CERA: Causal Event Reasoning and Attribution for Real-Time Surveillance

Anonymous Authors

Anonymous Institution
anonymous@eccv2026.org

Abstract. Real-time surveillance systems need more than anomaly flags or descriptive captions. Operational response requires causal accounts that connect events, agents, and outcomes under strict latency constraints. We introduce **CERA** (**C**ausal **E**vent **R**easoning and **A**ttribution), a framework direction for online surveillance analysis that prioritizes three requirements: causal fidelity, evidential grounding, and runtime efficiency. This manuscript presents an initial formalization with a structured objective, an online output contract, and an evaluation protocol that jointly considers reasoning quality and deployment-time feasibility.

Keywords: Causal Event Reasoning, Attribution, Surveillance, Efficient Inference

1 Introduction

Why Causality Matters in Surveillance. Modern surveillance models can detect anomalies or generate captions, but practical response requires more: a causal account of what happened, what triggered it, and which entities were responsible. In safety-critical settings, this distinction affects intervention priority, accountability, and prevention planning.

Problem Setting. We consider online surveillance streams where decisions must be both timely and explainable. For each event segment, the system should produce (1) an event statement, (2) an attribution statement linking causes to outcomes, and (3) supporting evidence references. We refer to this objective as *causal event reasoning and attribution*.

Current Gap. Existing pipelines are often optimized for a single objective [9,13]. Detection-focused systems emphasize recall but provide limited causal structure [14,2,16,3,17]. Caption-focused systems can describe scenes fluently but often blur causality and agency [8,5,11]. Offline reasoning approaches are expressive but difficult to deploy with strict real-time constraints [10,11].

CERA. We introduce **CERA**, a framework direction for real-time causal event reasoning and attribution. CERA is organized around three constraints: *causal fidelity* (correct cause-effect structure), *evidential grounding* (traceable support

in observed frames), and *runtime efficiency* (practical throughput for continuous streams). This paper starts by fixing these requirements as first-class design targets.

Scope of This Draft. This manuscript is built from a clean-room start. The current version includes method-level formalization, an implementation-ready reference stack, and a projected-results package for experiment planning. All quantitative tables in this draft are explicitly non-empirical placeholders and will be replaced by measured values after external-server execution.

Contributions.

- We define a real-time surveillance task for causal event reasoning and attribution with explicit outputs for event, cause, and evidence.
- We propose CERA as a framework direction centered on causal fidelity, evidential grounding, and runtime efficiency.
- We provide a full evaluation template with projected quantitative targets to guide external empirical runs.

2 Related Work

Surveillance Anomaly Detection. Prior surveillance literature has made strong progress on anomaly detection under weak supervision, including benchmark construction and learning objectives that improve anomaly recall [9,12,15,14,2]. More recent adaptations of vision-language models improve semantic coverage for anomaly understanding [16,3,17]. However, most systems still optimize anomaly scoring or descriptive diagnosis, not explicit cause-outcome attribution with verifiable evidence links.

Video-Language Reasoning. General video-language systems provide richer temporal descriptions and dialogue-style interaction [8,5,10,11]. These advances are valuable for open-ended understanding, but they are not primarily designed around causal direction consistency, evidence attachment, or deployment-time abstention behavior required in online surveillance settings.

Position of CERA. CERA is not another captioning or anomaly-score variant. It targets the missing intersection between surveillance deployment constraints and causal attribution requirements by enforcing structured outputs, evidence checks, and runtime-feasibility objectives in one contract.

3 Method

3.1 Task Formulation

Let a surveillance stream be $\mathcal{V} = \{\mathbf{X}_1, \mathbf{X}_2, \dots\}$, where $\mathbf{X}_t$ is the frame at time t. The system identifies event windows $\mathcal{W} = \{[t_s^k, t_e^k]\}_{k=1}^{K}$ and outputs one structured report per window.

For each window k, CERA predicts

$$\hat{\mathbf{y}}_k = \{\hat{e}_k, \hat{\mathcal{A}}_k, \hat{\mathcal{E}}_k\}, \tag{1}$$

where $\hat{e}_k$ is the event statement, $\hat{\mathcal{A}}_k$ is a set of causal attribution tuples, and $\hat{\mathcal{E}}_k$ is an evidence index set.

