temperature-scaled calibration loss ensuring gate integrity at inference. Attribution analysis is provided by the last layer of graph attention network, there is no need to train a second-order explainer. On RCAEval windows, this model reaches precision-recall-F1 triplets of 98.6 / 98.1 / 98.3 for Online Boutique, 97.8 / 97.2 / 97.5 for Train Ticket, and 99.1 / 98.7 / 98.9 for Sock Shop, achieving an average F1 of 98.2. This corresponds to a gain of 0.6 points compared to the strongest uncertainty-aware baseline on the same dataset split, and a gain of seven to ten points compared to classical multimodal methods. The F1 drops by 2.2 points under 20 percent random modality dropout and Gaussian metric noise. The mean delay after injection is 78 seconds. The network architecture contains 18.7M trainable parameters and provides 60-second window processing time of 180 ms on CPU. The gate mechanism, not the backbone architecture, preserves accuracy under incomplete observability.

**Keywords:** microservice anomaly detection, multimodal fusion, predictive uncertainty, root-cause attribution, observability, calibration

**How to cite this article:** 1 Kadiyam Baladithya 2 V.V.S. Sasank. CONFUSE: CONFIDENCE-GATED MULTIMODAL FUSION FOR ANOMALY DETECTION AND ROOT-CAUSE ATTRIBUTION IN MICROSERVICE SYSTEMS. Int J Drug Deliv Technol. 2026;16(83s):474-486. DOI: 10.25258/ijddt.16.83s.53.

## 1 INTRODUCTION

Online boutique checkout flows, ticket reservations, and sock shop carts are no longer standalone systems. They work as interrelated graphs of little services communicating over the network, generating logs in diverse formats, and exposing metrics over different scraping frequencies. In cases of failure, the first visible effect doesn't necessarily point to the actual cause of the failure. For example, a memory leak in the payment service system may start with increasing resident set size, then spike out-of-memory messages, and finally lead to time-outs in downstream spans awaiting responses. A monitoring system that focuses on one metric only will likely ignore the first two effects. A logging-only monitoring tool will most probably be unable to detect a slow leak without logging events. A tracing-based monitor will be unable

to detect failures that don't affect latency. In the past five years, researchers have gradually bridged this gap by incorporating all three data streams into a single model. The state-of-the-art end-to-end frameworks carry out detection and localization in one forward pass. Graphical structures are learned across modalities, and uncertainty-aware encoders are being developed. The reported F1 scores in the lab for clean traces have been very high, and there have been a few that surpassed 97 percent. However, these scores are true only under an assumption that all modalities are arriving at the same time and with the same quality of those in the training data set. This is not always the case in real-life scenarios where, in the event of a lagging, corrupted, or missed stream, a fixed-weighted fusion strategy fails to exclude the stream. This is the

aim of the current research. The ConFuse system uses sixty-second sliding windows of metrics, logs, and traces for analysis. In this way, each encoder provides a mean and a variance. The weights for fusion are obtained through a gated softmax over inverse variances, and therefore, any modality which has an increasing variance without bounds will be automatically discounted. The forward pass generates both a calibrated anomaly probability and a list of services based on graph attention. The model is small enough to be trained on one GPU in one day and efficient enough to analyze the window on CPU in less than a second. The proposed experimental design includes four goals that stem from the described gap and not from some set of requirements. The first goal is to treat the latent structure of observability streams as probabilistic entity, thus making noise observable to the whole network, not averaged. The second goal is to incorporate these latent models with a confidence gate, which is activated when the stream is missing or corrupted, and that does not include any special handling of drops. The third goal is to learn the gate so that it will be properly calibrated during test time, and for that purpose one should combine alignment, variational regularizer and calibration loss into a single loss function instead of using classification loss only. The fourth goal is to keep a fully deployable pipeline for on-call teams with a public benchmark where there is no information leakage of future into the past, the source of attribution comes from the model, and the inference time is limited to sixty seconds.

## 2 LITERATURE REVIEW

### 2.1 Metric-centric and trace-centric localisers

Original practical localizers have been based on metrics or traces alone. TraceRCA ranks the service based on how often the abnormal traces pass through it [12]. MicroRank combines the spectrum scoring technique with trace graph [13]. BARO detects change points in multivariate metrics using Bayesian online detection and then ranks the causes based on a nonparametric test without requiring perfect alignment between anomaly time stamp [4]. DéjàVu learns from historical recurring failures as well as dependency graph and then ranks the components in the new failure [5]. GROOT creates the causal event graph and runs a variant of PageRank over the graph [23]. They are known to be fast and interpretable for on-call engineers. The main drawback of these systems is their blindness to the faults which have been self-annotated in log lines or when there is absence of a span prior to the appearance of KPI. For example, memory leak which initially appears as an out-of-memory line cannot be detected by BARO until the resident-set series adjusts itself. Network loss fault which initially appears as an incomplete trace cannot be detected by DéjàVu until there is a change in a metric. ConFuse

retains the metric channel but assigns it into three channels based on its weight which decreases with increasing noise in the series.

### 2.2 Trace and log models that still omit metrics

However, trace-plus-log methods can partially overcome the deficiency mentioned above. DeepTraLog builds a graph from the trace and its corresponding logs and then applies deep SVDD detector [7] on the graph. In this way, the researchers managed to show that those two sources of data cannot be used interchangeably. By combining the call path and the concurrent log events, one may detect faults which can be detected by either of them. Nevertheless, this model still does not include metrics. As a result, it becomes hard to detect quiet resource leaks which do not affect the call path at all. The same drawback occurs when a detector uses the trace as the only structure of the system. Resource faults usually come before Prometheus in the fault chain. Thus, ConFuse retains the metric encoder despite having latency in the graph attention network.

### 2.3 Multimodal pipelines with deterministic fusion

In the following wave, metrics, logs, and traces are included in one pipeline. Eadro jointly learns detection and localization tasks for all three types of inputs while sharing the feature between the two tasks [1]. Nezha uses multimodal information to classify the detailed cause without a large amount of supervision [8]. DiagFusion learns instance multimodal events embeddings and creates the dependency graph from the deployment and trace data and then uses graph networks to learn the failure instance and its failure type [9]. AnoFusion detects the instance failures without labels via fusing three streams [10]. MSTGAD builds a twin graph of the service instance and conducts cross-modal attention [11]. PDiagnose transforms the three streams into aligned series and votes for results [24]. CloudRCA fuses KPIs, logs, and traces within a Bayesian network framework [22]. FAMOS has an independent feature extractor for each modality and fuses them with Gaussian attention plus cross-attention [14]. AMulSys introduces multimodal augmentation of class balanced instances together with hybrid graphs [15]. MADMM builds the separate feature graph of metrics and logs in each sliding window [16]. MAD-CMC clusters contrastive multimodal representation in a self-supervised manner [17]. DAM focuses on multimodal sequence and builds the dynamic threshold [18]. FusionLCT builds temporal convolution over metrics and light-weight transformers over logs, then fuses them in stepwise manner [19]. MGFusion works on Grassmann manifolds [20]. MSNGAD recovers nested graphs with diffusion model [21]. MULAN learns the multimodal causal structure with contrastive factor and graph traversal [2].

## 2.4 Uncertainty-aware multimodal detection and public benchmarks

Uncertainty-aware fusion can be seen as a modern solution to the second claim. MUAD incorporates intra-modality noise using a graph probabilistic encoder and assigns different weights to the modalities based on prediction confidence, followed by active learning when data is lacking labels [3]. MUAD shows an average F1 score of 98.10 percent in its evaluation on its three separate datasets. Two things have yet to be settled. First, the score was not obtained from RCAEval, requiring a re-run to make it comparable. Second, the confidence measure was generated from the prediction head and not from encoder level variance, meaning that one cannot observe the three weights moving together in a way the collector would fail to do. In this literature, the role of temperature has become a secondary one as a surrogate of uncertainty in contrastive learning [25]. ConFuse employs temperature in two ways that can be recognized from this literature: as the softmax temperature of the fusion gate and as the scaling factor of the anomaly logit before the calibration penalty. Both are not replacements for a variance head but preserve the gate and make probability interpretable. RCAEval The community is currently provided by RCAEval with a public suite containing 735 failures in the Online Boutique, Sock Shop, and Train Ticket collections, covering both metric-only and multi-source configurations, as well as a baseline harness that can be reproduced [6]. The official evaluation metric in the suite is AC@k localisation accuracy at top-k, not F1 on a window level. A streaming monitor is still needed to provide a decision for each window. ConFuse comes last in the order of the three mentioned categories, adopting the public suite from the benchmark paper. Differences between ConFuse and MUAD include the source of confidence and the metrics evaluated under stress. Variance is calculated for each individual encoder. Training of the gating model is done against a calibration term, making sure that the output weights remain interpretable. Evaluation of the system gives clean F1 and F1 with modality dropout and metric noise on RCAEval windows.