Each attribution tuple is represented as

$$\hat{a}_{k,j} = (\hat{c}_{k,j}, \hat{r}_{k,j}, \hat{o}_{k,j}), \tag{2}$$

where $\hat{c}_{k,j}$ is a candidate cause, $\hat{o}_{k,j}$ is an outcome, and $\hat{r}_{k,j}$ is the causal relation type. The evidence set $\hat{\mathcal{E}}_k$ maps each tuple to supporting temporal spans or frame references.

3.2 CERA Design Requirements

CERA is designed around three joint requirements.

Causal Fidelity. Predictions should preserve cause-effect direction and avoid descriptive but non-causal statements. Operationally, this means the attribution tuples should match reference causal structure, not only lexical overlap.

Evidential Grounding. Every attribution claim should be linked to observable evidence in the same event window. Ungrounded claims are treated as low-trust outputs regardless of language fluency.

Runtime Efficiency. Inference must satisfy deployment latency constraints under continuous streams. Given an end-to-end latency L_{e2e} and deployment budget L_{max}, CERA should operate in the feasible region:

$$L_{e2e} \leq L_{max}. \tag{3}$$

3.3 Structured Objective

We model CERA as maximizing a utility that balances causal quality, evidence quality, and runtime feasibility:

$$\mathcal{U} = \lambda_c S_c + \lambda_e S_e + \lambda_r S_r, \tag{4}$$

where S_c measures causal fidelity, S_e measures evidence grounding quality, and S_r measures runtime fitness.

For runtime fitness, we use a normalized score:

$$S_r = \min\left(1, \frac{L_{max}}{L_{e2e}}\right). \tag{5}$$

This formulation encourages quality gains only when latency remains practical.

3.4 Online Inference Contract

Given an event window $\mathbf{X}_{t_s^k:t_e^k}$, CERA predicts:

$$\hat{\mathbf{y}}_k = \mathcal{F}_\theta\big(\mathbf{X}_{t_s^k:t_e^k}\big)\,, \tag{6}$$

with confidence score p_k. To reduce high-confidence hallucination risk in safety contexts, CERA uses abstention:

$$\hat{\mathbf{y}}_k = \begin{cases} \mathcal{F}_\theta(\mathbf{X}_{t_s^k:t_e^k}) & \text{if } p_k \geq \tau_{conf}, \\ \varnothing & \text{otherwise.} \end{cases} \tag{7}$$

This contract makes failure modes explicit and prevents forcing a causal narrative when evidence is weak.

3.5 Output Schema and Validation

For deployment integration, CERA exposes a fixed output schema:

$$\hat{\mathbf{y}}_k = \{\hat{e}_k, \hat{\mathcal{A}}_k, \hat{\mathcal{E}}_k, \hat{s}_k\}, \tag{8}$$

where $\hat{s}_k$ is a quality status flag (`valid`, `abstain`, or `insufficient_evidence`).

Before returning `valid`, CERA applies causal and evidence validators. For each predicted tuple $\hat{a}_{k,j} = (\hat{c}_{k,j}, \hat{r}_{k,j}, \hat{o}_{k,j})$, we define:

$$\mathbb{I}_{k,j}^{rel} = \mathbf{1}[\hat{r}_{k,j} \in \mathcal{R}_{allow}]\,, \quad \mathbb{I}_{k,j}^{dir} = \mathbf{1}[t(\hat{c}_{k,j}) \leq t(\hat{o}_{k,j}) + \delta_{dir}]\,, \tag{9}$$

$$\mathbb{I}_{k,j}^{ent} = \mathbf{1}\big[\hat{c}_{k,j}, \hat{o}_{k,j} \in \mathcal{V}_k^{track}\big]\,. \tag{10}$$

The causal validator is:

$$\text{CausalCheck}(\hat{\mathcal{A}}_k) = \prod_j \left(\mathbb{I}_{k,j}^{rel}\mathbb{I}_{k,j}^{dir}\mathbb{I}_{k,j}^{ent}\right). \tag{11}$$