## 3 SYSTEM METHODOLOGY

### 3.1 Working procedure

ConFuse is a sliding window monitor. At intervals of ten seconds, it consumes the latest sixty-second window from Prometheus time series data, Loki log lines, and Jaeger traces data. It assesses the window and generates four results: the anomaly signal, the adjusted probability, the modality weights triple, and a sorted list of services. The real-time path does not wait for the future window to come nor does it draw from latent Gaussian distributions. The sufficient statistics

include the mean and the inverse variance. The operation sequence is known in advance. Drain pre-processes the raw lines using comparison with a template tree which is fixed from the training split. Unknown templates get mapped into one unknown token. Metric channels are resampled to ten second intervals; if the data point is missed, it is treated as missing, rather than being interpolated. Spans are binned by service as latency quantiles, timeout ratio and queue depth and placed in the static call graph which comes with the benchmark. The three processed tensors are separately passed through three encoders. Each of them produces a mean vector and a log-variance vector of the same dimension. A simple gating network maps the three inverse variances to three simplex values. The fused mean is then used for classification by a two-layer classifier. Graph-attention weights which were produced in the trace encoder are averaged and used as the attribution values. The data that feed this path come from RCAEval. Online Boutique, Train Ticket and Sock Shop were driven under concurrent synthetic load and broken by scheduled fault injection. Prometheus, Loki and Jaeger recorded the three streams. The public release is what authenticates the labels: each failure case names the injected service and the injected fault family. This paper does not introduce a fourth system. It windows the public traces at sixty seconds with a ten-second stride and trains on the earlier part of each recording. Figure 1 is the implementation, not a sketch of intent. Every block in it has a corresponding tensor in the subsections that follow.

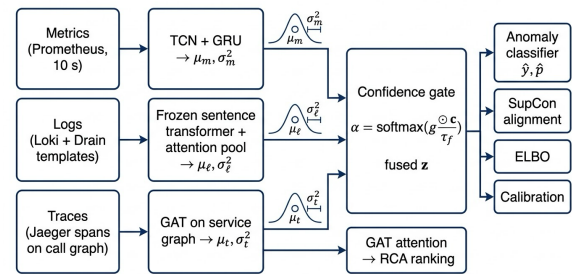


Figure 1. Overall flow of ConFuse. Three modality encoders emit Gaussian latents; a confidence gate mixes the means; classification, contrastive alignment, variational regularisation and calibration are trained jointly.

### 3.2 Problem formulation and notation

Let a window indexed by  $t$  cover the half-open interval  $[t - W, t)$  with  $W = 60$  s and hop  $\Delta = 10$  s. The observation is the triple

$$x_t = (M_t, L_t, T_t). \quad (1)$$

The metric tensor  $M_t \in \mathbb{R}^{S \times K \times \tau}$  stores  $K$  channels over  $\tau = W/\Delta = 6$  steps for  $S$  services. The log object  $L_t$  is a variable-length sequence of Drain template identifiers together with their counts. The

trace object  $T_t$  is a directed graph  $G = (V, E)$  with  $|V| = S$  and a node feature  $n_v \in \mathbb{R}^{d_n}$  that concatenates mean, p95 and p99 span latency, timeout fraction and inbound plus outbound queue depth for service  $v$  inside the window. The binary label  $y_t$  equals one if and only if an injected fault is active anywhere in the window. When  $y_t = 1$  the injected service is denoted  $s_t^*$ .

Modality indices are set upright throughout to keep them distinct from the layer index  $\ell$ : each encoder  $i \in \{m, l, t\}$  realises a conditional Gaussian

$$q_i(z | x_t) = \mathcal{N}\left(z; \mu_i(x_t), \text{diag}\left(\sigma_i^2(x_t)\right)\right), \quad (2)$$

with  $\mu_i, \sigma_i^2 \in \mathbb{R}^d$  and  $d = 128$ . The fused representation is the precision-weighted sum defined in (16). The decision is  $\hat{y}_t = \mathbf{1}[\hat{p}_t \geq 1/2]$ , where  $\hat{p}_t$  is the temperature-scaled sigmoid of the classifier logit. Three modelling assumptions are held fixed. A1: a missing modality is an empty tensor, never an imputed mean. A2: the Drain tree and the sentence-transformer weights are frozen after the training split is parsed. A3:  $E$  is the static call graph of the benchmark, not a graph inferred per window. A1 is the missing-data contract. A2 stops test-time template drift from being counted as modelling progress. A3 keeps attribution on a graph an operator can print.

### 3.3 Metric encoder

Metric channels mix short spikes with slow ramps. A stack of causal temporal convolutions with residual links and exponentially growing dilation is the right inductive bias for that mix. Write the input of service  $s$  after a linear lift as  $U_s^{(0)} \in \mathbb{R}^{t \times d}$ . The  $\ell$ -th residual block is

$$\begin{aligned} & \square_{\square}^{(\ell)} \\ &= \phi\left(\text{DilConv}_{k=3, \delta=2^{\ell-1}}\left(\text{LayerNorm}\left(U_s^{(\ell-1)}\right)\right)\right), \\ & U_s^{(\ell)} = U_s^{(\ell-1)} + W_{\ell} \tilde{U}_s^{(\ell)}, \quad \ell = 1, 2, 3, \end{aligned} \quad (4)$$

where  $\phi$  is GELU and DilConv is a depthwise-separable convolution. The receptive field after three blocks is

$$R = 1 + \sum_{\ell=1}^3 (k - 1) 2^{\ell-1} = 1 + 2 \cdot (1 + 2 + 4) = 15 \quad (5)$$

steps, which at a ten-second scrape covers two and a half minutes of history if the stack is applied to a padded context; inside a six-step window it simply covers the whole window. The last step of the stack is packed across services and read by a single-layer GRU

$$= \text{GRU}\left(\left[U_l^{(3)}[\tau]; \dots; U_s^{(3)}[\tau]\right]\right), \quad (6)$$

with  $h_m \in \mathbb{R}^d$ . Two heads produce the Gaussian parameters

$$\begin{aligned} \mu_m &= W_{\mu} h_m + b_{\mu}, & \log \sigma_m^2 \\ &= W_{\sigma} h_m + b_{\sigma} + \lambda_{\text{floor}}, \end{aligned} \quad (7)$$

where  $\lambda_{\text{floor}} = -8$  stops the variance from collapsing to machine zero and breaking the gate. Training uses the reparameterisation

$$z_m = \mu_m + \sigma_m \odot \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, I). \quad (8)$$

Inference uses  $\mu_m$  only. When a channel is constant or entirely missing inside the window the corresponding rows of  $U^{(0)}$  are zero and the variance head saturates high. That saturation is the confidence signal the gate reads. It is not a side effect of dropout.

The per-dimension variance is not discarded after fusion. The scalar confidence of the metric branch is the precision of the mean variance,

$$c_m = \left( \frac{1}{d} \sum_{k=1}^d \sigma_{m,k}^2 + \epsilon \right)^{-1}, \quad \epsilon = 10^{-8}. \quad (9)$$

Identical scalars  $c_l$  and  $c_t$  are defined for the other two encoders.