For evidence links $\hat{\mathcal{E}}_{k,j}$ and support scores $q_{k,j}$:

$$\mathbb{I}_{k,j}^{sup} = \mathbf{1}[q_{k,j} \geq \tau_{evd}]\,, \quad \mathbb{I}_{k,j}^{link} = \mathbf{1}\Big[|\hat{\mathcal{E}}_{k,j}| \geq 1\Big]\,, \tag{12}$$

with

$$\text{EvidenceCheck}(\hat{\mathcal{A}}_k, \hat{\mathcal{E}}_k) = \prod_j \left(\mathbb{I}_{k,j}^{sup}\mathbb{I}_{k,j}^{link}\right). \tag{13}$$

Final status is determined by confidence, validation, and runtime budget:

$$\hat{s}_k = \begin{cases} \texttt{valid}, & \text{if } p_k \geq \tau_{conf},\ \text{CausalCheck} = 1, \\ & \quad \text{EvidenceCheck} = 1,\ L_{e2e}^k \leq L_{max} \\ \texttt{insufficient\_evidence}, & \text{if } p_k \geq \tau_{conf},\ L_{e2e}^k \leq L_{max}, \\ & \quad (\text{CausalCheck} = 0 \vee \text{EvidenceCheck} = 0) \\ \texttt{abstain}, & \text{otherwise.} \end{cases} \tag{14}$$

This policy makes safety behavior explicit: uncertain or ungrounded claims are not promoted to `valid`.

3.6 Failure Taxonomy

To keep failure analysis actionable, we partition output errors into three types:

- **Causal Structure Violation**: cause-effect direction or relation type is wrong.
- **Evidence Support Failure**: claim is plausible but unsupported by linked evidence.
- **Runtime Budget Breach**: quality is acceptable but latency violates deployment budget.

This taxonomy maps directly to the three CERA requirements and guides where to improve: reasoning logic, evidence alignment, or system optimization.

3.7 Module-Level Design

We decompose CERA into four cooperative modules:

$$\hat{\mathbf{y}}_k = \mathcal{M}_{ctrl}\big(\mathcal{M}_{evd}\big(\mathcal{M}_{attr}\big(\mathcal{M}_{evt}\big(\mathbf{X}_{t_s^k:t_e^k}\big)\big)\big)\big)\,. \tag{15}$$

This decomposition keeps responsibilities explicit and aligns each module with one or more CERA requirements.

Event Proposal Module ($\mathcal{M}_{evt}$). The event proposal stage generates candidate temporal windows and tracked entities from the stream. Its goal is high recall under bounded latency, producing a compact event context for downstream reasoning instead of processing the full stream at maximum cost.

Attribution Construction Module ($\mathcal{M}_{attr}$). Given event context, CERA builds a directed causal interaction graph

$$\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k^c), \tag{16}$$

where nodes represent entities/outcomes and directed edges represent candidate causal relations. Final attribution tuples are decoded from validated graph edges, enforcing direction consistency by construction.

Evidence Alignment Module ($\mathcal{M}_{evd}$). For each candidate tuple, CERA links supporting evidence spans in the same event window. Each claim receives an evidence support score $q_{k,j}$ and is retained only if $q_{k,j} \geq \tau_{evd}$. This stage prevents fluent but ungrounded attributions from passing as valid outputs.

Budget-Aware Control Module ($\mathcal{M}_{ctrl}$). The control module manages compute allocation and output policy under runtime budgets. It applies adaptive depth, selective decoding, and abstention triggers to keep latency within L_{max} while preserving causal and evidential quality. When budget pressure is high, the controller prioritizes reliable partial outputs over unstable full attributions.

3.8 Reference Instantiation (CERA-Ref)

To make CERA implementation-ready, we define a reference stack (**CERA-Ref**) with concrete but replaceable components.

Event Backbone. CERA-Ref supports two detector backbones with different deployment priorities: (1) a lightweight one-stage detector (YOLOv8-Nano class) for strict latency budgets, and (2) a transformer detector (DETR-R50 class) for stronger global scene matching. These detector families are representative of efficient one-stage and global matching paradigms [4,1]. For frame $\mathbf{X}_t$, the detector returns boxes $\mathcal{B}_t$ and risk scores $\mathcal{S}_t$, and triggers downstream reasoning only when the event score exceeds a gate threshold.