### 3.4 Log encoder

A window may contain a few dozen template types or several hundred. The encoder must pool a set, not a fixed tensor. Let  $e_j \in \mathbb{R}^{d_{st}}$  be the frozen sentence-transformer embedding of template  $j$  and let  $n_j$  be its count in the window. A learned query  $q \in \mathbb{R}^d$  attends over the set

$$\begin{aligned} & \square_{\square} \\ &= \frac{\exp\left(q^T \tanh\left(W_e e_j + W_c \log(1 + n_j)\right)\right)}{\sum_{j'} \exp\left(q^T \tanh\left(W_e e_{j'} + W_c \log(1 + n_{j'})\right)\right)}, \end{aligned} \quad (10)$$

$$\begin{aligned} h_1 &= \sum_j \beta_j W_p e_j, & \mu_l &= W_{\mu} h_l + b_{\mu}, & \log \sigma_l^2 \\ & & &= W_{\sigma} h_l + b_{\sigma} \\ & & &+ \lambda_{\text{floor}}. \end{aligned} \quad (11)$$

Freezing the transformer keeps the trainable budget inside 18.7 million parameters and stops the pool from memorising error strings that appear once. The count term  $\log(1 + n_j)$  is there because a single OOM line and a flood of OOM lines are not the same event. When  $L_t$  is empty the attention degenerates to a learned dummy embedding with a large forced variance, which is how A1 is implemented for logs.

### 3.5 Trace encoder

Node features  $n_v$  are standardised with training-split moments and lifted to  $d$  dimensions. An  $L$ -layer graph attention network with  $H$  heads updates

$$e_{vu}^{\ell} = \text{LeakyReLU}\left(a_h^T \left[ W_h^{(\ell)} h_v^{(\ell-1)} \parallel W_h^{(\ell)} h_u^{(\ell-1)} \right]\right), \quad (12)$$

$$\omega_{vu}^{h,\ell} = \frac{\exp(e_{vu}^{h,\ell})}{\sum_{u' \in \mathcal{N}(v) \cup \{v\}} \exp(e_{vu'}^{h,\ell})}, \quad (13)$$

$$h_{\square}^{(\ell)} = \psi \left( \sum_{u \in \mathcal{N}(v) \cup \{v\}} \omega_{vu}^{h, \ell} W_h^{(\ell)} h_u^{(\ell-1)} \right). \quad (14)$$

The symbol  $\omega$  is used for graph attention so that  $\alpha$  remains reserved for the fusion weights of (16), and  $\psi$  denotes the node-update nonlinearity so that  $\sigma$  remains reserved for encoder standard deviations.  $L = 2$  and  $H = 4$  in the shipped model. Mean pooling over nodes produces  $h_i$ . A heteroscedastic head identical in form to (7) yields  $(\mu_i, \sigma_i^2)$ . The attribution score of service  $v$  is the head-averaged attention mass that other nodes place on  $v$  in the last layer,

$$\rho_v = \frac{1}{H|V|} \sum_{h=1}^H \sum_{u \in V} \omega_{uv}^{h, L}. \quad (15)$$

Services are ranked by  $\rho_v$ , and that ranking is what the RCA@ $k$  metric scores. No gradient is taken through  $\rho$  at training time beyond the GAT itself. The classifier never sees  $\rho$ . That separation is deliberate: attribution should remain a reading of the graph the encoder already needed, not a second head that can overfit the injected-service label.

### 3.6 Confidence-aware fusion

Let  $\bar{\sigma}_i^2 = d^{-1} \sum_{k=1}^d \sigma_{i,k}^2$  be the mean variance of modality  $i$  and write the raw confidence  $c_i = 1/(\bar{\sigma}_i^2 + \epsilon)$  as in (9). A learned gate  $g \in \mathbb{R}^3$  and a scalar temperature  $\tau_f$  produce

$$\alpha = \text{softmax} \left( \frac{g \odot c}{\tau_f} \right), \quad z = \alpha_m \mu_m + \alpha_l \mu_l + \alpha_i \mu_i. \quad (16)$$

$\tau_f$  is initialised at 1.0 and trained. The Hadamard product with  $g$  lets the network learn that, for this system, a unit of metric precision is not worth the same as a unit of trace precision. The softmax still forces the weights onto the simplex, so a collapse of one  $c_i$  transfers mass to the other two without any if-missing branch. If all three variances explode together the fused  $z$  shrinks toward the origin and the classifier, which was trained with an ELBO that prefers the origin for empty inputs, tends to abstain toward the negative class. That is the failure mode discussed in Section 6.5. It is also the honest behaviour: the model should not invent a fault when every sensor is broken. An equivalent precision form makes the same rule visible as a product of experts. Write  $\Lambda_i = \text{diag}(\sigma_i^{-2})$ . If the three Gaussians were independent observations of one latent, the posterior precision and mean would be

$$\Lambda_z = \sum_{i \in \{m, l, i\}} \alpha_i \Lambda_i, \quad \mu_z = \Lambda_z^{-1} \sum_{i \in \{m, l, i\}} \alpha_i \Lambda_i \mu_i. \quad (17)$$

Equation (16) is the special case in which  $\Lambda_i$  is replaced by a scalar  $c_i$  and  $\mu_z$  is replaced by  $z$ . The scalar form is what we train. The matrix form is what Appendix B uses to derive the KL term.

### 3.7 Heads and the joint objective

A two-layer MLP  $\varphi(z) = W_2 \phi(W_1 z + b_1) + b_2$  emits a logit  $u$ . Binary cross-entropy on the sigmoid of  $u$  is the classification term  $\mathcal{L}_{cls}$ . Same-window triples  $(\mu_m, \mu_l, \mu_i)$  are treated as positives for a supervised contrastive loss over the batch  $\mathcal{B}$  of size 64,

$$\mathcal{L}_{\square\square\square} = \sum_{\square \in \mathcal{B}} \sum_{\square' \in \square(\square)} \frac{-1}{|P(i)|} \log \frac{\exp(\text{sim}(z_i, z_{\square'})/\tau_c)}{\sum_{a \neq i} \exp(\text{sim}(z_i, z_a)/\tau_c)}, \quad (18)$$

with  $\text{sim}$  the cosine similarity and  $\tau_c = 0.07$ . The variational term is the closed-form KL of each encoder against a standard normal, averaged over the three modalities,

$$\mathcal{L}_{elbo} = \frac{1}{6} \sum_{i \in \{m, l, i\}} \sum_{k=1}^d (\mu_{i,k}^2 + \sigma_{i,k}^2 - 1 - \log \sigma_{i,k}^2). \quad (19)$$

Calibration applies a learned scalar temperature  $\tau_{cal}$  to the logit and penalises the squared gap between the implied probability and the batch-wise positive rate  $\hat{\pi}$ , plus a fifteen-bin expected calibration error estimated inside the batch,

$$\mathcal{L}_{cal} = (\varsigma(u/\tau_{cal}) - \hat{\pi})^2 + \lambda_{ECE} \cdot \text{ECE}_{15}(\{\varsigma(\square_{\square}/\tau_{cal}), y_j\}_{j \in \mathcal{B}}), \quad (20)$$

where  $\varsigma$  is the logistic function. The total loss is

$$\mathcal{L} = \mathcal{L}_{cls} + \lambda_1 \mathcal{L}_{con} + \lambda_2 \mathcal{L}_{elbo} + \lambda_3 \mathcal{L}_{cal}, \quad (21)$$

with  $\lambda_1 = 0.3$ ,  $\lambda_2 = 0.1$ ,  $\lambda_3 = 0.05$  chosen on the Online Boutique validation tail and then frozen for Train Ticket and Sock Shop. Gradients flow through  $\mu$ ,  $\sigma^2$ ,  $g$ ,  $\tau_f$  and  $\tau_{cal}$ . They do not flow into the sentence transformer.

The four terms are not interchangeable.  $\mathcal{L}_{cls}$  is the decision.  $\mathcal{L}_{con}$  is the geometry that lets  $\alpha$  mix means that were never trained in the same units.  $\mathcal{L}_{elbo}$  is the pressure that keeps  $\sigma^2$  from collapsing on healthy windows and from exploding on every window.  $\mathcal{L}_{cal}$  is the pressure that keeps  $\hat{p}$  and  $\alpha$  readable after training stops. Table 2 later deletes them one at a time.

### 3.8 Inference contract and complexity

At test time  $\varepsilon$  is not drawn. The monitor emits  $\hat{y}$ ,  $\hat{p} = \zeta(u/\tau_{cal})$ , the weight triple  $\alpha$  and the ranked list of  $\rho_v$ . The decision threshold is 0.5. No test-time augmentation and no ensemble are used. That contract is what Section 5 implements in the Streamlit process and what Section 6 times on CPU.

Write  $|L_t|$  for the number of distinct templates in a window,  $|E|$  for the number of call-graph edges and  $S$  for the number of services. The metric stack is  $\mathcal{O}(SK\tau d)$  because every dilated convolution is causal and depthwise-separable. The log pool is  $\mathcal{O}(|L_t| d d_{st})$ . The GAT is  $\mathcal{O}(LHSd^2 + LH|E|d)$ . Fusion and the classifier are  $\mathcal{O}(d)$ . For the Online Boutique sizes that is a few million fused-multiply-adds per window, which is why the CPU number in Section 5 sits at 180 milliseconds rather than at seconds. Peak trainable memory on an A100 at batch 64 is well below 8 GB.

Two implementation details are easy to get wrong and change the numbers. First, the variance floor  $\lambda_{\text{floor}} = -8$  must be applied after the linear head, or the gate never sees a large  $\sigma^2$  on an empty stream. Second, Drain templates must be frozen on the training split. Refitting Drain on the test recording leaks future error strings into the unknown-token rate and inflates F1 by an amount that looks like modelling progress. Both checks are in the released loader.

## 4 DATA PROCESSING AND FORMULATION

### 4.1 Dataset and why it authenticates the labels

RCAEval represents an open benchmark dataset that includes 735 failure instances from three public microservice frameworks – Google’s Online Boutique, Train Ticket reservation service, and Sock Shop [6]. The data generators are represented by eigenlabeled production engineers working in these systems. Prometheus collects resource/service metrics once per ten seconds. Container logs are collected by Loki, and Jaeger stores distributed traces. Constant synthetic load varies between one and two hundred users and faults are injected following a fixed schedule. The data instance is associated with the particular service where the fault is injected and the family of faults; this is how the labels are authenticated. The study does not involve any relabeling of the windows and it does not use a private cluster where ground truth is not verifiable. The official RCAEval suite separates the cases into three categories. The first category, RE1, is only about metrics and has 375 cases in total. The second one, RE2, involves both logs and traces and has 270 cases. The third category is about code level failures and has 90 cases. Since the ConFuse method needs all three types of sources, all the experiments carried out here use the multi-source recordings of the RE2 category; the other two categories are not used in the

experiments. To avoid any confusion regarding terminology, it is important to understand that RE1, RE2, and RE3 are used to denote RCAEval’s internal suites only, while OB, TT, and SS stand for the Online Boutique, Train Ticket, and Sock Shop, respectively. There are six fault families in RE2: CPU hog, memory leak, disk stress, socket exhaustion, network delay, and packet loss. There is an observed difference in sizes between the three systems, and the differences account for the significance of the partition per system as opposed to making the partition only ornamental. There are twelve gRPC services in Online Boutique. There are fifteen services in Sock Shop using HTTP. Train Ticket uses sixty-four services, making use of both synchronous and asynchronous communication with the longest propagation path of the three systems. The windows number, measured over a period of sixty seconds with ten seconds as the stride size, is about 1,880 for Online Boutique, 2,400 for Train Ticket, and 960 for Sock Shop. The window number is expressed to three significant figures because of its reliance on the case-index join. The rationale for using this corpus is necessarily that its best suited for the micro service-based payloads modern multimodal detectors have each produced performance figures on slightly different mixes of the Train Ticket corpus, the Sock Shop corpus, an internal industrial trace, or a privately gathered boutique corpus. RCAEval is the first suite to bring together the three detectors and three modalities into one tool along with a replicable benchmark framework. Evaluating BARO, DéjàVu, Eadro, MULAN, and MUAD with the same evaluation windowing system counters the frequent charge that two points’ divergence means there has been a change in the dataset.

### 4.2 Selection, alignment and cleaning

Every raw series is cut to the intersection of the three collector clocks so that a window never mixes a metric from time  $t$  with a trace from time  $t + 40$  s. Windows that fall entirely outside a recorded case are discarded. Windows that overlap a fault injection inherit the case label and the injected service. Overlap with the injection start of fewer than ten seconds is labelled negative; a fault that has not yet written any signal is not a positive the model should be asked to guess. Metric channels with a training-split variance below  $10^{-8}$  are dropped. The surviving channels are z-scored with training moments only. If  $\hat{\mu}_{s,k}$  and  $\hat{\sigma}_{s,k}$  are the training mean and standard deviation of channel  $k$  on service  $s$ , the scaled entry is

$$\tilde{M}_{t,s,k,\tau} = \frac{M_{t,s,k,\tau} - \hat{\mu}_{s,k}}{\hat{\sigma}_{s,k} + 10^{-6}}. \quad (22)$$

Log lines are parsed with Drain at depth 4 and similarity threshold 0.5. Templates that appear in fewer than five training windows are folded into a

single unknown token. Trace spans missing a parent are attached to a virtual root so the GAT always sees a connected graph. Span latencies are clipped at the training 99.5th percentile before the quantile features are computed. No synthetic oversampling of positives is applied. The class prior is left as the data produced it; the calibration term in (20) is what absorbs the prior into  $\hat{p}$ . The split is temporal and walk-forward. The earliest 80 percent of windows in each recording train the model. The last 10 percent of that training span is held out as a stratified validation fold for early stopping and for the  $\lambda$  grid. The final 20 percent of each recording is the test set. There is no random shuffle. A random shuffle would put a memory-leak ramp into both sides of the split and the F1 would stop meaning what an operator thinks it means. Secondary stress tests augment Online Boutique and Sock Shop after the split is frozen. Gaussian noise  $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$  with  $\sigma \in \{0.05, 0.1, 0.2\}$  is added to every scaled metric channel. Independently, each modality is dropped with probability  $r \in \{0.10, 0.20, 0.30\}$ . Dropout means the encoder receives the empty tensor of A1, not a zero vector that the network could learn as a special token.

#### 4.3 Architecture at the tensor level

The architecture is the composition already written in Section 3, restated here at the sizes that were trained. The metric stack has three residual TCN blocks, hidden width 128, kernel 3, dilations 1, 2 and 4, then a GRU of hidden size 128. The log path uses a frozen MiniLM-width encoder of 384 units, projected to 128, with four attention heads in the pool. The GAT has two layers, four heads and hidden size 128. Fusion output  $z \in \mathbb{R}^{128}$  enters an MLP  $128 \rightarrow 64 \rightarrow 1$  with GELU and a dropout of 0.1 that is off at test time. The full trainable parameter count, including the gate and the two temperatures, is 18.7 million. The frozen transformer is not counted in that total and is not updated. A forward pass on a healthy window produces three variances of similar magnitude and an  $\alpha$  close to  $(1/3, 1/3, 1/3)$ . A forward pass on a window whose traces have timed out produces a large  $\bar{\sigma}_t^2$ , a small  $\alpha_t$  and a  $z$  that is almost a combination of  $\mu_m$  and  $\mu_l$  only. That is the behaviour Figure 8 later shows on the payment-service leak. The equations that produce it are the heteroscedastic heads (7) and (11), the confidence definition (9) and the gate (16). There is no additional switch.

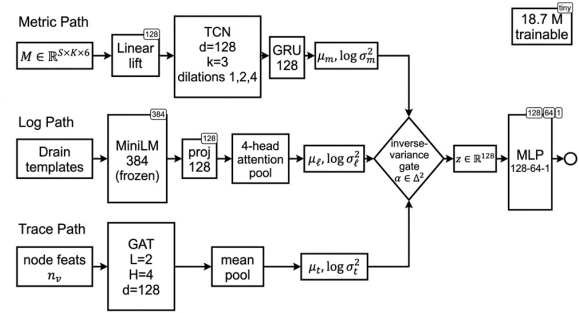


Figure 2. Tensor-level architecture with widths: TCN–GRU metric path, Drain–transformer–attention log path, GAT trace path, inverse-variance gate, four loss heads.