Reasoning Backbone. For attribution generation, CERA-Ref uses an open-source instruction-tuned 7B VLM with a 336px visual encoder setting. In our reference profile, this corresponds to a LLaVA-family backbone [7,6]. This choice keeps the reasoning backbone reproducible while preserving enough capacity for causal language outputs.

Detection-Guided Token Compaction. Before attribution decoding, CERA-Ref applies detection-guided visual token selection:

$$\mathcal{T}_t^{keep} = \mathcal{T}(\mathcal{R}_t) \cup \mathcal{T}(\text{Dilate}(\mathcal{R}_t, \alpha)), \tag{17}$$

where $\mathcal{R}_t$ is the detector-derived ROI set and α is a context dilation factor. This keeps hazard-centric tokens and a thin context ring while pruning background-heavy regions.

Budgeted Decoding. During generation, CERA-Ref constrains active KV usage by a budget ratio ρ_{kv}:

$$K_{active} = \lceil \rho_{kv} \cdot K_{full} \rceil, \quad 0 < \rho_{kv} \leq 1. \tag{18}$$

The controller selects the top-K_{active} entries via query-conditioned saliency and falls back to abstention or partial report if quality checks fail.

3.9 Optimization Strategy

CERA-Ref follows a staged optimization schedule that separates recall, reasoning, and runtime control.

Stage 1: Event Proposal Calibration. The detector is tuned with risk-aware weighting:

$$\mathcal{L}_{evt} = \sum_c \lambda_c \mathcal{L}_{det}^{(c)}, \tag{19}$$

where higher-risk classes use larger λ_c to preserve recall on safety-critical events.

Stage 2: Attribution and Evidence Learning. Given triggered windows, the reasoning module optimizes attribution and evidence objectives jointly:

$$\mathcal{L}_{reason} = \mathcal{L}_{attr} + \gamma \mathcal{L}_{evd}. \tag{20}$$

$\mathcal{L}_{attr}$ supervises relation direction/type and tuple content, while $\mathcal{L}_{evd}$ supervises support validity for each tuple.

Stage 3: Budget Controller Tuning. Controller parameters $(\tau_{conf}, \tau_{evd}, \rho_{kv})$ are tuned against deployment constraints by maximizing utility under latency limits:

$$\max \mathcal{U} \quad \text{s.t.} \quad L_{e2e} \leq L_{max}. \tag{21}$$

This stage determines the operating point between conservative abstention and aggressive throughput.

3.10 Online Inference Procedure

At deployment time, CERA-Ref runs the following sequence for each incoming stream window:

1. run event proposal and gate non-event windows early,
2. build attribution candidates from retained event context,
3. attach evidence spans and compute support scores,
4. apply schema and consistency checks,
5. enforce runtime budget policy to output `valid`, `insufficient_evidence`, or `abstain`.

This pipeline operationalizes CERA as a deterministic contract rather than free-form generation.

3.11 Current Scope

This manuscript now defines CERA at task and objective levels with concrete module design. It also specifies a reference instantiation for implementation and a projected result profile for planning. The remaining stage is external-server execution that replaces projected tables with measured outcomes.

4 Experiments

Draft Status Notice. All quantitative values in this section are **projected** for planning and discussion only. No claims in this section are from executed runs yet; real values will be filled after external-server experiments.

Table 1. Projected benchmark scale for planning (non-empirical).

Dataset	Train windows	Val windows	Test windows
UCF-Crime derived	1,920	640	960
XD-Violence derived	2,560	768	1,024

4.1 Evaluation Targets

We evaluate CERA along three axes aligned with the method objective:

- **Causal Quality**: correctness of predicted attribution tuples.
- **Evidence Quality**: correctness and completeness of linked evidence references.
- **Runtime**: end-to-end latency and effective throughput under fixed hardware.

Benchmark design is based on established surveillance anomaly datasets [12,15] with additional attribution/evidence labels.