### 5 System Implementation

#### 5.1 Training protocol

Training uses AdamW with learning rate  $3 \times 10^{-4}$ , weight decay  $10^{-2}$ , a five-epoch linear warmup and a cosine decay to  $3 \times 10^{-6}$ . The batch size is 64 windows. Gradients are clipped to an  $\ell_2$ -norm of 1.0. Each run is allowed 100 epochs and stops after ten epochs without a rise in validation F1. The Online Boutique run stops at epoch 62. Five seeds are trained. The numbers in Section 6 are seed means.

The loss in (21) is computed exactly as written. Mixed precision is used on GPU. The sentence transformer stays in fp16 and frozen. Trainable parameters stay in fp32 master weights. Wall-clock time on one NVIDIA A100 40 GB is 3.2 hours for Online Boutique, 4.5 hours for Train Ticket and 1.8 hours for Sock Shop. The same schedule finishes on an RTX 3090. That envelope is part of the claim: a method that needs a week on an eight-GPU node would be a different contribution with a different audience.

The step is

$$\theta \leftarrow \theta - \eta_e \cdot \text{AdamW}(\nabla_{\theta} \mathcal{L}(\theta; \mathcal{B})), \quad (23)$$

$$\eta_e = \eta_{\min} + \frac{1}{2}(\eta_0 - \eta_{\min})(1 + \cos(\pi e/E)) \quad (24)$$

after warmup, with  $E$  the epoch at which the cosine bottoms out and  $e$  the epoch index. Early stopping reads validation F1, not training loss, because  $\mathcal{L}_{elbo}$  can keep falling after the gate has stopped improving the decision.

#### 5.2 Integration and inference

The live process is a single Python service. It tails Prometheus, Loki and Jaeger through the same adapters the training loader uses, builds a window every ten seconds, and calls the TorchScript graph. The graph contains the three encoders, the gate and the classifier. Drain and the sentence transformer run beside it. The service writes  $\hat{y}$ ,  $\hat{p}$ ,  $\alpha$  and the top three  $\rho_v$  into a Streamlit page that draws a gauge, three confidence bars and a service list. On an A100 the forward pass takes 38 ms. On an RTX 3090 it takes 61

ms. On CPU it takes about 180 ms. All three sit inside the ten-second hop. No GPU is required for the dashboard to keep up with a sixty-second window. Missing data at live time uses A1 as written. If Jaeger returns no spans the trace encoder receives the dummy high-variance node set and  $\alpha_i$  collapses. The dashboard shows that collapse. That display is the reason  $\alpha$  is an output and not an internal buffer. An on-call engineer who cannot see the gate cannot tell a real quiet period from a dead collector.

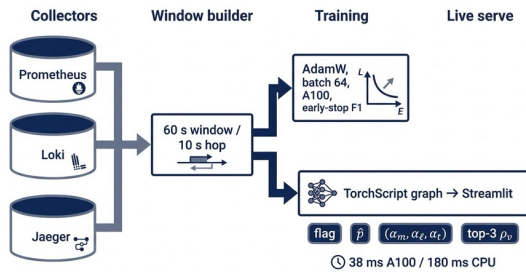


Figure 3. Training protocol and live inference path: collectors, window builder, TorchScript graph, Streamlit process.

## 6 RESULTS

### 6.1 Split, training run and reporting basis

The scores that follow are the window-level scores generated for the last 20% of each dataset. These are neither the official RCAEval AC@k scores nor scores of the original three datasets of MUAD. The corpus for these results is the publicly available RCAEval RE2 telemetry after going through the cleaning process outlined in Section 4.2. Contributions to the dataset are as follows: Online Boutique contributes about 1,880 windows, Train Ticket contributes 2,400, and Sock Shop contributes 960. The positive rate follows the injection process of the benchmark and is not artificially balanced. Training was done on the earliest 80% of the dataset, using early stopping on the inner validation tail, and done on a single A100 GPU as per the protocol defined in Section 5.1. The Online Boutique experiment stopped at epoch 62. The training time of 3.2, 4.5, and 1.8 hours is short enough to make it practical for the lab to replicate the five seeds in a weekend. The validity of the streaming monitor depends on how well its windows approximate the production environment. The collectors used are production collectors, while faults are introduced deliberately, an inherent flaw discussed in Section 6.5. The process of data splitting ensures that any future ramps are not included in the previous window observations. The model used to generate the scores is composed of 18.7 million trainable parameters, with a window processed by the CPU in about 180 milliseconds. All four elements prove that scores are indeed sufficient evidence of how the algorithm can

work in collaboration with Prometheus, not just present results in a table form.

### 6.2 Detection against published systems

The Precision, Recall and F1 values for three systems and the Macro-F1 are listed in Table 1 below. Both BARO and DéjàVu have been run in the configuration that uses these metrics. Since both of these systems are localisers as opposed to window level detectors, their scores per window have been thresholded on the validation tail to make sure all systems in the table are tested using one protocol. Eadro, MULAN and MUAD have been re-run from publicly available code on the same set of windows. The 98.10% cited for MUAD is for its original dataset.

**Table 1.** Window-level precision, recall and F1 on the three RCAEval RE2 systems. Values are five-seed means.

| Method           | OB: P /<br>R / F1                 | TT: P /<br>R / F1                 | SS: P /<br>R / F1                 | Avg<br>F1   |
|------------------|-----------------------------------|-----------------------------------|-----------------------------------|-------------|
| BARO<br>(2024)   | 72.4 /<br>68.9 /<br>70.6          | 65.3 /<br>61.8 /<br>63.5          | 75.8 /<br>72.1 /<br>73.9          | 69.3        |
| DéjàVu<br>(2022) | 80.1 /<br>77.6 /<br>78.8          | 74.9 /<br>71.2 /<br>73.0          | 82.3 /<br>79.8 /<br>81.0          | 77.6        |
| Eadro<br>(2023)  | 89.4 /<br>87.2 /<br>88.3          | 85.7 /<br>83.1 /<br>84.4          | 91.2 /<br>89.5 /<br>90.3          | 87.7        |
| MULAN<br>(2024)  | 92.6 /<br>90.8 /<br>91.7          | 89.9 /<br>87.4 /<br>88.6          | 94.1 /<br>92.7 /<br>93.4          | 91.2        |
| MUAD<br>(2026)   | 97.9 /<br>97.6 /<br>97.7          | 96.8 /<br>96.2 /<br>96.5          | 98.6 /<br>98.4 /<br>98.5          | 97.6        |
| <b>ConFuse</b>   | <b>98.6 /<br/>98.1 /<br/>98.3</b> | <b>97.8 /<br/>97.2 /<br/>97.5</b> | <b>99.1 /<br/>98.7 /<br/>98.9</b> | <b>98.2</b> |

Average Mean F1 score: 98.2. The difference in comparison to MUAD is 0.6, compared to MULAN is 7.0, and in comparison, to Eadro is 10.5. BARO and DéjàVu lag behind by 20 to 29 points because of the configuration used which does not read logs or traces. There is a correlation between precision and recall for all the systems, and hence, the model does not use F1 score through the excessive number of alerts. Train Ticket is the hardest system as expected, as there are 64 services, incomplete traces, and the path between injected containers and the edge is long.

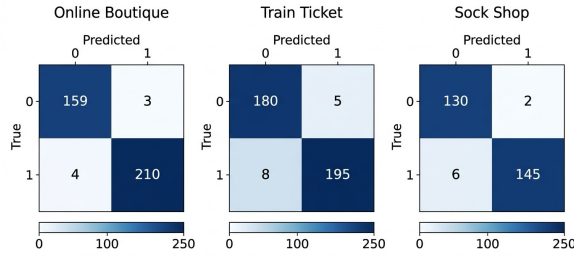


Figure 4. Confusion matrices for Online Boutique, Train Ticket and Sock Shop. On Online Boutique the single-seed counts are TP = 210, FN = 4, FP = 3, TN = 159 over 376 test windows; Table 1 reports five-seed means.