4.2 Benchmark and Annotation Protocol

For each video, we derive event windows with fixed temporal stride and event-aware boundary checks. Each evaluation window is annotated with:

- event statement target e^*,
- attribution tuple set $\mathcal{A}^*$ with directed relation labels,
- evidence span set $\mathcal{E}^*$ linking tuples to frame ranges.

To reduce annotation drift, each sample is reviewed by two annotators and adjudicated under a fixed causal direction guideline. For planning-scale projection, we set the benchmark size as follows. The same plan assumes projected annotation agreement of $\kappa = 0.81$ for relation direction and span-level overlap agreement of 0.74 for evidence links.

4.3 Baseline Families

We compare CERA against three baseline groups:

- **Detection-centric**: surveillance anomaly baselines focused on event scoring [14,2].
- **VLM-adapted anomaly systems**: models that couple language outputs with anomaly detection [16,3,17].
- **Video-language reasoning**: general long-video reasoning models [8,5]. We additionally include long-context variants [10,11].

Within CERA, we report profile variants (YOLO-class and DETR-class detectors) under identical downstream attribution/evidence modules.

4.4 Metric Definitions

Let $\hat{\mathcal{A}}$ and $\mathcal{A}^*$ be predicted and reference attribution tuples. Tuple matching requires aligned cause, outcome, and relation direction. With matched pair set $\mathcal{M}_A$:

$$P_A = \frac{|\mathcal{M}_A|}{|\hat{\mathcal{A}}|}, \quad R_A = \frac{|\mathcal{M}_A|}{|\mathcal{A}^*|}, \quad F1_A = \frac{2P_A R_A}{P_A + R_A}. \tag{22}$$

For evidence, let $\hat{\mathcal{E}}$ and $\mathcal{E}^*$ denote predicted and reference support links. We compute support precision/recall/F1 analogously as $P_E, R_E, F1_E$. Runtime is reported by latency, throughput, and budget-feasibility rate:

$$R_{feas} = \frac{1}{K} \sum_{k=1}^{K} \mathbf{1}\left[L_{e2e}^{k} \leq L_{max}\right]. \tag{23}$$

To evaluate status-policy behavior, we also report:

$$R_{valid} = \frac{1}{K} \sum_{k=1}^{K} \mathbf{1}[\hat{s}_k = \texttt{valid}], \tag{24}$$

$$R_{unsafe} = \frac{\sum_{k=1}^{K} \mathbf{1}[\hat{s}_k = \texttt{valid} \wedge (\text{TupleMatch}_k = 0 \vee \text{EvidenceMatch}_k = 0)]}{\sum_{k=1}^{K} \mathbf{1}[\hat{s}_k = \texttt{valid}] + \epsilon}. \tag{25}$$

Lower R_{unsafe} indicates safer deployment-time behavior under the same coverage. We additionally report mean utility $\bar{\mathcal{U}}$ to summarize quality–latency trade-offs in one score.

4.5 Protocol Principles

To keep comparisons meaningful, all compared systems are assigned the same runtime protocol: hardware class, input resolution, decoding length, and reporting policy. Detector-swap comparisons (YOLO-class vs DETR-class) keep the rest of the CERA pipeline fixed.

4.6 Implementation and Reproducibility

All reference settings are versioned in CERA configs. We use ‘base.yaml‘ for the YOLO profile and ‘base_detr.yaml‘ for the DETR profile. The default profile uses YOLO-class event proposal and an open 7B VLM backbone; the DETR profile swaps only detector-related parameters and adjusted latency budget. In the default reference profile, key control parameters are initialized as $\tau_{conf} = 0.45$, $\tau_{evd} = 0.55$, $\rho_{kv} = 0.25$, and $L_{max} = 120$ms. For reproducibility, we fix random seeds, report configuration files used per run, and release per-video prediction dumps for attribution and evidence outputs. The external execution target is a single high-memory GPU server with unchanged protocol settings.

Table 2. Projected main results for planning only (non-empirical). Higher is better for $F1_A$, $F1_E$, R_{feas}, R_{valid}; lower is better for latency and R_{unsafe}.

Method	$F1_A$	$F1_E$	R_{feas}	Latency (ms)	R_{valid}	R_{unsafe}
RTFM + template attribution	24.1	18.7	0.97	34	0.88	0.43
AnomalyGPT-style VLM adaptation	39.8	31.4	0.63	182	0.71	0.27
Video-LLaVA-style reasoning baseline	42.7	34.5	0.58	205	0.69	0.25
CERA-Ref (YOLO profile)	55.9	49.2	0.91	108	0.74	0.11
CERA-Ref (DETR profile)	58.1	51.0	0.78	142	0.72	0.09

Table 3. Projected ablation deltas against full CERA-Ref (YOLO), planning only (non-empirical).

Setting	$\Delta F1_A$	$\Delta F1_E$	ΔR_{feas}	ΔR_{unsafe}
No Event Gating	+0.8	+0.4	-0.29	+0.02
No Token Compaction	+0.3	+0.2	-0.17	+0.01
No Budgeted Decoding	+1.0	+0.7	-0.36	+0.00
No Evidence Gate	+1.4	-6.8	+0.01	+0.14

4.7 Statistical Reporting

This draft reports projected point estimates only. When measured runs become available, we will attach mean/std over multiple seeds, 95% bootstrap confidence intervals for $F1_A$, $F1_E$, and R_{feas}, and paired significance tests on per-video metrics.

4.8 Projected Main Results (Non-Empirical)

Table 2 summarizes projected outcomes under the unified protocol. The projected trend is that CERA-Ref improves attribution/evidence quality while preserving deployment feasibility, with DETR providing slightly higher quality at higher latency.

4.9 Projected Ablations (Non-Empirical)

We project component effects relative to full CERA-Ref (YOLO) in Table 3. These projected deltas indicate that evidence gating is the most important safety component, while event gating and budgeted decoding are primary contributors to runtime feasibility.

4.10 Planned Ablations

To validate each design choice, we plan a component-wise ablation suite:

– **Full CERA-Ref**: event proposal + attribution + evidence alignment + budget controller.

- **Detector Backbone Swap**: YOLO-class vs DETR-class under identical downstream settings.
- **No Event Gating**: process all windows to measure temporal-efficiency contribution.
- **No Token Compaction**: disable detection-guided token selection.
- **No Budgeted Decoding**: use dense decoding to isolate controller impact.
- **No Evidence Gate**: keep attribution without evidence-threshold filtering.

The external-server run will replace projected deltas with measured deltas and utility changes.

4.11 Projected Error Breakdown (Non-Empirical)

For failure analysis planning, we project error composition in invalid or downgraded cases as:

- **Evidence-link miss (46%)**: tuples are generated without sufficient grounded support.
- **Causal direction inversion (29%)**: cause/outcome order flipped under multi-agent interactions.
- **Entity-role ambiguity (25%)**: interacting entities are detected but role assignment is unstable.

This breakdown guides implementation priority for the first external run.

5 Threats, Limitations, and Deployment Notes

Projected-vs-measured gap. All tables in this draft are projected priors, not executed measurements. Therefore, relative orderings and magnitudes in Section 4 may change after external-server execution.

Dataset transfer risk. Projected numbers assume that annotation quality and event taxonomy are consistent across collection domains. Real deployments may present unseen camera placement, motion blur, and nighttime conditions that alter both attribution and evidence performance.

Annotation uncertainty. Causal tuple boundaries and evidence spans include inherent subjectivity even under adjudication rules. Measured reporting will include agreement summaries and per-category error slices to separate model error from annotation ambiguity.

Operational deployment constraints. Latency feasibility depends on hardware profile, queue pressure, and stream fan-out. The external runbook fixes these settings for reproducibility; production deviations should be audited against the same reporting contract.

Measured replacement protocol. When measured outputs are available, projected blocks marked by `MEASURED_SWAP_START/END` in Section 4 will be replaced first. This minimizes narrative drift and keeps the main manuscript aligned with execution artifacts.