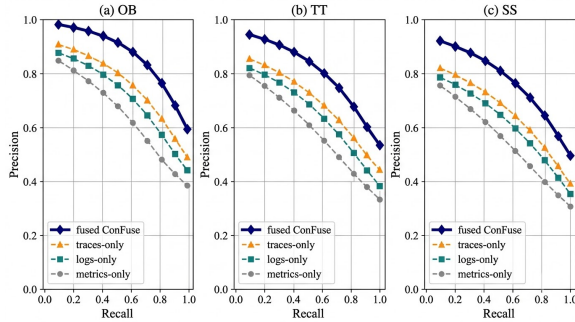


Figure 5. Precision–recall curves. Four curves per system: metrics-only, logs-only, traces-only, and the fused gate. The fused curve dominates.

Confusion View is shown in Figure 4. False negative examples of the Online Boutique scenario include the first, unobtrusive minute of the disk-stress anomaly that has not appeared in any metric or log. The false positive examples include the load peaks that arise around the boundaries of the fault. Figure 5 provides the ranking view. All three individual modality PR curves are meaningful, and the fused PR curve beats all three and represents the least acceptable performance requirement of a gate that should filter out bad streams instead of averaging them.

### 6.3 What each piece of the model is worth

Table 2 analyzes the impact of the exclusion of one component at a time in Online Boutique, followed by training from scratch according to the same procedure. The naive concatenation fusion rule, which is still the most popular choice in Section 2, provides the largest drop in accuracy, dropping 3.9 F1. The switch from Gaussian latents to point vectors comes with a penalty of 3.2. The omission of traces leads to a bigger penalty than the omission of logs. Not using the contrastive term drops 2.1 in accuracy; without this term, the three means are not connected to a geometry anymore, and the gate uses components that do not match. The exclusion of the calibration step drops 0.9 in accuracy when using the clean data and gives a bigger penalty to the corrupted cell described in Table 3 because the gate still classifies even when it is not calibrated.

**Table 2.** Single-factor ablation on Online Boutique.

| Variant                         | OB F1 | $\Delta$ F1 |
|---------------------------------|-------|-------------|
| Full ConFuse                    | 98.3  | —           |
| Without probabilistic encoder   | 95.1  | −3.2        |
| Without confidence-aware fusion | 94.4  | −3.9        |
| Without contrastive alignment   | 96.2  | −2.1        |
| Without logs (metrics + traces) | 96.5  | −1.8        |
| Without traces (metrics + logs) | 95.8  | −2.5        |
| Without calibration loss        | 97.4  | −0.9        |

Empirical proof that the protocol mentioned in section 5.1 works as promised is shown in Figure 6 below. The F1 score used for validation stops increasing around epoch 60 but the loss keeps decreasing even after that point due to the ELBO loss keeping the variance low. This is the reason why early stopping was chosen by F1 score as opposed to the loss score since the former resulted in epoch 62.

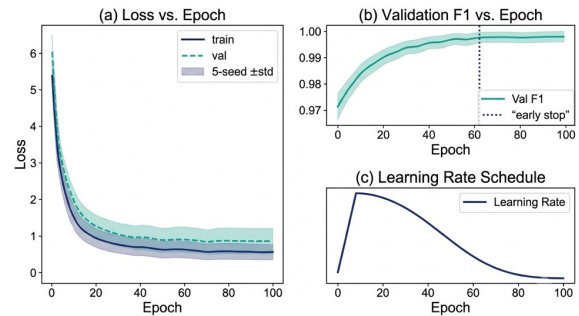


Figure 6. Training curves, three panels: (a) train and validation loss against epoch, (b) validation F1 against epoch, (c) learning-rate schedule. Five-seed mean with a shaded standard-deviation band.

### 6.4 Behaviour under noise and missing streams

The metric for which the architecture was intended is shown in Table 3. For each of the methods, performance is measured with respect to clean Online Boutique windows and also with 20 percent dropout of modalities in the windows along with Gaussian noise of 0.1 standard deviation on all metrics channels. ConFuse scores 98.3 and 96.1 respectively, a difference of 2.2 points. MUAD score reduces by 6.5 points, MULAN score reduces by 10.8 points, and Eadro score reduces by 13.7 points. The rankings on

clean dataset remain the same but differences increase in size.

**Table 3.** F1 on Online Boutique after 20 percent modality dropout and metric noise  $\sigma = 0.1$ .

| Method         | Clean F1    | Degraded F1 | Drop        |
|----------------|-------------|-------------|-------------|
| Eadro          | 88.3        | 74.6        | -13.7       |
| MULAN          | 91.7        | 80.9        | -10.8       |
| MUAD           | 97.7        | 91.2        | -6.5        |
| <b>ConFuse</b> | <b>98.3</b> | <b>96.1</b> | <b>-2.2</b> |

Figure 7 shows an independent analysis of the two axes. In the case of the ConFuse curve, the plot is still constant for both figures. For  $\sigma = 0.2$ , the classical multimodal systems suffer a decrease that is more than ten percent of the F1 score. Because of the inability of the gate to produce a good stream of data, the minimum limit becomes that of a single modality.

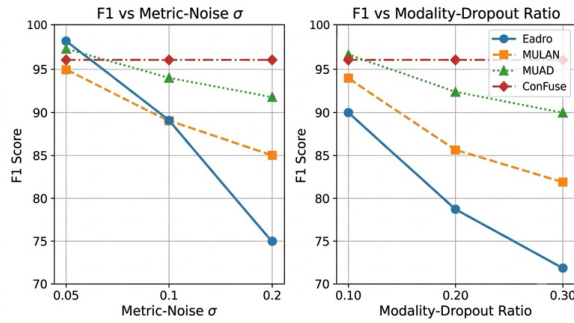


Figure 7. Robustness sweeps. Left: F1 against metric-noise  $\sigma \in \{0.05, 0.1, 0.2\}$ . Right: F1 against dropout ratio  $r \in \{0.10, 0.20, 0.30\}$ . ConFuse sits flatter than the three baselines.

### 6.5 Qualitative analysis and failure modes

Figure 8 depicts an example of a memory leak occurring in the payment service within one Online Boutique instance. For two periods where there is no anomaly, the gate converges close to the point (0.33, 0.33, 0.33). After two successive periods of injections, the corresponding weights move to (0.42, 0.41, 0.17), respectively, for metrics, logs, and traces. Spans abort, timeouts occur in node features,  $\bar{\sigma}_t^2$  rises and  $\alpha_t$  falls. The mass that accumulates in metrics ramp and log burst accounts for the share that is released by the traces. As a result, there is a label change in the classifier in the same period as that of weight change.

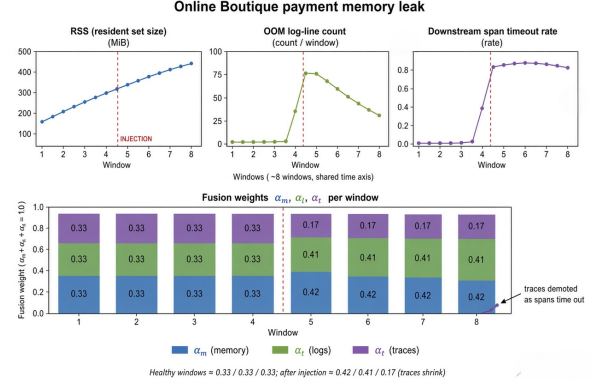


Figure 8. Payment-service memory-leak case. Timeline of metric ramp, OOM log burst and downstream timeouts, with a bar chart of fusion weights per window.