6 Conclusion

This paper established **CERA** as a practical objective for real-time surveillance understanding: produce causal event reports that are structurally correct, evidentially grounded, and operationally feasible. We formalized this objective through (1) a structured task definition, (2) a utility that balances causal quality, grounding quality, and runtime fitness, (3) an online contract with abstention and explicit status outputs, and (4) a concrete module-level design for event proposal, attribution construction, evidence alignment, and budget-aware control. We also completed a projected evaluation package with non-empirical benchmark scale, comparative tables, and ablation trends to guide execution.

Projected numbers indicate a consistent target pattern: stronger attribution/evidence quality than detection- or caption-centric baselines with controlled safety behavior (R_{unsafe} reduction), while maintaining feasible latency in the YOLO profile. These values are planning priors only. The remaining step is to run the same protocol on an external high-compute server and replace each projected figure with measured evidence.

References

1. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-End Object Detection with Transformers. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020) 6
2. Feng, J.C., Hong, F.T., Zheng, W.S.: MIST: Multiple Instance Self-Training Framework for Video Anomaly Detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2021) 1, 2, 8
3. Gu, Z., Zhu, B., Zhu, G., Chen, Y., Tang, M., Wang, J.: AnomalyGPT: Detecting Industrial Anomalies using Large Vision-Language Models. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI) (2024) 1, 2, 8
4. Jocher, G., Chaurasia, A., Qiu, J.: YOLOv8. GitHub repository (2023), https://github.com/ultralytics/ultralytics 6
5. Lin, B., Zhu, B., Ye, Y., Ning, M., Jin, P., Yuan, L.: Video-LLaVA: Learning United Visual Representation by Alignment Before Projection. In: Proceedings of the European Conference on Computer Vision (ECCV) (2024) 1, 2, 8
6. Liu, H., Li, C., Li, Y., Lee, Y.J.: Improved Baselines with Visual Instruction Tuning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2024) 6
7. Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual Instruction Tuning. In: Advances in Neural Information Processing Systems (NeurIPS) (2023) 6
8. Maaz, M., Rasheed, H., Khan, S., Khan, F.S.: Video-ChatGPT: Towards Detailed Video Understanding via Large Vision and Language Models. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL) (2024) 1, 2, 8

9. Nayak, R., Pati, U.C., Das, S.K.: A Survey on Deep Learning Based Video Anomaly Detection. IEEE Transactions on Circuits and Systems for Video Technology (2021) 1, 2
10. Ren, S., Yao, L., Li, S., Sun, X., Hou, L.: TimeChat: A Time-sensitive Multimodal Large Language Model for Long Video Understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2024) 1, 2, 8
11. Song, E., Chai, W., Wang, G., Zhang, Y., Zhou, H., Wu, F., Guo, X., Ye, T., Lu, Y., Hwang, J.N., Gao, G.: MovieChat: From Dense Token to Sparse Memory for Long Video Understanding. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2024) 1, 2, 8
12. Sultani, W., Chen, C., Shah, M.: Real-world Anomaly Detection in Surveillance Videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 2, 8
13. Tian, Y., Zhang, X., Werghi, N., Abdullah, A., et al.: A Survey of Video Analytics for Smart Cities. IEEE Access (2020) 1
14. Tian, Y., Pang, G., Chen, Y., Singh, R., Verjans, J.W., Carneiro, G.: Weakly-supervised Video Anomaly Detection with Robust Temporal Feature Magnitude Learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2021) 1, 2, 8
15. Wu, P., Liu, J., Shi, Y., Sun, Y., Shao, F., Wu, Z., Yang, Z.: Not Only Look, But Also Listen: Learning Multimodal Violence Detection Under Weak Supervision. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020) 2, 8
16. Wu, P., Zhou, X., Pang, G., Sun, Y., Liu, J., Wang, P., Zhang, Y.: VADCLIP: Adapting Vision-Language Models for Weakly Supervised Video Anomaly Detection. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI) (2024) 1, 2, 8
17. Zhang, H., Xu, X., Wang, X., Zeng, J., Li, C., Chen, X.: Holmes-VAD: Towards Unbiased and Explainable Video Anomaly Detection via Multi-modal LLM. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2025) 1, 2, 8