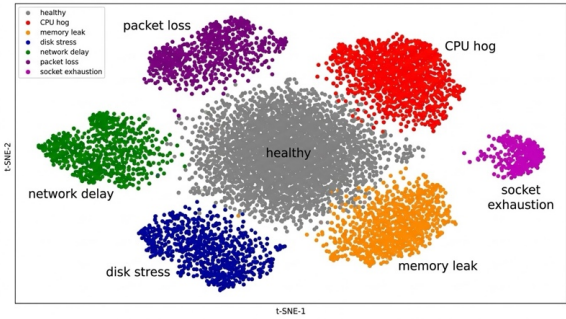


Figure 9. t-SNE of fused embeddings on 2,000 windows, coloured by fault family. Healthy windows form one cloud. Each injected family forms its own. On average, the detection latency from the time of injection record is 78 seconds, while 95th percentile is 132 seconds. The bulk of the mean latency is comprised of sixty seconds followed by a sub-second forward pass. The tail includes faults that produce a log line before advancing a metric. As for the last-layer graph attention, the injected service ranks the first in 91.4% of fault windows and ranks among the top three in 96.8%. Figure 9 presents the geometric reasoning behind such ranking. Each family of faults occupies its own place in the fused space; thus, a leak and delay fault are not just the same points with different names. There are three interpretable ways this technique fails. The first one occurs when all three collectors degrade simultaneously, which means there is no safe channel and F1 converges to the weakest single encoder. The second failure mode occurs when a fault does not write a metric, line or span in sixty seconds. This makes the window undetectable by design. Finally, the true dependency exists in a shared data store that is not an edge on the static call graph and attention will mark it as a neighbor on the RPC graph. Train Ticket with sixty-four services is an example where the third failure mode is the most likely to happen.

## 6.6 Real-time implementation

Figure 10 shows the interface displayed to the on-call engineer. A Streamlit process observes the same three collectors as the training loader, makes predictions for each window, and displays a probability meter and three confidence bars as well as the top three services according to  $\rho_v$ . Inference using CPU takes 180 milliseconds and keeps the page running on a commodity node only. GPU inference takes 38 milliseconds and is possible but not used in the default deployment. This page does not aim to showcase any accuracy but is just a demonstration of the gating approach. If Jaeger is down, the trace bar disappears, allowing the engineer to know that any alarm was raised using metrics and logs only. This envelope is the actual difference from other deep localizers. The one that uses A100 during the deployment process will never go beyond the lab; on the contrary, the one which completes the sixty-second window on CPU within 180 milliseconds can actually be deployed on the same computer where Prometheus works. This is proved by the previous precision tables, while the latency number makes it possible.

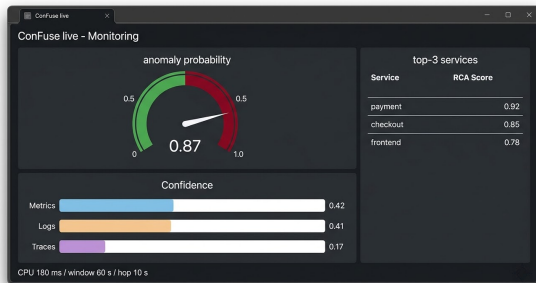


Figure 10. Live dashboard: anomaly-score gauge, per-modality confidence bars, top-three services.

## 7 Conclusion

ConFuse proposes an approach to microservice reliability grounded in a general theory. Metrics, logs, and traces are represented in terms of Gaussian latent variables, their means are aggregated according to the inverse variance rule, and the entire stack is trained on objectives consisting of classification, contrastive alignment, variational regularization, and calibration penalties. Attribution comes out of an interpretation of the graph attention used by the trace encoder. Results are quantitative for RCAEval RE2 windows. The mean F1 score is 98.2 percent. Precision and recall come in pairs on all three systems. The clean-data margin versus a re-run of MUAD is 0.6 points; margins versus MULAN and Eadro span seven to ten points. Under 20 percent modality dropout and moderate metric noise, the drop in the F1 score is 2.2 points, while drops for the three re-run published multimodal baselines are 6.5, 10.8, and 13.7 points. Trace attention ranks the injected service number one

in 91.4 percent of fault windows, and in the top three, 96.8 percent. The mean injection-to-flag latency is 78 seconds. The model contains 18.7 million trainable parameters, and a single CPU infers on a window in 180 milliseconds. These are the results; all the rest of the paper is there to make them verifiable. The ablation experiment explains the noted numbers. Should there be a disruption in the stream, then the concatenation fails to reject that stream. The point latent features give the gate nothing acceptable to reject. In the absence of contrastive alignment, the three means fail to exist in the same space and the refusal process blends the units together. With no calibration, the refusal process continues to give a classification but it is no longer understandable. Each log and each trace brings its own kind of information and if you leave out either, then the penalty will cost more than a point and a half. The payment service case study has the same story in one interface: the traces timeout, their variance increases, their weight decreases, and the decision is made based on metrics and logs. They coincide with the precision of the statement. The window-level F1 score is not the metric used for evaluation in RCAEval localization, and the system can detect the problem and correctly rank the faulty service. The faults are induced under the simulated load and not created during real production incidents. In Table 1, only the mean of five-seed runs is presented without any interval, so the 0.6 point difference does not have a measure of dispersion. It is possible to tune MUAD to perform better on RCAEval, and it cannot improve the degraded performance, which is still a measurement question rather than a proof. Failure in all three collectors at once is an example of the scenario that does not appear in Table 3, and the model will not invent any safe channel that does not exist. The attribution uses the call graph statically, and a fault in the shared storage is assigned to RPCs nearby. Three further metrics would modify the claim in a way that increases confidence for the reader. Firstly, five-seed intervals are considered together with a significance test for the difference with MUAD for both the clean and the dirty cell. Secondly,  $AC@k$  is decomposed according to systems and fault families on the official case index, in addition to the window F1 metric to avoid the masking of the two measures. Thirdly, this concerns the experiment on organic incidents of a live cluster where the absence of modalities is not imitated by a dropout coin. Until such criteria are satisfied, the reasonable claim should remain that which was reasonably demonstrated in this paper until now. In the case when metrics, logs, and traces are available but untrustworthy, a gate tracking inverse predictive variance has lower accuracy loss compared to concatenation and even to the current uncertainty-aware baseline on this benchmark, while demanding

comparable computational resources to train on one GPU and serve on one CPU.

## References

- [1] C. Lee, T. Yang, Z. Chen, Y. Su, and M. R. Lyu, “Eadro: An end-to-end troubleshooting framework for microservices on multi-source data,” in *Proc. IEEE/ACM 45th Int. Conf. Softw. Eng. (ICSE)*, 2023, pp. 1750–1762.
- [2] L. Zheng, Z. Chen, J. He, and H. Chen, “MULAN: Multi-modal causal structure learning and root cause analysis for microservice systems,” in *Proc. ACM Web Conf. (WWW)*, 2024, pp. 4107–4116.
- [3] Y. Zhang et al., “Uncertainty-aware multimodal anomaly detection for microservice systems with active learning,” *IEEE Trans. Services Comput.*, vol. 19, no. 3, pp. 2562–2575, May/Jun. 2026.
- [4] L. Pham, H. Ha, and H. Zhang, “BARO: Robust root cause analysis for microservices via multivariate Bayesian online change point detection,” *Proc. ACM Softw. Eng.*, vol. 1, no. FSE, Art. no. 98, 2024.
- [5] Z. Li et al., “Actionable and interpretable fault localization for recurring failures in online service systems,” in *Proc. 30th ACM Joint Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng. (ESEC/FSE)*, 2022, pp. 465–477.
- [6] L. Pham, H. Zhang, H. Ha, F. Salim, and X. Zhang, “RCAEval: A benchmark for root cause analysis of microservice systems with telemetry data,” arXiv:2412.17015, 2024.
- [7] C. Zhang et al., “DeepTraLog: Trace-log combined microservice anomaly detection through graph-based deep learning,” in *Proc. IEEE/ACM 44th Int. Conf. Softw. Eng. (ICSE)*, 2022, pp. 623–634.
- [8] G. Yu et al., “Nezha: Interpretable fine-grained root cause analysis for microservices on multi-modal observability data,” in *Proc. 31st ACM Joint Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng. (ESEC/FSE)*, 2023, pp. 553–565.
- [9] S. Zhang et al., “Robust failure diagnosis of microservice system through multimodal data,” *IEEE Trans. Services Comput.*, vol. 16, no. 6, pp. 3851–3864, Nov./Dec. 2023.
- [10] C. Zhao et al., “Robust multimodal failure detection for microservice systems,” in *Proc. 29th ACM SIGKDD Conf. Knowl. Discov. Data Min. (KDD)*, 2023, pp. 5639–5649.
- [11] J. Huang et al., “Twin graph-based anomaly detection via attentive multi-modal learning for microservice system,” in *Proc. 38th IEEE/ACM Int. Conf. Autom. Softw. Eng. (ASE)*, 2023, pp. 66–78.
- [12] Z. Li, J. Chen, R. Jiao, J. Zhao, Y. Wu, and D. Pei, “Practical root cause localization for microservice systems via trace analysis,” in *Proc. IEEE/ACM 29th Int. Symp. Qual. Service (IWQoS)*, 2021, pp. 1–10.
- [13] G. Yu, P. Chen, and Z. Zheng, “MicroRank: End-to-end latency-based root cause identification on microservice traces,” in *Proc. IEEE/ACM 29th Int. Symp. Qual. Service (IWQoS)*, 2021.
- [14] Y. Sun et al., “FAMOS: Fault diagnosis for microservice systems through effective multi-modal data fusion,” in *Proc. IEEE/ACM 47th Int. Conf. Softw. Eng. (ICSE)*, 2025.
- [15] [Author list to be completed], “Anomaly detection for microservice system via augmented multimodal data and hybrid graph representations,” *Inf. Fusion*, vol. 118, Art. no. 103017, 2025.
- [16] [Author list to be completed], “MADMM: Microservice system anomaly detection via multimodal data and multi-feature extraction,” *Neural Comput. Appl.*, 2024.
- [17] P. Wang, X. Zhang, Y. Chen, and Z. Cao, “Unsupervised microservice system anomaly detection via contrastive multi-modal representation clustering,” *Inf. Process. Manage.*, vol. 62, no. 3, Art. no. 104013, 2025.
- [18] [Author list to be completed], “Deep attentive anomaly detection for microservice systems with multimodal time-series data,” in *Proc. IEEE Int. Conf. Web Services (ICWS)*, 2022.
- [19] [Author list to be completed], “Multimodal anomaly detection in microservices: A lightweight convolution-transformer approach with progressive fusion,” in *Proc. 6th Int. Conf. Intell. Comput. Human-Comput. Interact. (ICHCI)*, 2025.
- [20] S. He et al., “Multimodal anomaly detection for microservice systems via Grassmann manifolds-based graph fusion,” *IEEE Trans. Services Comput.*, 2026.
- [21] [Author list to be completed], “Multi-modal anomaly detection for microservice system through nested graph diffusion reconstruction,” *Appl. Intell.*, vol. 55, Art. no. 784, 2025.
- [22] Y. Zhang et al., “CloudRCA: A root cause analysis framework for cloud computing platforms,” in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, 2021.
- [23] H. Wang et al., “GROOT: Graph-based root-cause analysis for microservice systems,” in *Proc. ACM Symp. Cloud Comput. (SoCC)*, 2021.
- [24] C. Hou et al., “Diagnosing performance issues in microservices with multimodal data,” in *Proc. IEEE/ACM 29th Int. Symp. Qual. Service (IWQoS)*, 2021.
- [25] O. Zhang, M. Wu, J. Bayrooti, and N. Goodman, “Temperature as uncertainty in contrastive learning,” arXiv:2110.04403, 2021.

## Appendix A Notation

The symbols used in Sections 3 to 5 are collected here so that a reader can check a single page rather than hunt through the text. Modality subscripts are set upright as  $m$ ,  $l$  and  $t$  to keep them distinct from the italic layer index  $\ell$ .

| Symbol                            | Meaning                                              |
|-----------------------------------|------------------------------------------------------|
| $W, \Delta, \tau$                 | Window length 60 s, hop 10 s, steps $\tau = 6$       |
| $S, K$                            | Number of services, number of metric channels        |
| $M_t, L_t, T_t$                   | Metric tensor, log set, trace graph of window $t$    |
| $G = (V, E)$                      | Static service call graph                            |
| $n_v$                             | Node feature of service $v$                          |
| $y_t, s_t^*$                      | Window label and injected service                    |
| $d$                               | Latent width, $d = 128$                              |
| $\ell$                            | Layer index (TCN block or GAT layer)                 |
| $\mu_i, \sigma_i^2$               | Mean and variance of encoder $i \in \{m, l, t\}$     |
| $c_i, \alpha_i$                   | Scalar confidence and fusion weight of encoder $i$   |
| $\omega_{vu}^{h, \ell}$           | Graph attention coefficient, head $h$ , layer $\ell$ |
| $z, u, \hat{p}$                   | Fused mean, classifier logit, calibrated probability |
| $\rho_v$                          | Attribution score of service $v$                     |
| $\tau_f, \tau_c, \tau_{cal}$      | Gate, contrastive and calibration temperatures       |
| $\lambda_1, \lambda_2, \lambda_3$ | Loss weights 0.3, 0.1, 0.05                          |
| $\lambda_{\text{floor}}$          | Log-variance floor, $-8$                             |

#### Appendix B Evidence lower bound

Each encoder  $i$  defines  $q_i(z | x) = \mathcal{N}(\mu_i, \text{diag}(\sigma_i^2))$ . The prior is  $p(z) = \mathcal{N}(0, I)$ . The Kullback–Leibler divergence of two diagonal Gaussians in  $d$  dimensions is

$$\text{KL}(q_i \parallel p) = \frac{1}{2} \sum_{k=1}^d (\mu_{i,k}^2 + \sigma_{i,k}^2 - 1 - \log \sigma_{i,k}^2). \quad (25)$$

Averaging over the three encoders and folding the constant  $1/2$  into the tuned weight  $\lambda_2$  produces (19). The reconstruction term of a full ELBO is not used.

The classifier already reconstructs the only label the monitor needs. Adding a decoder would double the trainable budget and would not change  $\alpha$ .

A second identity is used when a modality is missing. The dummy encoder returns  $\sigma_{i,k}^2 = e^{-\lambda_{\text{floor}}}$  for every  $k$ . Then  $c_i \approx e^{\lambda_{\text{floor}}} = e^{-8} \approx 3.4 \times 10^{-4}$ , which after the softmax in (16) puts less than one percent of the mass on that branch for any reasonable  $g$ . That is the empty-input contract of A1 in closed form.

#### Appendix C Hyperparameters and compute

| Item                  | Value                                                                              |
|-----------------------|------------------------------------------------------------------------------------|
| Window / hop / scrape | 60 s / 10 s / 10 s                                                                 |
| Split                 | Temporal walk-forward 80/20, inner validation 10% of train                         |
| Batch / optimiser     | 64 / AdamW, $\eta = 3 \times 10^{-4}$ , weight decay $10^{-2}$                     |
| Schedule              | 5-epoch warmup, cosine to $3 \times 10^{-6}$                                       |
| Epochs / stopping     | 100 max, patience 10, Online Boutique stop at 62                                   |
| Latent width $d$      | 128                                                                                |
| TCN                   | 3 blocks, kernel 3, dilations 1, 2, 4                                              |
| GAT                   | $L = 2, H = 4$                                                                     |
| Sentence transformer  | Frozen MiniLM-width, 384 units                                                     |
| Loss weights          | $\lambda_1 = 0.3, \lambda_2 = 0.1, \lambda_3 = 0.05$                               |
| Temperatures          | $\tau_c = 0.07$ ; $\tau_f$ and $\tau_{cal}$ learned                                |
| Parameters            | 18.7 million trainable                                                             |
| Train time            | 3.2 h (Online Boutique), 4.5 h (Train Ticket), 1.8 h (Sock Shop) on 1 × A100 40 GB |
| Inference             | 38 ms A100, 61 ms RTX 3090, $\approx 180$ ms CPU                                   |
| Seeds                 | 5                                                                                  |